Guide de l’utilisateur

AWS Glue

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

AWS Glue: Guide de l’utilisateur

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Les marques et la présentation commerciale d’Amazon ne peuvent être utilisées en relation avec un produit ou un service qui n’est pas d’Amazon, d’une manière susceptible de créer une confusion parmi les clients, ou d’une manière qui dénigre ou discrédite Amazon. Toutes les autres marques commerciales qui ne sont pas la propriété d’Amazon appartiennent à leurs propriétaires respectifs, qui peuvent ou non être affiliés ou connectés à Amazon, ou sponsorisés par Amazon.

Table of Contents

Qu'est-ce que c'est AWS Glue ? ... 1
Fonctionnalités d'AWS Glue ... 2
En savoir plus sur les innovations dans AWS Glue ... 4
Démarrer avec AWS Glue ... 4
Accès à AWS Glue ... 4
Services connexes ... 5
AWS Glue fin du support pour Ray ... 5
Migration d'une tâche Ray vers Amazon Elastic Kubernetes Service ... 6
Comment ça marche ... 18
Travaux ETL sans serveur exécutés en isolation ... 19
Concepts ... 20
AWS Glue terminologie ... 23
Éléments ... 26
AWS Glue console ... 27
AWS Glue Data Catalog ... 27
AWS Glue chenilles et classificateurs ... 28
AWS Glue Opérations ETL ... 29
Streaming d'ETL en AWS Glue ... 29
Le système AWS Glue d'emplois ... 29
Composants ETL visuels ... 30
AWS Glue pour Spark et AWS Glue pour Ray ... 36
Qu'est-ce que c'est AWS Glue pour Ray ? ... 37
Conversion de schémas semi-structurés en schémas relationnels ... 38
AWS Types de colle ... 40
AWS Types de catalogues de données Glue ... 40
Types dans les scripts AWS Glue with Spark ... 40
AWS Types de Glue Crawler ... 41
Prise en main ... 42
Présentation de l'utilisation AWS Glue ... 42
Configuration des autorisations IAM ... 44
Étapes suivantes ... 49
Configuration pour AWS Glue Studio ... 49
Mise en route avec les blocs-notes dans AWS Glue Studio ... 62
Configuration des profils d'utilisation ... 65

Gestion de profils d'utilisation 66
Profils d'utilisation et tâches 78
Mise en route avec l'AWS Glue Data Catalog 79
Présentation de 79
Étape 1 : Créer une base de données 79
Étape 2. Création d'une table 81
Étapes suivantes 82
Configuration de l'accès réseau aux magasins de données 86
Configuration d'un VPC pour se connecter à PyPI pour AWS Glue 87
Configuration du DNS de votre VPC 89
Configuration du chiffrement 90
Configuration du réseau pour le développement 94
Configuration de votre réseau pour un point de terminaison de développement 94
Configuration d'Amazon EC2 pour un serveur de bloc-notes 96
Découverte et catalogage des données 98
Remplissage du catalogue de données 101
À l'aide d'un AWS Glue crawler 102
Définition manuelle des métadonnées 194
Intégration aux tables Amazon S3 215
Intégration à d'autres AWS services 226
Paramètres du catalogue de données 227
Remplissage et gestion des tables transactionnelles 230
Création de tables Iceberg 231
Optimisation des tables Iceberg 234
Optimisation des performances des requêtes pour les tables Iceberg 277
Gestion du catalogue de données 280
Mise à jour du schéma et ajout de nouvelles partitions 281
Optimisation des performances des requêtes à l'aide des statistiques de colonne 288
Chiffrement de votre catalogue de données 315
Sécurisation de votre catalogue de données à l'aide de Lake Formation 315
Utilisation des AWS Glue Data Catalog vues 316
Accès au catalogue de données 320
Connexion au catalogue de données à l'aide du point de terminaison AWS Glue Iceberg REST 321
Connexion au catalogue de données à l'aide du point de terminaison de l'extension AWS Glue Iceberg REST 325

AWS Glue REST APIs pour Apache Iceberg 325
Connexion au catalogue de données à partir d'une application Spark autonome 346
Mappage de données entre Amazon Redshift et Apache Iceberg 347
Considérations et limites relatives à l'utilisation du catalogue REST d' AWS Glue Iceberg APIs 348
Bonnes pratiques en matière de catalogue de données 349
Surveillance des métriques d'utilisation du catalogue de données dans Amazon CloudWatch .. 350
Vue d'ensemble des métriques du catalogue de données 351
Ajouter des métriques à votre CloudWatch tableau de bord 351
Registre de schémas AWS Glue 353
Schémas 354
Registres 356
Gestion des versions et compatibilité des schémas 357
Bibliothèques Serde en open source 363
Quotas du registre des schémas 363
Comment ça marche 364
Prise en main 366
Connexion aux données 422
Présentation de l'utilisation des connecteurs et des connexions 422
Connexions unifiées 423
Types d'authentification pris en charge 427
Considérations 428
Connexions disponibles 428
Limitations 431
Connexions à l'API REST 432
Création d'un type de connexion REST dans AWS Glue 432
AWS Glue propriétés de connexion 434
Propriétés de connexion requises 435
Propriétés de connexion JDBC 436
Propriétés de connexion MongoDB et MongoDB Atlas 441
Propriétés de connexion Salesforce 442
Connexion Snowflake 443
Connexion Vertica 444
Connexion SAP HANA 445
Connexion Azure SQL 446
Connexion Teradata Vantage 446

OpenSearch Connexion au service 447
Connexion Azure Cosmos 448
Propriétés de connexion SSL 449
Propriétés de connexion Kafka pour l'authentification 452
BigQuery Connexion Google 453
Connexion Vertica 444
Stockage des informations de connexion dans AWS Secrets Manager 453
Ajouter une AWS Glue connexion 454
Connexion à Adobe Analytics 457
Connexion à Adobe Marketo Engage 465
Connexion à Amazon Redshift 477
Connexion à Asana 482
Connexion à Azure Cosmos DB 492
Connexion à Azure SQL 495
Connexion à Blackbaud Raiser's Edge NXT 498
Connexion à CircleCI 508
Connexion à Datadog 520
Connexion à Docusign Monitor 528
Connexion à Domo 535
Connexion à Dynatrace 544
Connexion à Facebook Ads 551
Connexion à Facebook Page Insights 559
Connexion à Freshdesk 581
Connexion à Freshsales 592
Connexion à Google Ads 598
Connexion à Google Analytics 4 626
Connexion à Google BigQuery 639
Connexion à Google Search Console 644
Connexion à Google Sheets 654
Connexion à HubSpot 661
Connexion à Instagram Ads 690
Connexion à Intercom 698
Connexion à Jira Cloud 720
Connexion à Kustomer 746
Connexion à LinkedIn 774
Connexion à Mailchimp. 783

Connexion à Microsoft Dynamics 365 CRM 793
Connexion à Microsoft Teams 802
Connexion à Mixpanel 813
Connexion à Monday 828
Connexion à MongoDB 837
Connexion à Oracle NetSuite 841
Connexion au OpenSearch service 854
Connexion à Okta 857
Connexion à PayPal 866
Connexion à Pendo 873
Connexion à Pipedrive 882
Connexion à Productboard 891
Connexion à QuickBooks 900
Connexion à une API REST 908
Connexion à Salesforce 916
Connexion à Salesforce Marketing Cloud 929
Connexion à Salesforce Commerce Cloud 954
Connexion à Salesforce Marketing Cloud Account Engagement 965
Connexion à SAP HANA 978
Connexion à SAP OData 982
Connexion à SendGrid 1006
Connexion à ServiceNow 1016
Connexion à Slack 1028
Connexion à Smartsheet 1040
Connexion à Snapchat Ads 1052
Connexion à Snowflake 1061
Connexion à Stripe 1065
Connexion à Teradata 1114
Connexion à Twilio 1118
Connexion à Vertica 1137
Connexion à WooCommerce 1141
Connexion à Zendesk 1169
Connexion à Zoho CRM 1198
Connexion à Zoom Meetings 1209
Ajout d'une connexion JDBC à l'aide de vos propres pilotes JDBC 1219
Utilisation de connecteurs personnalisés et de connexions 1224

Création de connecteurs personnalisés 1226
Création de connexions pour les connecteurs 1231
Création de tâches avec des connecteurs personnalisés 1239
Gestion des connecteurs et des connexions 1247
Développement de connecteurs personnalisés 1250
Restrictions d'utilisation des connecteurs et des connexions dans AWS Glue Studio Glue Studio 1253
Test d'une connexion AWS Glue 1253
Configuration des AWS appels pour qu'ils passent par votre VPC 1254
Connexion à un magasin de données JDBC dans un VPC 1255
Accès à des données de VPC par l'intermédiaire d'interfaces réseau Elastic 1255
Propriétés d'interface réseau Elastic 1256
Utilisation d'une connexion MongoDB ou MongoDB Atlas 1257
Analyse d'un magasin de données Amazon S3 à l'aide d'un point de terminaison d'un VPC ... 1258
Conditions préalables 1258
Création de la connexion à Amazon S3 1259
Test de la connexion à Amazon S3 1261
Création d'un crawler pour un magasin de données Amazon S3 1263
Création d'un crawler pour les tables du catalogue de données basées sur Amazon S3 1265
Exécution d'un crawler 1266
Résolution des problèmes 1267
Travailler avec des AWS Lake Formation données protégées 1267
Accès complet aux tables 1267
Dépannage des problèmes de connexion 1268
Tutoriel : Utilisation du AWS Glue connecteur pour Elasticsearch 1269
Conditions préalables 1270
Étape 1 : (Facultatif) Créez un AWS secret pour les informations de votre OpenSearch cluster 1270
Étape 2 : S'abonner au connecteur 1271
Étape 3 : activer le connecteur AWS Glue Studio et créer une connexion 1272
Étape 4 : Configurer un rôle IAM pour votre tâche ETL 1273
Étape 5 : créer une tâche qui utilise la OpenSearch connexion 1274
Étape 6 : Exécuter la tâche 1275
Création AWS Glue d'emplois grâce à des sessions interactives 1277
Présentation de séances interactives AWS Glue 1277
Limitations 1278

Démarrage avec séances interactives AWS Glue 1278
Conditions préalables à la configuration locale des séances interactives 1278
Installation de Jupyter et de sessions AWS Glue interactives (noyaux Jupyter) 1278
Exécution de Jupyter 1279
Configuration des informations d'identification de séance et de région 1279
Mise à niveau à partir de l'aperçu des séances interactives 1281
Utilisation de sessions interactives avec SageMaker AI Studio 1281
Utilisation des séances interactives avec Microsoft Visual Studio Code 1282
Séances interactives avec IAM 1285
Configuration des Séances interactives AWS Glue pour Jupyter et Blocs-notes AWS Glue Studio 1294
Présentation de Jupyter magics 1294
Des magics soutenus par les séances interactives AWS Glue pour Jupyter 1294
Désignation des séances 1313
Spécification d'un rôle IAM pour les séances interactives 1314
Configuration des séances avec des profils nommés 1315
Conversion d'un script ou d'un bloc-notes en AWS Glue tâche 1316
Utilisation des opérations de streaming dans le cadre de sessions interactives AWS Glue 1316
Changement de type de séance de streaming 1316
Échantillonnage du flux d'entrée pour un développement interactif 1316
Exécution d'applications de streaming dans des séances interactives 1318
Tarif des sessions interactives AWS Glue 1319
Configuration de vos sessions interactives AWS Glue 1319
Développement et test locaux 1320
Développement à l'aide AWS de Glue Studio 1321
Développement à l'aide de sessions interactives 1321
Développer des AWS Glue emplois au niveau local avec Docker 1321
Points d'extrémité du développeur 1331
Migration des points de terminaison de développement vers des séances interactives 1333
Développement de scripts à l'aide de points de terminaison de développement 1335
Gestion des blocs-notes 1365
Création de tâches ETL visuelles 1367
Créez des tâches ETL visuelles avec AWS Glue Studio 1367
Se connecter à la console 1367
Prochaines étapes de création d'une tâche dans AWS Glue Studio 1368
Créez des flux ETL visuels avec Amazon SageMaker 1368

Démarrage de tâches ETL visuelles dans AWS Glue Studio 1368
Démarrer une tâche dans AWS Glue Studio 1369
Fonctionnalités de l'éditeur de tâche 1371
Transformation de données avec des transformations gérées par AWS Glue 1377
Transformation des données avec des transformations visuelles personnalisées 1456
Utilisation d'infrastructures de lac de données avec AWS Glue Studio 1474
Connexion aux sources de données 1487
Configuration des nœuds de données cible 1497
Modification ou téléchargement d'un script de tâche 1503
Modification des noeuds parents d'un nœud dans le diagramme de tâche 1508
Suppression de noeuds du diagramme de tâches 1508
Ajout de paramètres source et cible au AWS Glue nœud Data Catalog 1513
Utiliser les systèmes de contrôle de version Git dans AWS Glue 1515
Créer du code avec des blocs-notes AWS Glue Studio 1524
Limitations 1525
Présentation de l'utilisation des blocs-notes 1525
Création d'une tâche ETL à l'aide de blocs-notes dans AWS Glue Studio 1526
Composants de l'éditeur de bloc-notes 1527
Enregistrement de votre bloc-notes et de votre script de tâche 1528
Gestion des séances de bloc-notes 1529
Utilisation d'Amazon Q Developer avec des blocs-notes AWS Glue Studio 1531
Surveillance de l'exécution des tâches 1531
Accès au tableau de bord de surveillance de tâche 1532
Présentation du tableau de bord de surveillance de tâche 1532
Vue des exécutions de tâche 1532
Afficher les journaux d'exécution de la tâche 1537
Affichage des détails d'une exécution de tâche 1538
Afficher Amazon CloudWatch les statistiques relatives à l'exécution d'une tâche Spark 1541
Afficher Amazon CloudWatch les statistiques relatives à l'exécution d'une tâche Ray 1542
Détecter et traiter les données sensibles 1544
Choix du mode d'analyse des données 1545
Choix des entités PII à détecter 1546
Spécification du niveau de sensibilité de détection 1550
Choix de ce qu'il faut faire avec les données PII identifiées 1551
Ajout des remplacements précis des actions 1552
Gestion des tâches 1553

Démarrer une exécution de tâche 1553
Planifier des exécutions de tâche 1554
Gestion des planifications de tâche 1555
Arrêter les exécutions de tâche 1556
Voir vos tâches 1556
Afficher les informations sur les exécutions de tâche récentes 1557
Afficher le script de tâche 1558
Modifier les propriétés de tâche 1559
Sauvegarder la tâche 1562
Cloner une tâche 1565
Supprimer une tâche 1565
Travail avec les tâches 1567
AWS Glue versions 1567
AWS Glue versions 1567
Politique de prise en charge des versions AWS Glue 1588
Migration AWS Glue des tâches Spark vers la AWS Glue version 5.1 1590
Migration AWS Glue des tâches Spark vers la AWS Glue version 5.0 1598
Migration AWS Glue des tâches Spark vers la AWS Glue version 4.0 1614
Analyse des mises à niveau avec l'IA 1629
Utilisation des tâches Spark 1646
Paramètres du job 1646
Spark et PySpark jobs 1656
Types de travailleur 1832
Tâches ETL en streaming 1837
Record correspondant à FindMatches 1852
Migration de programmes 1890
Utilisation des tâches Ray 1898
Commencer avec AWS Glue For Ray 1898
Environnements d'exécution Ray pris en charge 1899
Comptabilité pour les travailleurs dans les tâches Ray 1900
Paramètres de tâche Ray 1901
Métriques des tâches Ray 1905
Configuration des propriétés des tâches shell Python 1906
Limitations 1907
Environnement d'exécution 1907
Définition des propriétés pour les tâches shell Python 1908

Bibliothèques prises en charge pour les tâches shell Python 1910
Ajout de votre propre bibliothèque Python 1912
À utiliser AWS CloudFormation avec les jobs shell Python dans AWS Glue 1915
Migration du shell Python 3.6 vers le shell Python 3.9 1916
Migrer depuis les jobs shell AWS Glue Python 1916
Contrôle 1918
AWS balises 1919
Automatiser avec EventBridge 1925
Surveillance des ressources 1928
Journalisation à l'aide CloudTrail 1931
AWS Glue Streaming 1935
Cas d'utilisation pour le streaming 1935
Quels sont les avantages du AWS Glue streaming ? 1936
Quand utiliser le AWS Glue streaming ? 1937
Sources de données prises en charge 1938
Cibles de données prises en charge 1938
Tutoriel : Créez votre première charge de travail de streaming à l'aide de AWS Glue Studio .. 1939
Conditions préalables 1939
Utilisation des données de streaming à partir d'Amazon Kinesis 1939
Tutoriel : créez votre première charge de travail de streaming à l'aide des blocs-notes AWS Glue Studio 1950
Conditions préalables 1951
Utilisation des données de streaming à partir d'Amazon Kinesis 1951
Concepts du streaming 1958
Anatomie d'une tâche AWS Glue de streaming 1958
Connexions de Streaming 1962
Connexions Kafka 1963
Connexions Kinesis 1970
AWS Glue mise à l'échelle automatique du streaming 1977
Activer Auto Scaling dans AWS Glue Studio 1977
Activer Auto Scaling avec la AWS CLI ou le SDK 1978
Comment ça marche 1979
Fenêtres de maintenance 1981
Configuration d'une fenêtre de maintenance 1981
Comportement des fenêtres de maintenance 1982
Surveillance des tâches 1983

Gestion des pertes de données 1985
Concepts de streaming AWS Glue avancés 1986
Considérations temporelles lors du traitement des flux 1986
Fenêtrage 1987
Gestion des données tardives et des filigranes 1993
Surveillance des tâches AWS Glue de streaming 1994
Visualisation des statistiques de AWS Glue streaming 1995
Utilisation des métriques AWS Glue de streaming 1997
Comment obtenir les meilleures performances 2002
Intégrations sans ETL 2004
Fonctionnalités zéro ETL dans AWS Glue 2004
Conditions préalables 2006
Configuration des ressources source 2006
Configuration des ressources cible 2008
Création d'un entrepôt de données Amazon Redshift 2013
Configuration d'un VPC 2013
Configuration d'une intégration entre comptes 2017
Configuration d'une source 2018
Prise en charge pour les entités SAP spéciales 2018
Source Amazon DynamoDB 2019
Source Salesforce 2020
Source Salesforce MCAE 2021
Configuration Salesforce supplémentaire 2022
OData Source SAP 2022
ServiceNow source 2022
Source Zendesk 2023
Source Zoho CRM 2023
Source Facebook Ads 2025
Source Instagram Ads 2025
Champs Salesforce non pris en charge 2025
Entités SAP non prises en charge 2028
Champs non pris en charge ServiceNow 2029
Configuration d'une cible 2034
SageMaker Lakehouse avec espace de rangement S3 2035
Objectif de tableaux S3 2035
SageMaker Maison au bord du lac avec rangement Amazon Redshift 2040

Amazon Redshift cible de l'entrepôt de données ... 2041
Configuration de la cible d'intégration ... 2042
Configuration d'une intégration ... 2043
Synchronisation continue ... 2043
RefreshInterval ... 2044
Partitions et désimbrication et des schémas ... 2046
Comportement de désimbrication et de partitionnement par défaut ... 2047
Désimbrication du schéma ... 2047
Partitionnement de données ... 2051
Limitations ... 2059
Tâches courantes ... 2059
Création d'une intégration ... 2060
Modification d'une intégration ... 2061
Suppression d'une intégration ... 2062
En utilisant APIs ... 2062
Surveillance d'une intégration ... 2063
États d'intégration ... 2063
Affichage des journaux ... 2063
Affichage des métriques ... 2066
Notifications d'événements ... 2066
Limitations ... 2068
AWS Glue Qualité des données ... 2078
Avantages et fonctionnalités clés ... 2078
Comment ça marche ... 2079
Qualité des données pour AWS Glue Data Catalog ... 2079
Qualité des données pour les tâches AWS Glue ETL ... 2080
Comparaison des points d'entrée relatifs à la qualité des AWS Glue données ... 2080
Considérations ... 2082
Terminologie ... 2082
Restrictions ... 2084
Notes de mise à jour relatives à la qualité AWS Glue des données ... 2084
Disponibilité générale : nouvelles fonctionnalités ... 2084
27 novembre 2023 (aperçu) ... 2085
12 mars 2024 ... 2085
26 juin 2024 ... 2085
7 août 2024 ... 2085

22 novembre 2024 2086
6 déc. 2024 2086
7 juillet 2025 2086
21 novembre 2025 2086
Détection d'anomalies dans Qualité des données d'AWS Glue 2087
Comment ça marche 2088
Détails de l'algorithme de détection des anomalies 2092
Autorisations IAM pour AWS Glue Data Quality 2093
Autorisations IAM 2094
Configuration IAM requise pour planifier les exécutions d'évaluation 2097
Exemple de politiques IAM 2098
Commencer à utiliser AWS Glue Data Quality pour le Data Catalog 2103
Conditions préalables 2104
Step-by-step exemple 2105
Génération de recommandations de règles 2105
Recommandations sur les règles de surveillance 2106
Modification des ensembles de règles recommandés 2107
Création d'un ensemble de règles 2108
Exécution d'un jeu de règles pour évaluer la qualité des données 2110
Affichage du score de qualité des données et des résultats 2111
Utilisation de requêtes de prétraitement 2112
Types de sources pris en charge 2115
Rubriques en relation 2117
Évaluation de la qualité des données avec AWS Glue Studio 2117
Avantages 2118
Évaluation de la qualité des données pour les tâches ETL dans AWS Glue Studio 2118
Générateur de règles de qualité des données 2129
Configuration de la détection des anomalies dans les jobs AWS Glue ETL 2134
Affichage des scores de qualité des données et des anomalies 2137
Qualité des données pour les tâches ETL dans les blocs-notes AWS Glue Studio 2141
Conditions préalables 2142
Création d'une tâche ETL dans AWS Glue Studio 2142
Référence DQDL (Data Quality Definition Language) 2147
Syntaxe 2149
Référence du type de règle 2169
Utilisation APIs pour mesurer et gérer la qualité des données 2238

Conditions préalables 2238
Utilisation des recommandations relatives à AWS la qualité des données de Glue 2238
Travailler avec les ensembles de règles de qualité des données de AWS Glue 2241
Travailler avec AWS Glue Data Quality s'exécute 2244
Utilisation des résultats AWS de Glue Data Quality 2249
Configuration des alertes, des déploiements et de la planification 2255
Configuration des alertes et des notifications dans le cadre de EventBridge l'intégration Amazon 2255
Configurer des alertes et des notifications lors de l' CloudWatch intégration 2264
Interrogation des résultats de la qualité des données 2266
Déploiement des règles de qualité des données 2270
Planification de règles de qualité des données 2270
Le chiffrement des données est arrêté pour AWS Glue Data Quality 2270
AWS clés possédées 2270
Clés gérées par le client 2271
Création d'une clé gérée par le client 2274
Création d'une configuration de sécurité 2276
AWS Contexte de chiffrement de Glue Data Quality 2277
Surveillance de vos clés de chiffrement pour AWS Glue Data Quality 2278
En savoir plus 2282
Résolution des erreurs liées à la qualité des données de AWS Glue 2283
Erreur : module manquant 2284
Erreur : autorisations insuffisantes 2284
Erreur : les ensembles de règles ne sont pas uniques 2284
Erreur : tables contenant des caractères spéciaux 2284
Erreur : débordement avec un ensemble de règles volumineux 2284
Erreur : échec du statut de la règle 2285
AnalysisException: Impossible de vérifier l'existence de la base de données par défaut 2285
La clé de distribution fournie ne convient pas aux cadres de données fournies. 2286
java.lang. RuntimeException : Impossible de récupérer les données. 2286
ERREUR DE LANCEMENT : erreur lors du téléchargement depuis S3 pour le compartiment 2286
InvalidInputException (statut : 400) : DataQuality les règles ne peuvent pas être analysées 2287
Erreur : Eventbridge ne déclenche pas les tâches Glue DQ selon le calendrier que j'ai configuré. 2287

Erreurs CustomSQL 2288
Règles dynamiques 2288
Exception dans la classe utilisateur : org.apache.spark.sql. AnalysisException: org.apache.hadoop.hive.ql.metadata. HiveException 2290
UNCLASSIFIED_ERROR ; IllegalArgumentException : Erreur d'analyse : aucune règle ni analyseur fournis., aucune alternative viable en entrée 2291
Intégration des données Amazon Q dans AWS Glue 2292
Qu'est-ce qu'Amazon Q ? 2292
Intégration des données Amazon Q dans AWS Glue 2292
Utilisation de l'intégration des données Amazon Q 2293
Bonnes pratiques 2295
Amélioration des services 2296
Considérations 2296
Configuration de l'intégration des données Amazon Q 2296
Configuration des autorisations IAM 2296
Génération de code prise en charge 2299
Exemples d'interactions 2300
Interactions dans le chat Amazon Q 2300
AWS Glue Interactions avec Studio Notebook 2302
Utilisation de la prise en compte du contexte 2305
Exemple : interactions 2305
Limitations 2309
Orchestration 2310
Démarrage des tâches et des crawlers à l'aide de déclencheurs 2310
Déclencheurs AWS Glue 2310
Ajout de déclencheurs 2313
Activation et désactivation des déclencheurs 2317
Exécution d'activités ETL complexes à l'aide de plans et de flux de travail 2318
Présentation des flux de travail 2319
Création et développement d'un flux de travail manuellement 2323
Démarrer un flux de travail avec un EventBridge événement 2328
Afficher les EventBridge événements qui ont lancé un flux de travail 2336
Exécution et surveillance d'un flux de travail 2337
Arrêt d'une exécution de flux de travail 2339
Réparation et reprise de l'exécution d'un flux de travail 2340
Obtention et définition des propriétés d'exécution du flux de travail 2347

Interrogation des flux de travail à l'aide du AWS Glue API 2348
Restrictions du plan et du flux de travail 2353
Résolution des erreurs de plans 2354
Autorisations pour les personas et les rôles du plan 2359
Développement des plans 2364
Présentation des plans 2364
Développement des plans 2368
Enregistrement d'un plan 2393
Affichage des plans 2396
Mise à jour d'un plan 2397
Création d'un flux de travail à partir d'un plan 2400
Affichage des exécutions de plans 2402
AWS CloudFormation pour AWS Glue 2404
Exemple de base de données 2406
Exemple de base de données, de table, de partition 2407
Exemple de classificateur Grok 2412
Exemple de classifieur JSON 2413
Exemple de classifieur XLM 2414
Exemple d'un crawler Amazon S3 2415
Exemple de connexion 2417
Exemple de connexion 2418
Exemple de connexion 2419
Exemple d'un crawler JDBC 2421
Exemple de tâche pour Amazon S3 vers Amazon S3 2424
Exemple de tâche pour JDBC vers Amazon S3 2425
Exemple de déclencheur à la demande 2427
Exemple de déclencheur planifié 2428
Exemple de déclencheur conditionnel 2429
Exemple de transformation de machine learning 2431
Exemple d'ensemble de règles de qualité des données 2432
Exemple de règles de qualité des données avec planificateur EventBridge 2434
Exemple de point de terminaison de développement 2436
AWS Glue guide de programmation 2438
Fournir vos propres scripts personnalisés 2438
AWS Glue pour Spark 2439
Didacticiel : rédiger un script Spark 2440

ETL en PySpark ... 2453
ETL dans Scala ... 2706
Fonctionnalités et optimisations ... 2793
AWS Glue pour Ray ... 3078
Didacticiel : écrire un script Ray ... 3079
Utiliser Ray Core et Ray Data AWS Glue pour Ray ... 3086
Fournir des fichiers et des bibliothèques Python ... 3088
Connexion aux données ... 3093
Travailler avec AWS SDKs ... 3096
AWS Glue API ... 3098
Sécurité ... 3126
— les types de données — ... 3126
DataCatalogEncryptionSettings ... 3126
EncryptionAtRest ... 3127
ConnectionPasswordEncryption ... 3127
EncryptionConfiguration ... 3128
S3Encryption ... 3129
CloudWatchEncryption ... 3129
JobBookmarksEncryption ... 3129
SecurityConfiguration ... 3130
GluePolicy ... 3130
DataQualityEncryption ... 3131
— operations — ... 3131
GetDataCatalogEncryptionSettings (get_data_catalog_encryption_settings) ... 3132
PutDataCatalogEncryptionSettings (put_data_catalog_encryption_settings) ... 3132
PutResourcePolicy (put_resource_policy) ... 3133
GetResourcePolicy (get_resource_policy) ... 3135
DeleteResourcePolicy (supprimer_ressource_policy) ... 3136
CreateSecurityConfiguration (créer_configuration_sécurité) ... 3136
DeleteSecurityConfiguration (supprimer_configuration_sécurité) ... 3137
GetSecurityConfiguration (get_security_configuration) ... 3138
GetSecurityConfigurations (get_security_configurations) ... 3139
GetResourcePolicies (get_resource_policies) ... 3139
Objets du catalogue ... 3140
Catalogues ... 3141
Bases de données ... 3156

Tables 3166
Partitions 3215
Connexions 3241
Fonctions définies par l'utilisateur 3285
Importation d'un Athena catalogue 3292
Optimiseur de table 3294
— les types de données — 3294
TableOptimizer 3294
TableOptimizerConfiguration 3295
TableOptimizerVpcConfiguration 3296
CompactionConfiguration 3296
IcebergCompactionConfiguration 3297
TableOptimizerRun 3298
BatchGetTableOptimizerEntry 3299
BatchTableOptimizer 3300
BatchGetTableOptimizerError 3300
RetentionConfiguration 3301
IcebergRetentionConfiguration 3301
OrphanFileDeletionConfiguration 3302
IcebergOrphanFileDeletionConfiguration 3302
CompactionMetrics 3303
RetentionMetrics 3303
OrphanFileDeletionMetrics 3303
IcebergCompactionMetrics 3304
IcebergRetentionMetrics 3304
IcebergOrphanFileDeletionMetrics 3305
RunMetrics 3305
— operations — 3306
GetTableOptimizer (get_table_optimizer) 3306
BatchGetTableOptimizer (optimiseur de table par lots) 3307
ListTableOptimizerRuns (list_table_optimizer_runs) 3308
CreateTableOptimizer (create_table_optimizer) 3310
DeleteTableOptimizer (supprimer_table_optimiseur) 3311
UpdateTableOptimizer (optimiseur de table de mise à jour) 3312
crawlers et classifieurs 3313
Classifiers 3313

Crawlers 3327
Statistiques de colonne 3356
Planificateur 3372
Génération automatique de scripts ETL 3375
— les types de données — 3375
CodeGenNode 3375
CodeGenNodeArg 3376
CodeGenEdge 3376
Location 3377
CatalogEntry 3377
MappingEntry 3378
— operations — 3378
CreateScript (créer_script) 3379
GetDataflowGraph (get_dataflow_graph) 3379
GetMapping (get_mapping) 3380
GetPlan (get_plan) 3381
API Visual Job 3382
— les types de données — 3382
CodeGenConfigurationNode 3386
JDBCConnectorOptions 3394
StreamingDataPreviewOptions 3395
AthenaConnectorSource 3396
JDBCConnectorLa source 3397
SparkConnectorSource 3397
CatalogSource 3398
Ma SQLCatalog source 3399
Source de Postgrer SQLCatalog 3399
SQLCatalogSource d'Oracle 3400
Microsoft SQLServer CatalogSource 3400
CatalogKinesisSource 3400
DirectKinesisSource 3401
KinesisStreamingSourceOptions 3402
CatalogKafkaSource 3405
DirectKafkaSource 3406
KafkaStreamingSourceOptions 3406
RedshiftSource 3409

AmazonRedshiftSource 3409
AmazonRedshiftNodeData 3410
AmazonRedshiftAdvancedOption 3412
Option 3413
S3 CatalogSource 3413
S3 SourceAdditionalOptions 3414
S3 CsvSource 3414
Directement JDBCSource 3416
S3 DirectSourceAdditionalOptions 3417
S3 JsonSource 3418
S3 ParquetSource 3419
S3 DeltaSource 3421
S3 CatalogDeltaSource 3422
CatalogDeltaSource 3422
S3 HudiSource 3423
S3 CatalogHudiSource 3424
S3 ExcelSource 3424
CatalogHudiSource 3426
Source de dynamo DBCatalog 3426
RelationalCatalogSource 3427
JDBCConnectorCible 3427
SparkConnectorTarget 3428
BasicCatalogTarget 3429
Ma SQLCatalog cible 3430
Cible Postgrer SQLCatalog 3430
SQLCatalogCible Oracle 3431
Microsoft SQLServer CatalogTarget 3431
RedshiftTarget 3432
AmazonRedshiftTarget 3433
UpsertRedshiftTargetOptions 3433
S3 CatalogTarget 3433
S3 GlueParquetTarget 3434
CatalogSchemaChangePolicy 3435
S3 DirectTarget 3436
S3 HudiCatalogTarget 3437
S3 HudiDirectTarget 3438

S3 DeltaCatalogTarget 3439
S3 DeltaDirectTarget 3440
S3 HyperDirectTarget 3441
S3 IcebergDirectTarget 3442
DirectSchemaChangePolicy 3444
ApplyMapping 3444
Mappage 3445
SelectFields 3446
DropFields 3446
RenameField 3446
Spigot 3447
Join 3448
JoinColumn 3448
SplitFields 3449
SelectFromCollection 3449
FillMissingValues 3449
Filtre 3450
FilterExpression 3451
FilterValue 3451
CustomCode 3451
SparkSQL 3452
SqlAlias 3453
DropNullFields 3453
NullCheckBoxList 3454
NullValueField 3454
Datatype 3455
Fusionner 3455
Union 3456
PIIDetection 3456
Regrouper 3458
DropDuplicates 3459
GovernedCatalogTarget 3459
GovernedCatalogSource 3460
AggregateOperation 3460
GlueSchema 3461
GlueStudioSchemaColumn 3461

GlueStudioColumn 3461
DynamicTransform 3463
TransformConfigParameter 3463
EvaluateDataQuality 3464
DQResultsPublishingOptions 3465
DQStopJobOnFailureOptions 3466
EvaluateDataQualityMultiFrame 3466
Formule 3467
RecipeReference 3467
SnowflakeNodeData 3468
SnowflakeSource 3470
SnowflakeTarget 3471
ConnectorDataSource 3471
ConnectorDataTarget 3472
RecipeStep 3473
RecipeAction 3473
ConditionExpression 3474
S3 CatalogIcebergSource 3474
CatalogIcebergSource 3475
S3 IcebergCatalogTarget 3476
Source de dynamo DBELTConnector 3477
DDBELTConnectionOptions 3477
DDBELTCatalogAdditionalOptions 3478
Acheminement 3479
GroupFilters 3479
AutoDataQuality 3480
Tâches 3480
Tâches 3480
Job exécutés 3508
Triggers 3529
Intégration APIs 3544
— les types de données — 3544
Integration 3544
IntegrationConfig 3546
IntegrationPartition 3547
IntegrationError 3548

IntegrationFilter 3548
InboundIntegration 3549
SourceProcessingProperties 3550
TargetProcessingProperties 3550
SourceTableConfig 3550
TargetTableConfig 3551
— operations — 3551
CreateIntegration (créer_intégration) 3552
ModifyIntegration (modifier_intégration) 3555
DescribeIntegrations (décrive_intégrations) 3558
DeleteIntegration (supprimer_intégration) 3559
DescribeInboundIntegrations (décrivez les intégrations entrantes) 3561
CreateIntegrationTableProperties (create_integration_table_properties) 3563
UpdateIntegrationTableProperties (update_integration_table_properties) 3564
GetIntegrationTableProperties (get_integration_table_properties) 3565
DeleteIntegrationTableProperties (supprimer_integration_table_properties) 3566
CreateIntegrationResourceProperty (create_integration_resource_property) 3567
UpdateIntegrationResourceProperty (update_integration_resource_property) 3569
GetIntegrationResourceProperty (get_integration_resource_property) 3570
UntagResource (untag_resource) 3571
ListTagsForResource (liste_tags_pour_ressource) 3571
— exceptions — 3572
ResourceNotFoundException 3572
InternalServerException 3572
IntegrationAlreadyExistsFault 3573
IntegrationConflictOperationFault 3573
IntegrationQuotaExceededFault 3573
KMSKeyNotAccessibleFault 3573
IntegrationNotFoundFault 3574
TargetResourceNotFound 3574
InvalidIntegrationStateFault 3574
Sessions interactives 3575
— les types de données — 3575
Session 3575
SessionCommand 3577
Instruction 3578

StatementOutput 3579
StatementOutputData 3579
ConnectionsList 3579
— operations — 3580
CreateSession (créer_session) 3580
StopSession (stop_session) 3584
DeleteSession (supprimer_session) 3585
GetSession (get_session) 3585
ListSessions (liste_sessions) 3586
RunStatement (run_statement) 3587
CancelStatement (annuler_déclaration) 3588
GetStatement (get_statement) 3589
ListStatements (liste_déclarations) 3590
GetGlueIdentityCenterConfiguration (get_glue_identity_center_configuration) 3591
UpdateGlueIdentityCenterConfiguration (mise à jour de la configuration du centre d'identité de la colle) 3592
CreateGlueIdentityCenterConfiguration (create_glue_identity_center_configuration) 3593
DeleteGlueIdentityCenterConfiguration (delete_glue_identity_center_configuration) 3594
DevEndpoints 3595
— les types de données — 3595
DevEndpoint 3595
DevEndpointCustomLibraries 3599
— operations — 3600
CreateDevEndpoint (create_dev_endpoint) 3600
UpdateDevEndpoint (update_dev_endpoint) 3606
DeleteDevEndpoint (supprimer_dev_endpoint) 3607
GetDevEndpoint (get_dev_endpoint) 3608
GetDevEndpoints (get_dev_endpoints) 3609
BatchGetDevEndpoints (batch_get_dev_endpoints) 3610
ListDevEndpoints (list_dev_endpoints) 3611
Registre de schémas 3612
— les types de données — 3612
RegistryId 3612
RegistryListItem 3613
MetadataInfo 3614
OtherMetadataValueListItem 3614

SchemaListItem 3614
SchemaVersionListItem 3615
MetadataKeyValuePair 3616
SchemaVersionErrorItem 3616
ErrorDetails 3617
SchemaVersionNumber 3617
SchemaId 3617
— operations — 3618
CreateRegistry (créer_registre) 3619
CreateSchema (créer_schéma) 3620
GetSchema (get_schema) 3624
ListSchemaVersions (liste_schéma_versions) 3626
GetSchemaVersion (get_schema_version) 3628
GetSchemaVersionsDiff (get_schema_versions_diff) 3629
ListRegistries (list_registries) 3630
ListSchemas (liste_schémas) 3631
RegisterSchemaVersion (enregistre_schéma_version) 3632
UpdateSchema (schéma de mise à jour) 3634
CheckSchemaVersionValidity (check_schema_version_validity) 3635
UpdateRegistry (mise à jour du registre) 3636
GetSchemaByDefinition (get_schema_by_definition) 3637
GetRegistry (get_registry) 3639
PutSchemaVersionMetadata (put_schema_version_metadata) 3640
QuerySchemaVersionMetadata (query_schema_version_metadata) 3642
RemoveSchemaVersionMetadata (supprimer_schema_version_metadata) 3643
DeleteRegistry (supprimer_registre) 3645
DeleteSchema (supprimer_schéma) 3646
DeleteSchemaVersions (supprimer_les versions du schéma) 3647
Flux de travail 3648
— les types de données — 3648
JobNodeDetails 3649
CrawlerNodeDetails 3649
TriggerNodeDetails 3649
Crawl 3649
Nœud 3650
Edge 3651

Flux de travail 3651
WorkflowGraph 3653
WorkflowRun 3653
WorkflowRunStatistics 3654
StartingEventBatchCondition 3655
Plan 3656
BlueprintDetails 3657
LastActiveDefinition 3657
BlueprintRun 3658
— operations — 3659
CreateWorkflow (créer_flux de travail) 3660
UpdateWorkflow (flux de travail de mise à jour) 3661
DeleteWorkflow (supprimer_flux de travail) 3663
GetWorkflow (get_workflow) 3663
ListWorkflows (liste_flux de travail) 3664
BatchGetWorkflows (batch_get_workflows) 3665
GetWorkflowRun (get_workflow_run) 3666
GetWorkflowRuns (get_workflow_runs) 3666
GetWorkflowRunProperties (get_workflow_run_properties) 3667
PutWorkflowRunProperties (put_workflow_run_properties) 3668
CreateBlueprint (créer_un plan) 3670
UpdateBlueprint (mise à jour du plan) 3671
DeleteBlueprint (supprimer_le plan) 3672
ListBlueprints (liste_plans) 3672
BatchGetBlueprints (batch_get_blueprints) 3673
StartBlueprintRun (start_blueprint_run) 3674
GetBlueprintRun (get_blueprint_run) 3675
GetBlueprintRuns (get_blueprint_runs) 3676
StartWorkflowRun (start_workflow_run) 3677
StopWorkflowRun (stop_workflow_run) 3678
ResumeWorkflowRun (resume_workflow_run) 3678
Profils d'utilisation 3680
— les types de données — 3680
ProfileConfiguration 3680
ConfigurationObject 3680
UsageProfileDefinition 3681

— operations — 3682
CreateUsageProfile (créer un profil d'utilisation) 3682
GetUsageProfile (get_usage_profile) 3683
UpdateUsageProfile (mettre à jour le profil_d'utilisation) 3684
DeleteUsageProfile (supprimer_profil_d'utilisation) 3685
ListUsageProfiles (liste_profils_d'utilisation) 3685
Machine learning 3686
— les types de données — 3686
TransformParameters 3687
EvaluationMetrics 3688
MLTransform 3688
FindMatchesParameters 3691
FindMatchesMetrics 3693
ConfusionMatrix 3694
GlueTable 3695
TaskRun 3696
TransformFilterCriteria 3697
TransformSortCriteria 3698
TaskRunFilterCriteria 3698
TaskRunSortCriteria 3699
TaskRunProperties 3699
FindMatchesTaskRunProperties 3700
ImportLabelsTaskRunProperties 3700
ExportLabelsTaskRunProperties 3701
LabelingSetGenerationTaskRunProperties 3701
SchemaColumn 3701
TransformEncryption 3702
MLUserDataEncryption 3702
ColumnImportance 3703
— operations — 3703
Créer MLTransform (create_ml_transform) 3704
Mettre à jour MLTransform (update_ml_transform) 3707
Supprimer MLTransform (delete_ml_transform) 3710
Obtenir MLTransform (get_ml_transform) 3711
Obtenir MLTransforms (get_ml_transform) 3714
Liste MLTransforms (list_ml_transform) 3715

Démarrer MLEvaluation TaskRun (start_ml_evaluation_task_run) 3716
Démarrer MLLabeling SetGenerationTaskRun
(start_ml_labeling_set_generation_task_run) 3717
MLTaskExécuter (get_ml_task_run) 3718
Obtenir des MLTask courses (get_ml_task_runs) 3720
Annuler l'MLTaskexécution (cancel_ml_task_run) 3721
StartExportLabelsTaskRun (start_export_labels_task_run) 3722
StartImportLabelsTaskRun (start_import_labels_task_run) 3723
Qualité des données 3725
— les types de données — 3725
DataSource 3726
DataQualityRulesetListDetails 3726
DataQualityTargetTable 3727
DataQualityRulesetEvaluationRunDescription 3727
DataQualityRulesetEvaluationRunFilter 3728
DataQualityEvaluationRunAdditionalRunOptions 3728
DataQualityRuleRecommendationRunDescription 3729
DataQualityRuleRecommendationRunFilter 3729
DataQualityResult 3730
DataQualityAnalyzerResult 3732
DataQualityObservation 3732
MetricBasedObservation 3733
DataQualityMetricValues 3734
DataQualityRuleResult 3734
DataQualityResultDescription 3735
DataQualityResultFilterCriteria 3736
DataQualityRulesetFilterCriteria 3737
DataQualityAggregatedMetrics 3737
StatisticAnnotation 3738
TimestampedInclusionAnnotation 3739
AnnotationError 3739
DatapointInclusionAnnotation 3739
StatisticSummaryList 3740
StatisticSummary 3740
RunIdentifier 3741
StatisticModelResult 3742

DataQualityGlueTable 3742
— operations — 3743
StartDataQualityRulesetEvaluationRun (start_data_quality_ruleset_evaluation_run) 3745
CancelDataQualityRulesetEvaluationRun (cancel_data_quality_ruleset_evaluation_run) 3746
GetDataQualityRulesetEvaluationRun (get_data_quality_ruleset_evaluation_run) 3747
ListDataQualityRulesetEvaluationRuns (list_data_quality_ruleset_evaluation_runs) 3749
StartDataQualityRuleRecommendationRun (start_data_quality_rule_recommendation_run) 3750
CancelDataQualityRuleRecommendationRun
(cancel_data_quality_rule_recommendation_run) 3752
GetDataQualityRuleRecommendationRun (get_data_quality_rule_recommendation_run) .. 3752
ListDataQualityRuleRecommendationRuns (list_data_quality_rule_recommendation_runs) 3754
GetDataQualityResult (get_data_quality_result) 3755
BatchGetDataQualityResult (batch_get_data_quality_result) 3757
ListDataQualityResults (liste_données_qualité_résultats) 3758
CreateDataQualityRuleset (create_data_quality_ruleset) 3759
DeleteDataQualityRuleset (delete_data_quality_ruleset) 3760
GetDataQualityRuleset (get_data_quality_ruleset) 3761
ListDataQualityRulesets (liste_data_quality_rulesets) 3762
UpdateDataQualityRuleset (ensemble de règles de qualité des données de mise à jour) ... 3763
ListDataQualityStatistics (liste_données_qualité_statistiques) 3765
TimestampFilter 3766
CreateDataQualityRulesetRequest 3766
GetDataQualityRulesetResponse 3767
GetDataQualityResultResponse 3768
StartDataQualityRuleRecommendationRunRequest 3770
GetDataQualityRuleRecommendationRunResponse 3771
BatchPutDataQualityStatisticAnnotation (batch_put_data_quality_statistic_annotation) 3772
GetDataQualityModel (get_data_quality_model) 3773
GetDataQualityModelResult (get_data_quality_model_result) 3774
ListDataQualityStatisticAnnotations (list_data_quality_statistic_annotations) 3775
PutDataQualityProfileAnnotation (put_data_quality_profile_annotation) 3776
Données sensibles 3777
— les types de données — 3777
CustomEntityType 3777
— operations — 3778
CreateCustomEntityType (créer_custom_entity_type) 3778

DeleteCustomEntityType (supprimer_type_entité_personnalisé) 3779
GetCustomEntityType (get_custom_entity_type) 3780
BatchGetCustomEntityTypes (batch_get_custom_entity_types) 3781
ListCustomEntityTypes (list_custom_entity_types) 3782
Balisage APIs 3783
– types de données – 3783
Balise 3783
— operations — 3783
TagResource (tag_ressource) 3783
UntagResource (untag_resource) 3784
GetTags (get_tags) 3785
Types de données courants 3786
Balise 3786
DecimalNumber 3786
ErrorDetail 3786
PropertyPredicate 3787
ResourceUri 3787
ColumnStatistics 3788
ColumnStatisticsError 3788
ColumnError 3789
ColumnStatisticsData 3789
BooleanColumnStatisticsData 3790
DateColumnStatisticsData 3790
DecimalColumnStatisticsData 3791
DoubleColumnStatisticsData 3791
LongColumnStatisticsData 3792
StringColumnStatisticsData 3792
BinaryColumnStatisticsData 3793
Modèles de chaîne 3793
Exceptions 3796
AccessDeniedException 3796
AlreadyExistsException 3796
ConcurrentModificationException 3796
ConcurrentRunsExceededException 3797
CrawlerNotRunningException 3797
CrawlerRunningException 3797

CrawlerStoppingException 3797
EntityNotFoundException 3798
FederationSourceException 3798
FederationSourceRetryableException 3798
GlueEncryptionException 3799
IdempotentParameterMismatchException 3799
IllegalWorkflowStateException 3799
InternalServiceException 3799
InvalidExecutionEngineException 3800
InvalidInputException 3800
InvalidStateException 3800
InvalidTaskStatusTransitionException 3801
JobDefinitionErrorException 3801
JobRunInTerminalStateException 3801
JobRunInvalidStateTransitionException 3801
JobRunNotInTerminalStateException 3802
LateRunnerException 3802
NoScheduleException 3802
OperationTimeoutException 3803
ResourceNotReadyException 3803
ResourceNumberLimitExceededException 3803
SchedulerNotRunningException 3803
SchedulerRunningException 3804
SchedulerTransitioningException 3804
UnrecognizedRunnerException 3804
ValidationException 3804
VersionMismatchException 3805
AWS Glue Exemples de code API 3806
Principes de base 3807
Bonjour AWS Glue 3808
Principes de base 3817
Actions 3956
Sécurité 4116
Protection des données 4117
Chiffrement de données au repos 4117
chiffrement des données en transit 4136

Conformité FIPS 4136
Gestion des clés 4136
AWS Gluedépendance à l'égard d'autres AWS services 4137
Points de terminaison de développement 4138
Gestion des identités et des accès 4138
Public ciblé 4139
Authentification par des identités 4140
Gestion de l'accès à l'aide de politiques 4141
Comment AWS Glue fonctionne avec IAM 4143
Configuration des autorisations IAM pour AWS Glue 4150
AWS Exemples de politiques de contrôle d'accès Glue 4186
Octroi de politiques AWS gérées 4213
Octroi de politiques à portée dynamique 4223
Spécification de AWS Glue la ressource ARNs 4224
Octroi d'un accès intercompte 4234
Résolution des problèmes 4242
AWS Lake Formation modèles de contrôle d'accès 4244
Utilisation de AWS Glue with AWS Lake Formation pour un accès complet à la table 4244
Lake Formation pour FGAC 4257
Utilisation d'Amazon S3 Access Grants avec AWS Glue 4270
Comment AWS Glue fonctionne le S3 Access Grants 4270
Considérations relatives aux subventions d'accès S3 avec AWS Glue 4271
Configurez les autorisations d'accès S3 avec AWS Glue 4272
Propagation d'identité fiable 4274
Présentation de 4275
Fonctionnalités et avantages 4275
Cas d'utilisation 4275
Comment ça marche 4276
Intégrations 4277
Commencer à utiliser la propagation fiable des identités dans l' AWS Glue ETL 4278
Considérations et limites relatives à l'intégration d' AWS Glue ETL Trusted Identity Propagation 4284
Sessions d'arrière-plan pour les utilisateurs 4286
Journalisation et surveillance 4290
Validation de conformité 4291
Résilience 4291

Sécurité de l'infrastructure 4291
Configuration de points de terminaison d'un VPC d'interface (AWS PrivateLink) pour AWS Glue 4292
Configuration d'Amazon partagé VPCs 4295
Résolution des problèmes de AWS Glue 4296
Collecte d'informations AWS Glue de dépannage 4296
Résolution des erreurs de configuration courantes de Glue 4297
Erreur : Ressource non disponible 4299
Erreur : Impossible de trouver le point de terminaison S3 ou la passerelle NAT du subnetId dans VPC 4299
Erreur : Règle de trafic entrant obligatoire dans le groupe de sécurité 4299
Erreur : Règle de trafic sortant obligatoire dans le groupe de sécurité 4299
Erreur : L'exécution du Job a échoué car le rôle transmis doit être autorisé à assumer le rôle pour le AWS Glue service 4300
Erreur : l' DescribeVpcEndpoints action n'est pas autorisée. impossible de valider l'ID VPC vpc-id 4300
Erreur : l' DescribeRouteTables action n'est pas autorisée. impossible de valider l'identifiant du sous-réseau : ID du sous-réseau dans l'identifiant du VPC : vpc-id 4300
Erreur : Impossible d'appeler ec2 : DescribeSubnets 4300
Erreur : Impossible d'appeler ec2 : DescribeSecurityGroups 4301
Erreur : Impossible de trouver le sous-réseau pour la zone de disponibilité 4301
Erreur : Exception d'exécution de tâche lors d'écriture sur une cible JDBC 4301
Erreur : Amazon S3 : l'opération n'est pas valide pour la classe de stockage de l'objet 4302
Erreur : Délai Amazon S3 4302
Erreur : Accès à Amazon S3 refusé 4302
Erreur : L'identifiant de la clé d'accès Amazon S3 n'existe pas 4303
Erreur : l'exécution d'une tâche échoue lors de l'accès à Amazon S3 avec un URI `s3a://` 4303
Erreur : Le jeton de service Amazon S3 a expiré 4305
Erreur : Aucun DNS privé trouvé pour l'interface réseau 4305
Erreur : Échec de l'allocation du point de terminaison de développement 4305
Erreur : Serveur de bloc-notes CREATE_FAILED 4306
Erreur : Échec du démarrage du bloc-notes local 4306
Erreur : Échec de l'exécution du crawler 4306
Erreur : les partitions n'ont pas été mises à jour 4307
Erreur : la mise à jour du signet de tâche a échoué en raison d'une incompatibilité de version 4307

Erreur : Une tâche retraite des données lorsque les signets de tâche sont activés 4308
Erreur : comportement de basculement entre VPCs AWS Glue 4309
Erreurs du robot lorsqu'il utilise les autorisations de Lake Formation 4310
Erreur : l'emplacement S3 : s3://examplepath n'est pas enregistré 4310
Erreur : n' User/Role est pas autorisé à exécuter : lakeformation : GetDataAccess on resource 4310
Erreur : autorisation(s) insuffisante(s) sur Lake Formation (nom de la base de données : ExampleDatabase, nom de la table : ExampleTable) 4311
Erreur : autorisation(s) insuffisante(s) pour Lake Formation sur s3://examplepath 4311
Questions fréquemment posées sur la configuration du crawler à l'aide des informations d'identification Lake Formation 4312
Dépannage des erreurs Ray 4313
Inspection des journaux de tâches Ray 4313
Résolution des erreurs liées aux tâches Ray 4314
Exceptions AWS Glue liées au machine learning 4316
Annuler MLTask RunActivity 4316
Créez MLTask RunActivity 4316
Supprimer MLTransform l'activité 4318
Obtenez MLTask RunActivity 4318
Obtenez MLTask RunsActivity 4318
Obtenir de MLTransform l'activité 4319
Obtenir de MLTransforms l'activité 4319
GetSaveLocationForTransformArtifactActivity 4320
GetTaskRunArtifactActivity 4320
Publier MLTransform ModelActivity 4321
PullLatestMLTransformModelActivity 4321
PutJobMetadataForMLTransformActivité 4322
StartExportLabelsTaskRunActivity 4323
StartImportLabelsTaskRunActivity 4323
Démarrer MLEvaluation TaskRunActivity 4324
Démarrer MLLabeling SetGenerationTaskRunActivity 4325
MLTransformActivité de mise à jour 4326
AWS Glue quotas 4327
Améliorer les AWS Glue performances 4328
Réglage de stratégies pour votre type de tâche 4328
Amélioration de la performance Spark 4328

Optimiser les lectures avec pushdown 4329
Pushdown de prédicat sur les fichiers stockés dans Amazon S3 4329
Pushdown lors de l'utilisation de sources JDBC 4331
Remarques et limites relatives au pushdown dans Glue AWS 4333
Utilisation d'Auto Scaling pour AWS Glue 4334
Exigences 4335
Activation d'Auto Scaling dans AWS Glue Studio 1977
Activation de l'Auto Scaling avec la AWS CLI ou le SDK 1978
Activation d'Autoscaling avec des sessions interactives 4337
Conseils et considérations 4337
Surveillance d'Auto Scaling à l'aide CloudWatch des métriques Amazon 4338
Surveillance de l'Auto Scaling avec Amazon CloudWatch Logs 4339
Surveillance de Auto Scaling avec Spark UI 4340
Surveillance de l'utilisation du DPU d'exécution de la tâche Auto Scaling 4341
Limitations 4341
Partitionnement de la charge de travail avec exécution limitée 4341
Activation du partitionnement de la charge de travail 4342
Configuration d'un déclencheur AWS Glue pour exécuter automatiquement la tâche 4343
Problèmes connus 4344
Interdiction d'accès aux données inter-tâches 4344
Historique de la documentation 4347
Mises à jour antérieures 4422
AWS Glossaire 4424
..... mmmmcdxxv

Qu'est-ce que c'est AWS Glue ?

AWS Glueest un service d'intégration de données sans serveur qui facilite la découverte, la préparation, le déplacement et l'intégration de données provenant de plusieurs sources pour les utilisateurs d'analytique. Vous pouvez l'utiliser pour l'analyse, le machine learning et le développement d'applications. Il inclut également des outils de productivité et d'exploitation des données supplémentaires pour la création, l'exécution de tâches et la mise en œuvre de flux de travail.

avecAWS Glue, vous pouvez découvrir et vous connecter à plus de 70 sources de données diverses et gérer vos données dans un catalogue de données centralisé. Vous pouvez créer, exécuter et surveiller visuellement des pipelines d'extraction, de transformation et de chargement (ETL) pour charger les données dans vos lacs de données. Vous pouvez également rechercher et interroger immédiatement les données cataloguées à l'aide d'Amazon Athena, Amazon EMR et Amazon Redshift Spectrum.

AWS Gluerenforce les fonctionnalités principales d'intégration de données en un seul service. Il s'agit notamment de la découverte de données, de l'ETL moderne, du nettoyage, de la transformation et du catalogage centralisé. Il est également sans serveur, ce qui signifie qu'il n'y a aucune infrastructure à gérer. Avec une prise en charge flexible pour toutes les charges de travail telles que l'ETL, l'ELT et le streaming dans un seul service,AWS Glue prend en charge les utilisateurs pour différentes charges de travail et différents types d'utilisateurs.

Également,AWS Gluefacilite l'intégration des données au sein de votre architecture. Il s'intègre aux services AWS d'analyse et aux lacs de données Amazon S3. AWS Gluepropose des interfaces d'intégration et des outils de création de tâches faciles à utiliser pour tous les utilisateurs, des développeurs aux utilisateurs professionnels, avec des solutions adaptées à des compétences techniques variées.

Avec la possibilité d'évoluer à la demande,AWS Gluevous permet de vous concentrer sur des activités à forte valeur ajoutée qui optimisent la valeur de vos données. Il s'adapte à toutes les tailles de données et prend en charge tous les types de données et toutes les variances de schéma. Pour accroître l'agilité et optimiser les coûts, AWS Glue fournit une haute disponibilité et une pay-as-you-go facturation intégrées.

Pour en savoir plus sur la tarification, consultez Tarification AWS Glue.

AWS Glue Studio

AWS Glue Studioest une interface graphique qui facilite la création, l'exécution et le contrôle des tâches d'intégration de données dansAWS Glue. Vous pouvez composer visuellement des flux de transformation de données et les exécuter de manière transparente sur le moteur ETL sans serveur basé sur Apache Spark de AWS Glue.

Avec AWS Glue Studio, vous aide à créer ainsi qu'à gérer des tâches rassemblant, améliorant et nettoyant les données. Vous pouvez également utiliser AWS Glue Studio pour résoudre les problèmes et modifier les scripts de tâche.

Rubriques

- Fonctionnalités d'AWS Glue
- En savoir plus sur les innovations dans AWS Glue
- Démarrer avec AWS Glue
- Accès à AWS Glue
- Services connexes
- AWS Glue fin du support pour Ray

Fonctionnalités d'AWS Glue

AWS Glueles fonctionnalités se répartissent en trois catégories principales :

- Découvrez et organisez les données
- Transformez, préparez et nettoyez les données pour les analyser
- Créez et surveillez des pipelines de données

Découverte et organisation des données

- Unifiez et recherchez dans plusieurs banques de données : stockez, indexez et recherchez dans plusieurs sources et récepteurs de données en cataloguant toutes vos données. AWS
- Découverte automatique de données — Utilisation deAWS Gluerobots d'exploration afin de déduire automatiquement les informations du schéma et de les intégrer dans votre AWS Glue Data Catalog.
- Gestion des schémas et des autorisations— Validez et contrôlez l'accès à vos bases de données et tables.

- Connectez-vous à un large éventail de sources de données : exploitez plusieurs sources de données, sur site ou sur site AWS, en utilisant des AWS Glue connexions pour créer votre lac de données.

Transformez, préparez, et nettoyez les données afin de les analyser

- Transformez visuellement les données avec une interface de canevas de la tâche : définissez votre processus ETL dans l'éditeur de tâches visuel. Ensuite, générez automatiquement le code pour l'extraction, la transformation ainsi que le chargement de vos données.
- Grâce à une planification simple des tâches, créez des pipelines ETL complexes — InvoquezAWS Glueemplois en fonction d'un calendrier, d'une demande ou d'un événement.
- Nettoyez et transformez les données de streaming en transit— Permettez une consommation continue des données, nettoyez-les et transformez-les en transit. Cette démarche le rend disponible afin de l'analyser en quelques secondes dans votre magasin de données cible.
- Dédupliquez et nettoyez les données grâce au machine learning intégré— Nettoyez et préparez vos données pour analyse sans devenir un expert en machine learning en utilisant le`FindMatches`fonction. Cette fonctionnalité permet la déduplication ainsi que la recherche des enregistrements ne correspondant pas parfaitement les uns aux autres.
- Bloc-notes de travail intégrés–AWS Glueles carnets de tâches fournissent des ordinateurs portables sans serveur avec une configuration minimale dansAWS Glueafin de rendre le démarrage rapide.
- Modifier, déboguer et tester le code ETL— AvecAWS Gluesessions interactives, il est possible d'explorer et de préparer des données de manière interactive. Grâce à l'IDE ou à un bloc-notes de votre choix, vous pouvez explorer, expérimenter ainsi que traiter des données de manière interactive.
- Définissez, détectez et corrigez les données sensibles–AWS Gluela détection des données sensibles vous permet de définir, d'identifier et de traiter les données sensibles dans votre pipeline de données ainsi que dans votre lac de données.

Création et surveillance des pipelines de données

- Adaptation automatique en fonction de la charge— Augmentez et diminuez les ressources de manière dynamique en fonction de la charge Cette adaptation affecte les travailleurs à des emplois uniquement en cas de nécessité.

- Automatisez les tâches à l'aide de déclencheurs— démarrer les robots d'exploration ouAWS Gluetâches avec déclencheurs basés sur les événements, ensuite, concevoir une chaîne de tâches ainsi que de robots d'exploration dépendants.
- Exécuter et surveiller les tâches : exécutez les tâches AWS Glue avec le moteur de votre choix, Spark ou Ray. Surveillez-les à l'aide d'outils de surveillance automatisés, d'informations sur l'exécution des tâches AWS Glue et AWS CloudTrail. Améliorez votre surveillance des tâches soutenues par Spark avec l'interface utilisateur Apache Spark.
- Définition des flux de travail pour les activités ETL et d'intégration— Définissez les flux de travail pour l'ETL et les activités d'intégration pour plusieurs robots d'exploration, tâches et déclencheurs.

En savoir plus sur les innovations dans AWS Glue

Découvrez les dernières innovations AWS Glue et découvrez comment les clients les utilisent AWS Glue pour permettre la préparation des données en libre-service au sein de leur organisation.

Découvrez comment les clients évoluent AWS Glue au-delà de la configuration traditionnelle et comment ils se configurent AWS Glue pour le suivi des tâches et les performances.

Démarrer avec AWS Glue

La lecture de ces sections est indispensable:

- Présentation de l'utilisation AWS Glue
- AWS GlueConcepts
- Configuration des autorisations IAM pour AWS Glue
- Mise en route avec le kit AWS Glue Data Catalog
- Création des tâches AWS Glue
- Démarrage avec AWS Glue sessions interactives
- Orchestration dans AWS Glue

Accès à AWS Glue

Vous pouvez créer vos AWS Glue, y accéder et les gérer à l'aide des interfaces suivantes :

- AWS Glueconsole— Fournit une interface Web pour créer, afficher et gérer vosAWS Gluetâches. Pour accéder à la console, consultez AWS Glue.
- AWS Glue Studio— Fournit une interface graphique vous permettant de créer et de modifier visuellement vosAWS Glue tâches. Pour de plus amples informations, veuillez consulter Création de tâches ETL visuelles.
- AWS Gluesection de la AWS CLI référence — Fournit des AWS CLI commandes que vous pouvez utiliser avecAWS Glue. Pour plus d'informations, consultez la référence AWS CLI pour AWS Glue.
- AWS GlueAPI— Fournit une référence d'API complète pour les développeurs. Pour plus d'informations, consultez AWS Glue API.

Services connexes

Utilisateurs deAWS Glueutilisez également :

- AWS Lake Formation — Un service qui est une couche d'autorisation fournissant un contrôle précis des accès aux ressources duAWS Glue Data Catalog.
- AWS Glue DataBrew : un outil de préparation visuelle de données que vous pouvez utiliser afin de nettoyer et de normaliser les données sans écrire aucun code.

AWS Glue fin du support pour Ray

> **Important**
>
> AWS Glue for Ray ne sera plus ouvert aux nouveaux clients à compter du 30 avril 2026. Si vous souhaitez utiliser AWS Glue for Ray, inscrivez-vous avant cette date. Les clients existants peuvent continuer à utiliser le service normalement. Pour des fonctionnalités similaires à celles de AWS Glue for Ray, explorez Amazon EKS. Pour plus d'informations, consultez la section AWS Glue relative à la fin du support de Ray.

Après mûre réflexion, nous avons décidé de fermer AWS Glue les portes de Ray à de nouveaux clients à compter du 30 avril 2026. Si vous souhaitez utiliser AWS Glue for Ray, inscrivez-vous avant cette date. Les clients existants peuvent continuer à utiliser le service normalement.

AWS continue d'investir dans l'amélioration de la sécurité et de la disponibilité AWS Glue pour for Ray. Notez que nous ne prévoyons pas d'introduire de nouvelles fonctionnalités dans AWS Glue for Ray, à l'exception des améliorations de sécurité et de disponibilité.

Comme alternative à AWS Glue for Ray, nous vous recommandons d'utiliser Amazon Elastic Kubernetes Service. Amazon Elastic Kubernetes Service est un service entièrement géré et certifié conforme à Kubernetes qui simplifie le processus de création, de sécurisation, d'exploitation et de maintenance de clusters Kubernetes sur. AWS Il s'agit d'une option hautement personnalisable qui repose sur l'open source KubeRay Operator pour déployer et gérer les clusters Ray sur Kubernetes, offrant une meilleure utilisation des ressources, une gestion simplifiée de l'infrastructure et un support complet des fonctionnalités Ray.

Migration d'une tâche Ray vers Amazon Elastic Kubernetes Service

Cette section décrit les étapes de migration de AWS Glue for Ray vers Ray sur Amazon Elastic Kubernetes Service. Ces étapes sont utiles pour deux scénarios de migration :

- Migration standard (x86/amd64) : pour ces cas d'utilisation, la stratégie de migration utilise le conteneur OpenSource Ray pour les implémentations de base et exécute les scripts directement sur le conteneur de base.
- ARM64 Migration : pour ces cas d'utilisation, la stratégie de migration prend en charge les versions de conteneurs personnalisées pour ARM64 des dépendances et des exigences d'architecture spécifiques.

Conditions préalables à la migration

Installez les outils CLI suivants :aws,,kubectl, eksctlhelm, Python 3.9+. Ces outils CLI sont nécessaires pour provisionner et gérer votre environnement Ray on EKS. eksctlsimplifie la création et la gestion de clusters EKS. kubectlest la CLI Kubernetes standard pour le déploiement et le dépannage des charges de travail sur votre cluster. helmest utilisé pour installer et gérer KubeRay (l'opérateur qui exécute Ray sur Kubernetes). Python 3.9+ est requis pour Ray lui-même et pour exécuter des scripts de soumission de tâches localement.

Installer eksctl

Suivez les instructions des options d'installation pour Eksctl ou utilisez les instructions ci-dessous pour l'installation.

Pour Mac OS :

```
brew tap weaveworks/tap
brew install weaveworks/tap/eksctl
```

Pour Linux :

```
curl --silent --location "https://github.com/weaveworks/eksctl/releases/latest/
download/eksctl_$(uname -s)_amd64.tar.gz" | tar xz -C /tmp

# Move the extracted binary to /usr/local/bin
sudo mv /tmp/eksctl /usr/local/bin

# Test the installation
eksctl version
```

Installer kubectl

Suivez les instructions de la section Configurer kubectl et eksctl ou utilisez les instructions ci-dessous pour l'installation.

Pour Mac OS :

```
brew install kubectl
```

Pour Linux :

```
curl -LO "https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)/
bin/linux/amd64/kubectl"
chmod +x kubectl
sudo mv kubectl /usr/local/bin/
```

Installer le casque

Suivez les instructions d'installation de Helm ou utilisez les instructions ci-dessous pour l'installation.

Pour Mac OS :

```
brew install helm
```

Pour Linux :

```
curl https://raw.githubusercontent.com/helm/helm/main/scripts/get-helm-3 | bash
```

Étape 1. Créez ou choisissez une image Docker pour Ray

Option 1 : utiliser l'image officielle de Ray (aucune compilation requise)

Cette option utilise l'image officielle de Ray Docker sur Docker Hub, par exemple`rayproject/ray:2.4.0-py39`, qui est maintenue par le projet Ray.

> **Note**
>
> Cette image est uniquement destinée à AMD64. Utilisez-le si vos dépendances sont compatibles avec amd64 et que vous n'avez pas besoin de versions spécifiques à ARM.

Option 2 : créez et publiez votre propre image arm64 Ray 2.4.0

Cette option est utile lorsque vous utilisez des nœuds Graviton (ARM), conformément à ce que AWS Glue Ray utilise en interne. Vous pouvez créer une image personnalisée épinglée aux mêmes versions de dépendance que AWS Glue pour Ray, afin de réduire les incompatibilités.

Créez un Dockerfile localement :

```
# Build an ARM64 image
FROM --platform=linux/arm64 python:3.9-slim-bullseye
# Handy tools: wget for KubeRay probes; CA certs; keep image small
RUN apt-get update && apt-get install -y --no-install-recommends \
    wget ca-certificates \
    && rm -rf /var/lib/apt/lists/*

# Keep pip/setuptools modern enough for wheels resolution
RUN python -m pip install -U "pip<24" "setuptools<70" wheel

# ---- Install Ray 2.4.0 (ARM64 / Py3.9) and Glue-like dependencies ----
# 1) Download the exact Ray 2.4.0 wheel for aarch64 (no network at runtime)
RUN python -m pip download --only-binary=:all: --no-deps --dest /tmp/wheels ray==2.4.0

# 2) Core libs used in Glue (pin to Glue-era versions)
#    + the dashboard & jobs API dependencies compatible with Ray 2.4.0.
#    (Pins matter: newer major versions break 2.4.0's dashboard.)
RUN python -m pip install --no-cache-dir \
```

```
    /tmp/wheels/ray-2.4.0-*.whl \
    "pyarrow==11.0.0" \
    "pandas==1.5.3" \
    "boto3==1.26.133" \
    "botocore==1.29.133" \
    "numpy==1.24.3" \
    "fsspec==2023.4.0" \
    "protobuf<4" \
    # --- dashboard / jobs server deps ---
    "aiohttp==3.8.5" \
    "aiohttp-cors==0.7.0" \
    "yarl<1.10" "multidict<7.0" "frozenlist<1.4" "aiosignal<1.4" "async_timeout<5" \
    "pydantic<2" \
    "opencensus<0.12" \
    "prometheus_client<0.17" \
    # --- needed if using py_modules ---
    "smart_open[s3]==6.4.0"

# Optional: prove Ray & arch at container start
ENV PYTHONUNBUFFERED=1
WORKDIR /app

# KubeRay overrides the start command; this is just a harmless default
CMD ["python","-c","import ray,platform; print('Ray', ray.__version__, 'on',
 platform.machine())"]
```

```
# Set environment variables
export AWS_REGION=us-east-1
export AWS_ACCOUNT=$(aws sts get-caller-identity --query Account --output text)
export REPO=ray-2-4-arm64
export IMAGE=${AWS_ACCOUNT}.dkr.ecr.${AWS_REGION}.amazonaws.com/${REPO}:v1

# Create repository and login
aws ecr create-repository --repository-name $REPO >/dev/null 2>&1 || true
aws ecr get-login-password --region $AWS_REGION \
  | docker login --username AWS --password-stdin ${AWS_ACCOUNT}.dkr.ecr.
${AWS_REGION}.amazonaws.com

# Enable Buildx (for cross-builds on non-ARM hosts)
docker buildx create --name multi --driver docker-container --use 2>/dev/null || true

# Build & push ARM64 image
docker buildx build \
```

```
  --platform linux/arm64 \
  -t "$IMAGE" \
  . --push

# Verify the image architecture remotely
aws ecr batch-get-image \
  --repository-name $REPO \
  --image-ids imageTag=v1 \
  --accepted-media-types application/vnd.docker.distribution.manifest.v2+json \
  | jq -r '.images[0].imageManifest' \
  | jq -r 'fromjson.config.digest'
```

Une fois terminé, référencez cette ARM64 image dans les RayCluster spécifications avec`nodeSelector: { kubernetes.io/arch: arm64 }`.

```
spec:
  rayVersion: "2.4.0"
  headGroupSpec:
    template:
      spec:
        containers:
        - name: ray-head
          image: <your ECR image>
```

Étape 2. Convertir AWS Glue la configuration de for Ray Job en Ray sur Amazon Elastic Kubernetes Service

AWS Glue les tâches for Ray prennent en charge un ensemble d'arguments de tâche qui configurent les travailleurs, les dépendances, la mémoire et la journalisation. Lors de la migration vers Amazon Elastic Kubernetes KubeRay Service with, ces arguments doivent être RayCluster traduits en champs de spécification ou en paramètres d'environnement d'exécution Ray Job.

Cartographie des arguments du poste

Cartographie AWS Glue des arguments de Ray avec les équivalents de Ray sur EKS

AWS Glue pour Ray argument	Ce que cela apporte AWS Glue à Ray	Ray sur l'équivalent d'Amazon Elastic Kubernetes Service
`--min-workers`	Nombre minimum de travailleurs que le poste doit affecter.	`workerGroupSpecs[].minReplicas` dans votre RayCluster

AWS Glue pour Ray argument	Ce que cela apporte AWS Glue à Ray	Ray sur l'équivalent d'Amazon Elastic Kubernetes Service
`--working-dir`	Distribue un zip (URI S3) à tous les nœuds.	Utilisez Ray Runtime Env : `working_dir` si vous soumettez à partir de fichiers locaux ; `py_modules` pour que les zips S3 pointent vers l'artefact S3
`--s3-py-modules`	Ajoute Python wheels/dists à partir de S3.	Utilisez Ray Runtime Env : `py_modules: ["s3://.../xxx.whl", ...]`
`--pip-install`	Installe des packages PyPI supplémentaires pour le travail.	Environnement d'exécution Ray : `pip: ["pkg==ver", ...]` (Ray Job CLI `--runtime-env-json` ou RayJob`runtimeEnvYAML`).
`--object_store_memory_head`	% de mémoire pour le magasin de plasma du nœud principal.	`headGroupSpec[].rayStartParams.object-store-memory` dans votre RayCluster. Notez que cela doit être exprimé en octets. AWS Glue utilise un pourcentage, tandis que Ray utilise des octets.
`--object_store_memory_worker`	% de mémoire pour le stockage de plasma des nœuds de travail.	Identique à celui ci-dessus mais défini dans celui de chaque groupe de travail `rayStartParams.object-store-memory` (octets).
`--object_spilling_config`	Configurez le déversement d'objets Ray.	`headGroupSpec[].rayStartParams.object-spilling-config`

AWS Glue pour Ray argument	Ce que cela apporte AWS Glue à Ray	Ray sur l'équivalent d'Amazon Elastic Kubernetes Service
`--logging_configuration`	AWS Glue-journaux gérés (CloudWatch, S3).	Vérifiez pod stdout/stderr : utiliser. `kubectl -n ray logs <pod-name> --follow` Consultez les journaux depuis Ray Dashboard (transfert de port vers : 8265), vous pouvez également y consulter les journaux des tâches et des tâches.

Cartographie de la configuration des tâches

Cartographie AWS Glue des configurations Ray Job avec les équivalents Ray on EKS

Configuration	Ce que cela apporte AWS Glue à Ray	Ray sur l'équivalent EKS
Type d'employé	Définissez le type de travailleur prédéfini autorisé lors de l'exécution d'une tâche. La valeur par défaut est Z 2X (8vCPU, 64 Go de RAM).	Type d'instance de groupe de nœuds dans EKS (par exemple, r7g.2xlarge ≈ 8 vCPU /64 Go pour ARM, r7a.2xlarge pour x86).
Nombre maximum de travailleurs	Le nombre de travailleurs que vous AWS Glue souhaitez affecter à cette tâche.	Réglez `workerGroupSpecs[].maxReplicas` le même numéro que celui dans lequel vous l'avez utilisé AWS Glue. Il s'agit de la limite supérieure de la mise à l'échelle automatique. Défini de la même manière `minReplicas` comme limite inférieure. Vous pouvez commencer par`replicas: 0`,`minReplicas: 0` .

Étape 3. Configuration d'Amazon Elastic Kubernetes Service

Vous pouvez soit créer un nouveau cluster Amazon Elastic Kubernetes Service, soit réutiliser un cluster Amazon Elastic Kubernetes Service existant. Si vous utilisez un cluster existant, ignorez

les commandes de création de cluster et passez à Ajouter un groupe de nœuds, IRSA, et installez KubeRay.

Création d'un cluster Amazon Elastic Kubernetes Service

> **Note**
>
> Si vous possédez déjà un cluster Amazon Elastic Kubernetes Service, ignorez les commandes pour créer un nouveau cluster et ajoutez simplement un groupe de nœuds.

```
# Environment Variables
export AWS_REGION=us-east-1
export CLUSTER=ray-eks
export NS=ray # namespace for your Ray jobs (you can reuse another if you like)

# Create a cluster (OIDC is required for IRSA)
eksctl create cluster \
  --name $CLUSTER \
  --region $AWS_REGION \
  --with-oidc \
  --managed
```

Ajouter un groupe de nœuds

```
# ARM/Graviton (matches Glue's typical runtime):
eksctl create nodegroup \
  --cluster $CLUSTER \
  --region $AWS_REGION \
  --name arm64-ng \
  --node-type m7g.large \
  --nodes 2 --nodes-min 1 --nodes-max 5 \
  --managed \
  --node-labels "workload=ray"

# x86/amd64 (use if your image is amd64-only):
eksctl create nodegroup \
  --cluster $CLUSTER \
  --region $AWS_REGION \
  --name amd64-ng \
  --node-type m5.large \
  --nodes 2 --nodes-min 1 --nodes-max 5 \
```

```
  --managed \
  --node-labels "workload=ray"
```

> **Note**
>
> Si vous utilisez un cluster Amazon Elastic Kubernetes Service existant, `--with-oidc` utilisez-le pour activer OIDC lors de l'ajout d'un groupe de nœuds.

Créer un espace de noms et un rôle IAM pour les comptes de service (IRSA) pour S3

Un espace de noms Kubernetes est un regroupement logique de ressources (pods, services, rôles, etc.). Vous pouvez créer ou réutiliser un espace de noms existant. Vous devrez également créer une politique IAM pour S3 qui reflète l'accès à votre AWS Glue tâche. Utilisez les mêmes autorisations personnalisées que celles dont bénéficiait votre AWS Glue rôle professionnel (généralement S3 read/write pour des compartiments spécifiques). Pour accorder des autorisations similaires à Amazon Elastic Kubernetes Service, créez un compte de service (IRSA) lié à AWSGlue ServiceRole cette politique IAM. Reportez-vous à la section Rôles IAM pour les comptes de service pour obtenir des instructions sur la configuration de ce compte de service.

```
# Create (or reuse) namespace
kubectl create namespace $NS || true
```

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": ["s3:PutObject", "s3:GetObject", "s3:ListBucket"],
    "Resource": [
      "arn:aws:s3:::YOUR-BUCKET",
      "arn:aws:s3:::YOUR-BUCKET/*"
    ]
  }]
}
```

```
# Create the IAM policy and wire IRSA:
aws iam create-policy \
  --policy-name RayS3Policy \
  --policy-document file://example.json || true
```

```
# Create a service account (IRSA) bound to that policy.
eksctl create iamserviceaccount \
  --cluster $CLUSTER \
  --region $AWS_REGION \
  --namespace $NS \
  --name ray-s3-access \
  --attach-policy-arn arn:aws:iam::${AWS_ACCOUNT}:policy/RayS3Policy \
  --approve \
  --override-existing-serviceaccounts
```

KubeRay Opérateur d'installation (contrôleur qui exécute Ray sur K8s)

```
helm repo add kuberay https://ray-project.github.io/kuberay-helm/
helm repo update
helm upgrade --install kuberay-operator kuberay/kuberay-operator \
  --namespace kuberay-system \
  --create-namespace

# Validate the operator pod Running
kubectl -n kuberay-system get pods
```

Étape 4 : Créez un cluster de rayons

Créez un fichier YAML pour définir le cluster de rayons. Voici un exemple de configuration (raycluster.yaml) :

```
apiVersion: ray.io/v1
kind: RayCluster
metadata:
  name: glue-like
  namespace: ray
spec:
  rayVersion: "2.4.0"
  headGroupSpec:
    template:
      spec:
        nodeSelector:
          kubernetes.io/arch: amd64
        serviceAccountName: ray-s3-access
        containers:
        - name: ray-head
```

```
          image: rayproject/ray:2.4.0-py39
          imagePullPolicy: Always
          resources:
            requests: { cpu: "1", memory: "2Gi" }
            limits:   { cpu: "1", memory: "2Gi" }
  workerGroupSpecs:
  - groupName: workers
    replicas: 0 # start with just a head (like small Glue dev job) and turn number of
 replicas later
    minReplicas: 0
    maxReplicas: 5
    template:
      spec:
        nodeSelector:
          kubernetes.io/arch: amd64
        serviceAccountName: ray-s3-access
        containers:
        - name: ray-worker
          image: rayproject/ray:2.4.0-py39
          imagePullPolicy: Always
          resources:
            requests: { cpu: "1", memory: "2Gi" }
            limits:   { cpu: "1", memory: "2Gi" }
```

Déployez le cluster Ray sur le cluster Amazon Elastic Kubernetes Service

```
kubectl apply -n $NS -f raycluster.yaml

# Validate that the head pod turns to READY/ RUNNING state
kubectl -n $NS get pods -l ray.io/cluster=glue-like -w
```

S'il est nécessaire de modifier le yaml déployé, supprimez d'abord le cluster, puis réappliquez le yaml mis à jour :

```
kubectl -n $NS delete raycluster glue-like
kubectl -n $NS apply -f raycluster.yaml
```

Accès au tableau de bord Ray

Vous pouvez accéder au tableau de bord Ray en activant le transfert de port à l'aide de kubectl :

```
# Get service
```

```
SVC=$(kubectl -n $NS get svc -l ray.io/cluster=glue-like,ray.io/node-type=head -o
 jsonpath='{.items[0].metadata.name}')

# Make the Ray dashboard accessible at http://localhost:8265 on your local machine.
kubectl -n $NS port-forward svc/$SVC 8265:8265
```

Étape 5. Soumettre Ray Job

Pour soumettre une tâche Ray, utilisez la CLI Ray jobs. La version de la CLI peut être plus récente que le cluster, elle est rétrocompatible. Comme condition préalable, stockez votre script de tâche localement dans un fichier, par exemple `job.py`

```
python3 -m venv ~/raycli && source ~/raycli/bin/activate
pip install "ray[default]==2.49.2"

# Submit your ray job by supplying all python dependencies that was added to your Glue
 job
ray job submit --address http://127.0.0.1:8265 --working-dir . \
  --runtime-env-json '{
    "pip": ["boto3==1.28.*","pyarrow==12.*","pandas==2.0.*"]
  }' \
  -- python job.py
```

La tâche peut être surveillée sur le tableau de bord Ray.

AWS Glue : comment ça marche

AWS Glue utilise d'autres AWS services pour orchestrer vos tâches ETL (extraction, transformation et chargement) afin de créer des entrepôts de données et des lacs de données et de générer des flux de sortie. AWS Glueappelle les opérations d'API pour transformer vos données, créer des journaux d'exécution, stocker la logique de vos tâches et créer des notifications pour vous aider à surveiller l'exécution de vos tâches. La console AWS Glue connecte ces services dans une application gérée, ce qui vous permet de vous concentrer sur la création et la surveillance de vos tâches ETL. La console effectue les opérations de développement et de tâches administratives en votre nom. Vous fournissez les informations d'identification et d'autres propriétés à AWS Glue pour accéder à vos sources de données et écrire dans vos cibles de données.

AWS Glue s'occupe de la mise en service et de la gestion des ressources requises pour exécuter votre charge de travail. Vous n'avez pas besoin de créer l'infrastructure pour un outil ETL car AWS Glue le fait pour vous. Lorsque les ressources sont requises, afin de réduire le temps de démarrage, AWS Glue utilise une instance à partir de son pool d'instances « à chaud » pour exécuter votre charge de travail.

AWS Glue vous permet de créer des tâches en utilisant des définitions de tables dans votre catalogue de données. Les tâches consistent en des scripts contenant les instructions qui exécutent les tâches de transformation de données souhaitées. Vous pouvez utiliser des déclencheurs pour initier des tâches sur un calendrier ou en tant que résultat d'un événement spécifié. Vous déterminez où résident vos données cible et les données source qui alimentent votre cible. Sur la base de vos entrées, AWS Glue transforme vos données du format source au format cible. Vous pouvez également fournir des scripts personnalisés dans la console ou l’API AWS Glue pour traiter vos données en fonction de vos besoins spécifiques.

Sources et destinations des données

AWS Glue pour Spark vous permet de lire et d'écrire des données issues de plusieurs systèmes et bases de données, notamment :

- Amazon S3
- Amazon DynamoDB
- Amazon Redshift
- Amazon Relational Database Service (Amazon RDS)
- Bases de données tierces accessibles à JDBC

- MongoDB et Amazon DocumentDB (compatible avec MongoDB)
- Autres connecteurs de place de marché et plug-ins Apache Spark

Flux de données

AWS Glue pour Spark peut diffuser des données à partir des systèmes suivants :

- Amazon Kinesis Data Streams
- Apache Kafka

AWS Glue est disponible dans plusieurs AWS régions. Pour de plus amples informations, veuillez consulter AWS Régions et points de terminaison dans le manuel Référence générale d'Amazon Web Services.

Rubriques

- Travaux ETL sans serveur exécutés en isolation
- AWS Glue concepts
- AWS Glue composants
- AWS Glue pour Spark et AWS Glue pour Ray
- Conversion de schémas semi-structurés en schémas relationnels avec AWS Glue
- AWS Systèmes de type Glue

Travaux ETL sans serveur exécutés en isolation

AWS Glue exécute vos tâches ETL dans un environnement sans serveur avec le moteur de votre choix, Spark ou Ray. AWS Glue exécute ces tâches sur les ressources virtuelles qu'il met en service et gère dans son propre compte de service.

AWS Glue est conçu pour effectuer les tâches suivantes :

- Isoler les données clients.
- Protéger les données client en transit et au repos.
- Accéder aux données des clients uniquement en cas de besoin, en réponse à leurs demandes, à l'aide d'informations d'identification temporaires et réduites, ou aux rôles IAM de leur compte lorsqu'ils y consentent.

Lors de la mise en service d'une tâche ETL, vous fournissez des sources de données d'entrée et des cibles de données de sortie dans votre cloud privé virtuel (VPC). En outre, vous fournissez le rôle IAM, l'ID VPC, l'ID de sous-réseau et le groupe de sécurité nécessaires pour accéder aux sources et aux cibles de données. Pour chaque tuple (ID de compte client, rôle IAM, ID de sous-réseau et groupe de sécurité), AWS Glue crée un environnement, isolé au niveau du réseau et de la gestion de tous les autres environnements au sein de votre compte de service AWS Glue.

Vous créez et configurez AWS Glue des ressources, telles que des catalogues de données, des tâches et des robots d'exploration au sein de votre AWS compte. Ces ressources sont ensuite associées au rôle IAM et aux paramètres réseau (sous-réseau et groupe de sécurité) que vous spécifiez lors du processus de création.

AWS Glue crée des interfaces réseau Elastic dans votre sous-réseau à l'aide d'adresses IP privées. Les tâches utilisent ces interfaces réseau élastiques pour accéder à vos sources et à vos cibles de données. Le trafic entrant, sortant et au sein de l'environnement d'exécution de la tâche est régi par votre VPC et vos stratégies de mise en réseau, avec une exception : les appels effectués vers des bibliothèques AWS Glue peuvent acheminer le trafic vers des opérations d'API AWS Glue via le VPC AWS Glue. Tous les appels d'API AWS Glue sont consignés ; par conséquent, les propriétaires de données peuvent effectuer un audit de l'accès aux API en activant AWS CloudTrail, qui fournit des journaux d'audit pour votre compte.

AWS Glueles environnements gérés qui exécutent vos tâches ETL sont protégés par les mêmes pratiques de sécurité que celles appliquées AWS aux autres services. Pour un aperçu des pratiques et des responsabilités partagées en matière de sécurité, consultez le livre blanc Introduction aux processus AWS de sécurité.

AWS Glue concepts

AWS Glue est un service ETL (extraction, transformation, chargement) entièrement géré qui vous permet de déplacer facilement des données entre différentes sources de données et cibles. Les composants clés sont les suivants :

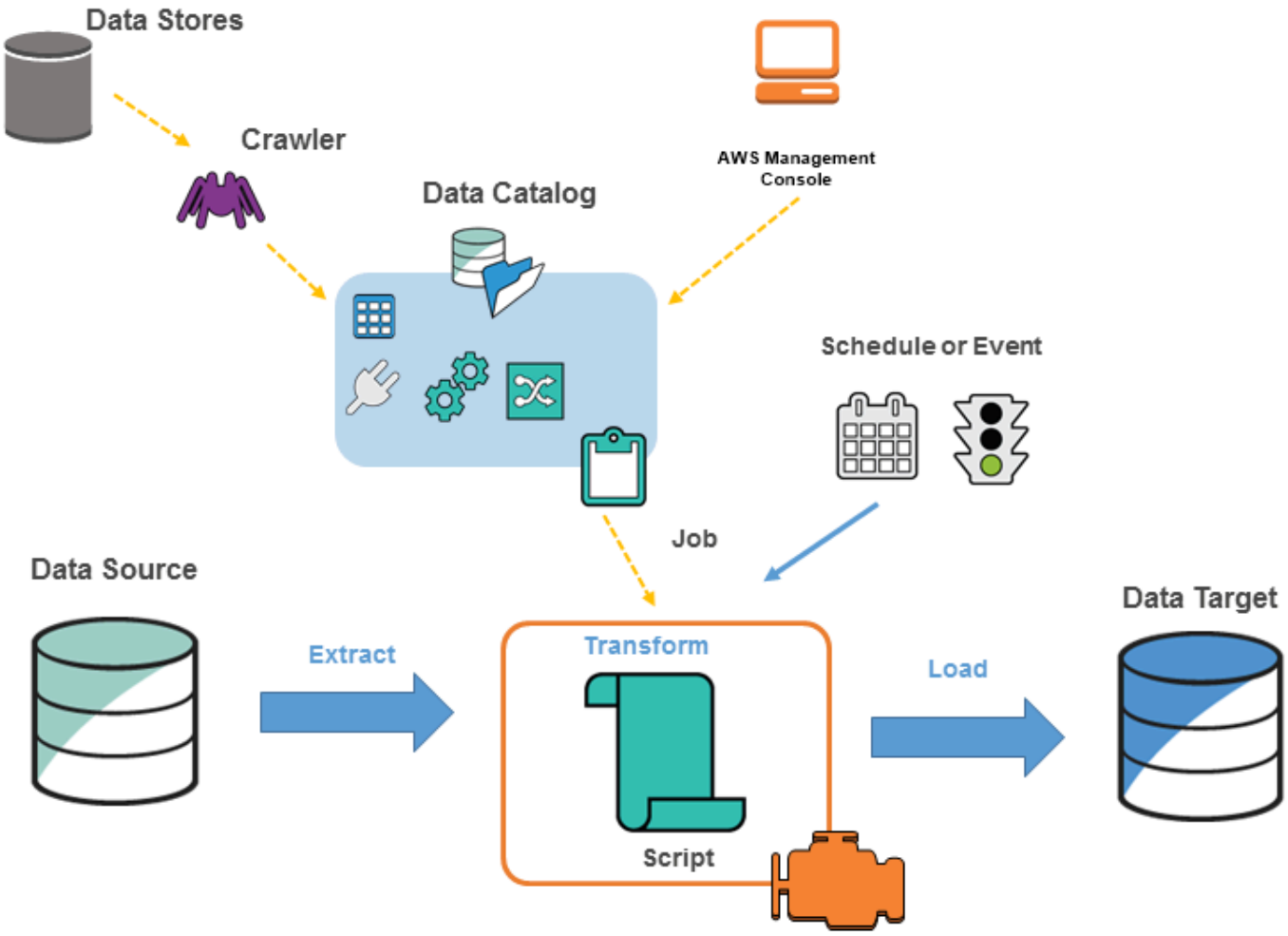
- Catalogue de données : magasin de métadonnées contenant des définitions de tables, des définitions de tâches et d'autres informations de contrôle pour vos flux de travail ETL.
- Robots : programmes qui se connectent à des sources de données, déduisent des schémas de données et créent des définitions de tables de métadonnées dans le catalogue de données.
- Tâches ETL : logique métier permettant d'extraire les données des sources, de les transformer à l'aide de scripts Apache Spark et de les charger dans des cibles.

- Déclencheurs : mécanismes permettant de lancer des exécutions de tâches en fonction de calendriers ou d’événements.

Le flux de travail type implique :

1. Définissez les sources de données et les cibles dans le catalogue de données.
2. Utilisez les robots pour renseigner le catalogue de données avec des métadonnées de table provenant de sources de données.
3. Définissez des tâches ETL avec des scripts de transformation pour déplacer et traiter les données.
4. Exécutez des tâches à la demande ou en fonction de déclencheurs.
5. Surveillez les performances des tâches à l’aide de tableaux de bord.

Le schéma suivant montre l'architecture d'un AWS Glue environnement.



Vous définissez des tâches AWS Glue pour accomplir le travail requis pour extraire, transformer et charger des données (ETL) d'une source de données vers une cible de données. Généralement, vous effectuez les actions suivantes :

- Pour les sources de magasins de données, vous définissez un crawler pour remplir votre AWS Glue Data Catalog avec les définitions de table des métadonnées. Vous associez votre crawler à un magasin de données, et le crawler crée les définitions de table dans le catalogue de données. Pour les sources en streaming, vous définissez manuellement les tables du catalogue de données et spécifiez les propriétés du flux des données.

  Outre les définitions de tables, le AWS Glue Data Catalog contient d'autres métadonnées nécessaires pour définir les tâches ETL. Vous utilisez ces métadonnées lorsque vous définissez une tâche pour transformer vos données.

- AWS Glue peut générer un script pour transformer vos données. Vous pouvez également fournir le script dans la AWS Glue console ou dans l'API.
- Vous pouvez exécuter votre tâche à la demande ou la configurer pour qu'elle démarre lorsqu'un déclencheur spécifié est mis en action. Le déclencheur peut être une planification temporelle ou un événement.

  Lorsque votre tâche s'exécute, un script extrait les données de la source de données, les transforme et les charge sur la cible de données. Le script s'exécute dans un environnement Apache Spark, dans AWS Glue.

> **Important**
>
> Les tables et les bases de données dans AWS Glue sont des objets dans le AWS Glue Data Catalog. Elles contiennent des métadonnées, mais aucune donnée d'un magasin de données.

Les données textuelles, telles que CSVs, doivent être codées **`UTF-8`** AWS Glue pour être traitées correctement. Pour plus d'informations, veuillez consulter la page Wikipédia UTF-8.

AWS Glue terminologie

AWS Glue repose sur l'interaction de plusieurs composants pour créer et gérer votre flux de travail d'extraction, de transformation et de chargement (ETL).

AWS Glue Data Catalog

Les métadonnées persistantes sont stockées dans AWS Glue. Il contient des définitions de tables, des définitions de tâches et d'autres informations de contrôle pour gérer votre AWS Glue environnement. Chaque AWS compte en possède un AWS Glue Data Catalog par région.

Classifieur

Détermine le schéma de vos données. AWS Glue fournit des classificateurs pour les types de fichiers courants, tels que CSV, JSON, AVRO, XML, etc. Il fournit également les classifieurs des systèmes de gestion de base de données relationnelle (SGBDR) utilisant une connexion JDBC. Vous pouvez

écrire votre propre classificateur à l'aide d'un modèle grok ou en spécifiant une balise de ligne dans un document XML.

Connexion

Un objet de catalogue de données qui contient les propriétés requises pour se connecter à un magasin de données particulier.

crawler

Programme qui se connecte à un magasin de données (source ou cible), parcourt la liste hiérarchisée des classificateurs pour déterminer le schéma de vos données, puis crée des tables de métadonnées dans l' AWS Glue Data Catalog.

Base de données

Ensemble de définitions de tables du catalogue de données associées, organisées en un groupe logique.

Stockage de données, source de données, cible de données

Un magasin de données est un référentiel qui permet de stocker vos données de façon permanente. Les exemples incluent les compartiments Amazon S3 et les bases de données relationnelles. Une source de données est un magasin de données qui est utilisé comme entrée d'un processus ou d'une transformation. Une cible de données est un magasin de données dans lequel écrit un processus ou une transformation.

Point de terminaison de développement

Un environnement que vous pouvez utiliser pour développer et tester vos scripts AWS Glue ETL.

Trame dynamique

Table distribuée qui prend en charge les données imbriquées telles que les structures et les tableaux. Chaque enregistrement est auto-descriptif, conçu pour une flexibilité de schéma avec des données semi-structurées. Chaque enregistrement contient à la fois les données et le schéma qui les décrit. Vous pouvez utiliser à la fois des cadres dynamiques et Apache Spark DataFrames dans vos scripts ETL, et effectuer des conversions entre eux. Les trames dynamiques fournissent un ensemble de transformations avancées pour le nettoyage des données, ainsi que pour l'extraction, le transfert et le chargement.

Tâche

Logique métier requise pour exécuter un travail d’extraction, de transformation et de chargement (ETL). Elle se compose d’un script de transformation, de sources de données et de cibles de données. Les exécutions des tâches sont initiées par les déclencheurs qui peuvent être planifiés ou mis en action par des événements.

Tableau de bord de performance de tâche

AWS Glue fournit un tableau de bord d'exécution complet pour vos tâches ETL. Le tableau de bord affiche des informations sur les exécutions de tâche à partir d’une période spécifique.

Interface de bloc-notes

Une expérience de bloc-notes améliorée grâce à une configuration en un clic pour faciliter la création des tâches et l’exploration des données. Le bloc-notes et les connexions sont configurés automatiquement pour vous. Vous pouvez utiliser l'interface du bloc-notes basée sur Jupyter Notebook pour développer, déboguer et déployer des scripts et des flux de travail de manière interactive à l'aide de l'infrastructure ETL Apache Spark AWS Glue sans serveur. Vous pouvez également effectuer des requêtes ad hoc, des analyses de données et des visualisations (par exemple, des tableaux et des graphiques) dans l’environnement de bloc-notes.

Script

Code qui extrait les données des sources, les transforme et les charge dans des cibles. AWS Glue génère PySpark ou des scripts Scala.

Table

Définition de métadonnées qui représente vos données. Que les données se trouvent dans un fichier Amazon Simple Storage Service (Amazon S3), une table Amazon Relational Database Service (Amazon RDS) ou un autre jeu de données, une table définit le schéma de vos données. Un tableau AWS Glue Data Catalog contient les noms des colonnes, les définitions des types de données, les informations de partition et les autres métadonnées relatives à un jeu de données de base. Le schéma de vos données est représenté dans la définition de votre AWS Glue table. Les données réelles restent dans leur magasin de données d'origine, qu'il s'agisse d'un fichier ou d'une table de base de données relationnelle. AWS Glue catalogue vos fichiers et vos tables de base de données relationnelle dans le. AWS Glue Data Catalog Ils sont utilisés comme sources et cibles lorsque vous créez une tâche ETL.

Transformation

Logique de code qui permet de manipuler les données dans un format différent.

Déclencheur

Démarre une tâche ETL. Les déclencheurs peuvent être définis selon une heure planifiée ou un événement.

Éditeur de tâche visuel

L'éditeur visuel est une interface graphique qui facilite la création, l'exécution et le contrôle des tâches d'Extraction, transformation et chargement (ETL) dans AWS Glue. Vous pouvez composer visuellement des flux de travail de transformation des données, les exécuter AWS Glue de manière fluide sur le moteur ETL sans serveur basé sur Apache Spark et inspecter le schéma et les résultats des données à chaque étape du travail.

Nœuds

Avec AWS Glue, vous ne payez que pour le temps nécessaire à l'exécution de votre tâche ETL. Il n'y a pas de ressources à gérer, pas de coûts initiaux et les temps de démarrage ou d'arrêt ne vous sont pas facturés. Un taux horaire vous est facturé en fonction du nombre d'unités de traitement des données (ou DPUs) utilisées pour exécuter votre tâche ETL. Une seule unité de traitement des données (DPU) est également appelée « travailleur ». AWS Glue propose plusieurs types de travailleurs pour vous aider à sélectionner la configuration qui répond à vos exigences en matière de latence et de coûts. Les travailleurs sont disponibles dans des configurations standard, G.1X, G.2X, G.4X, G.8X, G.12X, G.16X, G.025X, et des configurations R.1X, R.2X, R.4X, R.8X optimisées pour la mémoire.

AWS Glue composants

AWS Glue fournit une console et des opérations d'API pour configurer et gérer votre charge de travail d'extraction, de transformation et de chargement (ETL). Vous pouvez utiliser les opérations d'API via plusieurs langages spécifiques SDKs et le AWS Command Line Interface ()AWS CLI. Pour plus d'informations sur l'utilisation du AWS CLI, reportez-vous à la section Référence des AWS CLI commandes.

AWS Glue utilise le AWS Glue Data Catalog pour stocker les métadonnées relatives aux sources de données, aux transformations et aux cibles. Le Catalogue de données est un remplacement

instantané du métastore Apache Hive. AWS Glue Jobs system Il fournit une infrastructure gérée pour définir, planifier et exécuter des opérations ETL sur vos données. Pour plus d'informations sur l' AWS Glue API, consultezAWS Glue API.

AWS Glue console

Vous utilisez la AWS Glue console pour définir et orchestrer votre flux de travail ETL. La console appelle plusieurs opérations d'API dans AWS Glue Data Catalog et AWS Glue Jobs system pour effectuer les tâches suivantes :

- Définissez AWS Glue des objets tels que des tâches, des tables, des robots d'exploration et des connexions.
- Planifier l'exécution des crawlers.
- Définir les événements ou planifications des déclencheurs de tâche.
- Recherchez et filtrez des listes d' AWS Glue objets.
- Modifier les scripts de transformation.

AWS Glue Data Catalog

AWS Glue Data Catalog Il s'agit de votre magasin de métadonnées techniques permanent dans le AWS cloud.

Chaque AWS compte en possède un AWS Glue Data Catalog par AWS région. Chaque catalogue de données est un ensemble hautement évolutif de tableaux organisés en bases de données. Une table est une représentation par métadonnées d'un ensemble de données structurées ou semi-structurées stockées dans des sources telles qu'Amazon RDS, le système de fichiers distribué Apache Hadoop, Amazon OpenSearch Service, etc. AWS Glue Data Catalog Il fournit un référentiel uniforme dans lequel des systèmes disparates peuvent stocker et rechercher des métadonnées afin de suivre les données dans des silos de données. Vous pouvez ensuite utiliser les métadonnées pour interroger et transformer ces données de manière cohérente dans une grande variété d'applications.

Vous utilisez le catalogue de données ainsi que les Gestion des identités et des accès AWS politiques et Lake Formation pour contrôler l'accès aux tables et aux bases de données. Pour ce faire, vous pouvez autoriser différents groupes de votre entreprise à publier les données en toute sécurité à l'échelle de l'organisation tout en protégeant au maximum les informations sensibles.

Le catalogue de données, ainsi que Lake Formation, vous fournissent également des fonctionnalités complètes d'audit et de gouvernance, avec un suivi des modifications de schéma et des contrôles

d'accès aux données. CloudTrail Cela permet de s'assurer que les données ne sont pas modifiées de façon inappropriée ou partagées par inadvertance.

Pour plus d'informations sur la sécurisation et l'audit du AWS Glue Data Catalog, consultez :

- AWS Lake Formation – Pour en savoir plus, consultez Présentation d' AWS Lake Formation dans le Guide du développeur AWS Lake Formation .
- CloudTrail— Pour plus d'informations, voir Qu'est-ce que c'est CloudTrail ? dans le guide de AWS CloudTrail l'utilisateur.

Voici d'autres AWS services et projets open source qui utilisent : AWS Glue Data Catalog

- Amazon Athena – Pour plus d'informations, veuillez consulter la rubrique Présentation des tables, des bases de données et du catalogue de données dans le Guide de l'utilisateur Amazon Athena.
- Amazon Redshift Spectrum Pour en savoir plus, consultez la rubrique Utilisation d'Amazon Redshift Spectrum pour interroger des données externes dans le Guide du développeur de base de données Amazon Redshift.
- Amazon EMR - Pour plus d'informations, veuillez consulter la rubrique Utiliser de stratégies basées sur les ressources pour l'accès d'Amazon EMR à AWS Glue Data Catalog dans le Guide de gestion Amazon EMR.
- AWS Glue Data Catalog client pour le métastore Apache Hive — Pour plus d'informations sur ce GitHub projet, voir AWS Glue Data Catalog Client pour le métastore Apache Hive.

AWS Glue chenilles et classificateurs

AWS Glue vous permet également de configurer des robots d'exploration capables de scanner les données dans toutes sortes de référentiels, de les classer, d'en extraire des informations de schéma et de stocker les métadonnées automatiquement dans le. AWS Glue Data Catalog Ils AWS Glue Data Catalog peuvent ensuite être utilisés pour guider les opérations ETL.

Pour de plus amples informations sur la configuration des crawlers et des classifieurs, veuillez consulter Utilisation de robots pour alimenter le catalogue de données. Pour plus d'informations sur la programmation des robots d'exploration et des classificateurs à l'aide de l' AWS Glue API, consultez. API de classifieurs et d'crawlers

AWS Glue Opérations ETL

À l'aide des métadonnées du catalogue de données, AWS Glue vous pouvez générer automatiquement des scripts Scala ou PySpark (l'API Python pour Apache Spark) avec des AWS Glue extensions que vous pouvez utiliser et modifier pour effectuer diverses opérations ETL. Par exemple, vous pouvez extraire, nettoyer et transformer les données brutes, puis stocker le résultat dans un référentiel différent, où il peut être interrogé et analysé. Un tel script peut convertir un fichier CSV sous une forme relationnelle et l'enregistrer dans Amazon Redshift.

Pour plus d'informations sur l'utilisation des fonctionnalités AWS Glue ETL, consultezProgrammation de scripts Spark.

Streaming d'ETL en AWS Glue

AWS Glue vous permet d'effectuer des opérations ETL sur des données en streaming à l'aide de tâches exécutées en continu. AWS Glue L'ETL de streaming repose sur le moteur de streaming structuré Apache Spark et peut ingérer des flux provenant d'Amazon Kinesis Data Streams, d'Apache Kafka et d'Amazon Managed Streaming for Apache Kafka (Amazon MSK). ETL de streaming peut nettoyer et transformer les données de streaming et les charger dans des magasins de données Amazon S3 ou JDBC. Utilisez Streaming ETL in AWS Glue pour traiter les données d'événements telles que les flux IoT, les flux de clics et les journaux réseau.

Si vous connaissez le schéma de la source de données de streaming, vous pouvez le spécifier dans une table du catalogue de données. Sinon, vous pouvez activer la détection de schéma dans la tâche ETL de streaming. La tâche détermine ensuite automatiquement le schéma à partir des données entrantes.

La tâche ETL de streaming peut utiliser à la fois des transformations AWS Glue intégrées et des transformations natives d'Apache Spark Structured Streaming. Pour plus d'informations, consultez la section Opérations sur le DataFrames streaming/Datasets sur le site Web d'Apache Spark.

Pour de plus amples informations, veuillez consulter the section called "Tâches ETL en streaming".

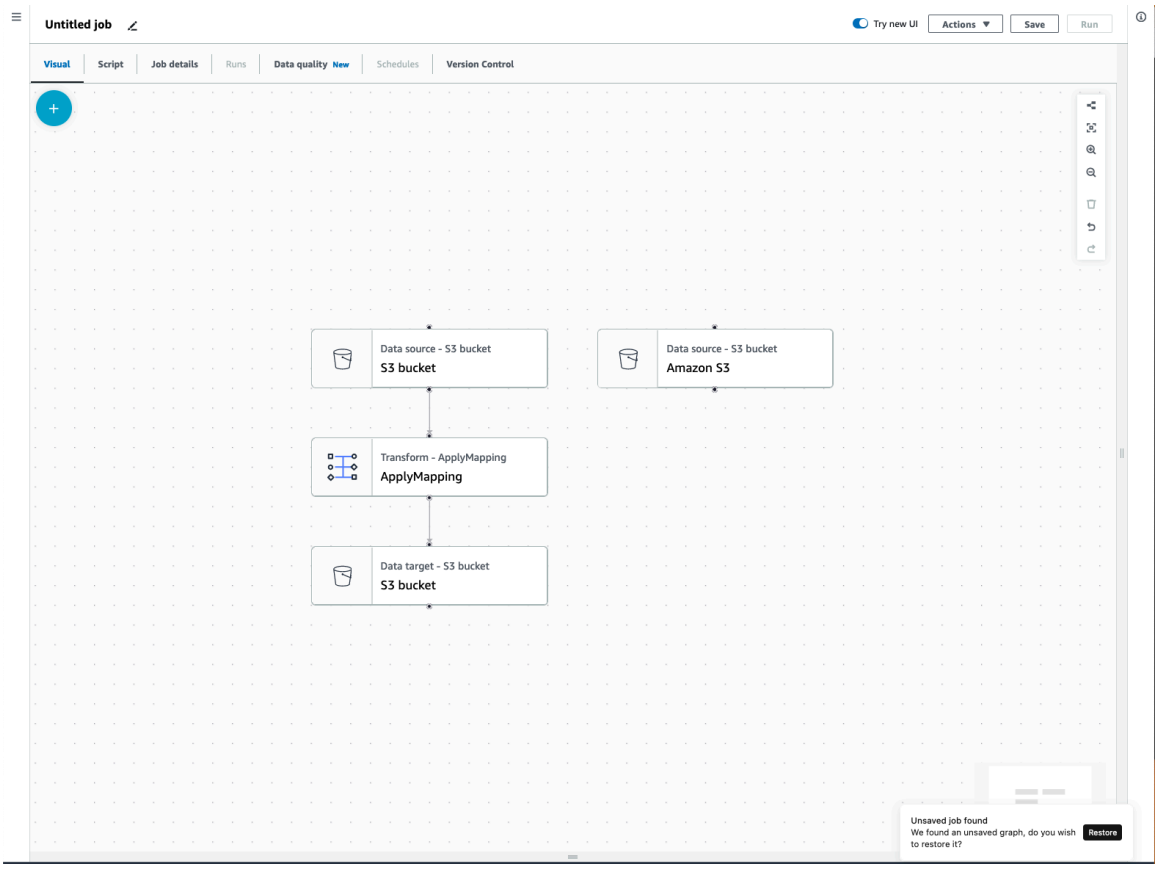
Le système AWS Glue d'emplois

AWS Glue Jobs system Il fournit une infrastructure gérée pour orchestrer votre flux de travail ETL. Vous pouvez créer des tâches AWS Glue qui automatisent les scripts que vous utilisez pour extraire, transformer et transférer des données vers différents emplacements. Les tâches peuvent être programmées et chaînées, ou elles peuvent être déclenchées par des événements tels que l'arrivée de nouvelles données.

Pour plus d'informations sur l'utilisation du AWS Glue Jobs system, consultezSurveillance AWS Glue. Pour plus d'informations sur la programmation à l'aide de l'API AWS Glue Jobs system , consultez API de tâches.

Composants ETL visuels

AWS Glue vous permet de créer des tâches ETL via un canevas visuel que vous pouvez manipuler.



Menu de tâches ETL

Les options du menu situées en haut du canevas vous permettent d'accéder aux différentes vues et aux détails de configuration de votre tâche.

- Visuel : le canevas de l'éditeur de tâche visuel. C'est ici que vous pouvez ajouter des nœuds pour créer une tâche.
- Script : représentation par script de votre tâche ETL. AWS Glue génère le script en fonction de la représentation visuelle de votre travail. Vous pouvez également modifier votre script ou le télécharger.

> **Note**
>
> Si vous choisissez de modifier le script, l'expérience de création de tâches est définitivement convertie en mode script uniquement. Ensuite, vous ne pourrez plus utiliser l'éditeur visuel pour modifier la tâche. Vous devez ajouter toutes les sources de tâches, les transformations et les cibles, et apporter toutes les modifications nécessaires à l'aide de l'éditeur visuel avant de choisir de modifier le script.

- Détails de la tâche : l'onglet Informations de la tâche vous permet de configurer votre tâche en définissant les propriétés de la tâche. Il existe des propriétés de base, telles que le nom et la description de votre tâche, le rôle IAM, le type de tâche, la version AWS Glue, la langue, le type de travailleur, le nombre de travailleurs, le signet de la tâche, l'exécution flexible, le nombre de suppressions et le délai d'expiration de la tâche, ainsi que des propriétés avancées, telles que les connexions, les bibliothèques, les paramètres de la tâche et les balises.
- Exécutions : après l'exécution de votre tâche, vous pouvez accéder à cet onglet pour consulter vos précédentes exécutions de tâches.
- Qualité des données : la qualité des données évalue et contrôle la qualité de vos ressources de données. Vous pouvez en savoir plus sur l'utilisation de la qualité des données dans cet onglet et ajouter une transformation de qualité des données à votre tâche.
- Calendriers : les tâches que vous avez planifiées apparaissent dans cet onglet. Si aucun calendrier n'est associé à cette tâche, cet onglet n'est pas accessible.
- Contrôle de version : vous pouvez utiliser Git avec votre tâche en la configurant dans un référentiel Git.

Panneaux ETL visuels

Lorsque vous travaillez dans le canevas, plusieurs panneaux sont disponibles pour vous aider à configurer vos nœuds ou à prévisualiser vos données et à visualiser le schéma de sortie.

- Propriétés : le panneau Propriétés apparaît lorsque vous choisissez un nœud sur votre canevas.
- Aperçu des données : le panneau Aperçu des données fournit un aperçu de la sortie des données afin que vous puissiez prendre des décisions avant d'exécuter votre tâche et d'examiner votre sortie.
- Schéma de sortie : l'onglet Schéma de sortie vous permet de visualiser et de modifier le schéma de vos nœuds de transformation.

Redimensionnement des panneaux

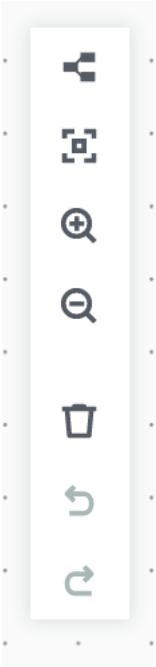
Vous pouvez redimensionner le panneau Propriétés sur le côté droit de l'écran et le panneau inférieur qui contient les onglets Aperçu des données et Schéma de sortie en cliquant sur le bord du panneau et en le faisant glisser vers la gauche et la droite ou vers le haut et le bas.

- Panneau Propriétés : redimensionnez le panneau Propriétés en cliquant et en faisant glisser le bord du canevas sur le côté droit de l'écran, puis en le faisant glisser vers la gauche pour augmenter sa largeur. Par défaut, le panneau est réduit et lorsqu'un nœud est sélectionné, le panneau Propriétés s'ouvre à sa taille par défaut.
- Panneau Aperçu des données et Schéma de sortie : redimensionnez le panneau inférieur en cliquant et en faisant glisser le bord inférieur du canevas en bas de l'écran, puis en le faisant glisser vers le haut pour augmenter sa hauteur. Par défaut, le panneau est réduit et lorsqu'un nœud est sélectionné, le panneau inférieur s'ouvre à sa taille par défaut.

Canevas de tâche

Vous pouvez ajouter, supprimer et des move/reorder nœuds directement sur le canevas Visual ETL. Considérez-le comme votre espace de travail pour créer une tâche ETL entièrement fonctionnelle qui commence par une source de données et peut se terminer par une cible de données.

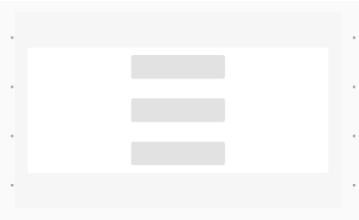
Lorsque vous travaillez avec des nœuds sur le canevas, vous disposez d'une barre d'outils qui vous permet d'effectuer des zooms avant et arrière, de supprimer des nœuds, d'établir ou de modifier des connexions entre les nœuds, de modifier l'orientation du flux de travail et d'annuler ou de rétablir une action.



La barre d'outils flottante est ancrée dans le coin supérieur droit du canevas et contient plusieurs images qui effectuent des actions :

- Icône de mise en page : la première icône de la barre d'outils est l'icône de mise en page. Par défaut, la direction des tâches visuelles est de haut en bas. Elle réorganise la direction de votre tâche visuelle en disposant les nœuds horizontalement de gauche à droite. Cliquez à nouveau sur l'icône de mise en page pour revenir de haut en bas.
- Icône de recentrage : l'icône de recentrage modifie l'affichage du canevas en le centrant. Vous pouvez l'utiliser pour des tâches importantes pour revenir à la position centrale.
- Icône de zoom avant : l'icône de zoom avant agrandit la taille des nœuds du canevas.
- Icône de zoom arrière : l'icône de zoom avant réduit la taille des nœuds du canevas.
- Icône corbeille : l'icône de corbeille supprime un nœud de la tâche visuelle. Vous devez d'abord sélectionner un nœud.
- Icône d'annulation : l'icône d'annulation annule la dernière action effectuée sur la tâche visuelle.
- Icône de rétablissement : l'icône de rétablissement répète la dernière action effectuée sur la tâche visuelle.

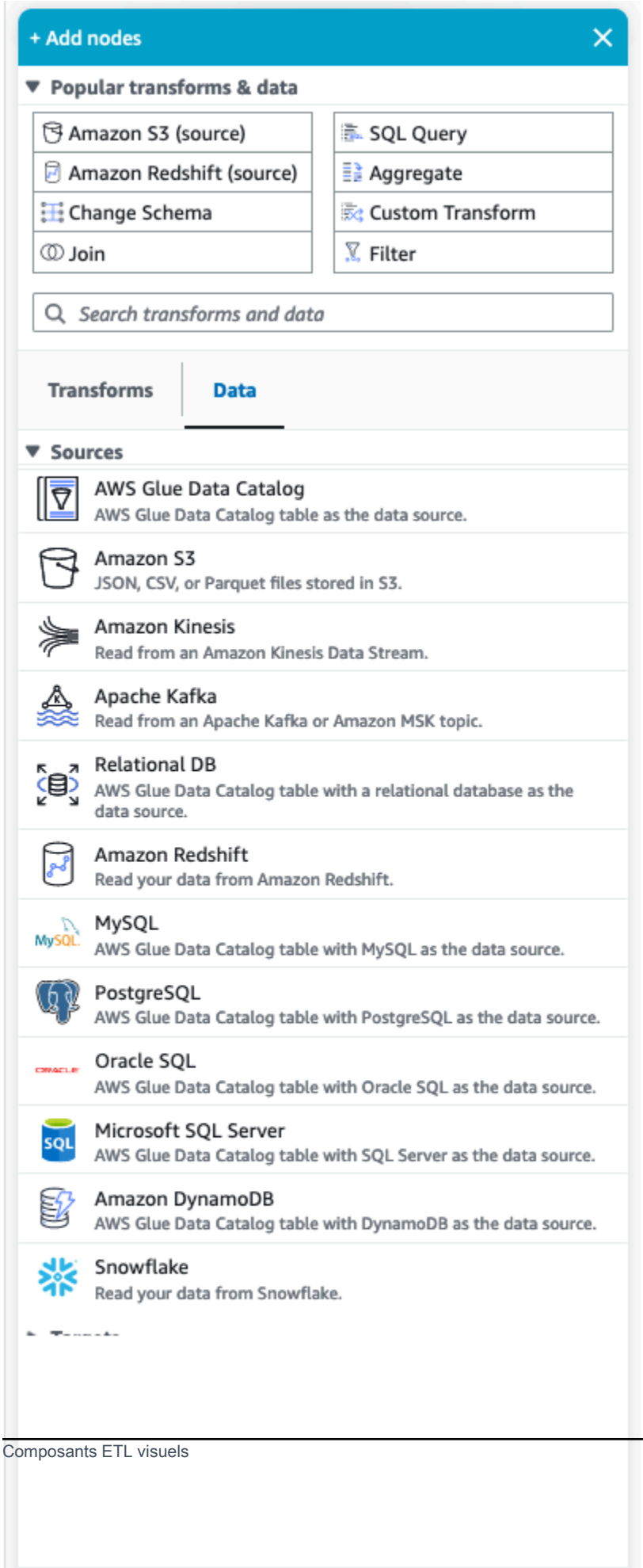
Utilisation de la mini-carte



Panneau des ressources

Le panneau des ressources contient toutes les sources de données, les actions de transformation et les connexions disponibles. Ouvrez le panneau des ressources sur le canevas en cliquant sur l’icône « + ». Cela ouvre le panneau des ressources.

Pour fermer le panneau des ressources, cliquez sur la croix, X, dans le coin supérieur droit du panneau des ressources. Le panneau est ainsi masqué jusqu’à ce que vous souhaitiez l’ouvrir à nouveau.


+ Add nodes
Popular transforms & data
Amazon S3 (source)
Amazon Redshift (source)
Change Schema
Join
SQL Query
Aggregate
Custom Transform
Filter
Search transforms and data
Transforms
Data
Sources
AWS Glue Data Catalog
AWS Glue Data Catalog table as the data source.
Amazon S3
JSON, CSV, or Parquet files stored in S3.
Amazon Kinesis
Read from an Amazon Kinesis Data Stream.
Apache Kafka
Read from an Apache Kafka or Amazon MSK topic.
Relational DB
AWS Glue Data Catalog table with a relational database as the data source.
Amazon Redshift
Read your data from Amazon Redshift.
MySQL
AWS Glue Data Catalog table with MySQL as the data source.
PostgreSQL
AWS Glue Data Catalog table with PostgreSQL as the data source.
Oracle SQL
AWS Glue Data Catalog table with Oracle SQL as the data source.
Microsoft SQL Server
AWS Glue Data Catalog table with SQL Server as the data source.
Amazon DynamoDB
AWS Glue Data Catalog table with DynamoDB as the data source.
Snowflake
Read your data from Snowflake.

Transformations et données populaires

En haut du panneau se trouve une collection de Transformations et de données populaires. Ces nœuds sont couramment utilisés dans AWS Glue. Choisissez-en un pour l'ajouter au canevas. Vous pouvez également masquer les Transformations et données populaires en cliquant sur le triangle situé à côté de l'en-tête Transformations et données populaires.

Dans la section Transformations et données populaires, vous pouvez rechercher des nœuds de transformations et de source de données. Les résultats apparaissent au fur et à mesure que vous saisissez du texte. Plus vous saisissez de caractères lors de votre requête de recherche, plus la liste des résultats diminuera. Les résultats de recherche sont renseignés à partir de la and/or description du nom du nœud. Choisissez un nœud pour l'ajouter à votre canevas.

Transformations et données

Deux onglets organisent les nœuds en Transformations et Données.

Transformations : lorsque vous choisissez l'onglet Transformations, toutes les transformations disponibles peuvent être sélectionnées. Choisissez une transformation pour l'ajouter au canevas. Vous pouvez également choisir Ajouter une transformation au bas de la liste des transformations, ce qui ouvrira une nouvelle page de documentation sur la création de Transformations visuelles personnalisées. En suivant les étapes, vous pourrez créer vos propres transformations. Vos transformations apparaîtront alors dans la liste des transformations disponibles.

Données : l'onglet de données contient tous les nœuds des Sources et des Cibles. Vous pouvez masquer les sources et les cibles en cliquant sur le triangle situé à côté de l'en-tête sources ou cibles. Vous pouvez afficher les sources et les cibles en cliquant à nouveau sur le triangle. Choisissez un nœud source ou cible pour l'ajouter au canevas. Vous pouvez également choisir Gérer les connexions pour ajouter une nouvelle connexion. Cela ouvrira la page Connecteurs dans la console.

AWS Glue pour Spark et AWS Glue pour Ray

Dans AWS Glue Apache Spark (AWS Glue ETL), vous pouvez PySpark écrire du code Python pour gérer les données à grande échelle. Spark est une solution familière à ce problème, mais les ingénieurs de données formés à Python peuvent trouver la transition peu intuitive. Le DataFrame modèle Spark n'est pas parfaitement « pythonique », ce qui reflète le langage Scala et le runtime Java sur lesquels il repose.

Dans AWS Glue, vous pouvez utiliser des tâches shell Python pour exécuter des intégrations de données Python natives. Ces tâches s'exécutent sur une seule EC2 instance Amazon et sont limitées

par la capacité de cette instance. Elle limite le débit des données que vous pouvez traiter et devient coûteuse à gérer lorsqu'il s'agit de Big Data.

AWS Glue for Ray vous permet d'augmenter les charges de travail en Python sans investir de manière substantielle dans l'apprentissage de Spark. Vous pouvez tirer parti de certains scénarios dans lesquels Ray est plus performant. En vous offrant le choix, vous pouvez utiliser les points forts de Spark et de Ray.

AWS Glue ETL et AWS Glue for Ray sont différents en dessous, ils supportent donc des fonctionnalités différentes. Consultez la documentation pour identifier les fonctionnalités prises en charge.

Qu'est-ce que c'est AWS Glue pour Ray ?

Ray est une infrastructure de calcul distribuée open source que vous pouvez utiliser pour augmenter les charges de travail, en mettant l'accent sur Python. Pour plus d'informations sur Ray, consultez le site Web de Ray. AWS Glue Les tâches de Ray et les sessions interactives vous permettent d'utiliser Ray dans Ray AWS Glue.

Vous pouvez utiliser AWS Glue for Ray pour écrire des scripts Python pour des calculs qui s'exécuteront en parallèle sur plusieurs machines. Dans les tâches et les sessions interactives Ray, vous pouvez utiliser des bibliothèques Python familières telles que Pandas, afin de faciliter l'écriture et l'exécution de vos flux de travail. Pour plus d'informations sur les jeux de données Ray, veuillez consulter la rubrique Jeux de données Ray dans la documentation Ray. Pour plus d'informations sur pandas, veuillez consulter le site Web Pandas.

Avec AWS Glue for Ray, vous pouvez exécuter vos flux de travail Pandas sur des mégadonnées à l'échelle de l'entreprise, avec seulement quelques lignes de code. Vous pouvez créer une tâche Ray à partir de la AWS Glue console ou du AWS SDK. Vous pouvez également ouvrir une session AWS Glue interactive pour exécuter votre code dans un environnement Ray sans serveur. Les tâches visuelles dans ne AWS Glue Studio sont pas encore prises en charge.

AWS Glue les jobs for Ray vous permettent d'exécuter un script selon un calendrier ou en réponse à un événement d'Amazon EventBridge. Les jobs stockent des informations de journal et des statistiques de surveillance CloudWatch qui vous permettent de comprendre l'état et la fiabilité de votre script. Pour plus d'informations sur le système des AWS Glue tâches, consultezthe section called “Utilisation des tâches Ray”.

Ray automatise la mise à l'échelle du code Python en répartissant le traitement sur un cluster de machines qu'il reconfigure en temps réel, en fonction de la charge. Cette opération peut entraîner

une amélioration des performances par dollar pour certaines charges de travail. Avec Ray Jobs, nous avons intégré la mise à l'échelle automatique de manière native dans le modèle de AWS Glue travail, afin que vous puissiez tirer pleinement parti de cette fonctionnalité. Les travaux de Ray sont exécutés sur AWS Graviton, ce qui se traduit par une meilleure performance globale en termes de prix.

Outre les économies de coûts qu'elle permet, vous pouvez utiliser la mise à l'échelle automatique native pour exécuter les charges de travail Ray sans consacrer de temps à la maintenance, au réglage et à l'administration des clusters. Vous pouvez utiliser des bibliothèques open source familières prêtes à l'emploi, telles que Pandas, et le AWS SDK pour Pandas. Elles améliorent la vitesse d'itération pendant que vous développez sur AWS Glue pour Ray. Lorsque vous utiliserez AWS Glue for Ray, vous serez en mesure de développer et d'exécuter rapidement des charges de travail d'intégration de données rentables.

Conversion de schémas semi-structurés en schémas relationnels avec AWS Glue

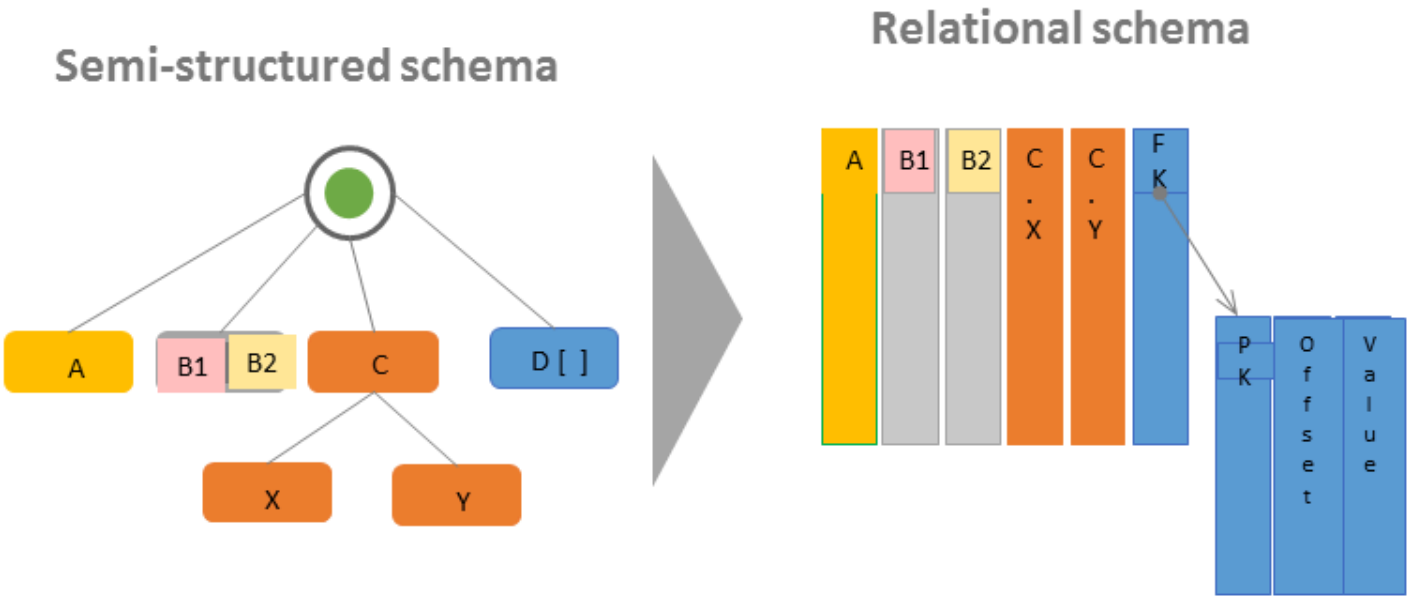
Il est courant de vouloir convertir des données semi-structurées en tables relationnelles. Conceptuellement, vous aplatissez un schéma hiérarchique pour en faire un schéma relationnel. AWS Gluepeut effectuer cette conversion pour vous on-the-fly.

Les données semi-structurées contiennent généralement un balisage pour identifier les entités au sein des données. Il peut inclure des structures de données imbriquées sans schéma fixe. Pour en savoir plus sur les données semi-structurées, consultez Semi-structured data dans Wikipédia.

Les données relationnelles sont représentées par des tables composées de lignes et de colonnes. Les relations entre les tables peuvent être représentées par une relation entre clé primaire et clé étrangère. Pour en savoir plus, consultez Base de données relationnelle dans Wikipedia.

AWS Glue utilise des crawlers pour déduire des schémas pour les données semi-structurées. Il transforme ensuite les données en un schéma relationnel à l'aide d'une tâche ETL (extraction, transformation et chargement). Par exemple, vous souhaiterez peut-être analyser les données JSON des fichiers source Amazon Simple Storage Service (Amazon S3) vers les tables Amazon Relational Database Service (Amazon RDS). Comprendre comment AWS Glue gère les différences entre les schémas peut vous aider à comprendre le processus de transformation.

Ce diagramme montre comment AWS Glue transforme un schéma semi-structuré en schéma relationnel.



Le diagramme illustre les éléments suivants :

- La valeur unique `A` est convertie directement en une colonne relationnelle.
- La paire de valeurs `B1` et `B2` est convertie en deux colonnes relationnelles.
- La structure `C` et ses enfants `X` et `Y` sont convertis en deux colonnes relationnelles.
- Le tableau `D[]` est converti en une colonne relationnelle avec une clé étrangère qui pointe vers une autre table relationnelle. En plus d'une clé primaire, la seconde table relationnelle comporte des colonnes qui contiennent le décalage et la valeur des éléments du tableau.

AWS Systèmes de type Glue

AWS Glue utilise plusieurs types de systèmes pour fournir une interface polyvalente sur des systèmes de données qui stockent les données de manières très différentes. Ce document met fin à l'ambiguïté des systèmes de type AWS Glue et des normes de données.

AWS Types de catalogues de données Glue

Le catalogue de données est un metastore, un registre de tables et de champs stockés dans divers systèmes de données. Lorsque les composants AWS Glue, tels que les robots AWS Glue et les tâches AWS Glue with Spark, écrivent dans le catalogue de données, ils le font à l'aide d'un système de typage interne permettant de suivre les types de champs. Ces valeurs sont affichées dans la colonne Type de données du schéma de table de la console AWS Glue. Ce système de types est basé sur le système de types d'Apache Hive. Pour plus d'informations sur le système de types Apache Hive, veuillez consulter la rubrique Types du wiki Apache Hive. Pour plus d'informations sur les types spécifiques et leur support, des exemples sont fournis dans la console AWS Glue, dans le cadre du Schema Builder.

Validation, compatibilité et autres utilisations

Le catalogue de données ne valide pas les types écrits dans des champs de type. Lorsque les composants de AWS Glue lisent et écrivent dans le catalogue de données, ils sont compatibles entre eux. AWS Les composants Glue visent également à préserver un haut degré de compatibilité avec les types Hive. Cependant, les composants AWS Glue ne garantissent pas la compatibilité avec tous les types de Hive. Cela permet l'interopérabilité avec des outils tels que les DDL Athena lorsque vous utilisez des tables dans le catalogue de données.

Le catalogue de données ne validant pas les types, d'autres services peuvent utiliser le catalogue de données pour suivre les types à l'aide de systèmes strictement conformes au système de types Hive ou à tout autre système.

Types dans les scripts AWS Glue with Spark

Lorsqu'un script AWS Glue with Spark interprète ou transforme un ensemble de données, nous fournissons `DynamicFrame` une représentation en mémoire de votre ensemble de données tel qu'il est utilisé dans votre script. L'objectif d'un `DynamicFrame` est similaire à celui du `DataFrame` Spark : il modélise votre jeu de données afin que Spark puisse planifier et exécuter des transformations sur vos données. Nous garantissons que la représentation du type de `DynamicFrame` est compatible avec un `DataFrame` en fournissant les méthodes `toDF` et `fromDF`.

Si les informations de type peuvent être déduites ou fournies à un `DataFrame`, elles peuvent être déduites ou fournies à un `DynamicFrame`, sauf indication contraire. Lorsque nous fournissons des lecteurs ou des rédacteurs optimisés pour des formats de données spécifiques, si Spark peut lire ou écrire vos données, les lecteurs et rédacteurs pourront le faire également, sous réserve de restrictions documentées. Pour plus d'informations sur les lecteurs et les rédacteurs, veuillez consulter la rubrique the section called “Options de format de données”.

Le type de choix

Les `DynamicFrames` fournissent un mécanisme pour modéliser les champs d'un jeu de données dont les valeurs peuvent présenter des types incohérents sur le disque entre les lignes. Par exemple, un champ peut contenir un nombre stocké sous forme de chaîne dans certaines lignes et un entier dans d'autres. Ce mécanisme est un type en mémoire appelé `Choice`. Nous proposons des transformations, telles que la `ResolveChoice` méthode, pour résoudre les colonnes Choice en un type concret. AWS Glue ETL n'écrit pas le type Choice dans le catalogue de données dans le cours normal des opérations ; les types Choice n'existent que dans le contexte des modèles de DynamicFrame mémoire des ensembles de données. Pour un exemple d'utilisation du type de choix, veuillez consulter la rubrique the section called “Exemple de préparation des données”.

AWS Types de Glue Crawler

Les crawlers visent à produire un schéma cohérent et utilisable pour votre ensemble de données, puis à le stocker dans Data Catalog pour l'utiliser dans d'autres composants de AWS Glue et Athena. Les crawlers traitent les types comme indiqué dans la section précédente du catalogue de données, the section called “AWS Types de catalogues de données Glue”. Pour produire un type utilisable dans les scénarios de type « choix », où une colonne contient des valeurs de deux types ou plus, les Crawlers créeront un type `struct` qui modélise les types potentiels.

Démarrer avec AWS Glue

Les sections suivantes présentent des informations sur la configuration AWS Glue. Toutes les sections de configuration ne sont pas nécessaires pour commencer à utiliser AWS Glue. Vous pouvez utiliser les instructions nécessaires pour configurer les autorisations IAM, le chiffrement et le DNS (si vous utilisez un VPC Environnement pour accéder aux centres de stockage des données ou si vous utilisez des séances interactives).

Rubriques

- Présentation de l'utilisation AWS Glue
- Configuration des autorisations IAM pour AWS Glue
- Configuration des profils AWS Glue d'utilisation
- Mise en route avec l'AWS Glue Data Catalog
- Configuration de l'accès réseau aux magasins de données
- Configuration du chiffrement dans AWS Glue
- Configuration du réseau pour le développement de AWS Glue

Présentation de l'utilisation AWS Glue

Avec AWS Glue, vous stockez les métadonnées dans le AWS Glue Data Catalog. Vous utilisez ces métadonnées pour orchestrer des tâches ETL qui transforment des sources de données et chargent votre entrepôt de données ou votre lac de données. Les étapes suivantes décrivent le flux de travail général et certains des choix qui s'offrent à vous lorsque vous travaillez avec AWS Glue.

> **Note**
>
> Vous pouvez suivre les étapes ci-dessous, ou créer un flux de travail qui exécute automatiquement les étapes 1 à 3. Pour de plus amples informations, veuillez consulter the section called "Exécution d'activités ETL complexes à l'aide de plans et de flux de travail".

1. Renseignez le AWS Glue Data Catalog avec les définitions des tables.

   Dans la console, pour les magasins de données persistantes, vous pouvez ajouter un crawler pour remplir le AWS Glue Data Catalog. Vous pouvez lancer l'assistant Add crawler (Ajout d'un crawler)

à partir de la liste des tables ou de la liste des crawlers. Vous choisissez un ou plusieurs magasins de données auxquels votre crawler accèdera. Vous pouvez également créer un calendrier pour déterminer la fréquence d'exécution de votre crawler. Pour les flux de données, vous pouvez créer manuellement la définition de table et définir les propriétés de flux.

Si vous le souhaitez, vous pouvez fournir un classifieur personnalisé qui déduit le schéma de vos données. Vous pouvez créer des classifieurs personnalisés à l'aide d'un modèle grok. Toutefois, AWS Glue fournit des classifieurs intégrés qui sont automatiquement utilisés par les crawlers si un classifieur personnalisé ne reconnaît pas vos données. Lorsque vous définissez un crawler, vous n'avez pas besoin de sélectionner un classifieur. Pour plus d'informations sur les classifieurs dans AWS Glue, consultez Définition et gestion des classificateurs.

L'analyse de certains types de magasins de données nécessite une connexion qui fournit des informations de localisation et d'authentification. Si nécessaire, vous pouvez créer une connexion qui fournit ces informations requises dans la console AWS Glue.

L'crawler lit votre magasin de données et crée des définitions de données et des tables nommées dans le AWS Glue Data Catalog. Ces tables sont organisées dans une base de données de votre choix. Vous pouvez également remplir le catalogue de données avec des tables créées manuellement. Avec cette méthode, vous fournissez le schéma et d'autres métadonnées pour créer des définitions de table dans le catalogue de données. Cette méthode pouvant être un peu fastidieuse et source d'erreurs, il est souvent préférable de faire créer les définitions de table par un crawler.

Pour plus d'informations sur le AWS Glue Data Catalog remplissage des définitions de tables, consultezCréation de tables.

2. Définissez une tâche qui décrit la transformation de données de la source vers la cible.

   En général, pour créer une tâche, vous devez faire les choix suivants :

   - Choisissez un tableau dans le tableau AWS Glue Data Catalog qui sera la source de la tâche. Votre tâche utilise cette définition de table pour accéder à votre source de données et interpréter le format de ces dernières.
   - Choisissez une table ou un emplacement parmi ceux AWS Glue Data Catalog qui seront la cible de la tâche. Votre tâche utilise cette information pour accéder à votre magasin de données.
   - Demandez à AWS Glue de générer un script pour transformer votre source en cible. AWS Glue génère le code pour appeler des transformations intégrées destinées à convertir les données de son schéma source au format du schéma cible. Ces transformations réalisent des opérations

comme copier des données, renommer des colonnes et filtrer des données pour transformer les données si nécessaire. Vous pouvez modifier ce script dans la console AWS Glue.

Pour en savoir plus sur la définition des tâches dans AWS Glue, consultez Création de tâches ETL visuelles.

3. Exécutez votre tâche pour transformer vos données.

   Vous pouvez exécuter votre tâche à la demande, ou la démarrer en fonction d'un des types de déclencheurs suivants :

   - Un déclencheur basé sur une planification cron.
   - Un déclencheur basé sur un événement ; par exemple, la réussite de l'exécution d'une autre tâche peut démarrer une tâche AWS Glue.
   - Un déclencheur qui lance une tâche à la demande.

   Pour en savoir plus sur les déclencheurs dans AWS Glue, consultez Démarrage des tâches et des crawlers à l'aide de déclencheurs.

4. Surveillez vos crawlers planifiés et vos tâches déclenchées.

   Utilisez la console AWS Glue pour afficher les informations suivantes :

   - Les détails et les erreurs de l'exécution d'une tâche.
   - Les détails et les erreurs de l'exécution d'un crawler.
   - Toutes les notifications sur les activités AWS Glue

   Pour en savoir plus sur la surveillance de vos crawlers et de vos tâches dans AWS Glue, consultez Surveillance AWS Glue.

Configuration des autorisations IAM pour AWS Glue

Les instructions de cette rubrique vous aident à configurer rapidement des autorisations Gestion des identités et des accès AWS (IAM) pour AWS Glue. Vous réaliserez les tâches suivantes :

- Accordez à vos identités IAM l'accès aux AWS Glue ressources.
- Créez un rôle de service pour exécuter des tâches, accéder aux données et exécuter AWS Glue des tâches de qualité des données.

Pour obtenir des instructions détaillées que vous pouvez utiliser pour personnaliser les autorisations IAM AWS Glue, consultezConfiguration des autorisations IAM pour AWS Glue.

Pour configurer les autorisations IAM pour AWS Glue AWS Management Console

1. Connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/.
2. Choisissez Mise en route.
3. Sous Préparer votre compte pour AWS Glue, choisissez Configurer les autorisations IAM.
4. Choisissez les identités IAM (rôles ou utilisateurs) auxquelles vous souhaitez accorder AWS Glue des autorisations. AWS Glue attache la politique `AWSGlueConsoleFullAccess` gérée à ces identités. Vous pouvez ignorer cette étape si vous souhaitez définir ces autorisations manuellement ou uniquement définir une fonction du service par défaut.
5. Choisissez Suivant.
6. Choisissez le niveau d'accès Amazon S3 dont vos rôles et utilisateurs ont besoin. Les options que vous choisissez à cette étape sont appliquées à toutes les identités que vous avez sélectionnées.

   a. Sous Choisir les emplacements S3, choisissez les emplacements Amazon S3 auxquels vous souhaitez accorder l'accès.

   b. Ensuite, indiquez si vos identités doivent avoir un accès en lecture seule (recommandé) ou en lecture et écriture aux emplacements que vous avez sélectionnés précédemment. AWS Glue ajoute des politiques d'autorisation à vos identités en fonction de la combinaison des emplacements et des autorisations de lecture ou d'écriture que vous sélectionnez.

   Le tableau suivant indique les autorisations associées AWS Glue à l'accès à Amazon S3.

Si vous choisissez…	AWS Glue attache...
Pas de modification	Aucune autorisation. AWS Glue n'apporte ra aucune modification aux autorisations de votre identité.
Accorder l'accès à des emplacements Amazon S3 spécifiques (lecture seule)	La politique gérée par le client, `AWSGlueConsole-S3-read-only-policy` , accorde l'accès à des

Si vous choisissez…	AWS Glue attache...
	emplacements Amazon S3 spécifiques avec des autorisations en lecture seule. JSON `{ "Version":"2012-10-17", "Statement": [{ "Effect": "Allow", "Action": ["s3:GetObject"], "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket1/*", "arn:aws:s3:::amzn-s3-demo-bucket2/*", "arn:aws:s3:::amzn-s3-demo-bucket3", "arn:aws:s3:::amzn-s3-demo-bucket"], "Condition": { "StringEquals": { "aws:ResourceAccount": "000000000000" } } }] }`

Si vous choisissez…	AWS Glue attache...
Accorder l'accès à des emplacements Amazon S3 spécifiques (lecture et écriture)	La `AWSGlueConsole-S3-read-and-write-policy` accorde l'accès à des emplacements Amazon S3 spécifiques avec des autorisations en lecture et en écriture. JSON **{** **"Version":"2012-10-17",** **"Statement": [** **{** **"Effect": "Allow",** **"Action": [** **"s3:GetObject",** **"s3:PutObject"** **],** **"Resource": [** **"arn:aws:s3:::aes-siem-00000000000-log/*",** **"arn:aws:s3:::aes-siem-00000000000-snapshot/*",** **"arn:aws:s3:::aes-siem-00000000000-log",** **"arn:aws:s3:::aes-siem-00000000000-snapshot"** **],** **"Condition": {** **"StringEquals": {** **"aws:ResourceAccount": "000000000000"** **}** **}** **}** **]**

Si vous choisissez…	AWS Glue attache...
	`}`

7. Choisissez Suivant.
8. Choisissez un rôle AWS Glue de service par défaut pour votre compte. Un rôle de service est un rôle IAM qui AWS Glue permet d'accéder aux ressources d'autres AWS services en votre nom. Pour de plus amples informations, veuillez consulter Rôles de service pour AWS Glue.

   - Lorsque vous choisissez le rôle de AWS Glue service standard, vous AWS Glue créez un nouveau rôle IAM dans votre Compte AWS nom `AWSGlueServiceRole` avec les politiques gérées suivantes jointes. Si votre compte possède déjà un rôle IAM nommé`AWSGlueServiceRole`, AWS Glue associe ces politiques au rôle existant.
     - AWSGlueServiceRole— Cette politique gérée est requise pour accéder AWS Glue aux ressources et les gérer en votre nom. Il permet AWS Glue de créer, de mettre à jour et de supprimer diverses ressources telles que des AWS Glue jobs, des crawlers et des connexions. Cette politique accorde également des autorisations pour accéder AWS Glue aux Amazon CloudWatch journaux à des fins de journalisation. Pour commencer, nous vous recommandons d'utiliser cette politique pour savoir comment utiliser AWS Glue. Au fur et à mesure que vous vous y AWS Glue habituerez, vous pourrez créer des politiques qui vous permettront d'ajuster l'accès aux ressources selon vos besoins.
     - AWSGlueConsoleFullAccess— Cette politique gérée accorde un accès complet au AWS Glue service via le AWS Management Console. Cette politique accorde des autorisations pour effectuer n'importe quelle opération dans ce cadre AWS Glue, vous permettant de créer, de modifier et de supprimer n'importe quelle AWS Glue ressource selon vos besoins. Cependant, il est important de noter que cette politique n'accorde pas d'autorisations pour accéder aux magasins de données sous-jacents ou à d'autres AWS services susceptibles d'être impliqués dans le processus ETL. En raison de la large portée des autorisations accordées par la politique `AWSGlueConsoleFullAccess`, celles-ci doivent être attribuées avec prudence et en respectant le principe du moindre privilège. Il est généralement recommandé de créer et d'utiliser des politiques plus détaillées adaptées à des cas d'utilisation et à des exigences spécifiques dans la mesure du possible.
     - AWSGlueConsole-S3- read-only-policy — Cette politique permet de lire des données AWS Glue à partir de compartiments Amazon S3 spécifiés, mais elle n'accorde pas l'autorisation d'écrire ou de modifier des données dans Amazon S3 ou

AWSGlueConsole-S3- read-and-write — Cette politique permet de AWS Glue lire et d'écrire des données dans des compartiments Amazon S3 spécifiés dans le cadre du processus ETL.

- Lorsque vous choisissez un rôle IAM existant, AWS Glue définissez-le comme rôle par défaut, mais n'y ajoutez pas d'`AWSGlueServiceRole`autorisations. Assurez-vous d'avoir configuré le rôle à utiliser en tant que rôle de service pour AWS Glue. Pour plus d'informations, consultez Étape 1 : créer une politique IAM pour le service AWS Glue et Étape 2 : créer un rôle IAM pour AWS Glue.

9. Choisissez Suivant.
10. Enfin, passez en revue les autorisations que vous avez sélectionnées, puis choisissez Appliquer les modifications. Lorsque vous appliquez les modifications, AWS Glue ajoute des autorisations IAM aux identités que vous avez sélectionnées. Vous pouvez consulter ou modifier les nouvelles autorisations dans la console IAM à https://console.aws.amazon.com/iam/l'adresse.

Vous avez maintenant terminé la configuration des autorisations IAM minimales pour AWS Glue. Dans un environnement de production, nous vous recommandons de vous familiariser avec les AWS ressources Sécurité dans AWS Glue adaptées Gestion des identités et des accès pour AWS Glue à votre cas d'utilisation et de vous aider à les sécuriser.

Étapes suivantes

Maintenant que les autorisations IAM sont configurées, vous pouvez explorer les rubriques suivantes pour commencer à utiliser AWS Glue :

- Commencer avec AWS GlueAWS Skill Builder
- Mise en route avec l'AWS Glue Data Catalog

Configuration pour AWS Glue Studio

Effectuez les tâches de cette section lorsque vous utilisez AWS Glue pour la tâche ETL visuelle pour la première fois :

Rubriques

- Autorisations IAM nécessaires pour l'utilisateur AWS Glue Studio
- Examinez les autorisations IAM nécessaires pour les tâches ETL

- Définition des autorisations IAM pour AWS Glue Studio
- Configuration d'un VPC pour votre tâche ETL

Autorisations IAM nécessaires pour l'utilisateur AWS Glue Studio

Pour l'utiliserAWS Glue Studio, l'utilisateur doit avoir accès à différentes AWS ressources. L'utilisateur doit être en mesure d'afficher et de sélectionner des compartiments Amazon S3, des politiques et des rôles IAM, de même que des objets AWS Glue Data Catalog.

Autorisations de service AWS Glue

AWS Glue Studio utilise les actions et les ressources du service AWS Glue. Votre utilisateur a besoin d'autorisations sur ces actions et ressources pour utiliser efficacement AWS Glue Studio. Vous pouvez allouer à l'utilisateur de AWS Glue Studio la politique gérée `AWSGlueConsoleFullAccess`, ou créer une politique personnalisée avec un ensemble d'autorisations plus restreint.

> **Important**
>
> Conformément aux bonnes pratiques en matière de sécurité, il est recommandé de restreindre l'accès en limitant les politiques afin de réduire davantage l'accès au compartiment Amazon S3 et aux groupes de journaux Amazon CloudWatch . Pour obtenir un exemple de politique Amazon S3, consultez la rubrique Writing IAM Policies: How to Grant Access to an Amazon S3 Bucket (Rédaction de politiques IAM : comment accorder l'accès à un compartiment Amazon S3).

Création de politiques IAM personnalisées pour AWS Glue Studio

Vous pouvez créer une politique personnalisée avec un ensemble d'autorisations plus réduit pour AWS Glue Studio. La politique peut accorder des autorisations pour un sous-ensemble d'objets ou d'actions. Utilisez les informations suivantes lors de la création d'une politique personnalisée.

Pour utiliser le AWS Glue Studio APIs, incluez `glue:UseGlueStudio` dans la politique d'action vos autorisations IAM. A l'aide de `glue:UseGlueStudio`, vous pourrez accéder à toutes les actions AWS Glue Studio même si d'autres actions sont ajoutées à l'API au fil du temps.

Pour plus d'informations sur les actions définies par AWS Glue, consultez la section Actions définies par AWS Glue.

Préparation des données et création d’actions

- SendRecipeAction
- GetRecipeAction

Actions associées aux graphes orientés acycliques (DAG)

- CreateDag
- UpdateDag
- GetDag
- DeleteDag

Actions associées aux tâches

- SaveJob
- GetJob
- CreateJob
- DeleteJob
- GetJobs
- UpdateJob

Actions associées à l'exécution des tâches

- StartJobRun
- GetJobRuns
- BatchStopJobRun
- GetJobRun
- QueryJobRuns
- QueryJobs
- QueryJobRunsAggregated

Actions associées aux schémas

- GetSchema

- GetInferredSchema

Actions associées aux bases de données

- GetDatabases

Actions associées aux plans

- GetPlan

Actions associées aux tableaux

- SearchTables
- GetTables
- GetTable

Actions associées aux connexions

- CreateConnection
- DeleteConnection
- UpdateConnection
- GetConnections
- GetConnection

Actions associées au mappage

- GetMapping

Actions associées aux proxy S3

- ListBuckets
- ListObjectsV2
- GetBucketLocation

Actions associées aux configurations de sécurité

- GetSecurityConfigurations

Actions associées aux scripts

- CreateScript (différent de l'API du même nom dansAWS Glue)

Accès à AWS Glue Studio APIs

Pour accéder à AWS Glue Studio, ajoutez `glue:UseGlueStudio` dans la liste des stratégies d'actions des autorisations IAM.

Dans l'exemple ci-dessous, `glue:UseGlueStudio` est inclus dans la politique d'action, mais AWS Glue Studio APIs ils ne sont pas identifiés individuellement. En effet, lorsque vous incluez`glue:UseGlueStudio`, vous êtes automatiquement autorisé à accéder à l'interne APIs sans avoir à spécifier la personne AWS Glue Studio APIs dans les autorisations IAM.

Dans l'exemple, les politiques d'action supplémentaires répertoriées (par exemple,`glue:SearchTables`) ne le sont pas AWS Glue Studio APIs. Elles devront donc être incluses dans les autorisations IAM selon les besoins. Vous pouvez également inclure des actions de proxy Amazon S3 pour spécifier le niveau d'accès Amazon S3 à accorder. L'exemple de politique ci-dessous fournit un accès pour ouvrirAWS Glue Studio, créer une tâche visuelle, et save/run ce, si le rôle IAM sélectionné dispose d'un accès suffisant.

Autorisations relative au bloc-notes et à la prévisualisation des données

Les prévisualisations de données et les blocs-notes vous autorisent à visualiser un échantillon de vos données à n'importe quelle étape de votre tâche (lecture, transformation, écriture), sans avoir à exécuter le travail. Vous spécifiez un rôle Gestion des identités et des accès AWS (IAM) AWS Glue Studio à utiliser lors de l'accès aux données. Les rôles IAM sont prévus pour être assumables et ne disposent pas d'informations d'identification standard à long terme comme un mot de passe ou des clés d'accès associées. Au lieu de cela, lorsque AWS Glue Studio adopte le rôle, IAM lui fournit des informations d'identification de sécurité temporaires.

Pour s'assurer que les prévisualisations de données et les commandes du bloc-notes fonctionnent correctement, utilisez un rôle dont le nom commence par la chaîne `AWSGlueServiceRole`. Si vous choisissez d'utiliser un nom différent pour votre rôle, vous devez alors ajouter l'autorisation `iam:passrole` et configurer une politique pour le rôle dans IAM. Pour de plus amples informations, veuillez consulter Créez une politique IAM pour les rôles non nommés « AWSGlue ServiceRole * ».

> **Warning**
>
> Si un rôle accorde l'autorisation `iam:passrole` pour un bloc-notes et que vous mettez en œuvre le chaînage des rôles, un utilisateur pourrait accéder involontairement à ce bloc-notes. Actuellement, aucun système d'audit n'est mis en place pour vous vous permettre de contrôler les utilisateurs qui ont été autorisés à accéder au bloc-notes.

Si vous souhaitez refuser à une identité IAM la possibilité de créer des sessions d'aperçu des données, consultez l'exemple suivant the section called "Refus de la possibilité de créer des sessions d'aperçu des données à une identité".

Amazon CloudWatch autorisations

Vous pouvez surveiller vos AWS Glue Studio tâches à l'aide Amazon CloudWatch d' near-real-timeindicateurs lisibles qui AWS Glue collectent et traitent les données brutes. Par défaut, AWS Glue les données des métriques sont envoyées CloudWatch automatiquement à. Pour plus d'informations, consultez Qu'est-ce qu'Amazon CloudWatch ? dans le guide de CloudWatch l'utilisateur Amazon, et AWS GlueMetrics dans le guide du AWS Glue développeur.

Pour accéder aux CloudWatch tableaux de bord, l'utilisateur AWS Glue Studio doit disposer de l'un des éléments suivants :

- La politique `AdministratorAccess`
- La politique `CloudWatchFullAccess`
- Une politique personnalisée qui inclut une ou plusieurs de ces autorisations spécifiques :
  - `cloudwatch:GetDashboard` et `cloudwatch:ListDashboards` pour afficher les tableaux de bord
  - `cloudwatch:PutDashboard` pour pouvoir créer ou modifier des tableaux de bord
  - `cloudwatch:DeleteDashboards` pour supprimer des tableaux de bord

Pour plus d'informations sur la modification des autorisations pour un utilisateur IAM à l'aide de politiques, veuillez consulter la rubrique Modification des autorisations pour un utilisateur IAM dans le Guide de l'utilisateur IAM.

Examinez les autorisations IAM nécessaires pour les tâches ETL

Lorsque vous créez une tâche à l'aide de AWS Glue Studio, la tâche dispose des autorisations du rôle IAM que vous spécifiez quand vous la créez. Ce rôle IAM doit être autorisé à extraire des données de votre source de données, à écrire des données sur votre cible et à accéder aux AWS Glue ressources.

Le nom du rôle que vous créez pour la tâche doit démarrer par la chaîne `AWSGlueServiceRole` pour être utilisé correctement par AWS Glue Studio. Par exemple, vous pouvez nommer votre rôle `AWSGlueServiceRole-FlightDataJob`.

Autorisations de source et de cible de données

Une tâche AWS Glue Studio doit avoir accès à l'intégralité des sources, cibles, scripts et répertoires temporaires Amazon S3 que vous utilisez pour celle-ci. Vous pouvez créer une politique pour fournir un accès détaillé à des ressources Amazon S3 spécifiques.

- Les sources de données nécessitent les autorisations `s3:ListBucket` et `s3:GetObject`.
- Les sources de données nécessitent les autorisations `s3:ListBucket`, `s3:PutObject` et `s3:DeleteObject`.

> **Note**
>
> Votre politique IAM doit autoriser `s3:GetObject` les compartiments spécifiques utilisés pour héberger AWS Glue des transformations.
> Les compartiments suivants appartiennent au compte de AWS service et sont lisibles dans le monde entier. Ces compartiments servent de référentiel pour le code source correspondant à un sous-ensemble de transformations accessibles via l'éditeur AWS Glue Studio visuel. Les autorisations sur le compartiment sont configurées pour refuser toute autre action d'API sur le compartiment. Tout le monde peut lire les scripts que nous fournissons pour les transformations, mais personne en dehors de notre équipe de service ne peut y « placer » quoi que ce soit. Lorsque votre AWS Glue tâche s'exécute, ce fichier est extrait sous forme d'importation locale afin qu'il soit téléchargé dans le conteneur local. Après cela, il n'y a plus de communication avec ce compte.

Région : nom du compartiment

- af-south-1 : -762339736633- aws-glue-studio-transforms -1 prod-af-south
- ap-east-1 : -125979764932- aws-glue-studio-transforms -1 prod-ap-east
- ap-northeast-2 : -673535381443- -2 aws-glue-studio-transforms prod-ap-northeast
- ap-northeast-3 : -149976050262- -3 aws-glue-studio-transforms prod-ap-northeast
- ap-south-1 : -584702181950- aws-glue-studio-transforms -1 prod-ap-south
- ap-south-2 : -380279651983- -2 aws-glue-studio-transforms prod-ap-south
- ap-southeast-1 : -737106620487- -1 aws-glue-studio-transforms prod-ap-southeast
- ap-southeast-2 : -234881715811- -2 aws-glue-studio-transforms prod-ap-southeast
- ap-southeast-3 : -151265630221- -3 aws-glue-studio-transforms prod-ap-southeast
- ap-southeast-4 : -052235663858- -4 aws-glue-studio-transforms prod-ap-southeast
- ca-central-1 : -622716468547- aws-glue-studio-transforms -1 prod-ca-central
- ca-west-1 : -915795495192- aws-glue-studio-transforms -1 prod-ca-west
- eu-central-1 : -560373232017- aws-glue-studio-transforms -1 prod-eu-central
- aws-glue-studio-transformseu-central-2 : -907358657121- -2 prod-eu-central
- eu-north-1 : -312557305497- -1 aws-glue-studio-transforms prod-eu-north
- eu-south-1 : -939684186351- -1 aws-glue-studio-transforms prod-eu-south
- eu-south-2 : -239737454084- -2 aws-glue-studio-transforms prod-eu-south
- eu-west-1 : aws-glue-studio-transforms -244479516193- -1 prod-eu-west
- eu-west-2 : aws-glue-studio-transforms -804222392271- -2 prod-eu-west
- eu-west-3 : aws-glue-studio-transforms -371299348807- -3 prod-eu-west
- aws-glue-studio-transformsil-central-1 : -806964611811- -1 prod-il-central
- me-central-1 : -733304270342- aws-glue-studio-transforms -1 prod-me-central
- me-south-1 : -112120182341- -1 aws-glue-studio-transforms prod-me-south
- sa-east-1 : -881619130292- aws-glue-studio-transforms -1 prod-sa-east
- us-east-1 : -510798373988- aws-glue-studio-transforms -1 prod-us-east
- aws-glue-studio-transformsus-east-2 : -251189692203- -2 prod-us-east
- us-west-1 : aws-glue-studio-transforms -593230150239- -1 prod-us-west
- us-west-2 : aws-glue-studio-transforms -818035625594- -2 prod-us-west

- ap-northeast-1 : -200493242866- -1 aws-glue-studio-transforms prod-ap-northeast
- cn-north-1 : aws-glue-studio-transforms -071033555442- -1 prod-cn-north
- cn-northwest-1 : -070947029561- -1 aws-glue-studio-transforms prod-cn-northwest
- us-gov-west-1 : aws-glue-studio-transforms -227493901923- -1-2604 prod-us-gov-west
- eusc-de-east-1 : aws-glue-studio-transforms -780995497573- -1-555 prod-eusc-de-east

Si vous le souhaitez Amazon Redshift comme source de données, vous pouvez attribuer un rôle aux autorisations de cluster. Les tâches exécutées sur un Amazon Redshift cluster émettent des commandes qui accèdent à Amazon S3 pour un stockage temporaire à l'aide d'informations d'identification temporaires. Si votre tâche s'exécute pendant plus d'une heure, ces informations d'identification expirent, ce qui entraîne l'échec de la tâche. Pour éviter ce problème, vous pouvez attribuer un rôle au cluster Amazon Redshift lui-même qui accorde les autorisations nécessaires aux tâches utilisant des informations d'identification temporaires. Pour de plus amples informations, veuillez consulter la rubrique Déplacement de données vers et depuis Amazon Redshift dans le Guide du développeur AWS Glue .

Si la tâche utilise des sources ou des cibles de données autres qu'Amazon S3, vous devez attacher les autorisations nécessaires au rôle IAM utilisé par la tâche pour accéder à ces sources et cibles de données. Pour de plus amples informations, veuillez consulter Configuration de votre environnement pour accéder aux magasins de données dans le Guide du développeur AWS Glue .

Si vous utilisez des connecteurs et des connexions pour votre magasin de données, vous avez besoin d'autorisations supplémentaires, comme décrit dans the section called “Autorisations requises pour l'utilisation de connecteurs”.

Autorisations requises pour supprimer des tâches

Dans AWS Glue Studio, vous pouvez sélectionner plusieurs tâches à supprimer dans la console. Pour effectuer cette action, vous devez disposer de l'autorisation `glue:BatchDeleteJob`. Ceci est différent de la console AWS Glue, qui nécessite une autorisation `glue:DeleteJob` pour supprimer des tâches.

AWS Key Management Service autorisations

Si vous prévoyez d'accéder aux sources et aux cibles Amazon S3 qui utilisent le chiffrement côté serveur avec AWS Key Management Service (AWS KMS), associez une politique au AWS Glue Studio rôle utilisé par la tâche qui permet à la tâche de déchiffrer les données. Le rôle de la tâche

nécessite les autorisations `kms:ReEncrypt`, `kms:GenerateDataKey` et `kms:DescribeKey`. En outre, le poste doit être `kms:Decrypt` autorisé à charger ou à télécharger un objet Amazon S3 chiffré à l'aide d'une clé principale AWS KMS du client (CMK).

L'utilisation entraîne des frais supplémentaires AWS KMS CMKs. Pour plus d'informations, consultez les sections AWS Key Management Service Concepts - Customer Master Keys (CMKs) et AWS Key Management Service Tarification dans le guide du AWS Key Management Service développeur.

Autorisations requises pour l'utilisation de connecteurs

Si vous utilisez un connecteur personnalisé AWS Glue et une connexion pour accéder à un magasin de données, le rôle utilisé pour exécuter la tâche ETL AWS Glue nécessite l'attribution d'autorisations supplémentaires :

- Politique gérée par AWS `AmazonEC2ContainerRegistryReadOnly` pour accéder aux connecteurs achetés auprès de AWS Marketplace.
- Les autorisations `glue:GetJob` et `glue:GetJobs`.
- AWS Secrets Manager autorisations d'accès aux secrets utilisés avec les connexions. Consultez Example: Permission to retrieve secret values (Exemple : Autorisation pour récupérer des valeurs secrètes) pour obtenir des exemples de politiques IAM.

Si votre tâche ETL AWS Glue s'exécute dans un VPC exécutant Amazon VPC, alors le VPC doit être configuré comme décrit dans the section called “Configuration d'un VPC pour votre tâche ETL”.

Définition des autorisations IAM pour AWS Glue Studio

Vous pouvez créer des rôles et affecter des politiques aux utilisateurs et aux rôles de tâches en recourant à l'utilisateur administrateur AWS .

Vous pouvez utiliser la politique AWSGlueConsoleFullAccess AWS gérée pour fournir les autorisations nécessaires à l'utilisation de la AWS Glue Studio console.

Pour créer votre propre politique, suivez les étapes décrites dans Créez une politique IAM pour le service AWS Glue dans le Guide du développeur AWS Glue . Incluez les autorisations IAM décrites précédemment dans Autorisations IAM nécessaires pour l'utilisateur AWS Glue Studio.

Rubriques

- Attacher des politiques à l'utilisateur AWS Glue Studio

- Créez une politique IAM pour les rôles non nommés « AWSGlue ServiceRole * »

Attacher des politiques à l'utilisateur AWS Glue Studio

Tout AWS utilisateur qui se connecte à la AWS Glue Studio console doit être autorisé à accéder à des ressources spécifiques. Vous fournissez ces autorisations par le biais des politiques IAM attribuées à l'utilisateur.

Attachez la politique AWSGlueConsoleFullAccess gérée à l'utilisateur

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, choisissez Politiques.
3. Dans la liste des politiques, cochez la case à côté de AWSGlueConsoleFullAccess. Vous pouvez utiliser le menu Filtre et la zone de recherche pour filtrer la liste de politiques.
4. Sélectionnez Policy Actions (Actions de politique), puis sélectionnez Attach (Attacher).
5. Sélectionnez l'utilisateur auquel attacher la politique. Vous pouvez utiliser le menu Filtre et la zone de recherche pour filtrer la liste des entités du principal. Après avoir choisi l'utilisateur auquel attacher la politique, sélectionnez Attach policy (Attacher la politique).
6. Répétez les étapes précédentes pour attacher des politiques supplémentaires à l'utilisateur, selon vos besoins.

Créez une politique IAM pour les rôles non nommés « AWSGlue ServiceRole * »

Pour configurer une politique IAM pour les rôles utilisés par AWS Glue Studio

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Ajoutez une nouvelle politique IAM. Vous pouvez ajouter à une politique existante ou créer une nouvelle politique en ligne IAM. Pour créer une politique IAM :
    1. Sélectionnez Policies (Politiques), puis Create Policy (Créer une politique). Si un bouton Get Started (Mise en route) est affiché, sélectionnez-le, puis sélectionnez Create Policy (Créer une politique).
    2. En regard de Create Your Own Policy (Créez votre politique), choisissez Select (Sélectionner).

3. Dans le champ Nom de la politique, saisissez une valeur facile à mémoriser pour vous y reporter ultérieurement. Le cas échéant, entrez un texte descriptif dans Description.
4. Dans le champ Policy Document (Document de la politique), saisissez une déclaration de politique au format suivant, puis sélectionnez Create Policy (Créer la politique) :

3. Copiez et collez les blocs suivants dans la politique sous le tableau « Statement », en les *my-interactive-session-role-prefix* remplaçant par le préfixe de tous les rôles courants pour lesquels les autorisations doivent être associées. AWS Glue

```
{
    "Action": [
        "iam:PassRole"
    ],
    "Effect": "Allow",
    "Resource": "arn:aws:iam::*:role/my-interactive-session-role-prefix*",
    "Condition": {
        "StringLike": {
            "iam:PassedToService": [
                "glue.amazonaws.com "
            ]
        }
    }
}
```

Voici l'exemple complet avec les rangs Version et Déclaration inclus dans la politique

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Action": [
        "iam:PassRole"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:iam::*:role/my-interactive-session-role-prefix*",
      "Condition": {
        "StringLike": {
          "iam:PassedToService": [
            "glue.amazonaws.com "
```

```
            ]
          }
        }
      }
    ]
}
```

4. Pour activer la politique pour un utilisateur, choisissez Users (Utilisateurs).
5. Sélectionnez l'utilisateur auquel vous souhaitez attacher la stratégie.

Configuration d'un VPC pour votre tâche ETL

Vous pouvez utiliser Amazon Virtual Private Cloud (Amazon VPC) pour définir un réseau virtuel dans votre propre zone logiquement isolée au sein de ce que l' AWS Cloud on appelle un cloud privé virtuel (VPC). Vous pouvez lancer vos ressources AWS , comme des instances, dans votre VPC. Votre VPC ressemble beaucoup à un réseau traditionnel que vous pourriez exécuter dans votre propre centre de données, et présente l'avantage d'utiliser l'infrastructure évolutive d' AWS. Vous pouvez configurer votre VPC en sélectionnant sa plage d'adresses IP, en créant des sous-réseaux et en configurant des tables de routage, des passerelles réseau et des paramètres de sécurité. Vous pouvez connecter les instances de votre VPC à internet. Vous pouvez connecter votre VPC à votre propre centre de données d'entreprise, pour en faire AWS Cloud une extension de votre centre de données. Pour protéger les ressources dans chaque sous-réseau, vous pouvez utiliser plusieurs couches de sécurité, y compris des groupes de sécurité et des listes de contrôle d'accès réseau. Pour de plus amples informations, veuillez consulter le Guide de l'utilisateur Amazon VPC.

Vous pouvez configurer vos tâches ETL AWS Glue pour qu'elles s'exécutent dans un VPC lors de l'utilisation de connecteurs. Vous devez configurer votre VPC pour les éléments suivants, selon vos besoins :

- Accès au réseau public pour les banques de données non incluses AWS. Tous les magasins de données auxquels la tâche peut accéder doivent être disponibles à partir du sous-réseau VPC.
- Si votre tâche a besoin d'accéder aux ressources du VPC et au réseau Internet public, le VPC doit disposer d'une passerelle de traduction d'adresses réseau (NAT) dans le VPC.

  Pour de plus amples informations, veuillez consulter Configuration de votre environnement pour accéder aux magasins de données dans le Guide du développeur AWS Glue .

Mise en route avec les blocs-notes dans AWS Glue Studio

Lorsque vous démarrez un bloc-notes via AWS Glue Studio, toutes les étapes de configuration sont effectuées pour vous afin que vous puissiez explorer vos données et commencer à développer votre script de tâche après quelques secondes seulement.

Les sections suivantes décrivent comment créer un rôle et accorder les autorisations appropriées pour utiliser les blocs-notes dans AWS Glue Studio pour les tâches ETL.

Pour plus d'informations sur les actions définies par AWS Glue, voir Actions définies par AWS Glue.

Rubriques

- Octroi d'autorisations pour le rôle IAM

Octroi d'autorisations pour le rôle IAM

La configuration de AWS Glue Studio est une condition préalable à l'utilisation de carnets de notes.

Pour utiliser des blocs-notes dans AWS Glue, votre rôle nécessite les éléments suivants :

- Une relation d'approbation avec AWS Glue pour l'action `sts:AssumeRole` et, si vous voulez lui attribuer une étiquette, alors `sts:TagSession`.
- Une politique IAM contenant toutes les autorisations pour les blocs-notes, AWS Glue, et des sessions interactives.
- Une politique IAM pour un rôle de passe, car le rôle doit pouvoir se transmettre du bloc-notes aux séances interactives.

Par exemple, lorsque vous créez un nouveau rôle, vous pouvez ajouter une stratégie AWS gérée standard similaire `AWSGlueConsoleFullAccessRole` au rôle, puis ajouter une nouvelle stratégie pour les opérations du bloc-notes et une autre pour la PassRole stratégie IAM.

Actions nécessaires pour une relation d'approbation avec AWS Glue

Lorsque vous démarrez une séance de bloc-notes, vous devez ajouter `sts:AssumeRole` à la relation d'approbation du rôle transmis au bloc-notes. Si votre séance inclut des identifications, vous devez également transmettre l'action `sts:TagSession`. Sans ces actions, la séance de bloc-notes ne peut pas démarrer.

Par exemple :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "glue.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Politiques contenant des autorisations IAM pour les blocs-notes

L'exemple de politique suivant décrit les autorisations AWS IAM requises pour les ordinateurs portables. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:StartNotebook",
        "glue:TerminateNotebook",
        "glue:GlueNotebookRefreshCredentials",
        "glue:DeregisterDataPreview",
        "glue:GetNotebookInstanceStatus",
        "glue:GlueNotebookAuthorize"
      ],
      "Resource": "*"
    }
  ]
```

```
}
```

Vous pouvez utilisez les politiques IAM suivantes pour autoriser l'accès à des ressources spécifiques :

- AwsGlueSessionUserRestrictedNotebookServiceRole: fournit un accès complet à toutes les AWS Glue ressources, à l'exception des sessions. Permet aux utilisateurs de créer et d'utiliser uniquement les séances de bloc-notes associées à l'utilisateur. Cette politique inclut également d'autres autorisations nécessaires AWS Glue pour gérer les AWS Glue ressources dans d'autres AWS services.
- AwsGlueSessionUserRestrictedNotebookPolicy: fournit des autorisations qui permettent aux utilisateurs de créer et d'utiliser uniquement les sessions de bloc-notes associées à l'utilisateur. Cette politique inclut également des autorisations permettant explicitement aux utilisateurs de passer un rôle de séance AWS Glue restreint.

Politique IAM pour transmettre un rôle

Lorsque vous créez un bloc-notes avec un rôle, ce rôle est ensuite transmis à des séances interactives afin que le même rôle puisse être utilisé aux deux endroits. En tant que tel, l'autorisation `iam:PassRole` doit faire partie de la politique du rôle.

Créez une politique pour votre rôle à l'aide de l'exemple suivant. Remplacez le numéro de compte par le vôtre et le nom du rôle.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iam:PassRole",
      "Resource": "arn:aws:iam::111122223333:role/<role_name>"
    }
  ]
}
```

Configuration des profils AWS Glue d'utilisation

L’un des principaux avantages de l’utilisation d’une plateforme cloud est sa flexibilité. Cependant, cette facilité de création de ressources informatiques risque de faire grimper les coûts du cloud en l’absence de gestion et de barrières de protection. Par conséquent, les administrateurs doivent trouver un équilibre entre éviter les coûts d’infrastructure élevés et permettre aux utilisateurs de travailler sans friction inutile.

Avec les profils AWS Glue d'utilisation, les administrateurs peuvent créer différents profils pour différentes catégories d'utilisateurs au sein du compte, telles que les développeurs, les testeurs et les équipes produit. Chaque profil est un ensemble unique de paramètres qui peuvent être attribués à différents types d’utilisateurs. Par exemple, les développeurs peuvent avoir besoin de plus de personnel et peuvent avoir un nombre maximum de travailleurs plus élevé, tandis que les équipes produit peuvent avoir besoin de moins de personnel et d’une valeur de délai d’attente ou d’inactivité inférieure.

Exemple de comportement des tâches et d’exécutions de tâches

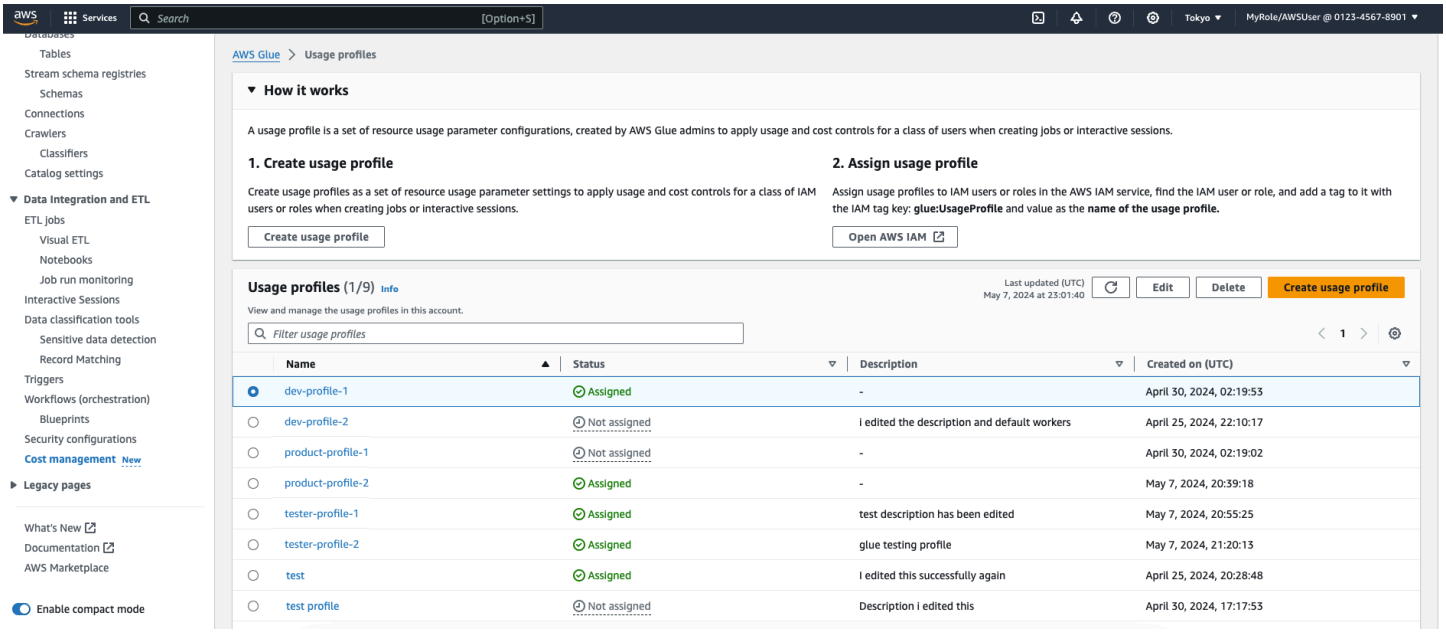
Supposons qu’une tâche soit créée par l’utilisateur A avec le profil A. La tâche est enregistrée avec certaines valeurs de paramètres. L’utilisateur B avec le profil B essaiera d’exécuter la tâche.

Lorsque l’utilisateur A a créé la tâche, s’il n’a pas défini un nombre spécifique de travailleurs, le paramètre par défaut défini dans le profil de l’utilisateur A a été appliqué et enregistré avec les définitions de la tâche.

Lorsque l’utilisateur B exécute la tâche, elle s’exécute avec les valeurs enregistrées pour celle-ci. Si le profil de l’utilisateur B est plus restrictif et n’est pas autorisé à s’exécuter avec autant de travailleurs, l’exécution de la tâche échouera.

Profil d’utilisation en tant que ressource

Un profil AWS Glue d'utilisation est une ressource identifiée par un Amazon Resource Name (ARN). Tous les contrôles IAM (Gestion des identités et des accès) par défaut s’appliquent, y compris les autorisations basées sur les actions et les ressources. Les administrateurs doivent mettre à jour la politique IAM des utilisateurs qui créent des AWS Glue ressources, en leur accordant l'accès à l'utilisation des profils.



Rubriques

- Création et gestion de profils d'utilisation
- Profils d'utilisation et tâches

Création et gestion de profils d'utilisation

Création d'un profil AWS Glue d'utilisation

Les administrateurs doivent créer des profils d'utilisation, puis les attribuer aux différents utilisateurs. Lors de la création d'un profil d'utilisation, vous spécifiez des valeurs par défaut, ainsi qu'une plage de valeurs autorisées pour divers paramètres de tâche et de session. Vous devez configurer au moins un paramètre pour les tâches ou les sessions interactives. Vous pouvez personnaliser la valeur par défaut à utiliser lorsqu'aucune valeur de paramètre n'est fournie pour la tâche, and/or définir une limite de plage ou un ensemble de valeurs autorisées pour validation si un utilisateur fournit une valeur de paramètre lors de l'utilisation de ce profil.

Les valeurs par défaut sont une bonne pratique définie par l'administrateur pour aider les auteurs de tâches. Lorsqu'un utilisateur crée une tâche sans définir de valeur de délai d'expiration, le délai d'expiration par défaut du profil d'utilisation s'applique. Si l'auteur n'a pas de profil, les valeurs par défaut du AWS Glue service s'appliqueront et seront enregistrées dans la définition de la tâche. Lors de l' AWS Glue exécution, applique les limites définies dans le profil (min, max, travailleurs autorisés).

Une fois qu’un paramètre est configuré, tous les autres paramètres sont facultatifs. Les paramètres qui peuvent être personnalisés pour les tâches ou les sessions interactives sont les suivants :

- Nombre d’exécuteurs : limitez le nombre d’exécuteurs pour éviter une utilisation excessive des ressources de calcul. Vous pouvez définir une valeur par défaut, une valeur minimale et une valeur maximale. La valeur minimale est de 1.
- Type d’exécuteur : limitez les types d’exécuteurs pertinents pour vos charges de travail. Vous pouvez définir un type par défaut et autoriser les types d’exécuteurs pour un profil utilisateur.
- Délai d’expiration : définissez la durée maximale pendant laquelle une tâche ou une session interactive peut être exécutée et consommer des ressources avant d’être terminée. Définissez des valeurs de délai d’expiration pour éviter les tâches de longue durée.

  Vous pouvez définir une valeur par défaut, minimale et maximale en minutes. La durée minimale est de 1 (1 minute). Bien que le délai d'expiration AWS Glue par défaut soit de 2880 minutes, vous pouvez définir n'importe quelle valeur par défaut dans le profil d'utilisation.

  Il est recommandé de définir une valeur pour « par défaut ». Cette valeur sera utilisée pour la création de la tâche ou de la session si aucune valeur n’a été définie par l’utilisateur.
- Délai d’inactivité : définissez le nombre de minutes pendant lesquelles une session interactive est inactive avant d’arriver à expiration après l’exécution d’une cellule. Définissez le délai d’inactivité pour que les sessions interactives se terminent une fois le travail terminé. La plage de délai d’inactivité doit être comprise dans la limite du délai d’expiration.

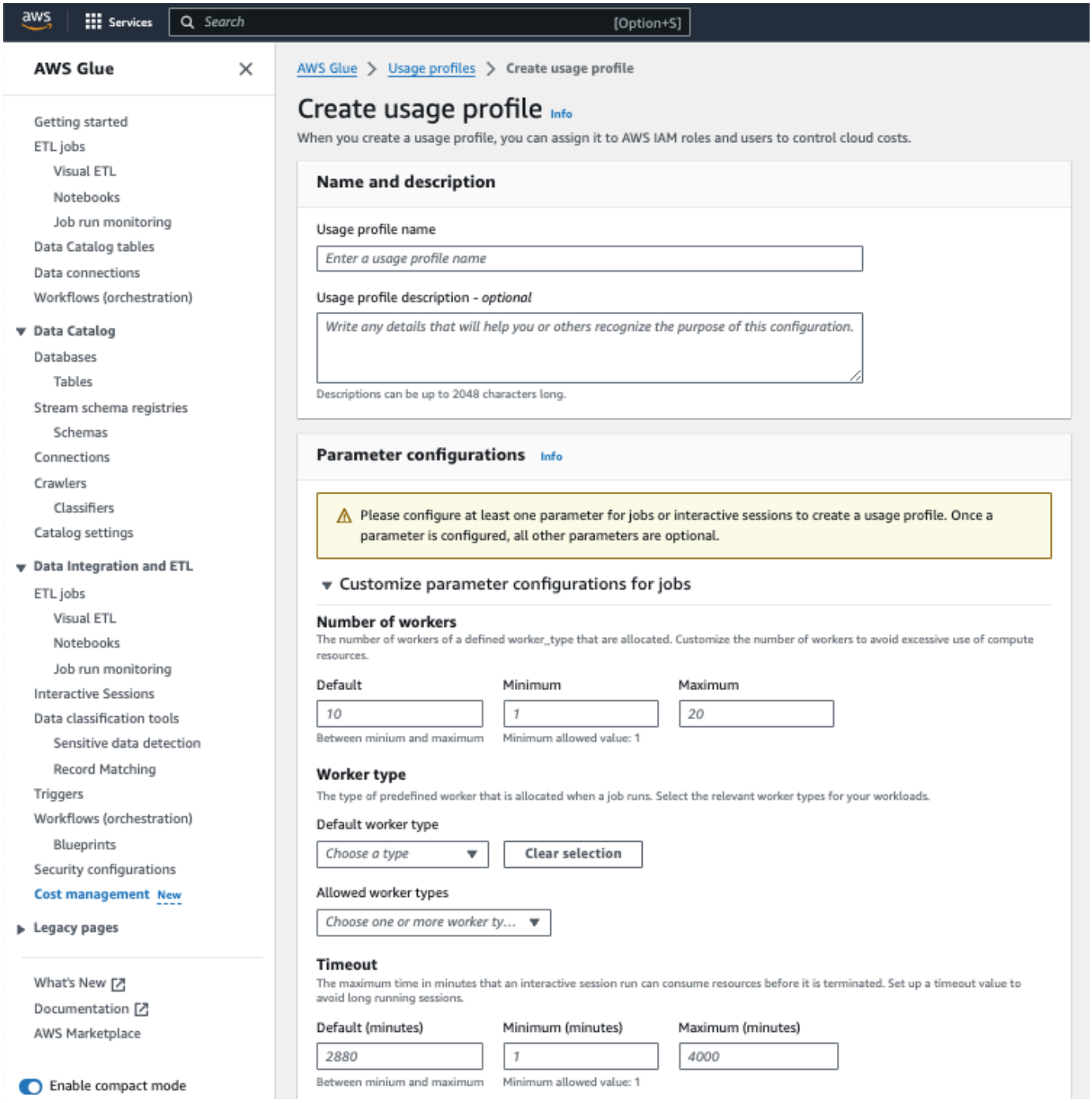
  Vous pouvez définir une valeur par défaut, minimale et maximale en minutes. La durée minimale est de 1 (1 minute). Bien que le délai d'expiration AWS Glue par défaut soit de 2880 minutes, vous pouvez définir n'importe quelle valeur par défaut dans le profil d'utilisation.

  Il est recommandé de définir une valeur pour « par défaut ». Cette valeur sera utilisée pour la création de session si aucune valeur n’a été définie par l’utilisateur.

Pour créer un profil AWS Glue d'utilisation en tant qu'administrateur (console)

1. Dans le menu de navigation de gauche, choisissez Gestion des coûts.
2. Choisissez Créer un profil d’utilisation.
3. Saisissez le nom du profil d’utilisation pour le profil d’utilisation.
4. Saisissez une description facultative qui aidera les autres utilisateurs à reconnaître l’objectif du profil d’utilisation.

5. Définissez au moins un paramètre dans le profil. Tout champ du formulaire est un paramètre. Par exemple, le délai d'inactivité minimal de la session.
6. Définissez les balises facultatives qui s'appliquent au profil d'utilisation.
7. Choisissez Enregistrer.



Pour créer un profil d'utilisation (AWS CLI)

1. Entrez la commande suivante.

```
aws glue create-usage-profile --name profile-name --configuration file://
config.json --tags list-of-tags
```

où le fichier config.json peut définir des valeurs de paramètres pour les sessions interactives (`SessionConfiguration`) et les tâches (`JobConfiguration`) :

```
//config.json (There is a separate blob for session/job configuration
{
    "SessionConfiguration": {
        "timeout": {
            "DefaultValue": "2880",
            "MinValue": "100",
            "MaxValue": "4000"
        },
        "idleTimeout": {
            "DefaultValue": "30",
            "MinValue": "10",
            "MaxValue": "4000"
        },
        "workerType": {
            "DefaultValue": "G.2X",
            "AllowedValues": [
                "G.1X",
                "G.2X",
                "G.4X",
                "G.8X",
                "G.12X",
                "G.16X",
                "R.1X",
                "R.2X",
                "R.4X",
                "R.8X"
            ]
        },
        "numberOfWorkers": {
            "DefaultValue": "10",
            "MinValue": "1",
            "MaxValue": "10"
```

```
        }
    },
    "JobConfiguration": {
        "timeout": {
            "DefaultValue": "2880",
            "MinValue": "100",
            "MaxValue": "4000"
        },
        "workerType": {
            "DefaultValue": "G.2X",
            "AllowedValues": [
                "G.1X",
                "G.2X",
                "G.4X",
                "G.8X",
                "G.12X",
                "G.16X",
                "R.1X",
                "R.2X",
                "R.4X",
                "R.8X"
            ]
        },
        "numberOfWorkers": {
            "DefaultValue": "10",
            "MinValue": "1",
            "MaxValue": "10"
        }
    }
}
```

2. Saisissez la commande suivante pour afficher le profil d'utilisation créé :

```
aws glue get-usage-profile --name profile-name
```

La réponse :

```
{
    "ProfileName": "foo",
    "Configuration": {
        "SessionConfiguration": {
            "numberOfWorkers": {
                "DefaultValue": "10",
```

```
                "MinValue": "1",
                "MaxValue": "10"
            },
            "workerType": {
                "DefaultValue": "G.2X",
                "AllowedValues": [
                    "G.1X",
                    "G.2X",
                    "G.4X",
                    "G.8X",
                    "G.12X",
                    "G.16X",
                    "R.1X",
                    "R.2X",
                    "R.4X",
                    "R.8X"
                ]
            },
            "timeout": {
                "DefaultValue": "2880",
                "MinValue": "100",
                "MaxValue": "4000"
            },
            "idleTimeout": {
                "DefaultValue": "30",
                "MinValue": "10",
                "MaxValue": "4000"
            }
        },
        "JobConfiguration": {
            "numberOfWorkers": {
                "DefaultValue": "10",
                "MinValue": "1",
                "MaxValue": "10"
            },
            "workerType": {
                "DefaultValue": "G.2X",
                "AllowedValues": [
                    "G.1X",
                    "G.2X",
                    "G.4X",
                    "G.8X",
                    "G.12X",
                    "G.16X",
```

```
                    "R.1X",
                    "R.2X",
                    "R.4X",
                    "R.8X"
                ]
            },
            "timeout": {
                "DefaultValue": "2880",
                "MinValue": "100",
                "MaxValue": "4000"
            }
        }
    },
    "CreatedOn": "2024-01-19T23:15:24.542000+00:00"
}
```

Commandes CLI supplémentaires utilisées pour gérer les profils d'utilisation :

- colle à cire list-usage-profiles
- aws glue update-usage-profile --name --configuration *profile-name file://config.json*
- aws glue delete-usage-profile --name *profile-name*

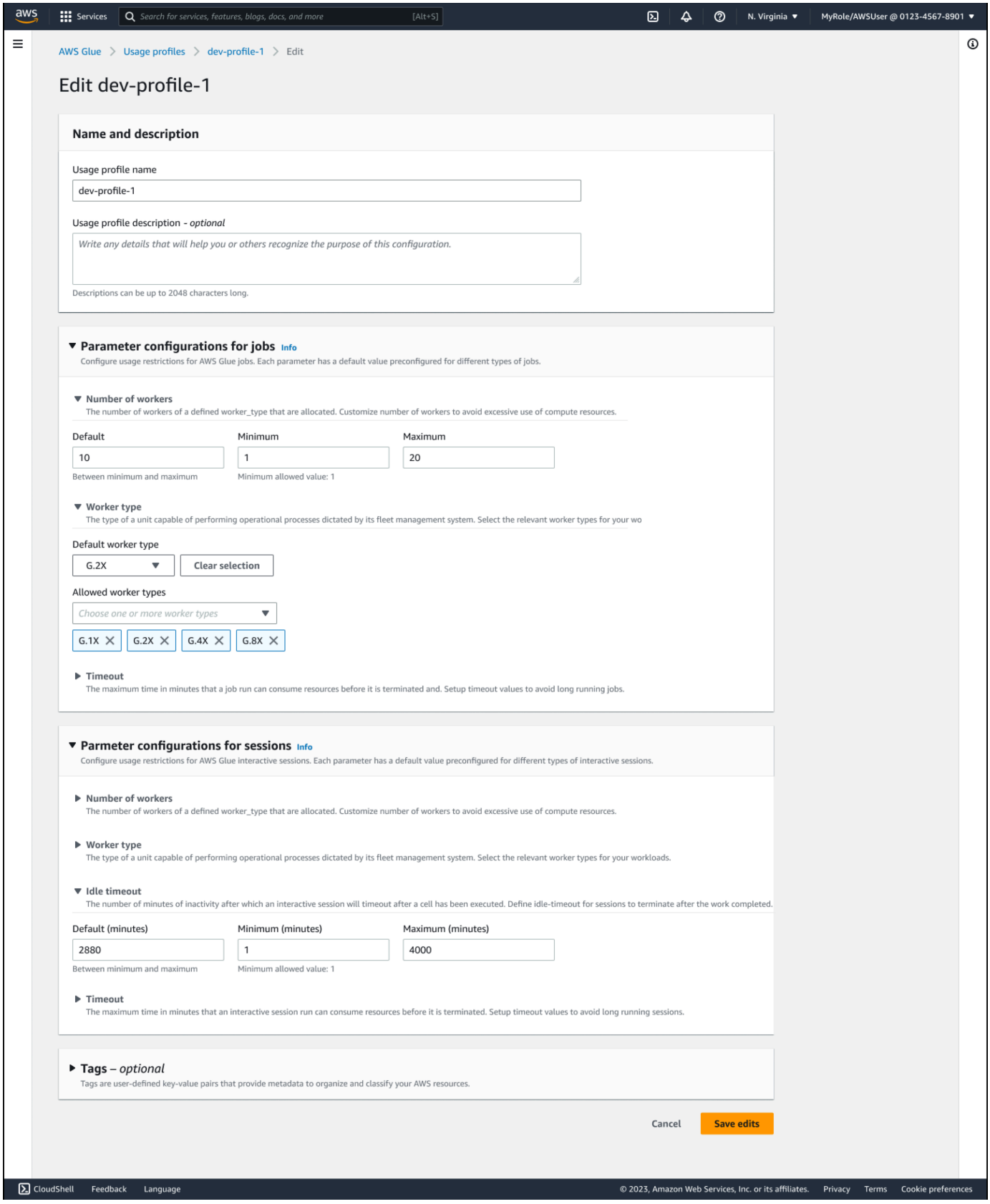
Modification d'un profil d'utilisation

Les administrateurs peuvent modifier les profils d'utilisation qu'ils ont créés afin de modifier les valeurs des paramètres de profil pour les tâches et les sessions interactives.

Pour modifier un profil d'utilisation

Pour modifier un profil AWS Glue d'utilisation en tant qu'administrateur (console)

1. Dans le menu de navigation de gauche, choisissez Gestion des coûts.
2. Choisissez un profil d'utilisation que vous êtes autorisé à modifier, puis cliquez sur Modifier.
3. Apportez les modifications nécessaires au profil. Par défaut, les paramètres qui comportent déjà des valeurs sont développés.
4. Sélectionnez Enregistrer les modifications.


Services
Search for services, features, blogs, docs, and more [Alt+S]
N. Virginia
MyRole/AWSUser @ 0123-4567-8901
AWS Glue > Usage profiles > dev-profile-1 > Edit
Edit dev-profile-1
Name and description
Usage profile name
dev-profile-1
Usage profile description - optional
Write any details that will help you or others recognize the purpose of this configuration.
Descriptions can be up to 2048 characters long.
Parameter configurations for jobs Info
Configure usage restrictions for AWS Glue jobs. Each parameter has a default value preconfigured for different types of jobs.
Number of workers
The number of workers of a defined worker_type that are allocated. Customize number of workers to avoid excessive use of compute resources.
Default
10
Between minimum and maximum
Minimum
1
Minimum allowed value: 1
Maximum
20
Worker type
The type of a unit capable of performing operational processes dictated by its fleet management system. Select the relevant worker types for your wo
Default worker type
G.2X
Clear selection
Allowed worker types
Choose one or more worker types
G.1X
G.2X
G.4X
G.8X
Timeout
The maximum time in minutes that a job run can consume resources before it is terminated and. Setup timeout values to avoid long running jobs.
Parmeter configurations for sessions Info
Configure usage restrictions for AWS Glue interactive sessions. Each parameter has a default value preconfigured for different types of interactive sessions.
Number of workers
The number of workers of a defined worker_type that are allocated. Customize number of workers to avoid excessive use of compute resources.
Worker type
The type of a unit capable of performing operational processes dictated by its fleet management system. Select the relevant worker types for your workloads.
Idle timeout
The number of minutes of inactivity after which an interactive session will timeout after a cell has been executed. Define idle-timeout for sessions to terminate after the work completed.
Default (minutes)
2880
Between minimum and maximum
Minimum (minutes)
1
Minimum allowed value: 1
Maximum (minutes)
4000
Timeout
The maximum time in minutes that an interactive session run can consume resources before it is terminated. Setup timeout values to avoid long running sessions.
Tags – optional
Tags are user-defined key-value pairs that provide metadata to organize and classify your AWS resources.
Cancel
Save edits
CloudShell
Feedback
Language
© 2023, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

Pour modifier un profil d'utilisation (AWS CLI)

- Entrez la commande suivante. La même syntaxe de fichier `--configuration` est utilisée comme indiqué ci-dessus dans la commande de création.

  ```
  aws glue update-usage-profile --name profile-name --configuration file://
  config.json
  ```

  où le fichier config.json définit des valeurs de paramètres pour les sessions interactives (`SessionConfiguration`) et les tâches (`JobConfiguration`) :

Affectation d'un profil d'utilisation

La colonne État d'utilisation de la page Profils d'utilisation indique si un profil d'utilisation est attribué aux utilisateurs. Le survol de l'état indique les entités IAM assignées.

L'administrateur peut attribuer un profil AWS Glue d'utilisation aux users/roles personnes qui créent AWS Glue les ressources. L'attribution d'un profil est une combinaison de deux actions :

- Mettre à jour le user/role tag IAM avec la `glue:UsageProfile` clé, puis
- Mise à jour de la politique IAM de l'utilisateur/du rôle.

Pour les utilisateurs qui utilisent AWS Glue Studio pour créer jobs/interactive des sessions, l'administrateur attribue les rôles suivants :

- Pour les restrictions sur les tâches, l'administrateur balise le rôle de console connecté.
- Pour les restrictions sur les sessions interactives, l'administrateur balise le rôle fourni par l'utilisateur lors de la création du bloc-notes.

Voici un exemple de politique que l'administrateur doit mettre à jour sur l'IAM users/roles qui crée des AWS Glue ressources :

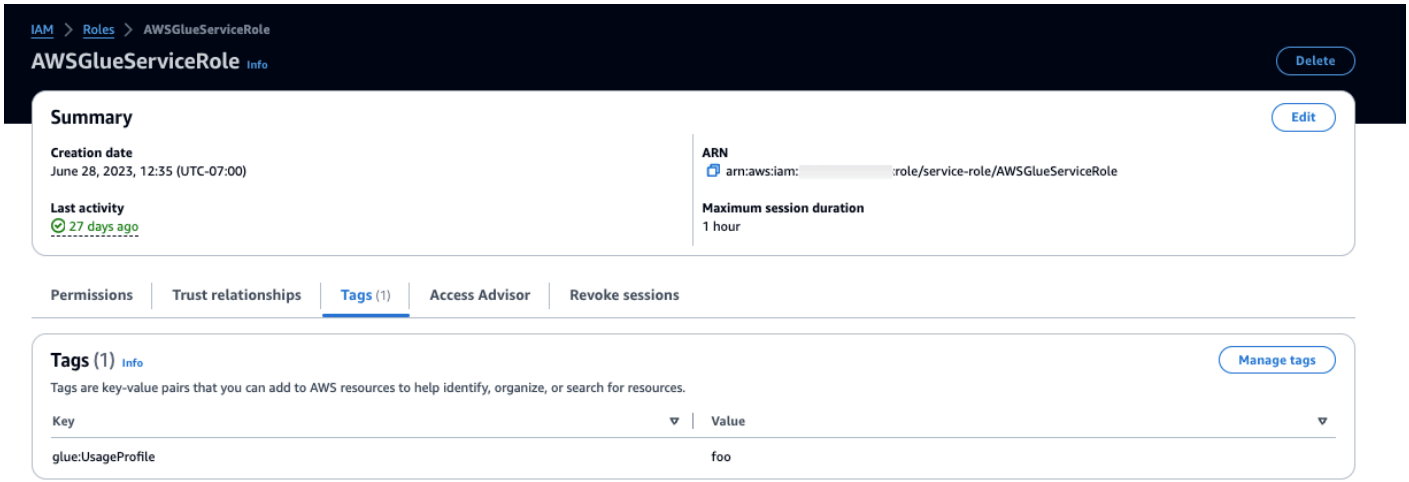
```
{
    "Effect": "Allow",
    "Action": [
        "glue:GetUsageProfile"
    ],
    "Resource": [
```

```
        "arn:aws:glue:us-east-1:123456789012:usageProfile/foo"
    ]
}
```

AWS Glue valide les demandes de travail, d'exécution de tâches et de session en fonction des valeurs spécifiées dans le profil AWS Glue d'utilisation et déclenche une exception si la demande est refusée. Pour le mode synchrone APIs, une erreur sera renvoyée à l'utilisateur. Pour les chemins asynchrones, une exécution de tâche échouée est créée avec le message d'erreur indiquant que le paramètre d'entrée se situe en dehors de la plage autorisée pour le profil attribué à l'utilisateur/au rôle.

Pour attribuer un profil d'utilisation à un utilisateur/rôle :

1. Connectez-vous à la console IAM (Gestion des identités et des accès).
2. Dans le panneau de navigation de gauche, sélectionnez Utilisateurs ou Rôles.
3. Choisissez un utilisateur ou un rôle.
4. Sélectionnez l'onglet Tags (Identifications).
5. Sélectionnez Ajouter une nouvelle balise.
6. Ajoutez une balise avec la clé `glue:UsageProfile` et la valeur du nom de votre profil d'utilisation.
7. Choisissez Enregistrer les modifications



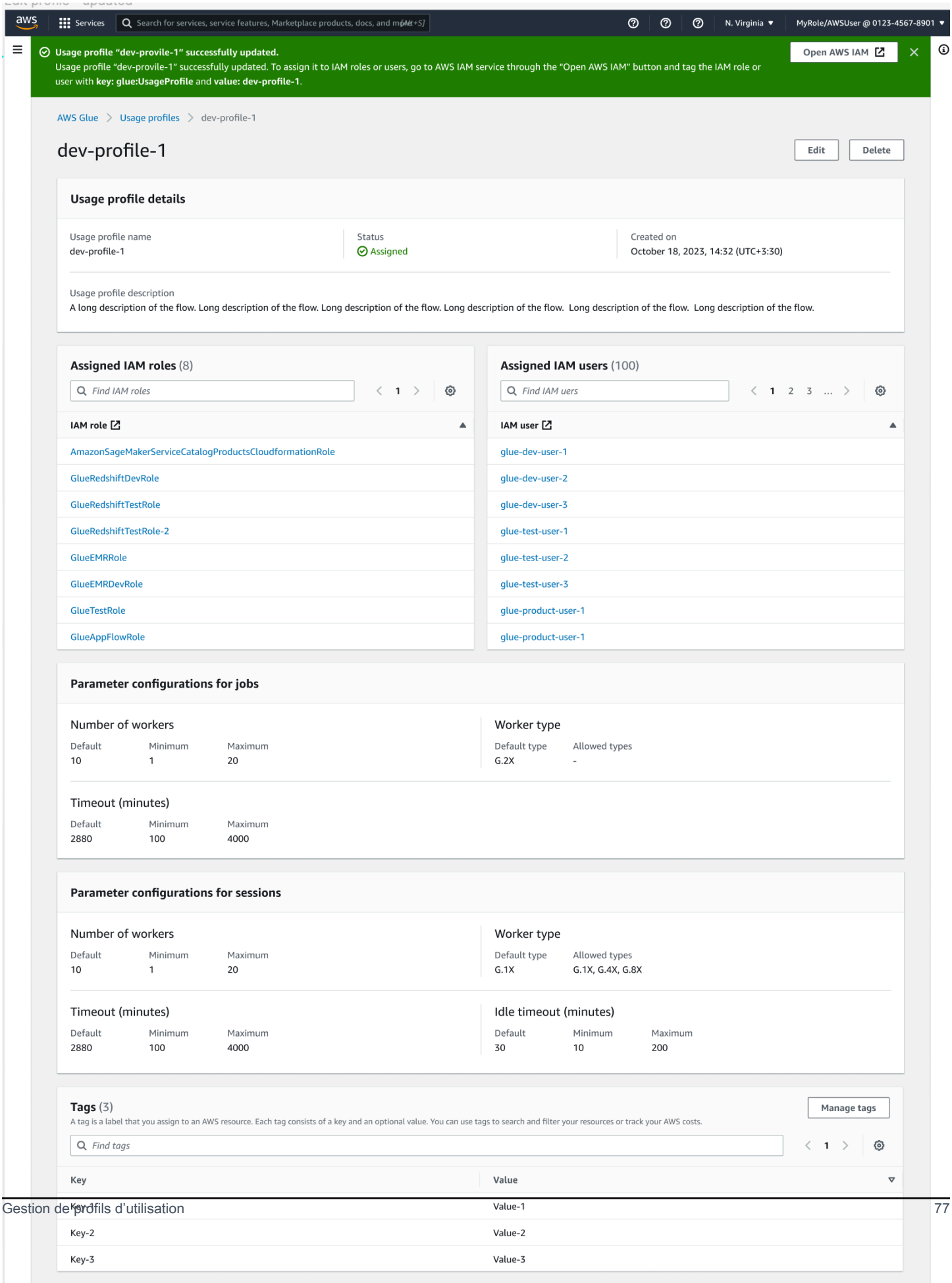
Affichage du profil d’utilisation qui vous a été attribué

Les utilisateurs peuvent consulter les profils d'utilisation qui leur sont assignés et les utiliser lorsqu'ils effectuent des appels d'API pour créer des ressources de AWS Glue tâche et de session, ou lorsqu'ils démarrent une tâche.

Les autorisations de profil sont fournies dans les politiques IAM. Tant que la politique de l’appelant dispose de l’autorisation `glue:UsageProfile`, un utilisateur peut voir le profil. Sinon, vous recevrez un message d’erreur « accès refusé ».

Pour consulter un profil d’utilisation attribué :

1. Dans le menu de navigation de gauche, choisissez Gestion des coûts.
2. Choisissez un profil d’utilisation que vous êtes autorisé à consulter.


Services
Search for services, service features, Marketplace products, docs, and more [Alt+S]
N. Virginia
MyRole/AWSUser @ 0123-4567-8901
Usage profile "dev-provile-1" successfully updated.
Usage profile "dev-provile-1" successfully updated. To assign it to IAM roles or users, go to AWS IAM service through the "Open AWS IAM" button and tag the IAM role or user with key: glue:UsageProfile and value: dev-profile-1.
Open AWS IAM
AWS Glue > Usage profiles > dev-profile-1
dev-profile-1
Edit
Delete
Usage profile details
Usage profile name
dev-profile-1
Status
Assigned
Created on
October 18, 2023, 14:32 (UTC+3:30)
Usage profile description
A long description of the flow. Long description of the flow. Long description of the flow. Long description of the flow. Long description of the flow. Long description of the flow.
Assigned IAM roles (8)
Find IAM roles
IAM role
AmazonSageMakerServiceCatalogProductsCloudformationRole
GlueRedshiftDevRole
GlueRedshiftTestRole
GlueRedshiftTestRole-2
GlueEMRRole
GlueEMRDevRole
GlueTestRole
GlueAppFlowRole
Assigned IAM users (100)
Find IAM uers
IAM user
glue-dev-user-1
glue-dev-user-2
glue-dev-user-3
glue-test-user-1
glue-test-user-2
glue-test-user-3
glue-product-user-1
glue-product-user-1
Parameter configurations for jobs
Number of workers
Default 10
Minimum 1
Maximum 20
Worker type
Default type G.2X
Allowed types -
Timeout (minutes)
Default 2880
Minimum 100
Maximum 4000
Parameter configurations for sessions
Number of workers
Default 10
Minimum 1
Maximum 20
Worker type
Default type G.1X
Allowed types G.1X, G.4X, G.8X
Timeout (minutes)
Default 2880
Minimum 100
Maximum 4000
Idle timeout (minutes)
Default 30
Minimum 10
Maximum 200
Tags (3)
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.
Manage tags
Find tags
Key
Value
Key-1
Value-1
Key-2
Value-2
Key-3
Value-3

Profils d'utilisation et tâches

Création de tâches avec des profils d'utilisation

Lors de la création de tâches, les limites et les valeurs par défaut définies dans votre profil d'utilisation s'appliquent. Votre profil sera attribué à la tâche une fois enregistré.

Exécution de tâches avec des profils d'utilisation

Lorsque vous lancez une tâche, AWS Glue applique les limites définies dans le profil de votre appelant. S'il n'y a pas d'appelant direct, Glue appliquera alors les limites du profil attribué à la tâche par son auteur.

> **Note**
>
> Lorsqu'une tâche est exécutée selon un calendrier (par des AWS Glue flux de travail ou des AWS Glue déclencheurs), le profil attribué à la tâche que l'auteur postule.
> Lorsqu'une tâche est exécutée par un service externe (Step Functions, MWAA) ou une API `StartJobRun`, la limite de profil de l'appelant est appliquée.

Pour les AWS Glue flux de travail ou les AWS Glue déclencheurs : les tâches préexistantes doivent être mises à jour pour enregistrer le nouveau nom du profil afin que les limites du profil (nombre minimum, maximum et travailleurs autorisés) soient appliquées lors de l'exécution pour les exécutions planifiées.

Affichage d'un profil d'utilisation attribué à des tâches

Pour consulter le profil attribué à vos tâches (qui sera utilisé lors de l'exécution avec des AWS Glue flux de travail ou des AWS Glue déclencheurs planifiés), vous pouvez consulter l'onglet Détails de la tâche. Vous pouvez également consulter le profil utilisé lors des exécutions précédentes dans l'onglet Détails des exécutions de tâches.

Mise à jour ou suppression d'un profil d'utilisation associé à une tâche

Le profil attribué à une tâche est modifié lors de la mise à jour. Si aucun profil d'utilisation n'est attribué à l'auteur, tout profil précédemment associé à la tâche en sera supprimé.

Mise en route avec l'AWS Glue Data Catalog

AWS Glue Data Catalog Il s'agit de votre magasin de métadonnées techniques permanent. Il s'agit d'un service géré que vous pouvez utiliser pour stocker, annoter et partager des métadonnées dans le AWS cloud. Pour de plus amples informations, veuillez consulter AWS Glue Data Catalog.

La console AWS Glue et certaines interfaces utilisateur ont été récemment mises à jour.

Présentation de

Vous pouvez utiliser ce didacticiel pour créer votre premier Catalogue de données AWS Glue, qui utilise un compartiment Amazon S3 comme source de données.

Dans le cadre de ce didacticiel, vous effectuerez les tâches suivantes à l'aide de la console AWS Glue :

1. Créer une base de données
2. Création d'une table
3. Utilisation d'un compartiment Amazon S3 comme source de données

Une fois ces étapes terminées, vous aurez utilisé avec succès un compartiment Amazon S3 comme source de données pour remplir le Catalogue de données AWS Glue.

Étape 1 : Créer une base de données

Pour commencer, connectez-vous à la AWS Glueconsole AWS Management Console et ouvrez-la.

Pour créer une base de données à l'aide de la console AWS Glue :

1. Dans la console AWS Glue, choisissez Databases (Bases de données) dans le menu de gauche Data catalog (Catalogue de données).
2. Choisissez Ajouter une base de données.
3. Sur la page Créer une base de données, saisissez un nom pour la base de données. Dans la section Emplacement – facultatif, définissez l'emplacement de l'URI à utiliser par les clients du catalogue de données. Si vous ne connaissez pas cette information, vous pouvez poursuivre la création de la base de données.
4. (Facultatif) Saisissez une description pour la base de données.

5. Choisissez Créer une base de données.

Félicitations, vous venez de configurer votre première base de données à l'aide de la console AWS Glue. Votre nouvelle base de données apparaîtra dans la liste des bases de données disponibles. Vous pouvez modifier la base de données en choisissant le nom de la base de données dans le tableau de bord des Bases de données.

Étapes suivantes

Autres méthodes pour créer une base de données :

Vous venez de créer une base de données à l'aide de la console AWS Glue, mais il existe d'autres moyens de créer une base de données :

- Vous pouvez utiliser des crawlers pour créer automatiquement une base de données et des tableaux pour vous. Pour configurer une base de données à l'aide d'crawlers, reportez-vous à Travailler avec des crawlers dans la console AWS Glue.
- Vous pouvez utiliser des CloudFormation modèles. Voir Création de AWS Glue ressources à l'aide AWS Glue Data Catalog de modèles.
- Vous pouvez également créer une base de données à l'aide du module Opérations d'API de base de données AWS Glue.

  Pour créer une base de données à l'aide de l'opération `create`, structurez la demande en incluant les paramètres `DatabaseInput` (obligatoires).

  Par exemple :

  Voici des exemples illustrant comment utiliser l'interface de ligne de commande, Boto3 ou DDL pour définir un tableau basé sur le même fichier flights_data.csv à partir du compartiment S3 que vous avez utilisé dans le didacticiel.

  CLI

  ```
  aws glue create-database --database-input "{\"Name\":\"clidb\"}"
  ```

  Boto3

  ```
  glueClient = boto3.client('glue')
  ```

```
response = glueClient.create_database(
    DatabaseInput={
        'Name': 'boto3db'
    }
)
```

Pour en savoir plus sur les types de données, la structure et les opérations de l'API de base de données, consultez API d'une base de données.

Étapes suivantes

Dans la section suivante, vous allez créer un tableau et l'ajouter à votre base de données.

Vous pouvez également consulter les paramètres et les autorisations de votre catalogue de données. Consultez Travailler avec des paramètres de catalogue de base de données dans la console AWS Glue.

Étape 2. Création d'une table

Au cours de cette étape, vous créez un tableau en utilisant la console AWS Glue.

1. Dans la console AWS Glue, choisissez Tables (tableaux) dans le menu de gauche.
2. Choisissez Add table (Ajouter une table).
3. Configurez les propriétés de votre table en saisissant un nom pour votre table dans Table details (Détails de la table).
4. Dans la section Databases (Bases de données), choisissez la base de données que vous avez créée à l'étape 1 dans le menu déroulant.
5. Dans la section Add a data store (Ajouter un magasin de données), S3sera sélectionné par défaut comme type de source.
6. Pour Data is located in (Les données se trouvent dans), choisissez Specified path in another account (Chemin d'accès spécifié dans un autre compte).
7. Copiez et collez le chemin pour le champ de saisie Include path (Chemin à inclure) :

   ```
   s3://crawler-public-us-west-2/flight/2016/csv/
   ```
8. Dans la section Data format (Format de données), pour Classification (Catégorie), choisissez CSV et pour Delimiter (Délimiteur), choisissez comma (,) (Virgule (,)). Choisissez Suivant.

9. Il vous est demandé de définir un schéma. Un schéma définit la structure et le format d'un enregistrement de données. Choisissez Ajouter une colonne. Pour en savoir plus, consultez Registre de schémas .
10. Spécifiez les propriétés de la colonne :
    a. Entrez un nom de colonne.
    b. Pour Type de colonne, « chaîne » est déjà sélectionnée par défaut.
    c. Pour Numéro de colonne, « 1 » est déjà sélectionné par défaut.
    d. Choisissez Ajouter.
11. Il vous est demandé d'ajouter des index de partition. Ce nom est facultatif. Pour sauter cette étape, choisissez Next (Suivant).
12. Une synthèse des propriétés du tableau s'affiche. Si tout se présente comme prévu, choisissez Créer. Sinon, choisissez Back (Retour) et apportez les modifications nécessaires.

Félicitations, vous avez créé manuellement un tableau et l'avez associé à une base de données. Le tableau nouvellement créé apparaît dans le tableau de bord des tableaux. Dans le tableau de bord, vous pouvez gérer et modifier tous vos tableaux.

Pour en savoir plus, consultez la rubrique Travailler avec des tableaux dans la console AWS Glue.

Étapes suivantes

Étapes suivantes

Maintenant que le catalogue de données est renseigné, vous pouvez commencer à autoriser des tâches dans AWS Glue. Consultez Building visual ETL jobs with AWS Glue Studio.

Outre l'utilisation de la console, il existe d'autres moyens de définir des tableaux dans le catalogue de données, notamment :

- Création et exécution d'un crawler
- Ajouter des classificateurs à un crawler dans AWS Glue
- Utilisation de l'API du tableau AWS Glue
- Utilisation du modèle AWS Glue Data Catalog
- Migrer un métastore Apache Hive.
- Utilisation du AWS CLI Boto3 ou du langage de définition de données (DDL)

Voici des exemples illustrant comment utiliser l'interface de ligne de commande, Boto3 ou DDL pour définir un tableau basé sur le même fichier flights_data.csv à partir du compartiment S3 que vous avez utilisé dans le didacticiel.

Consultez la documentation pour savoir comment structurer une AWS CLI commande. L'exemple de CLI contient la syntaxe JSON pour la valeur « aws glue create-table --table-input ».

CLI

```
{
        "Name": "flights_data_cli",
        "StorageDescriptor": {
            "Columns": [
                {
                    "Name": "year",
                    "Type": "bigint"
                },
                {
                    "Name": "quarter",
                    "Type": "bigint"
                }
            ],
            "Location": "s3://crawler-public-us-west-2/flight/2016/csv",
            "InputFormat": "org.apache.hadoop.mapred.TextInputFormat",
            "OutputFormat":
 "org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat",
            "Compressed": false,
            "NumberOfBuckets": -1,
            "SerdeInfo": {
                "SerializationLibrary":
 "org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe",
                "Parameters": {
                    "field.delim": ",",
                    "serialization.format": ","
                }
            }
        },
        "PartitionKeys": [
            {
                "Name": "mon",
                "Type": "string"
```

```
            }
        ],
        "TableType": "EXTERNAL_TABLE",
        "Parameters": {
            "EXTERNAL": "TRUE",
            "classification": "csv",
            "columnsOrdered": "true",
            "compressionType": "none",
            "delimiter": ",",
            "skip.header.line.count": "1",
            "typeOfData": "file"
        }
    }
```

Boto3

```
import boto3

glue_client = boto3.client("glue")

response = glue_client.create_table(
    DatabaseName='sampledb',
    TableInput={
        'Name': 'flights_data_manual',
    'StorageDescriptor': {
      'Columns': [{
        'Name': 'year',
        'Type': 'bigint'
      }, {
        'Name': 'quarter',
        'Type': 'bigint'
      }],
      'Location': 's3://crawler-public-us-west-2/flight/2016/csv',
      'InputFormat': 'org.apache.hadoop.mapred.TextInputFormat',
      'OutputFormat':
 'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat',
      'Compressed': False,
      'NumberOfBuckets': -1,
      'SerdeInfo': {
        'SerializationLibrary':
 'org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe',
        'Parameters': {
```

```
          'field.delim': ',',
          'serialization.format': ','
        }
      },
    },
    'PartitionKeys': [{
      'Name': 'mon',
      'Type': 'string'
    }],
    'TableType': 'EXTERNAL_TABLE',
    'Parameters': {
      'EXTERNAL': 'TRUE',
      'classification': 'csv',
      'columnsOrdered': 'true',
      'compressionType': 'none',
      'delimiter': ',',
      'skip.header.line.count': '1',
      'typeOfData': 'file'
    }
    }
)
```

DDL

```
CREATE EXTERNAL TABLE `sampledb`.`flights_data` (
  `year` bigint,
  `quarter` bigint)
PARTITIONED BY (
  `mon` string)
ROW FORMAT DELIMITED
  FIELDS TERMINATED BY ','
STORED AS INPUTFORMAT
  'org.apache.hadoop.mapred.TextInputFormat'
OUTPUTFORMAT
  'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION
  's3://crawler-public-us-west-2/flight/2016/csv/'
TBLPROPERTIES (
  'classification'='csv',
  'columnsOrdered'='true',
  'compressionType'='none',
  'delimiter'=',',
```

```
  'skip.header.line.count'='1',
  'typeOfData'='file')
```

Configuration de l'accès réseau aux magasins de données

Pour exécuter vos tâches d'extraction, de transformation et de chargement (ETL), AWS Glue doit être en mesure d'accéder à vos magasins de données. Si une tâche n'a pas besoin de s'exécuter dans le sous-réseau de votre Virtual Private Cloud (VPC), par exemple, si elle transforme les données d'Amazon S3 vers Amazon S3, aucune configuration supplémentaire n'est nécessaire.

Si une tâche doit s'exécuter dans votre sous-réseau VPC, par exemple, en transformant des données d'un magasin de données JDBC dans un sous-réseau privé, AWS Glue configure les interfaces réseau Elastic qui permettent à vos tâches de se connecter en toute sécurité à d'autres ressources au sein de votre VPC. À chaque interface réseau Elastic est affectée une adresse IP privée comprise dans la plage d'adresses IP du sous-réseau que vous avez spécifié. Aucune adresse IP publique n'est attribuée. Les groupes de sécurité spécifiés dans la connexion AWS Glue sont appliqués sur chacune des interfaces réseau Elastic. Pour de plus amples informations, veuillez consulter Configuration d'Amazon VPC pour les connexions JDBC aux entrepôts de données Amazon RDS depuis AWS Glue.

Tous les magasins de données JDBC auxquels la tâche peut accéder doivent être disponibles à partir du sous-réseau du VPC. Pour accéder à Amazon S3 à partir de votre VPC, un point de terminaison d'un VPC est requis. Si votre tâche a besoin d'accéder aux ressources du VPC et au réseau Internet public, le VPC doit disposer d'une passerelle de traduction d'adresses réseau (NAT) dans le VPC.

Une tâche ou un point de terminaison de développement ne peut accéder qu'à un seul VPC (et sous-réseau) à la fois. Si vous devez accéder aux banques de données de différentes manières VPCs, les options suivantes s'offrent à vous :

- Utilisez l'appairage de VPC pour accéder aux magasins de données. Pour plus d'informations sur l'appairage de VPC, consultez Principes de base de l'appairage de VPC.
- Utilisez un compartiment Amazon S3 comme emplacement de stockage intermédiaire. Fractionnez le travail en deux tâches, avec la sortie Amazon S3 de la tâche 1 comme entrée de la tâche 2.

Pour plus d'informations sur la façon de se connecter à un magasin de données Amazon Redshift à l'aide d'Amazon VPC, consultez the section called “Configurer Redshift”.

Pour plus d'informations sur la façon de se connecter aux magasins de données Amazon RDS à l'aide d'Amazon VPC, consultez the section called "Configuration d'Amazon VPC pour se connecter aux entrepôts de données Amazon RDS".

Une fois les règles nécessaires définies dans Amazon VPC, vous créez une connexion dans AWS Glue avec les propriétés nécessaires pour vous connecter à vos magasins de données. Pour plus d'informations sur la connexion, consultez Connexion aux données.

> **Note**
>
> Veillez à configurer votre environnement DNS pour AWS Glue. Pour de plus amples informations, veuillez consulter Configuration du DNS de votre VPC.

Rubriques

- Configuration d'un VPC pour se connecter à PyPI pour AWS Glue
- Configuration du DNS de votre VPC

Configuration d'un VPC pour se connecter à PyPI pour AWS Glue

Le Python Package Index (PyPI) est un référentiel de logiciels pour le langage de programmation Python. Cette rubrique aborde les détails nécessaires pour prendre en charge l'utilisation des packages installés par pip (comme spécifié par le créateur de la session à l'aide de l'indicateur `--additional-python-modules`).

L'utilisation de sessions AWS Glue interactives avec un connecteur entraîne l'utilisation du réseau VPC via le sous-réseau spécifié pour le connecteur. Par conséquent, les AWS services et autres destinations réseau ne sont pas disponibles, sauf si vous configurez une configuration spéciale.

Les solutions à ce problème incluent :

- L'utilisation d'une passerelle Internet accessible par votre session.
- Configuration et utilisation d'un compartiment S3 avec un PyPI/simple dépôt contenant la fermeture transitive des dépendances d'un ensemble de packages.
- Utilisation d'un CodeArtifact référentiel qui met en miroir PyPI et qui est connecté à votre VPC.

Configuration d'une passerelle Internet

Les aspects techniques sont détaillés dans Cas d'utilisation de la passerelle NAT, mais notez ces exigences pour l'utilisation de `--additional-python-modules`. Plus précisément, `--additional-python-modules` nécessite l'accès à pypi.org, qui est déterminé par la configuration de votre VPC. Notez les critères suivants :

1. L'obligation d'installer des modules Python supplémentaires via l'installation pip pour la session d'un utilisateur. Si la session utilise un connecteur, votre configuration peut être affectée.
2. Lorsqu'un connecteur est utilisé avec `--additional-python-modules`, lors du démarrage de la session, le sous-réseau associé aux `PhysicalConnectionRequirements` du connecteur doit fournir un chemin réseau pour atteindre pypi.org.
3. Vous devez déterminer si votre configuration est correcte ou non.

Configuration d'un compartiment Amazon S3 pour héberger un PyPI/simple dépôt ciblé

Cet exemple configure un miroir PyPI dans Amazon S3 pour un ensemble de packages et leurs dépendances.

Pour configurer le miroir PyPI pour un ensemble de packages :

```
# pip download all the dependencies
pip download -d s3pypi --only-binary :all: plotly gglplot
pip download -d s3pypi --platform manylinux_2_17_x86_64 --only-binary :all: psycopg2-
binary
# create and upload the pypi/simple index and wheel files to the s3 bucket
s3pypi -b test-domain-name --put-root-index -v s3pypi/*
```

Si vous possédez déjà un référentiel d'artefacts, il contiendra une URL d'index pour l'utilisation de pip que vous pourrez fournir à la place de l'exemple d'URL pour le compartiment Amazon S3 comme indiqué ci-dessus.

Pour utiliser l'URL d'index personnalisée, avec quelques exemples de packages :

```
%%configure
{
    "--additional-python-modules": "psycopg2_binary==2.9.5",
    "python-modules-installer-option": "--no-cache-dir --verbose --index-url https://
test-domain-name.s3.amazonaws.com/ --trusted-host test-domain-name.s3.amazonaws.com"
```

```
}
```

Configuration d'un CodeArtifact miroir de pypi connecté à votre VPC

Pour configurer un miroir :

1. Créez un référentiel dans la même région que le sous-réseau utilisé par le connecteur.

   Sélectionnez `Public upstream repositories` et choisissez `pypi-store`.
2. Fournissez un accès au référentiel depuis le VPC pour le sous-réseau.
3. Spécifiez l'`--index-url` correcte en utilisant le `python-modules-installer-option`.

```
%%configure
{
    "--additional-python-modules": "psycopg2_binary==2.9.5",
    "python-modules-installer-option": "--no-cache-dir --verbose --index-url https://
test-domain-name.s3.amazonaws.com/ --trusted-host test-domain-name.s3.amazonaws.com"
}
```

Pour plus d'informations, consultez la section Utilisation CodeArtifact depuis un VPC.

Configuration du DNS de votre VPC

Le DNS (Domain Name System) est une norme permettant la résolution des noms utilisés sur Internet en leurs adresses IP correspondantes. Un nom d'hôte DNS nomme de façon unique un ordinateur et se compose d'un nom d'hôte et d'un nom de domaine. Les serveurs DNS résolvent les noms d'hôte DNS en adresses IP correspondantes.

Pour configurer le DNS de votre VPC, assurez-vous que les noms d'hôte DNS et la résolution DNS sont activés dans votre VPC. Les attributs de réseau VPC `enableDnsHostnames` et `enableDnsSupport` doivent être définis sur `true`. Pour afficher et modifier ces attributs, accédez à la console VPC à l'adresse https://console.aws.amazon.com/vpc/.

Pour plus d'informations, consultez Utilisation de DNS avec votre VPC. Vous pouvez également utiliser la modify-vpc-attributecommande AWS CLI and call pour configurer les attributs du réseau VPC.

> **Note**
>
> Si vous utilisez Route 53, vérifiez que votre configuration ne remplace pas les attributs réseau DNS.

Configuration du chiffrement dans AWS Glue

L'exemple suivant de flux de travail met en évidence les options à configurer lorsque vous utilisez le chiffrement avec AWS Glue. L'exemple montre l'utilisation de touches spécifiques AWS Key Management Service (AWS KMS), mais vous pouvez choisir d'autres paramètres en fonction de vos besoins particuliers. Ce flux de travail met en évidence uniquement les options liées au chiffrement lors de la configuration d'AWS Glue.

1. Si l'utilisateur de la console AWS Glue n'utilise pas une stratégie d'autorisation autorisant toutes les opérations d'API AWS Glue (par exemple, `"glue:*"`), vérifiez que les actions suivantes sont permises :
   - `"glue:GetDataCatalogEncryptionSettings"`
   - `"glue:PutDataCatalogEncryptionSettings"`
   - `"glue:CreateSecurityConfiguration"`
   - `"glue:GetSecurityConfiguration"`
   - `"glue:GetSecurityConfigurations"`
   - `"glue:DeleteSecurityConfiguration"`
2. Tout client qui accède ou écrit à un catalogue chiffré, c'est-à-dire, tout utilisateur de console, crawler, tâche ou point de terminaison de développement, nécessite les autorisations suivantes.
3. Tout utilisateur ou rôle qui accède à un mot de passe de connexion chiffré nécessite les autorisations suivantes.
4. Le rôle de toute tâche Extract-transform-load (ETL) qui écrit des données chiffrées sur Amazon S3 a besoin des autorisations suivantes.

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
```

```
    "Action": [
      "kms:Decrypt",
      "kms:Encrypt",
      "kms:GenerateDataKey"
    ],
    "Resource": "arn:aws:kms:us-east-1:111122223333:key/key-id"
  }
}
```

5. Toute tâche ou robot d'exploration ETL qui écrit des Amazon CloudWatch Logs chiffrés nécessite les autorisations suivantes dans la clé et les politiques IAM.

   Dans la stratégie de clé (et non la politique IAM) :

```
{
  "Effect": "Allow",
  "Principal": {
   "Service": "logs.region.amazonaws.com"
  },
  "Action": [
   "kms:Encrypt*",
   "kms:Decrypt*",
   "kms:ReEncrypt*",
   "kms:GenerateDataKey*",
   "kms:Describe*"
  ],
  "Resource": "<arn of key used for ETL/crawler cloudwatch encryption>"
 }
```

   Pour de plus amples informations sur les stratégies de clé, veuillez consulter la section Utilisation de stratégies de clé dans AWS KMS du Guide du développeur AWS Key Management Service .

   Dans la politique IAM, attachez l'autorisation `logs:AssociateKmsKey` :

```
{
  "Effect": "Allow",
  "Principal": {
   "Service": "logs.region.amazonaws.com"
  },
  "Action": [
   "logs:AssociateKmsKey"
  ],
```

```
  "Resource": "<arn of key used for ETL/crawler cloudwatch encryption>"
}
```

6. Toute tâche ETL qui utilise un signet de tâche chiffré a besoin des autorisations suivantes.

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "kms:Decrypt",
      "kms:Encrypt"
    ],
    "Resource": "arn:aws:kms:us-east-1:111122223333:key/*"
  }
}
```

7. Dans la console AWS Glue, choisissez Paramètres dans le volet de navigation.
   a. Sur la page Data catalog settings (Paramètres du catalogue de données), chiffrez votre catalogue de données en sélectionnant Metadata encryption (Chiffrement des métadonnées). Cette option chiffre tous les objets du catalogue de données avec la AWS KMS clé de votre choix.
   b. Pour CléAWS KMS , choisissez aws/glue. Vous pouvez également choisir une AWS KMS clé que vous avez créée.

> **Important**
>
> AWS Gluene prend en charge que les clés principales symétriques du client (CMKs). La liste de AWS KMS key (clé KMS) n'affiche que des clés symétriques. Toutefois, si vous sélectionnez Choisir un ARN de AWS KMS clé, la console vous permet de saisir un ARN pour n'importe quel type de clé. Assurez-vous de n'entrer que ARNs pour les clés symétriques.

   Lorsque le chiffrement est activé, le client qui accède au catalogue de données doit avoir les autorisations AWS KMS .

8. Dans le volet de navigation, choisissez Security configurations (Configurations de sécurité). Une configuration de sécurité est un ensemble de propriétés de sécurité pouvant être utilisées pour

configurer les processus AWS Glue. Ensuite, choisissez Ajouter une configuration de sécurité. Dans la configuration, choisissez l'une des options suivantes :

a. Sélectionnez S3 encryption (Chiffrement S3). Pour Mode de chiffrement, choisissez SSE-KMS. Pour la CléAWS KMS , choisissez aws/s3 (assurez-vous que l'utilisateur a l'autorisation d'utiliser cette clé). Cela permet aux données écrites par la tâche sur Amazon S3 d'utiliser la AWS Glue AWS KMS clé AWS gérée.

b. Sélectionnez le chiffrement CloudWatch des journaux, puis choisissez une clé CMK. (Assurez-vous que l'utilisateur a l'autorisation d'utiliser cette clé). Pour plus d'informations, consultez la section Chiffrer les données des CloudWatch journaux dans les journaux AWS KMSà l'aide du guide du AWS Key Management Service développeur.

> **Important**
>
> AWS Gluene prend en charge que les clés principales symétriques du client (CMKs). La liste de AWS KMS key (clé KMS) n'affiche que des clés symétriques. Toutefois, si vous sélectionnez Choisir un ARN de AWS KMS clé, la console vous permet de saisir un ARN pour n'importe quel type de clé. Assurez-vous de n'entrer que ARNs pour les clés symétriques.

c. Choisissez Advanced properties (Propriétés avancées) et sélectionnez Job bookmark encryption (Chiffrement de signet de tâche). Pour la CléAWS KMS , choisissez aws/glue (assurez-vous que l'utilisateur a l'autorisation d'utiliser cette clé). Cela permet de chiffrer les signets de tâches écrits sur Amazon S3 avec la AWS Glue AWS KMS clé.

9. Dans le volet de navigation, choisissez Connections (Connexions).

   a. Choisissez Ajouter une connexion pour créer une connexion avec le magasin de données Java Database Connectivity (JDBC) qui est la cible de votre tâche ETL.

   b. Pour appliquer le chiffrement Secure Sockets Layer (SSL), sélectionnez Require SSL connection (Connexion SSL obligatoire) et testez votre connexion.

10. Dans le volet de navigation, sélectionnez Tâches.

    a. Choisissez Ajouter une tâche pour créer une tâche qui transforme les données.

    b. Dans la définition de tâche, choisissez la configuration de sécurité que vous avez créée.

11. Sur la console AWS Glue, exécutez votre travail à la demande. Vérifiez que toutes les données Amazon S3 écrites par la tâche, les CloudWatch journaux écrits par la tâche et les favoris de la tâche sont tous chiffrés.

Configuration du réseau pour le développement de AWS Glue

Pour exécuter vos scripts ETL (extraction, transformation et chargement) avec AWS Glue, vous pouvez parfois développer et tester vos scripts à l'aide d'un point de terminaison de développement. Les points de terminaison de développement ne sont pas pris en charge pour une utilisation avec les tâches AWS Glue version 2.0. Pour les versions 2.0 et ultérieures, la méthode de développement préférée consiste à utiliser Jupyter Notebook avec l'un des noyaux AWS Glue. Pour de plus amples informations, veuillez consulter the section called “Démarrage avec séances interactives AWS Glue”.

Configuration de votre réseau pour un point de terminaison de développement

Lorsque vous configurez un point de terminaison de développement, vous spécifiez un cloud privé virtuel (VPC), un sous-réseau et des groupes de sécurité.

> **Note**
>
> Veillez à configurer votre environnement DNS pour AWS Glue. Pour de plus amples informations, veuillez consulter Configuration du DNS de votre VPC.

Pour permettre à AWS Glue d'accéder aux ressources nécessaires, ajoutez une ligne dans votre table de routage de sous-réseau pour associer une liste de préfixes pour Amazon S3 au point de terminaison d'un VPC. Un ID de liste de préfixe est requis pour créer une règle de groupe de sécurité sortant qui autorise le trafic en provenance d'un VPC à accéder à un service AWS par le biais d'un point de terminaison d'un VPC. Pour faciliter la connexion à un serveur de bloc-notes associé à ce point de terminaison de développement, à partir de votre ordinateur local, ajoutez une ligne dans la table de routage pour ajouter un ID de passerelle Internet. Pour plus d'informations, consultez la page Points de terminaison d'un VPC. Mettez à jour la table de routage du sous-réseau de sorte qu'elle soit similaire à la table suivante :

Destination	Cible		
10.0.0.0/16	locale		
pl-id pour Amazon S3	vpce-id		
0.0.0.0/0	igw-xxxx		

Pour permettre à AWS Glue de communiquer entre ses composants, spécifiez un groupe de sécurité avec une règle entrante avec référence circulaire pour tous les ports TCP. En créant une règle avec référence circulaire, vous pouvez restreindre la source au même groupe de sécurité dans le VPC et ainsi, ne pas l'ouvrir à tous les réseaux. Le groupe de sécurité par défaut pour votre VPC peut déjà avoir une règle entrante avec référence circulaire pour ALL Traffic.

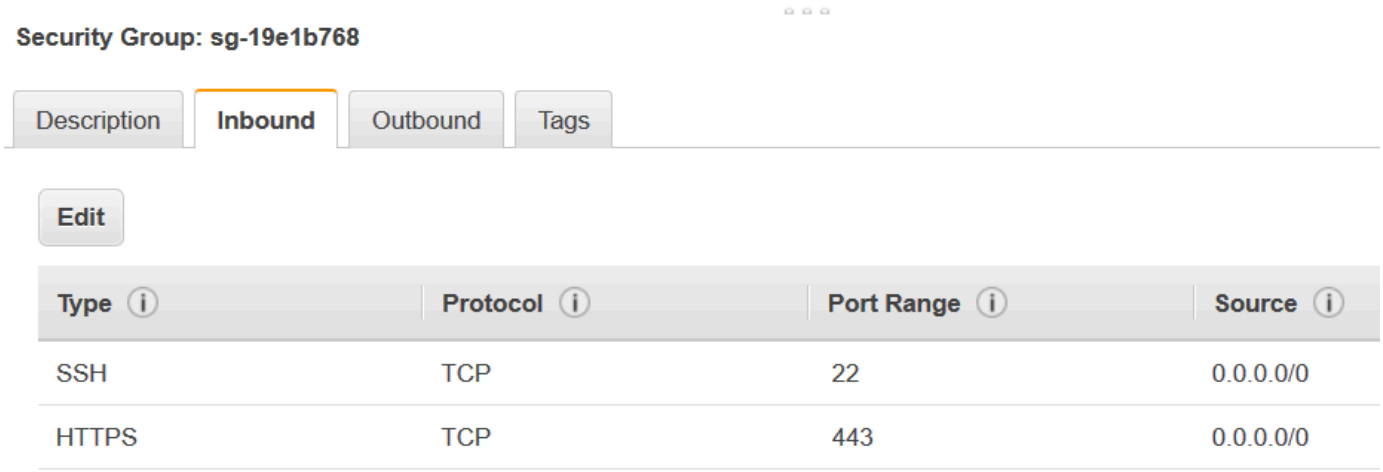
Pour configurer un groupe de sécurité

1. Connectez-vous à la console Amazon EC2 AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/ec2/
2. Dans le volet de navigation de gauche, sélectionnez Security Groups.
3. Sélectionnez un groupe de sécurité existant dans la liste, ou choisissez Create Security Group (Créer un groupe de sécurité) pour définir le groupe de sécurité à utiliser avec le point de terminaison de développement.
4. Dans le volet du groupe de sécurité, accédez à l'onglet Inbound (Entrant).
5. Ajoutez une règle avec référence circulaire afin de permettre aux composants AWS Glue de communiquer. Plus spécifiquement, ajoutez ou confirmez qu'il existe une règle Type `All TCP`, que Protocol (Protocole) est `TCP`, que Port Range (Plage de ports) comprend tous les ports et que la valeur de Source est le même nom de groupe de sécurité que la valeur de Group ID (ID du groupe).

   La règle entrante ressemble à ce qui suit :

Type	Protocole	Plage de ports	Source
Tous les TCP	TCP	0-65535	*`security-group`*

   L'exemple suivant présente une règle entrante avec référence circulaire :



6. Ajoutez également une règle pour le trafic sortant. Ouvrez le trafic sortant pour tous les ports ou créez une règle avec référence circulaire de Type `All TCP`, de Protocol (Protocole) `TCP`, dont la valeur Port Range (Plage de ports) comprend tous les ports et que la valeur de Source est le même nom de groupe de sécurité que la valeur de Group ID (ID du groupe).

   La règle sortante ressemble à l'une de ces règles :

Type	Protocole	Plage de ports	Destination
Tous les TCP	TCP	0-65535	*`security-group`*
Tout le trafic	ALL	ALL	0.0.0.0/0

Configuration d'Amazon EC2 pour un serveur de bloc-notes

Un point de terminaison de développement permet de créer un serveur de bloc-notes pour tester vos scripts ETL avec les blocs-notes Jupyter. Pour activer la communication avec votre bloc-notes, spécifiez un groupe de sécurité avec des règles entrantes pour HTTPS (port 443) et SSH (port 22). Assurez-vous que la source de la règle est 0.0.0.0/0 ou l'adresse IP de la machine qui se connecte au bloc-notes.

Pour configurer un groupe de sécurité

1. Connectez-vous à la console Amazon EC2 AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/ec2/
2. Dans le volet de navigation de gauche, sélectionnez Security Groups.

3. Sélectionnez un groupe de sécurité existant dans la liste, ou choisissez Create Security Group (Créer un groupe de sécurité) pour définir le groupe de sécurité à utiliser avec votre serveur de bloc-notes. Le groupe de sécurité associé à votre point de terminaison de développement est également utilisé pour créer votre serveur de bloc-notes.
4. Dans le volet du groupe de sécurité, accédez à l'onglet Inbound (Entrant).
5. Ajoutez des règles entrantes semblables à ce qui suit :

Type	Protocole	Plage de ports	Source
SSH	TCP	22	0.0.0.0/0
HTTPS	TCP	443	0.0.0.0/0

L'image suivante présente un exemple de règles entrantes pour le groupe de sécurité :

Découverte et catalogage des données dans AWS Glue

AWS Glue Data Catalog Il s'agit d'un référentiel centralisé qui stocke les métadonnées relatives aux ensembles de données de votre organisation. Il sert d'index pour l'emplacement, le schéma et les métriques d'exécution de vos sources de données. Les métadonnées sont stockées dans des tables de métadonnées, chaque table représentant un seul entrepôt de données.

Vous pouvez remplir le catalogue de données à l'aide d'un robot, qui analyse automatiquement vos sources de données et extrait les métadonnées. Un robot peut se connecter à des sources de données internes (basées sur AWS) et externes à AWS.

Pour plus d'informations sur les sources de données prises en charge, consultez Sources de données prises en charge pour l'indexation de site web.

Vous pouvez également créer des tables manuellement dans le catalogue de données en définissant la structure des tables, le schéma et la structure de partitionnement en fonction de vos besoins spécifiques.

Pour plus d'informations sur la création manuelle de tables de métadonnées, consultez Définition manuelle des métadonnées.

Les informations du catalogue de données vous permettent de créer et de surveiller vos tâches ETL. Le catalogue de données s'intègre à d'autres services d' AWS analyse, fournissant une vue unifiée des sources de données, ce qui facilite la gestion et l'analyse des données.

- Amazon Athena : stockez et interrogez les métadonnées des tables dans le catalogue de données pour les données Amazon S3 à l'aide de SQL.
- AWS Lake Formation — Définissez et gérez de manière centralisée des politiques d'accès aux données précises et auditez l'accès aux données.
- Amazon EMR : accédez aux sources de données définies dans le catalogue de données pour le traitement de big data.
- Amazon SageMaker AI — Créez, formez et déployez des modèles d'apprentissage automatique rapidement et en toute confiance.

Principales fonctionnalités du catalogue de données

Les principaux aspects du catalogue de données sont les suivants.

Référentiel de métadonnées

Le catalogue de données sert de référentiel de métadonnées central, stockant des informations sur l'emplacement, le schéma et les propriétés de vos sources de données. Ces métadonnées sont organisées en bases de données et en tables, comme dans un catalogue de base de données relationnelle traditionnel.

Découverte automatique des données

AWS Glue crawler s peut automatiquement découvrir et cataloguer des sources de données nouvelles ou mises à jour, réduisant ainsi la charge de travail liée à la gestion manuelle des métadonnées et garantissant la pérennité de votre catalogue de données up-to-date. En cataloguant vos sources de données, le catalogue de données permet aux utilisateurs et aux applications de découvrir et de comprendre plus facilement les ressources de données disponibles au sein de votre organisation, et favorise ainsi la réutilisation des données et la collaboration.

Le catalogue de données prend en charge un large éventail de sources de données, notamment Amazon S3, Amazon RDS, Amazon Redshift, Apache Hive, etc. Il peut automatiquement déduire et stocker les métadonnées de ces sources à l'aide de AWS Glue crawler s.

Pour plus d'informations, voir,Utilisation de robots pour alimenter le catalogue de données.

Gestion des schémas

Le catalogue de données capture et gère automatiquement le schéma de vos sources de données, y compris l'inférence, l'évolution et la gestion des versions des schémas. Vous pouvez mettre à jour votre schéma et vos partitions dans le catalogue de données à l'aide de tâches AWS Glue ETL.

Optimisation des tables

Pour améliorer les performances de lecture des services AWS d'analyse tels qu'Amazon Athena et Amazon EMR, ainsi que pour les tâches AWS Glue ETL, le catalogue de données fournit un compactage géré (un processus qui compacte de petits objets Amazon S3 en objets plus grands) pour les tables Iceberg du catalogue de données. Vous pouvez utiliser une AWS Glue console, une AWS Lake Formation console ou une AWS API pour activer ou désactiver le compactage de tables Iceberg individuelles figurant dans le catalogue de données. AWS CLI

Pour de plus amples informations, veuillez consulter Optimisation des tables Iceberg.

Statistiques de colonne

Vous pouvez calculer des statistiques de colonne pour des tables du catalogue des données dans des formats de données tels que Parquet, ORC, JSON, ION, CSV et XML sans définir des pipelines de données supplémentaires. Les statistiques de colonne vous aident à comprendre les profils de données en obtenant des informations sur les valeurs d'une colonne. Le catalogue de données prend en charge la génération de statistiques pour les valeurs de colonne telles que la valeur minimale, la valeur maximale, le total des valeurs nulles, le total des valeurs distinctes, la longueur moyenne des valeurs et le nombre total d'occurrences de valeurs vraies.

Pour de plus amples informations, veuillez consulter Optimisation des performances des requêtes à l'aide des statistiques de colonne.

Traçabilité des données

Le catalogue de données conserve un enregistrement des transformations et des opérations effectuées sur vos données, fournissant des informations sur la traçabilité des données. Ces informations de traçabilité sont précieuses pour la vérification, la conformité et la compréhension de la provenance des données.

Intégration avec d'autres AWS services

Le catalogue de données s'intègre parfaitement à d'autres AWS services AWS Lake Formation, tels qu'Amazon Athena, Amazon Redshift Spectrum et Amazon EMR. Cette intégration vous permet d'interroger et d'analyser des données dans différents entrepôts de données à l'aide d'une couche de métadonnées unique et cohérente.

Sécurité et contrôle d'accès

AWS Glue s'intègre AWS Lake Formation pour prendre en charge un contrôle d'accès précis aux ressources du catalogue de données, ce qui vous permet de gérer les autorisations et de sécuriser l'accès à vos actifs de données en fonction des politiques et des exigences de votre organisation. AWS Glue s'intègre à AWS Key Management Service (AWS KMS) pour chiffrer les métadonnées stockées dans le catalogue de données.

Vues matérialisées

Le catalogue de données prend en charge les vues matérialisées Apache Iceberg, qui sont des tables gérées qui stockent les résultats précalculés des requêtes SQL et s'actualisent automatiquement à mesure que les données sources sous-jacentes changent. Les vues matérialisées simplifient les pipelines de transformation des données et accélèrent les performances des requêtes en éliminant les calculs redondants.

Vous pouvez créer des vues matérialisées à l'aide d'Apache Spark SQL dans AWS Glue version 5.1 et ultérieure, Amazon EMR version 7.12.0 et versions ultérieures, et Amazon Athena. Le catalogue de données surveille automatiquement les tables Apache Iceberg sources et actualise les vues matérialisées à l'aide d'une infrastructure informatique gérée. Les moteurs Spark de AWS Glue, Amazon EMR et Amazon Athena peuvent automatiquement réécrire les requêtes afin d'utiliser des vues matérialisées lorsqu'elles offrent de meilleures performances.

Les vues matérialisées sont stockées sous forme de tables Apache Iceberg dans des compartiments Amazon S3 Tables ou dans des compartiments à usage général Amazon S3 au sein de votre compte, ce qui les rend accessibles depuis plusieurs moteurs de requêtes. Le catalogue de données gère tous les aspects du cycle de vie des vues matérialisées, notamment la planification automatique des actualisations, les mises à jour incrémentielles et la gestion des métadonnées.

Pour plus d'informations, consultez les sections Utilisation de vues matérialisées avec AWS Glue et Utilisation de vues matérialisées avec Amazon EMR.

Rubriques

- Remplissage du catalogue de AWS Glue données
- Remplissage et gestion des tables transactionnelles
- Gestion du catalogue de données
- Accès au catalogue de données
- AWS Glue Bonnes pratiques en matière de catalogue de données
- Surveillance des métriques d'utilisation du catalogue de données dans Amazon CloudWatch
- Registre de schémas AWS Glue

Remplissage du catalogue de AWS Glue données

Vous pouvez remplir le fichier à l' AWS Glue Data Catalog aide des méthodes suivantes :

- AWS Glue crawler — An AWS Glue crawler peut automatiquement découvrir et cataloguer des sources de données telles que des bases de données, des lacs de données et des données de streaming. Les robots constituent la méthode la plus courante et la plus recommandée pour remplir le catalogue de données, car ils peuvent automatiquement découvrir et déduire les métadonnées à partir d'une grande variété de sources de données.

- Ajout manuel de métadonnées : vous pouvez définir manuellement les bases de données, les tables et les détails de connexion et les ajouter au catalogue de données à l'aide de la AWS Glue console, de la console Lake Formation ou AWS Glue APIs. AWS CLI La saisie manuelle est utile lorsque vous souhaitez cataloguer des sources de données qui ne peuvent pas être analysées.
- Intégration à d'autres AWS services : vous pouvez remplir le catalogue de données avec des métadonnées provenant de services tels qu'Amazon AWS Lake Formation Athena. Ces services peuvent découvrir et enregistrer des sources de données dans le catalogue de données.
- Remplissage à partir d'un référentiel de métadonnées existant : si vous disposez d'un magasin de métadonnées existant tel qu'Apache Hive Metastore, vous pouvez l'utiliser AWS Glue pour importer ces métadonnées dans le catalogue de données. Pour plus d'informations, consultez la section Migration entre le métastore Hive et le on. AWS Glue Data Catalog GitHub

Rubriques

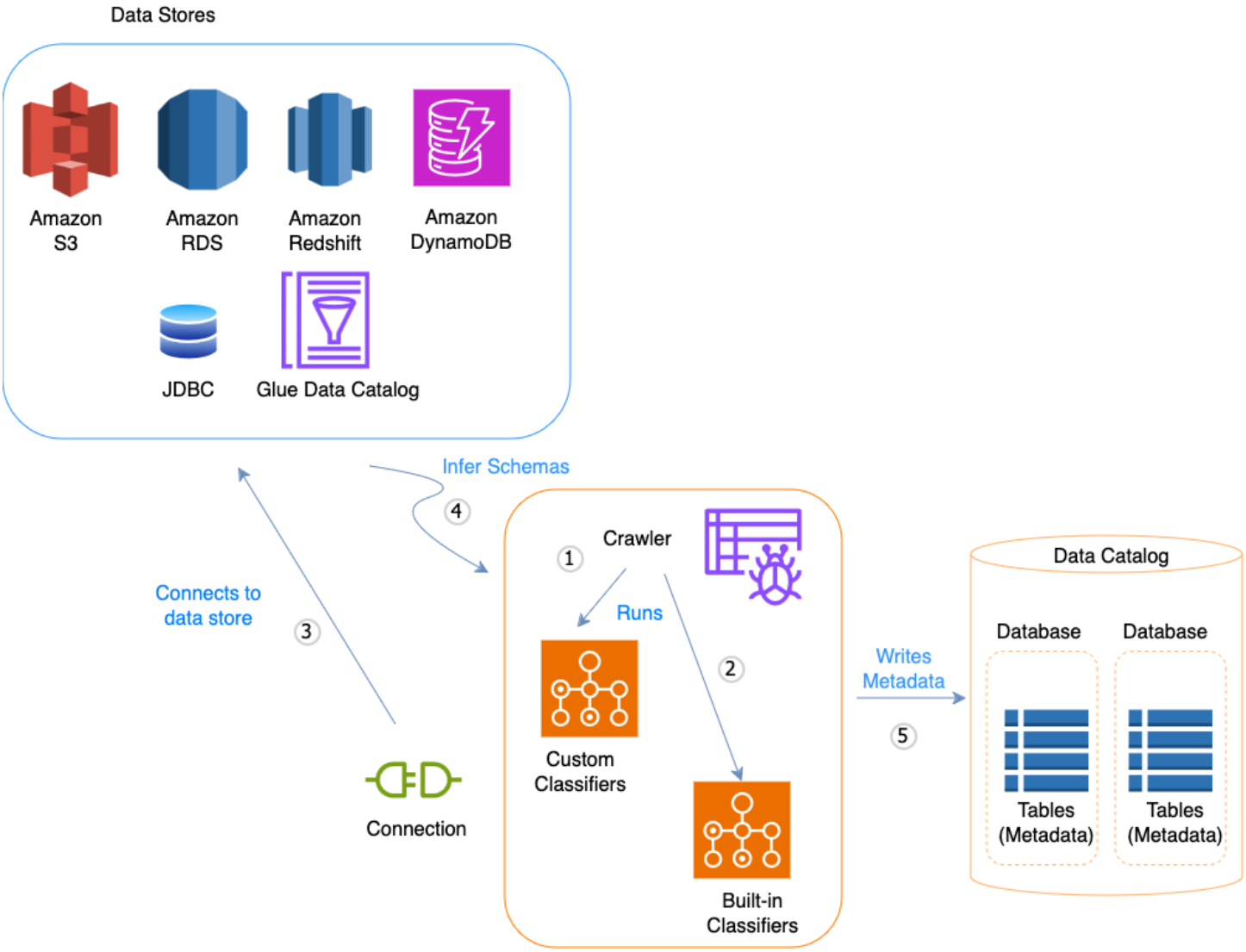
- Utilisation de robots pour alimenter le catalogue de données
- Définition manuelle des métadonnées
- Intégration aux tables Amazon S3
- Intégration à d'autres AWS services
- Paramètres du catalogue de données

Utilisation de robots pour alimenter le catalogue de données

Vous pouvez utiliser an AWS Glue crawler pour les renseigner AWS Glue Data Catalog avec des bases de données et des tables. Il s'agit de la principale méthode utilisée par la plupart des AWS Glue utilisateurs. Un crawler peut analyser plusieurs magasins de données en une seule fois. À la fin de cette opération, l'crawler crée ou met à jour une ou plusieurs tables dans votre Data Catalog. Les tâches Extract-transform-load (ETL) que vous définissez dans AWS Glue utilisent ces tables Data Catalog en tant que sources et cibles. La tâche ETL lit et écrit dans les magasins de données qui sont spécifiés dans les tables Data Catalog sources et cibles.

Flux de travail

Le diagramme de flux de travail suivant montre comment les crawlers AWS Glue interagissent avec magasins de données et d'autres éléments pour remplir le catalogue de données.



Voici comment un crawler remplit le AWS Glue Data Catalog :

1. Un crawler exécute tous les classifieurs personnalisés que vous choisissez pour déduire le format et le schéma de vos données. Vous fournissez le code pour les classificateurs personnalisés, lesquels s'exécutent dans l'ordre que vous spécifiez.

   Le premier classifieur personnalisé qui reconnaît avec succès la structure de vos données est utilisé pour créer un schéma. Les classifieurs personnalisés en bas de la liste sont ignorés.
2. Si aucun classifieur ne correspond au schéma de vos données, les classifieurs intégrés essaient de reconnaître le schéma de données. Un exemple de classifieur intégré est un classifieur qui reconnaît JSON.
3. L'crawler se connecte au magasin de données. Certains magasins de données nécessitent les propriétés de connexion pour l'accès de l'crawler.

4. Le schéma déduit est créé pour vos données.
5. L'crawler écrit les métadonnées dans le catalogue de données. Une définition de table contient les métadonnées sur les données de votre magasin de données. La table est écrite dans une base de données, qui est un conteneur de tables du catalogue de données. Les attributs d'une table incluent la classification, qui est une étiquette créé par le classificateur ayant déduit du schéma de la table.

Rubriques

- Fonctionnement des crawlers
- Comment un crawler détermine quand créer des partitions ?
- Sources de données prises en charge pour l'indexation de site web
- Prérequis pour le crawler
- Définition et gestion des classificateurs
- Configuration d'un robot
- Planification d'un crawler
- Affichage des résultats et des détails du crawler
- Personnalisation du comportement d'un robot
- Didacticiel : Ajout d'un crawler AWS Glue

Fonctionnement des crawlers

Lorsqu'un crawler s'exécute, il prend les actions suivantes pour interroger un magasin de données :

- Classe les données pour déterminer le format, le schéma et les propriétés associées des données brutes – Vous pouvez configurer les résultats de la classification en créant un classifieur personnalisé.
- Groupes les données en tables ou en partitions – Les données sont regroupées en fonction de l'heuristique de l'crawler.
- Writes metadata to the Data Catalog (Écrit les métadonnées sur Data Catatlog) – vous pouvez configurer la façon dont l'crawler ajoute, met à jour et supprime les tables et les partitions.

Lorsque vous définissez un crawler, vous choisissez un ou plusieurs classifieurs qui évaluent le format de vos données pour déduire un schéma. Lorsque l'crawler s'exécute, le premier classificateur

de la liste qui reconnaît correctement votre magasin de données est utilisé pour créer un schéma pour votre table. Vous pouvez utiliser des classifieurs intégrés ou définir les vôtres. Vous définissez vos classifieurs personnalisés dans une autre opération, avant de définir les crawlers. AWS Glue fournit des classifieurs intégrés pour déduire les schémas les plus répandus qui incluent les formats de fichiers JSON, CSV et Apache Avro. Pour obtenir la liste actuelle des classifieurs intégrés dans AWS Glue, consultez Classifieurs intégrés.

Les tables de métadonnées créées par un crawler sont contenues dans une base de données lorsque vous définissez un crawler. Si votre crawler ne spécifie pas de base de données, vos tables sont placées dans la base de données par défaut. En outre, chaque table dispose d'une colonne de classification remplie par le premier classificateur qui reconnaît correctement le magasin de données.

Si le fichier analysé est compressé, l'crawler doit le télécharger afin de le traiter. Lorsqu'un crawler s'exécute, il interroge les fichiers pour déterminer leur format et leur type de compression, et écrit ces propriétés dans Data Catalog. Certains formats de fichier (Apache Parquet, par exemple), vous permettent de compresser des parties du fichier au moment de son écriture. Pour ces fichiers, les données compressées sont un composant interne du fichier et AWS Glue ne renseigne pas la propriété `compressionType` lors de l'écriture des tables dans Data Catalog. En revanche, si un fichier complet est compressé par un algorithme de compression (par exemple, gzip), la propriété `compressionType` est alors renseignée lorsque les tables sont écrites dans Data Catalog.

L'crawler génère les noms pour les tables qu'il crée. Les noms des tables stockées dans le sont conformes aux AWS Glue Data Catalog règles suivantes :

- Seuls les caractères alphanumériques et les traits de soulignement (_) sont autorisés.
- Un préfixe personnalisé ne doit pas comporter plus de 64 caractères.
- La longueur maximale du nom ne doit pas dépasser 128 caractères. L'crawler tronque les noms générés pour les ajuster à la limite.
- En cas de noms de table en double, l'crawler ajoute un suffixe de chaîne de hachage au nom.

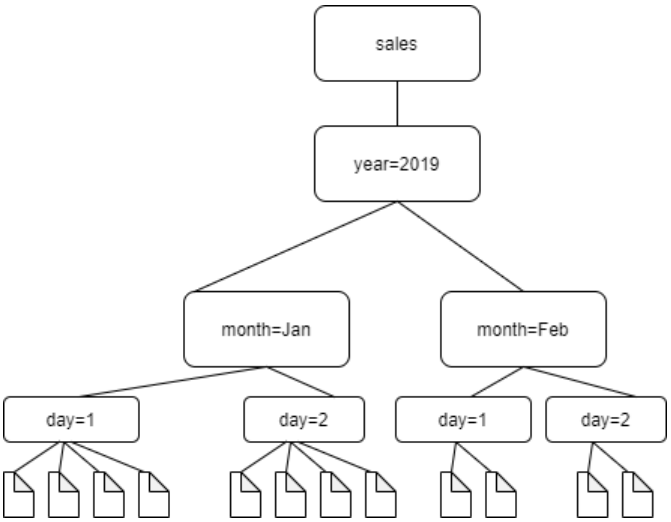
Si votre crawler s'exécute plusieurs fois, par exemple sur un calendrier, il recherche les tables et les fichiers nouveaux ou modifiés dans votre magasin de données. La sortie de l'crawler inclut les nouvelles tables et partitions trouvées depuis l'exécution précédente.

Comment un crawler détermine quand créer des partitions ?

Lorsqu'un AWS Glue robot d'exploration analyse le magasin de données Amazon S3 et détecte plusieurs dossiers dans un compartiment, il détermine la racine d'une table dans la structure des

dossiers et les dossiers qui sont des partitions d'une table. Le nom de la table est basé sur le préfixe Amazon S3 ou le nom de dossier. Vous fournissez un chemin Include qui pointe vers le niveau de dossier à analyser. Lorsque la majorité des schémas au niveau d'un dossier sont similaires, l'crawler crée les partitions d'une table au lieu de tables distinctes. Pour influencer l'crawler et créer des tables distincts, ajoutez le dossier racine de chaque table en tant que magasin de données séparé lorsque vous définissez l'crawler.

Prenons l'exemple de la structure de dossiers Amazon S3 suivante.



Les chemins d'accès aux quatre dossiers de niveau inférieur sont les suivants :

```
S3://sales/year=2019/month=Jan/day=1
S3://sales/year=2019/month=Jan/day=2
S3://sales/year=2019/month=Feb/day=1
S3://sales/year=2019/month=Feb/day=2
```

Supposons que la cible de l'crawler est définie sur `Sales` et que tous les fichiers des dossiers `day=`*n* ont le même format (par exemple, JSON, non chiffré) et ont des schémas identiques ou très similaires. L'crawler créera une seule table avec quatre partitions, avec les clés de partition `year`,`month` et `day`.

Dans l'exemple suivant, observez la structure Amazon S3 suivante :

```
s3://bucket01/folder1/table1/partition1/file.txt
s3://bucket01/folder1/table1/partition2/file.txt
s3://bucket01/folder1/table1/partition3/file.txt
s3://bucket01/folder1/table2/partition4/file.txt
```

```
s3://bucket01/folder1/table2/partition5/file.txt
```

Si les schémas pour les fichiers sous `table1` et `table2` sont similaires, et qu'un seul magasin de données est défini dans l'crawler avec Include path `s3://bucket01/folder1/`, l'crawler crée une seule table avec deux colonnes de clés de partition. La première colonne de clé de partition contient `table1` et `table2`, et la deuxième colonne de clé de partition contient `partition1` à `partition3` pour la partition `table1`, et `partition4` et `partition5` pour la partition `table2`. Pour créer deux tables distinctes, définissez l'crawler avec deux magasins de données. Dans cet exemple, définissez le premier Include path comme `s3://bucket01/folder1/table1/` et le second comme `s3://bucket01/folder1/table2`.

> **Note**
>
> Dans Amazon Athena, chaque table correspond à un préfixe Amazon S3 avec tous les objets qu'il contient. Si les objets ont des schémas différents, Athena ne reconnaît pas les différents objets au sein du même préfixe comme des tables distinctes. Cela peut se produire si un crawler crée plusieurs tables à partir du même préfixe Amazon S3. Il peut en résulter que des requêtes dans Athena ne renvoient aucun résultat. Pour qu'Athena reconnaisse et interroge correctement les tables, créez l'crawler avec un Include path (chemin d'inclusion) distinct pour chaque schéma de table différent dans la structure de dossier Amazon S3. Pour plus d'informations, consultez Bonnes pratiques lors de l'utilisation d'Athena avec AWS Glue et cet article du Centre de connaissances AWS.

Sources de données prises en charge pour l'indexation de site web

Des crawlers peuvent analyser les magasins de données basés sur les fichiers et sur les tables suivants.

Type d'accès utilisé par l'crawler	Magasins de données
Client en mode natif	• Amazon Simple Storage Service (Amazon S3) • Amazon DynamoDB • Delta Lake 2.0.x • Apache Iceberg 1.5 • Apache Hudi 0.14

Type d'accès utilisé par l'crawler	Magasins de données
JDBC	Amazon Redshift Snowflake Dans Amazon Relational Database Service (Amazon RDS) ou externe à Amazon RDS : • Amazon Aurora • MariaDB • Microsoft SQL Server • MySQL • Oracle • PostgreSQL
Client MongoDB	• MongoDB • Atlas MongoDB • Amazon DocumentDB (compatible avec MongoDB)

> **Note**
>
> Actuellement, AWS Glue ne prend pas en charge les crawlers pour les flux de données.

Pour les magasins de données JDBC, MongoDB, MongoDB Atlas et Amazon DocumentDB (compatible avec MongoDB), vous devez spécifier une connexion AWS Glue que le crawler peut utiliser pour se connecter au magasin de données. Pour Amazon S3, vous pouvez éventuellement spécifier une connexion de type Network (Réseau). Une connexion est un objet Data Catalog qui stocke les informations de connexion, telles que les informations d'identification, l'URL, les informations de Virtual Private Cloud Amazon, etc. Pour de plus amples informations, veuillez consulter Connexion aux données.

Voici les versions des pilotes prises en charge par le robot :

Produit (langue française non garantie)	Pilote pris en charge par le robot
PostgreSQL	42.2.1
Amazon Aurora	Identique aux pilotes de robots natifs
MariaDB	8.0.13
Microsoft SQL Server	6.1.0
MySQL	8.0.13
Oracle	11.2.2
Amazon Redshift	4.1
Snowflake	3,13,20
MongoDB	4.7.2
Atlas MongoDB	4.7.2

Voici quelques remarques sur les différents magasins de données.

Amazon S3

Vous pouvez choisir d'explorer un chemin dans votre compte ou dans un autre compte. Si tous les fichiers Amazon S3 d'un dossier ont le même schéma, l'crawler crée une table. De plus, si l'objet Amazon S3 est partitionné, une seule table de métadonnées est créée et les informations de partition sont ajoutées à Data Catalog pour cette table.

Amazon S3 et Amazon DynamoDB

Les robots d'exploration utilisent un rôle Gestion des identités et des accès AWS (IAM) pour obtenir l'autorisation d'accéder à vos magasins de données. Le rôle que vous transmettez à l'crawler doit avoir l'autorisation d'accéder aux chemins Amazon S3 et aux tables Amazon DynamoDB analysés.

Amazon DynamoDB

Lorsque vous définissez un crawler à l'aide de la console AWS Glue, vous spécifiez une table DynamoDB. Si vous utilisez l'API AWS Glue, vous pouvez spécifier une liste des tables. Vous

pouvez choisir d'analyser un petit échantillon des données uniquement pour réduire les temps d'exécution de l'crawler.

Delta Lake

Pour chaque magasin de données Delta Lake, vous précisez comment créer les tables Delta :

- Créer des tables natives : permettre l'intégration avec les moteurs de requêtes qui prennent directement en charge l'interrogation du journal de transactions Delta. Pour plus d'informations, veuillez consulter la rubrique Interrogation des tables Delta Lake.
- Créer des tables avec des liens symboliques : créer un dossier `_symlink_manifest` avec les fichiers manifestes partitionnés par les clés de partition, en fonction des paramètres de configuration spécifiés.

Iceberg

Pour chaque magasin de données Iceberg, vous spécifiez un chemin Amazon S3 qui contient les métadonnées de vos tables Iceberg. Si le Crawler découvre des métadonnées de table Iceberg, il les enregistre dans le catalogue de données. Vous pouvez définir un calendrier pour que le Crawler mette à jour les tables.

Vous pouvez définir les paramètres suivants pour le magasin de données :

- Exclusions : permet d'ignorer certains dossiers.
- Profondeur maximale d'indexation : définit la limite de profondeur que le Crawler peut indexer dans votre compartiment Amazon S3. La profondeur maximale d'indexation par défaut est de 10 et la profondeur maximale que vous pouvez définir est de 20.

Hudi

Pour chaque magasin de données Hudi, vous spécifiez un chemin Amazon S3 qui contient les métadonnées de vos tables Hudi. Si le Crawler découvre des métadonnées de table Hudi, il les enregistre dans le catalogue de données. Vous pouvez définir un calendrier pour que le Crawler mette à jour les tables.

Vous pouvez définir les paramètres suivants pour le magasin de données :

- Exclusions : permet d'ignorer certains dossiers.
- Profondeur maximale d'indexation : définit la limite de profondeur que le Crawler peut indexer dans votre compartiment Amazon S3. La profondeur maximale d'indexation par défaut est de 10 et la profondeur maximale que vous pouvez définir est de 20.

> **Note**
>
> Les colonnes d'horodatage avec `millis` en tant que types logiques seront interprétées comme `bigint`, en raison d'une incompatibilité avec Hudi 0.13.1 et les types d'horodatage. Une solution pourrait être fournie dans la prochaine version de Hudi.

Les tables Hudi sont classées comme suit, avec des implications spécifiques pour chacune d'entre elles :

- Copie sur écriture (CoW) : les données sont stockées dans un format en colonnes (Parquet) et chaque mise à jour crée une version des fichiers lors d'une écriture.
- Fusion sur lecture (MoR) : les données sont stockées à l'aide d'une combinaison des formats en colonnes (Parquet) et basés sur les lignes (Avro). Les mises à jour sont consignées dans des fichiers delta basés sur les lignes et sont compressées si nécessaire pour créer de nouvelles versions des fichiers en colonnes.

Avec les ensembles de données CoW, chaque fois qu'une mise à jour est apportée à un enregistrement, le fichier qui contient l'enregistrement est réécrit avec les valeurs mises à jour. Avec un jeu de données MoR, chaque fois qu'une mise à jour a lieu, Hudi écrit uniquement la ligne du registre modifié. Le type de stockage MoR est mieux adapté aux charges de travail donnant lieu à de nombreuses écritures ou modifications avec moins de lectures. Le type de stockage CoW est mieux adapté aux charges de travail lourdes en lecture sur des données qui changent moins fréquemment.

Hudi propose trois types de requêtes pour accéder aux données :

- Requêtes d'instantané : requêtes qui affichent le dernier instantané de la table à partir d'une action de validation ou de compactage donnée. Pour les tables MoR, les requêtes d'instantané exposent l'état le plus récent de la table en fusionnant les fichiers de base et delta de la dernière tranche de fichiers au moment de la requête.
- Requêtes progressives : les requêtes ne portent que sur les nouvelles données écrites dans la table, depuis une validation/un compactage donné. Cela fournit efficacement des flux de modifications pour activer les pipelines de données (Data Pipelines) progressives.
- Requêtes à lecture optimisée : pour les tables MoR, les requêtes portent sur les dernières données compactées. Pour les tables CoW, les requêtes portent sur les dernières données validées.

Pour Copy-On-Write les tables, les robots créent une seule table dans le catalogue de données avec le ReadOptimized `org.apache.hudi.hadoop.HoodieParquetInputFormat` serde.

Pour Merge-On-Read les tables, le robot crée deux tables dans le catalogue de données pour le même emplacement de table :

- Une table avec un suffixe `_ro` qui utilise le ReadOptimized `org.apache.hudi.hadoop.HoodieParquetInputFormat` serde.
- Une table avec suffixe `_rt` qui utilise le RealTime Serde pour permettre les requêtes Snapshot :. `org.apache.hudi.hadoop.realtime.HoodieParquetRealtimeInputFormat`

MongoDB and Amazon DocumentDB (compatible avec MongoDB)

Les versions 3.2 et ultérieures de MongoDB sont prises en charge. Vous pouvez choisir d'analyser un petit échantillon des données uniquement pour réduire les temps d'exécution de l'crawler.

Base de données relationnelle

L'authentification est effectuée à l'aide d'un nom d'utilisateur et d'un mot de passe de base de données. En fonction du type de moteur de base de données, vous pouvez choisir quels objets sont analysés, comme les bases de données, les schémas et les tables.

Snowflake

Le crawler JDBC Snowflake prend en charge l'analyse de la table, de la table externe, de la vue et de la vue matérialisée. La définition de la vue matérialisée n'est pas renseignée.

Pour les tables externes Snowflake, le crawler n'effectue l'analyse que s'il pointe vers un emplacement Amazon S3. Outre le schéma de la table, le crawler analyse également l'emplacement, le format de fichier et la sortie d'Amazon S3 sous forme de paramètres de table dans la table du catalogue de données. Notez que les informations de partition de la table externe partitionnée ne sont pas renseignées.

Actuellement, ETL n'est pas pris en charge pour les tables du catalogue de données créées à l'aide du crawler Snowflake.

Prérequis pour le crawler

Le robot d'exploration assume les autorisations du rôle Gestion des identités et des accès AWS (IAM) que vous spécifiez lorsque vous le définissez. Ce rôle IAM doit disposer d'autorisations pour

extraire les données de votre magasin de données et écrire sur Data Catalog. La console AWS Glue répertorie uniquement les rôles IAM auxquels une politique d'approbation est attachée pour le service principal AWS Glue. Dans la console, vous pouvez également créer un rôle IAM avec une politique IAM permettant d'accéder aux magasins de données Amazon S3 auxquels l'crawler accède. Pour plus d'informations sur les rôles pour AWS Glue, consultez Politiques basées sur l'identité pour Glue AWS.

> **Note**
>
> Lorsque vous explorez un magasin de données de Delta Lake, vous devez être Read/Write autorisé à accéder à l'emplacement Amazon S3.

Pour votre crawler, vous pouvez créer un rôle et attacher les politiques suivantes :

- La politique `AWSGlueServiceRole` AWS gérée, qui accorde les autorisations requises sur le catalogue de données
- Politique en ligne qui accorde des autorisations sur la source de données.
- Politique en ligne qui accorde l'autorisation `iam:PassRole` sur le rôle.

Si vous préférez une approche plus rapide, vous pouvez laisser l'assistant de l'crawler de la console AWS Glue créer un rôle pour vous. Le rôle qu'il crée est spécifique au robot d'exploration et inclut la politique `AWSGlueServiceRole` AWS gérée ainsi que la politique en ligne requise pour la source de données spécifiée.

Si vous spécifiez un rôle existant pour un crawler, assurez-vous qu'il inclut la politique `AWSGlueServiceRole` ou l'équivalent (ou une version limitée de cette politique), ainsi que les politiques en ligne requises. Par exemple, pour un magasin de données Amazon S3, la politique en ligne serait au minimum la suivante :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
```

```
      "Action": [
        "s3:GetObject"
      ],
      "Resource": [
        "arn:aws:s3:::bucket/object*"
      ]
    }
  ]
}
```

Pour un magasin de données Amazon DynamoDB, la politique serait au minimum la suivante :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "dynamodb:DescribeTable",
        "dynamodb:Scan"
      ],
      "Resource": [
        "arn:aws:dynamodb:us-east-1:111122223333:table/table-name*"
      ]
    }
  ]
}
```

En outre, si le robot lit AWS Key Management Service (AWS KMS) des données Amazon S3 chiffrées, le rôle IAM doit disposer d'une autorisation de déchiffrement sur la clé. AWS KMS Pour de plus amples informations, veuillez consulter Étape 2 : créer un rôle IAM pour AWS Glue.

Définition et gestion des classificateurs

Un classificateur lit les données d'un magasin de données. S'il reconnaît le format des données, il génère un schéma. Le classificateur renvoie également un pourcentage de certitude pour indiquer jusqu'à quel degré la reconnaissance du format était certaine.

AWS Glue fournit un ensemble de classificateurs intégrés, mais vous pouvez également créer classificateurs personnalisés. AWS Glue appelle les classificateurs personnalisés en premier, dans l'ordre que vous spécifiez dans votre définition d'crawler. Selon les résultats renvoyés par les classifieurs personnalisés, AWS Glue peut également appeler les classifieurs intégrés. Si un classificateur renvoie `certainty=1.0` pendant le traitement, cela indique qu'il est sûr à 100 % de pouvoir créer le schéma correct. AWS Glue utilise ensuite la sortie de ce classificateur.

Si aucun classifieur ne renvoie `certainty=1.0`, AWS Glue utilise la sortie du classifieur qui a la certitude la plus élevée. Si aucun classifieur ne renvoie une certitude supérieure à `0.0`, AWS Glue renvoie la chaîne de classification par défaut `UNKNOWN`.

Quand dois-je utiliser un classifieur ?

Vous utilisez les classificateurs lorsque vous analysez un magasin de données pour définir les tables de métadonnées dans l' AWS Glue Data Catalog. Vous pouvez configurer votre crawler avec un ensemble ordonné de classifieurs. Lorsque l'crawler appelle un classificateur, le classificateur détermine si les données sont reconnues. Si le classificateur ne peut pas reconnaître les données ou n'est pas certain à 100 %, l'crawler appelle le prochain classificateur de la liste pour déterminer s'il peut reconnaître les données.

Pour plus d'informations sur la création d'un classificateur à l'aide de la console AWS Glue, consultez Création de classificateurs à l'aide de la console AWS Glue.

Classifieurs personnalisés

La sortie d'un classificateur comprend une chaîne qui indique la classification du fichier ou le format (par exemple, `json`) et le schéma du fichier. Pour les classifieurs personnalisés, vous définissez la logique de création du schéma en fonction du type de classifieur. Les types de classificateur incluent la définition de schémas basés sur les modèles grok, les balises XML et les chemins d'accès JSON.

Si vous modifiez une définition de classificateur, toutes les données précédemment analysées à l'aide du classificateur ne sont pas reclassées. Un crawler garde trace des données précédemment analysées. Les nouvelles données sont classées avec le classifieur mis à jour, ce qui peut entraîner une mise à jour du schéma. Si le schéma de vos données a évolué, mettez à jour le classificateur pour prendre en compte les modifications de schéma lorsque votre crawler s'exécute. Pour reclasser les données et corriger un classificateur incorrect, créez un nouvel crawler avec le classificateur mis à jour.

Pour plus d'informations sur la création de classificateurs personnalisés dans AWS Glue, consultez Rédaction de classificateurs personnalisés pour divers formats de données.

> **Note**
>
> Si votre format de données est reconnu par l'un des classificateurs intégrés, vous n'avez pas besoin de créer un classificateur personnalisé.

Classifieurs intégrés

AWS Glue fournit des classifieurs intégrés pour différents formats, y compris JSON, CSV, les journaux web et de nombreux systèmes de bases de données.

Si AWS Glue ne trouve pas de classificateur personnalisé qui correspond au format des données en entrée avec 100 % de certitude, il fait appel aux classificateurs intégrés dans l'ordre indiqué dans le tableau suivant. Les classifieurs intégrés renvoient un résultat pour indiquer si le format correspond à (`certainty=1.0`) ou ne correspond à (`certainty=0.0`). Le premier classifieur qui a `certainty=1.0` fournit la chaîne de classification et le schéma pour une table de métadonnées de votre Data Catalog.

Type de classificateur	Chaîne de classification	Remarques
Apache Avro	`avro`	Lit le schéma au début du fichier pour déterminer le format.
Apache ORC	`orc`	Lit les métadonnées du fichier pour déterminer le format.
Apache Parquet	`parquet`	Lit le schéma à la fin du fichier pour déterminer le format.
JSON	`json`	Lit le début du fichier pour déterminer le format.
Binaire JSON	`bson`	Lit le début du fichier pour déterminer le format.
xml	`xml`	Lit le début du fichier pour déterminer le format. AWS Glue détermine le schéma de table en fonction des balises XML du document. Pour plus d'informations sur la création d'un classificateur XML personnalisé pour spécifier les

Type de classificateur	Chaîne de classification	Remarques
		lignes du document, reportez-vous à la section Écriture de classifieurs XML personnalisés.
Amazon Ion	`ion`	Lit le début du fichier pour déterminer le format.
Journal Apache combiné	`combined_apache`	Détermine les formats de journaux par le biais d'un modèle grok.
Journal Apache	`apache`	Détermine les formats de journaux par le biais d'un modèle grok.
Journal du noyau Linux	`linux_kernel`	Détermine les formats de journaux par le biais d'un modèle grok.
Journal Microsoft	`microsoft_log`	Détermine les formats de journaux par le biais d'un modèle grok.
Journal Ruby	`ruby_logger`	Lit le début du fichier pour déterminer le format.
Journal Squid 3.x	`squid`	Lit le début du fichier pour déterminer le format.
Journal de surveillance Redis	`redismonlog`	Lit le début du fichier pour déterminer le format.
Journal Redis	`redislog`	Lit le début du fichier pour déterminer le format.
CSV	`csv`	Recherche les séparateurs suivants : virgule (,), barre verticale (|), tabulation (\t), point-virgule (;) et Ctrl-A (\u0001). Ctrl-A est le caractère de contrôle Unicode pour `Start Of Heading`.
Amazon Redshift	`redshift`	Utilise la connexion JDBC pour importer les métadonnées.
MySQL	`mysql`	Utilise la connexion JDBC pour importer les métadonnées.

Type de classificateur	Chaîne de classification	Remarques
PostgreSQL	`postgresql`	Utilise la connexion JDBC pour importer les métadonnées.
Oracle Database	`oracle`	Utilise la connexion JDBC pour importer les métadonnées.
Microsoft SQL Server	`sqlserver`	Utilise la connexion JDBC pour importer les métadonnées.
Amazon DynamoDB	`dynamodb`	Lit les données de la table DynamoDB.

Les fichiers aux formats compressés suivants peuvent être classés :

- ZIP (pris en charge pour les archives contenant uniquement un fichier unique). Notez que Zip n'est pas correctement pris en charge dans d'autres services (en raison de l'archive).
- BZIP
- GZIP
- LZ4
- Snappy (pris en charge pour les formats Snappy standard et natifs Hadoop)

Classifieur CSV intégré

Le classificateur CSV intégré analyse le contenu du fichier CSV pour déterminer le schéma d'une table AWS Glue. Le classifieur vérifie les délimiteurs suivants :

- Virgule (,)
- Pipe (|)
- Tabulation (\t)
- Point-virgule (;)
- Ctrl-A (\u0001)

  Ctrl-A est le caractère de contrôle Unicode pour `Start Of Heading`.

Pour être classé comme CSV, le schéma de table doit avoir au moins deux colonnes et deux lignes de données. Le classifieur CSV utilise un certain nombre de méthodes heuristiques pour déterminer si un en-tête est présent dans un fichier donné. Si le classifieur ne peut pas déterminer un en-tête à partir de la première ligne de données, les en-têtes de colonne sont affichés en tant que `col1`, `col2`, `col3`, et ainsi de suite. Le classificateur CSV intégré détermine s'il convient de déduire un en-tête en évaluant les caractéristiques suivantes du fichier :

- Chaque colonne d'un en-tête potentiel s'analyse en tant que type de données STRING.
- À l'exception de la dernière colonne, chaque colonne d'un en-tête potentiel a un contenu de moins de 150 caractères. Pour autoriser un délimiteur de fin, la dernière colonne peut être vide dans le fichier.
- Chaque colonne d'un en-tête potentiel doit répondre aux exigences AWS Glue `regex` pour un nom de colonne.
- La ligne d'en-tête doit être suffisamment différente des lignes de données. Pour le déterminer, une ou plusieurs lignes doivent s'analyser autrement que de type STRING. Si toutes les colonnes sont de type STRING, la première ligne de données n'est pas suffisamment différente des lignes suivantes pour être utilisée comme en-tête.

> **Note**
>
> Si le classificateur CSV intégré ne crée pas votre table AWS Glue comme vous le souhaitez, vous pouvez utiliser l'une des solutions suivantes :
>
> - Modifiez les noms de colonne du Data Catalog, définissez la structure `SchemaChangePolicy` sur LOG et définissez la configuration de sortie de la partition sur `InheritFromTable` pour les futures exécutions de l'crawler.
> - Créez un classifieur grok personnalisés pour analyser les données et attribuez les colonnes de votre choix.
> - Le classificateur CSV intégré crée les tables en faisant référence à `LazySimpleSerDe` comme bibliothèque de sérialisation, ce qui est un bon choix pour l'inférence du type. Toutefois, si les données CSV contiennent des chaînes entre guillemets, modifiez la définition de la table et remplacez la SerDe bibliothèque par`OpenCSVSerDe`. Ajustez les types déduits sur STRING, définissez la structure `SchemaChangePolicy` sur LOG et définissez la configuration de sortie des partitions sur `InheritFromTable` pour les

futures exécutions de l'crawler. Pour plus d'informations sur SerDe les bibliothèques, consultez la section SerDe Référence du guide de l'utilisateur Amazon Athena.

Rédaction de classificateurs personnalisés pour divers formats de données

Vous pouvez fournir un classifieur personnalisé pour classer vos données dans AWS Glue. Vous pouvez créer un classificateur personnalisé à l'aide d'un modèle grok, d'une balise XML, d'une notation d' JavaScript objet (JSON) ou de valeurs séparées par des virgules (CSV). Un crawler AWS Glue appelle un classifieur personnalisé. Si le classifieur reconnaît les données, il renvoie la classification et le schéma des données à l'crawler. Il se peut que vous ayez besoin de définir un classifieur personnalisé si vos données ne correspondent à aucun classifieur intégré ou si vous souhaitez personnaliser les tables créées par l'crawler.

Pour plus d'informations sur la création d'un classificateur à l'aide de la console AWS Glue, consultez Création de classificateurs à l'aide de la console AWS Glue.

AWS Glue exécute les classifieurs personnalisés avant les classifieurs intégrés, dans l'ordre que vous spécifiez. Lorsqu'un crawler détecte un classifieur qui correspond aux données, la chaîne de classification et le schéma sont utilisés dans la définition des tables qui sont écrites dans votre AWS Glue Data Catalog.

Rubriques

- Écriture de classifieurs personnalisés Grok
- Écriture de classifieurs XML personnalisés
- Écriture de classifieurs JSON personnalisés
- Écriture de classifieurs CSV personnalisés

Écriture de classifieurs personnalisés Grok

Grok est un outil qui est utilisé pour analyser les données textuelles correspondant à un modèle donné. Une modèle grok est un ensemble nommé d'expressions régulières (regex) qui sont utilisées pour mettre en correspondance des données ligne par ligne. AWS Glue utilise les modèles grok pour déduire le schéma de vos données. Lorsqu'un modèle grok correspond à vos données, AWS Glue l'utilise pour déterminer la structure de vos données et la mapper aux champs.

AWS Glue propose de nombreux modèles intégrés ; vous pouvez aussi définir les vôtres. Vous pouvez créer un modèle grok à l'aide de modèles intégrés et de modèles personnalisés de votre

définition de classifieur personnalisée. Vous pouvez personnaliser un modèle grok pour classer les formats de fichier texte personnalisés.

> **Note**
>
> Les classifieurs grok personnalisés AWS Glue utilisent la bibliothèque de sérialisation `GrokSerDe` pour les tables créées dans le AWS Glue Data Catalog. Si vous utilisez le AWS Glue Data Catalog avec Amazon Athena, Amazon EMR ou Redshift Spectrum, consultez la documentation relative à ces services pour obtenir des informations sur le support du. `GrokSerDe` Il est possible que vous rencontriez actuellement des problèmes lors de l'interrogation de tables créées avec le `GrokSerDe` à partir d'Amazon EMR et Redshift Spectrum.

Voici la syntaxe de base pour les composants d'un modèle grok :

```
%{PATTERN:field-name}
```

Les données correspondant au `PATTERN` nommé sont mises en correspondance avec la colonne `field-name` du schéma, avec le type de données par défaut `string`. Le cas échéant, le type de données du champ peut être converti en `byte`, `boolean`, `double`, `short`, `int`, `long`, ou `float` dans le schéma obtenu.

```
%{PATTERN:field-name:data-type}
```

Par exemple, pour convertir un champ `num` en type de données `int`, vous pouvez utiliser ce modèle :

```
%{NUMBER:num:int}
```

Les modèles peuvent être composés d'autres modèles. Par exemple, vous pouvez avoir un modèle pour un horodatage `SYSLOG` qui est défini par des modèles pour le mois, le jour du mois et l'heure (par exemple, `Feb 1 06:25:43`). Pour ces données, vous pourriez définir le schéma suivant:

```
SYSLOGTIMESTAMP %{MONTH} +%{MONTHDAY} %{TIME}
```

> **Note**
>
> Les modèles grok ne peuvent traiter qu'une ligne à la fois. Les modèles à plusieurs lignes ne sont pas pris en charge. De même, les sauts de ligne au sein d'un modèle ne sont pas pris en charge.

Valeurs personnalisées pour le classificateur grok

Lorsque vous définissez un classificateur grok, vous fournissez les valeurs suivantes pour créer le classificeur personnalisé.

Name

Nom du classifieur.

Classification

Chaîne de texte écrite pour décrire le format des données qui est classé ; par exemple, `special-logs`.

Modèle grok

Ensemble des modèles appliqués au magasin de données afin de déterminer s'il y a une correspondance. Ces modèles proviennent des modèles AWS Glue intégrés et de tous les modèles personnalisés que vous définissez.

Voici un exemple de modèle grok :

```
%{TIMESTAMP_ISO8601:timestamp} \[%{MESSAGEPREFIX:message_prefix}\]
 %{CRAWLERLOGLEVEL:loglevel} : %{GREEDYDATA:message}
```

Lorsque les données correspondent à `TIMESTAMP_ISO8601`, une colonne de schéma `timestamp` est créée. Le comportement est similaire aux autres modèles nommés de l'exemple.

Modèles personnalisés

Modèles personnalisés facultatifs que vous définissez. Ces modèles sont référencés par le modèle grok qui classifie vos données. Vous pouvez référencer ces modèles personnalisés dans le modèle grok appliqué à vos données. Chaque modèle de composant personnalisé doit être sur une ligne distincte. La syntaxe des expressions régulières (regex) est utilisée pour définir le modèle.

Voici un exemple d'utilisation des modèles personnalisés :

```
CRAWLERLOGLEVEL (BENCHMARK|ERROR|WARN|INFO|TRACE)
MESSAGEPREFIX .*-.*-.*-.*-.*
```

Le premier modèle nommé personnalisé, `CRAWLERLOGLEVEL`, est une correspondance dans laquelle les données correspondent à l'une des chaînes énumérées. Le second modèle personnalisé, `MESSAGEPREFIX`, essaie de faire correspondre une chaîne de préfixe de message.

AWS Glue garde trace de l'heure de création, de la dernière heure de mise à jour et de la version de votre classifieur.

Modèles intégrés

AWS Glue fournit de nombreux modèles courants que vous pouvez utiliser pour construire un classifieur personnalisé. Vous ajoutez un modèle nommé au `grok pattern` d'une définition de classifieur.

La liste suivante comprend une ligne pour chaque modèle. Dans chaque ligne, le nom du modèle est suivi de sa définition. La syntaxe des expressions régulières (regex) est utilisée pour définir le modèle.

```
#<noloc>&GLU;</noloc> Built-in patterns
 USERNAME [a-zA-Z0-9._-]+
 USER %{USERNAME:UNWANTED}
 INT (?:[+-]?(?:[0-9]+))
 BASE10NUM (?<![0-9.+-])(?>[+-]?(?:(?:[0-9]+(?:\.[0-9]+)?)|(?:\.[0-9]+)))
 NUMBER (?:%{BASE10NUM:UNWANTED})
 BASE16NUM (?<![0-9A-Fa-f])(?:[+-]?(?:0x)?(?:[0-9A-Fa-f]+))
 BASE16FLOAT \b(?<![0-9A-Fa-f.])(?:[+-]?(?:0x)?(?:(?:[0-9A-Fa-f]+(?:\.[0-9A-Fa-f]*)?)|
(?:\.[0-9A-Fa-f]+)))\b
 BOOLEAN (?i)(true|false)

 POSINT \b(?:[1-9][0-9]*)\b
 NONNEGINT \b(?:[0-9]+)\b
 WORD \b\w+\b
 NOTSPACE \S+
 SPACE \s*
 DATA .*?
 GREEDYDATA .*
```

```
 #QUOTEDSTRING (?:(?<!\\)(?:"(?:\\.|[^\\"])*"|(?:'(?:\\.|[^\\'])*')|(?:`(?:\\.|[^\
\`])*`)))
 QUOTEDSTRING (?>(?<!\\)(?>"(?>\\.|[^\\"]+)+"|""|(?>'(?>\\.|[^\\']+)+')|''|(?>`(?>\\.|
[^\\`]+)+`)|``))
 UUID [A-Fa-f0-9]{8}-(?:[A-Fa-f0-9]{4}-){3}[A-Fa-f0-9]{12}

 # Networking
 MAC (?:%{CISCOMAC:UNWANTED}|%{WINDOWSMAC:UNWANTED}|%{COMMONMAC:UNWANTED})
 CISCOMAC (?:(?:[A-Fa-f0-9]{4}\.){2}[A-Fa-f0-9]{4})
 WINDOWSMAC (?:(?:[A-Fa-f0-9]{2}-){5}[A-Fa-f0-9]{2})
 COMMONMAC (?:(?:[A-Fa-f0-9]{2}:){5}[A-Fa-f0-9]{2})
 IPV6 ((([0-9A-Fa-f]{1,4}:){7}([0-9A-Fa-f]{1,4}|:))|(([0-9A-Fa-f]{1,4}:){6}(:[0-9A-
Fa-f]{1,4}|((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d))
{3})|:))|(([0-9A-Fa-f]{1,4}:){5}(((:[0-9A-Fa-f]{1,4}){1,2})|:((25[0-5]|2[0-4]\d|1\d
\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)){3})|:))|(([0-9A-Fa-f]{1,4}:){4}(((:
[0-9A-Fa-f]{1,4}){1,3})|((:[0-9A-Fa-f]{1,4})?:((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.
(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)){3}))|:))|(([0-9A-Fa-f]{1,4}:){3}(((:[0-9A-Fa-f]
{1,4}){1,4})|((:[0-9A-Fa-f]{1,4}){0,2}:((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|
2[0-4]\d|1\d\d|[1-9]?\d)){3}))|:))|(([0-9A-Fa-f]{1,4}:){2}(((:[0-9A-Fa-f]{1,4}){1,5})|
((:[0-9A-Fa-f]{1,4}){0,3}:((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d
\d|[1-9]?\d)){3}))|:))|(([0-9A-Fa-f]{1,4}:){1}(((:[0-9A-Fa-f]{1,4}){1,6})|((:[0-9A-Fa-
f]{1,4}){0,4}:((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d))
{3}))|:))|(:(((:[0-9A-Fa-f]{1,4}){1,7})|((:[0-9A-Fa-f]{1,4}){0,5}:((25[0-5]|2[0-4]\d|
1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)){3}))|:))(%.+)?
 IPV4 (?<![0-9])(?:(?:25[0-5]|2[0-4][0-9]|[0-1]?[0-9]{1,2})[.](?:25[0-5]|2[0-4][0-9]|
[0-1]?[0-9]{1,2})[.](?:25[0-5]|2[0-4][0-9]|[0-1]?[0-9]{1,2})[.](?:25[0-5]|2[0-4][0-9]|
[0-1]?[0-9]{1,2}))(?![0-9])
 IP (?:%{IPV6:UNWANTED}|%{IPV4:UNWANTED})
 HOSTNAME \b(?:[0-9A-Za-z][0-9A-Za-z-_]{0,62})(?:\.(?:[0-9A-Za-z][0-9A-Za-z-_]
{0,62}))*(\.?|\b)
 HOST %{HOSTNAME:UNWANTED}
 IPORHOST (?:%{HOSTNAME:UNWANTED}|%{IP:UNWANTED})
 HOSTPORT (?:%{IPORHOST}:%{POSINT:PORT})

 # paths
 PATH (?:%{UNIXPATH}|%{WINPATH})
 UNIXPATH (?>/(?>[\w_%!$@:.,~-]+|\\.)*)+
 #UNIXPATH (?<![\w\/])(?:/[^\/\s?*]*)+
 TTY (?:/dev/(pts|tty([pq])?)(\w+)?/?(?:[0-9]+))
 WINPATH (?>[A-Za-z]+:|\\)(?:\\[^\\?*]*)+
 URIPROTO [A-Za-z]+(\+[A-Za-z+]+)?
 URIHOST %{IPORHOST}(?::%{POSINT:port})?
 # uripath comes loosely from RFC1738, but mostly from what Firefox
 # doesn't turn into %XX
```

```
 URIPATH (?:/[A-Za-z0-9$.+!*'(){},~:;=@#%_\-]*)+
 #URIPARAM \?(?:[A-Za-z0-9]+(?:=(?:[^&]*))?(?:&(?:[A-Za-z0-9]+(?:=(?:[^&]*))?)?)*)?
 URIPARAM \?[A-Za-z0-9$.+!*'|(){},~@#%&/=:;_?\-\[\]]*
 URIPATHPARAM %{URIPATH}(?:%{URIPARAM})?
 URI %{URIPROTO}://(?:%{USER}(?::[^@]*)?@)?(?:%{URIHOST})?(?:%{URIPATHPARAM})?

 # Months: January, Feb, 3, 03, 12, December
 MONTH \b(?:Jan(?:uary)?|Feb(?:ruary)?|Mar(?:ch)?|Apr(?:il)?|May|Jun(?:e)?|Jul(?:y)?|
Aug(?:ust)?|Sep(?:tember)?|Oct(?:ober)?|Nov(?:ember)?|Dec(?:ember)?)\b
 MONTHNUM (?:0?[1-9]|1[0-2])
 MONTHNUM2 (?:0[1-9]|1[0-2])
 MONTHDAY (?:(?:0[1-9])|(?:[12][0-9])|(?:3[01])|[1-9])

 # Days: Monday, Tue, Thu, etc...
 DAY (?:Mon(?:day)?|Tue(?:sday)?|Wed(?:nesday)?|Thu(?:rsday)?|Fri(?:day)?|
Sat(?:urday)?|Sun(?:day)?)

 # Years?
 YEAR (?>\d\d){1,2}
 # Time: HH:MM:SS
 #TIME \d{2}:\d{2}(?::\d{2}(?:\.\d+)?)?
 # TIME %{POSINT<24}:%{POSINT<60}(?::%{POSINT<60}(?:\.%{POSINT})?)?
 HOUR (?:2[0123]|[01]?[0-9])
 MINUTE (?:[0-5][0-9])
 # '60' is a leap second in most time standards and thus is valid.
 SECOND (?:(?:[0-5]?[0-9]|60)(?:[:.,][0-9]+)?)
 TIME (?!<[0-9])%{HOUR}:%{MINUTE}(?::%{SECOND})(?![0-9])
 # datestamp is YYYY/MM/DD-HH:MM:SS.UUUU (or something like it)
 DATE_US %{MONTHNUM}[/-]%{MONTHDAY}[/-]%{YEAR}
 DATE_EU %{MONTHDAY}[./-]%{MONTHNUM}[./-]%{YEAR}
 DATESTAMP_US %{DATE_US}[- ]%{TIME}
 DATESTAMP_EU %{DATE_EU}[- ]%{TIME}
 ISO8601_TIMEZONE (?:Z|[+-]%{HOUR}(?::?%{MINUTE}))
 ISO8601_SECOND (?:%{SECOND}|60)
 TIMESTAMP_ISO8601 %{YEAR}-%{MONTHNUM}-%{MONTHDAY}[T ]%{HOUR}:?%{MINUTE}(?::?
%{SECOND})?%{ISO8601_TIMEZONE}?
 TZ (?:[PMCE][SD]T|UTC)
 DATESTAMP_RFC822 %{DAY} %{MONTH} %{MONTHDAY} %{YEAR} %{TIME} %{TZ}
 DATESTAMP_RFC2822 %{DAY}, %{MONTHDAY} %{MONTH} %{YEAR} %{TIME} %{ISO8601_TIMEZONE}
 DATESTAMP_OTHER %{DAY} %{MONTH} %{MONTHDAY} %{TIME} %{TZ} %{YEAR}
 DATESTAMP_EVENTLOG %{YEAR}%{MONTHNUM2}%{MONTHDAY}%{HOUR}%{MINUTE}%{SECOND}
 CISCOTIMESTAMP %{MONTH} %{MONTHDAY} %{TIME}

 # Syslog Dates: Month Day HH:MM:SS
```

```
 SYSLOGTIMESTAMP %{MONTH} +%{MONTHDAY} %{TIME}
 PROG (?:[\w._/%-]+)
 SYSLOGPROG %{PROG:program}(?:\[%{POSINT:pid}\])?
 SYSLOGHOST %{IPORHOST}
 SYSLOGFACILITY <%{NONNEGINT:facility}.%{NONNEGINT:priority}>
 HTTPDATE %{MONTHDAY}/%{MONTH}/%{YEAR}:%{TIME} %{INT}

 # Shortcuts
 QS %{QUOTEDSTRING:UNWANTED}

 # Log formats
 SYSLOGBASE %{SYSLOGTIMESTAMP:timestamp} (?:%{SYSLOGFACILITY} )?%{SYSLOGHOST:logsource}
 %{SYSLOGPROG}:

 MESSAGESLOG %{SYSLOGBASE} %{DATA}

 COMMONAPACHELOG %{IPORHOST:clientip} %{USER:ident} %{USER:auth}
 \[%{HTTPDATE:timestamp}\] "(?:%{WORD:verb} %{NOTSPACE:request}(?: HTTP/
%{NUMBER:httpversion})?|%{DATA:rawrequest})" %{NUMBER:response} (?:%{Bytes:bytes=
%{NUMBER}|-})
 COMBINEDAPACHELOG %{COMMONAPACHELOG} %{QS:referrer} %{QS:agent}
 COMMONAPACHELOG_DATATYPED %{IPORHOST:clientip} %{USER:ident;boolean} %{USER:auth}
 \[%{HTTPDATE:timestamp;date;dd/MMM/yyyy:HH:mm:ss Z}\] "(?:%{WORD:verb;string}
 %{NOTSPACE:request}(?: HTTP/%{NUMBER:httpversion;float})?|%{DATA:rawrequest})"
 %{NUMBER:response;int} (?:%{NUMBER:bytes;long}|-)


 # Log Levels
 LOGLEVEL ([A|a]lert|ALERT|[T|t]race|TRACE|[D|d]ebug|DEBUG|[N|n]otice|NOTICE|[I|i]nfo|
INFO|[W|w]arn?(?:ing)?|WARN?(?:ING)?|[E|e]rr?(?:or)?|ERR?(?:OR)?|[C|c]rit?(?:ical)?|
CRIT?(?:ICAL)?|[F|f]atal|FATAL|[S|s]evere|SEVERE|EMERG(?:ENCY)?|[Ee]merg(?:ency)?)
```

Écriture de classifieurs XML personnalisés

Le langage XML définit la structure d'un document avec l'utilisation de balises dans le fichier. Avec un classifieur XML personnalisé, vous pouvez spécifier le nom de la balise utilisé pour définir une ligne.

Valeurs de classificateur personnalisées pour un classificateur XML

Lorsque vous définissez un classifieur XML, vous fournissez les valeurs suivantes à AWS Glue pour créer le classifieur. Le champ de classification du classifieur est défini sur `xml`.

Name

Nom du classifieur.

Balise de ligne

Nom de balise XML qui définit une ligne de table du document XML, sans crochets < >. Ce nom doit respecter les règles XML relatives aux balises.

> **Note**
>
> L'élément contenant les données de ligne ne peut pas être un élément auto-fermant vide. Par exemple, cet élément vide n'est pas analysé par AWS Glue :
>
> ```
> <row att1=”xx” att2=”yy” />
> ```
>
> Les éléments vides peuvent être écrits comme suit :
>
> ```
> <row att1=”xx” att2=”yy”> </row>
> ```

AWS Glue garde trace de l'heure de création, de la dernière heure de mise à jour et de la version de votre classifieur.

Par exemple, supposons que vous disposez du fichier XML suivant. Pour créer une table AWS Glue qui contient uniquement les colonnes pour l'auteur et le titre, créez un classifieur dans la console AWS Glue avec une balise de ligne ayant comme valeur `AnyCompany`. Ensuite, ajoutez et exécutez un crawler qui utilise ce classifieur personnalisé.

```
<?xml version="1.0"?>
<catalog>
   <book id="bk101">
     <AnyCompany>
       <author>Rivera, Martha</author>
       <title>AnyCompany Developer Guide</title>
```

```
      </AnyCompany>
    </book>
    <book id="bk102">
      <AnyCompany>
        <author>Stiles, John</author>
        <title>Style Guide for AnyCompany</title>
      </AnyCompany>
    </book>
</catalog>
```

Écriture de classifieurs JSON personnalisés

JSON est un format d'échange de données. Il définit les structures de données avec des paires nom-valeur ou une liste ordonnée de valeurs. Grâce à un classifieur JSON personnalisé, vous pouvez spécifier le chemin d'accès JSON vers une structure de données utilisée pour définir le schéma de votre table.

Valeurs de classifieur personnalisé dans AWS Glue

Lorsque vous définissez un classifieur JSON, vous fournissez les valeurs suivantes à AWS Glue pour créer le classifieur. Le champ de classification du classifieur est défini sur `json`.

Name

Nom du classifieur.

Chemin JSON

Chemin d'accès JSON qui pointe vers un objet utilisé pour définir un schéma de table. Le chemin JSON peut être écrit en notation point ou en notation crochet. Les opérateurs suivants sont pris en charge:

Description
Élément racine d'un objet JSON. Toutes les expressions de chemin commencent par cet élément.
Caractère générique. Disponible partout où un nom ou une valeur numérique sont obligatoires dans le chemin JSON.
Enfant à notation point. Spécifie un champ enfant dans un objet JSON.

Description
Enfant à notation crochet. Spécifie un champ enfant dans un objet JSON. Un seul champ enfant peut être spécifié.
Index de tableau. Spécifie la valeur d'un tableau par son index.

AWS Glue garde trace de l'heure de création, de la dernière heure de mise à jour et de la version de votre classifieur.

Example Utilisation d'un classifieur JSON pour extraire les enregistrements d'un tableau

Supposons que vos données JSON soient un tableau d'enregistrements. Par exemple, les premières lignes de votre fichier peuvent se présenter ainsi :

```
[
  {
    "type": "constituency",
    "id": "ocd-division\/country:us\/state:ak",
    "name": "Alaska"
  },
  {
    "type": "constituency",
    "id": "ocd-division\/country:us\/state:al\/cd:1",
    "name": "Alabama's 1st congressional district"
  },
  {
    "type": "constituency",
    "id": "ocd-division\/country:us\/state:al\/cd:2",
    "name": "Alabama's 2nd congressional district"
  },
  {
    "type": "constituency",
    "id": "ocd-division\/country:us\/state:al\/cd:3",
    "name": "Alabama's 3rd congressional district"
  },
  {
    "type": "constituency",
    "id": "ocd-division\/country:us\/state:al\/cd:4",
```

```
    "name": "Alabama's 4th congressional district"
  },
  {
    "type": "constituency",
    "id": "ocd-division\/country:us\/state:al\/cd:5",
    "name": "Alabama's 5th congressional district"
  },
  {
    "type": "constituency",
    "id": "ocd-division\/country:us\/state:al\/cd:6",
    "name": "Alabama's 6th congressional district"
  },
  {
    "type": "constituency",
    "id": "ocd-division\/country:us\/state:al\/cd:7",
    "name": "Alabama's 7th congressional district"
  },
  {
    "type": "constituency",
    "id": "ocd-division\/country:us\/state:ar\/cd:1",
    "name": "Arkansas's 1st congressional district"
  },
  {
    "type": "constituency",
    "id": "ocd-division\/country:us\/state:ar\/cd:2",
    "name": "Arkansas's 2nd congressional district"
  },
  {
    "type": "constituency",
    "id": "ocd-division\/country:us\/state:ar\/cd:3",
    "name": "Arkansas's 3rd congressional district"
  },
  {
    "type": "constituency",
    "id": "ocd-division\/country:us\/state:ar\/cd:4",
    "name": "Arkansas's 4th congressional district"
  }
]
```

Lorsque vous exécutez un crawler à l'aide du classifieur JSON intégré, le fichier entier est utilisé pour définir le schéma. Comme vous ne spécifiez pas de chemin JSON, l'crawler traite les données comme un objet, en l'occurrence un simple tableau. Par exemple, le schéma pourrait s'apparenter à ce qui suit :

```
root
|-- record: array
```

Cependant, pour créer un schéma basé sur chaque enregistrement du tableau JSON, créez un classifieur JSON personnalisé et spécifiez le chemin JSON comme `$[*]`. Lorsque vous spécifiez ce chemin JSON, le classifieur interroge tous les 12 enregistrements du tableau pour déterminer le schéma. Le schéma obtenu contient des champs séparés pour chaque objet, comme dans l'exemple qui suit :

```
root
|-- type: string
|-- id: string
|-- name: string
```

Example Utilisation d'un classifieur JSON pour n'examiner que certaines parties d'un fichier

Supposons que vos données JSON suivent le modèle de l'exemple de fichier JSON `s3://awsglue-datasets/examples/us-legislators/all/areas.json` extrait de http://everypolitician.org/. Les exemples d'objets dans le fichier JSON se présentent comme suit :

```
{
  "type": "constituency",
  "id": "ocd-division\/country:us\/state:ak",
  "name": "Alaska"
}
{
  "type": "constituency",
  "identifiers": [
    {
      "scheme": "dmoz",
      "identifier": "Regional\/North_America\/United_States\/Alaska\/"
    },
    {
      "scheme": "freebase",
      "identifier": "\/m\/0hjy"
    },
    {
      "scheme": "fips",
```

```
      "identifier": "US02"
    },
    {
      "scheme": "quora",
      "identifier": "Alaska-state"
    },
    {
      "scheme": "britannica",
      "identifier": "place\/Alaska"
    },
    {
      "scheme": "wikidata",
      "identifier": "Q797"
    }
  ],
  "other_names": [
    {
      "lang": "en",
      "note": "multilingual",
      "name": "Alaska"
    },
    {
      "lang": "fr",
      "note": "multilingual",
      "name": "Alaska"
    },
    {
      "lang": "nov",
      "note": "multilingual",
      "name": "Alaska"
    }
  ],
  "id": "ocd-division\/country:us\/state:ak",
  "name": "Alaska"
}
```

Lorsque vous exécutez un crawler à l'aide du classifieur JSON intégré, le fichier entier est utilisé pour créer le schéma. Vous pouvez vous retrouver avec un schéma tel que celui-ci :

```
root
|-- type: string
```

```
|-- id: string
|-- name: string
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- lang: string
|    |    |-- note: string
|    |    |-- name: string
```

Cependant, pour créer un schéma en n'utilisant que l'objet « `id` », créez un classifieur JSON personnalisé et spécifiez le chemin JSON comme `$.id`. Le schéma est donc basé uniquement sur le champ « `id` » :

```
root
|-- record: string
```

Les quelques premières lignes de données extraites avec ce schéma se présentent ainsi :

```
{"record": "ocd-division/country:us/state:ak"}
{"record": "ocd-division/country:us/state:al/cd:1"}
{"record": "ocd-division/country:us/state:al/cd:2"}
{"record": "ocd-division/country:us/state:al/cd:3"}
{"record": "ocd-division/country:us/state:al/cd:4"}
{"record": "ocd-division/country:us/state:al/cd:5"}
{"record": "ocd-division/country:us/state:al/cd:6"}
{"record": "ocd-division/country:us/state:al/cd:7"}
{"record": "ocd-division/country:us/state:ar/cd:1"}
{"record": "ocd-division/country:us/state:ar/cd:2"}
{"record": "ocd-division/country:us/state:ar/cd:3"}
{"record": "ocd-division/country:us/state:ar/cd:4"}
{"record": "ocd-division/country:us/state:as"}
{"record": "ocd-division/country:us/state:az/cd:1"}
{"record": "ocd-division/country:us/state:az/cd:2"}
{"record": "ocd-division/country:us/state:az/cd:3"}
{"record": "ocd-division/country:us/state:az/cd:4"}
{"record": "ocd-division/country:us/state:az/cd:5"}
{"record": "ocd-division/country:us/state:az/cd:6"}
```

```
{"record": "ocd-division/country:us/state:az/cd:7"}
```

Pour créer un schéma basé sur un objet profondément imbriqué, comme « `identifier` », dans le fichier JSON, vous pouvez créer un classifieur JSON personnalisé et spécifier le chemin JSON comme ainsi : `$.identifiers[*].identifier`. Bien que le schéma soit similaire à l'exemple précédent, il repose sur un autre objet du fichier JSON.

Le schéma se présente comme suit :

```
root
|-- record: string
```

L'affichage des quelques premières lignes de données de la table montre que le schéma est basé sur les données de l'objet « `identifier` » :

```
{"record": "Regional/North_America/United_States/Alaska/"}
{"record": "/m/0hjy"}
{"record": "US02"}
{"record": "5879092"}
{"record": "4001016-8"}
{"record": "destination/alaska"}
{"record": "1116270"}
{"record": "139487266"}
{"record": "n79018447"}
{"record": "01490999-8dec-4129-8254-eef6e80fadc3"}
{"record": "Alaska-state"}
{"record": "place/Alaska"}
{"record": "Q797"}
{"record": "Regional/North_America/United_States/Alabama/"}
{"record": "/m/0gyh"}
{"record": "US01"}
{"record": "4829764"}
{"record": "4084839-5"}
{"record": "161950"}
{"record": "131885589"}
```

Pour créer une table basée sur un autre objet profondément imbriqué, tel que le champ « `name` » du tableau « `other_names` » du fichier JSON, vous pouvez créer un classifieur JSON personnalisé et spécifier le chemin JSON ainsi : `$.other_names[*].name`. Bien que le schéma soit similaire

à l'exemple précédent, il repose sur un autre objet du fichier JSON. Le schéma se présente comme suit :

```
root
|-- record: string
```

L'affichage des quelques premières lignes de données de la table montre qu'il est basé sur les données de l'objet « `name` » du tableau « `other_names` » :

```
{"record": "Alaska"}
{"record": "Alaska"}
{"record": "Аляска"}
{"record": "Alaska"}
{"record": "Alaska"}
{"record": "Alaska"}
{"record": "Alaska"}
{"record": "Alaska"}
{"record": "Alaska"}
{"record": "######"}
{"record": "######"}
{"record": "######"}
{"record": "Alaska"}
{"record": "Alyaska"}
{"record": "Alaska"}
{"record": "Alaska"}
{"record": "Штат Аляска"}
{"record": "Аляска"}
{"record": "Alaska"}
{"record": "#######"}
```

Écriture de classifieurs CSV personnalisés

Les classifieurs CSV personnalisés vous permettent de spécifier des types de données pour chaque colonne dans le champ de classifieur CSV personnalisé. Vous pouvez spécifier le type de données de chaque colonne en les séparant par des virgules. En spécifiant les types de données, vous pouvez remplacer les types de données déduits par les Crawlers et vous assurer que les données seront classées de manière appropriée.

Vous pouvez définir le CSV SerDe pour le traitement dans le classificateur, qui sera appliqué dans le catalogue de données.

Lorsque vous créez un classifieur personnalisé, vous pouvez également le réutiliser pour différents Crawlers.

- Les fichiers CSV contenant uniquement des en-têtes (pas de données) sont classés comme UNKNOWN (INCONNUS), car les informations fournies ne sont pas suffisantes. Si vous spécifiez « Has headings » (Comporte des en-têtes) pour le fichier CSV dans l'option Column headings (En-têtes de colonne) et que vous fournissez les types de données, nous pouvons classer ces fichiers correctement.

Vous pouvez utiliser un classifieur CSV personnalisé pour déduire le schéma de différents types de données CSV. Les attributs personnalisés que vous pouvez fournir à votre classificateur incluent des délimiteurs, une SerDe option CSV, des options concernant l'en-tête et l'opportunité d'effectuer certaines validations sur les données.

Valeurs de classifieur personnalisé dans AWS Glue

Lorsque vous définissez un classifieur CSV, vous fournissez les valeurs suivantes à AWS Glue pour créer le classifieur. Le champ de classification du classifieur est défini sur `csv`.

Nom du classifieur

Nom du classifieur.

SerDe CSV

Définit le CSV SerDe pour le traitement dans le classificateur, qui sera appliqué dans le catalogue de données. Les options sont Open CSV SerDe, Lazy Simple SerDe et None. Vous pouvez spécifier la valeur None lorsque vous souhaitez que le Crawler effectue la détection.

Délimiteur de colonne

Symbole personnalisé pour indiquer ce qui sépare chaque entrée de colonne dans la ligne. Saisissez un caractère Unicode. Si vous ne parvenez pas à saisir votre délimiteur, vous pouvez le coller. Cela fonctionne pour les caractères imprimables, y compris ceux que votre système ne prend pas en charge (généralement affichés comme □).

Symbole de guillemets

Symbole personnalisé pour indiquer ce qui combine le contenu en une seule valeur de colonne. Doit être différent du délimiteur de colonne. Saisissez un caractère Unicode. Si vous ne parvenez pas à saisir votre délimiteur, vous pouvez le coller. Cela fonctionne pour les caractères

imprimables, y compris ceux que votre système ne prend pas en charge (généralement affichés comme □).

En-têtes de colonnes

Indique le comportement à suivre pour détecter les en-têtes de colonnes dans le fichier CSV. Si votre fichier CSV personnalisé a des en-têtes de colonnes, entrez une liste séparée par des virgules de ces en-têtes de colonnes.

Options de traitement : Autoriser les fichiers avec une seule colonne

Active le traitement des fichiers qui ne contiennent qu'une seule colonne.

Options de traitement : Supprimer l'espace blanc avant d'identifier les valeurs de colonne

Spécifie s'il convient de couper les valeurs avant d'identifier le type des valeurs de colonne.

Types de données personnalisés – facultatif

Saisissez le type de données personnalisé en le séparant par une virgule. Spécifie les types de données personnalisés dans le fichier CSV. Le type de données personnalisé doit être un type de données pris en charge. Les types de données pris en charge sont les suivants : « BINARY », « BOOLEAN », « DATE », « DECIMAL », « DOUBLE », « FLOAT », « INT », « LONG », « SHORT », « STRING », « TIMESTAMP ». Les types de données non pris en charge affichent une erreur.

Création de classificateurs à l’aide de la console AWS Glue

Un classifieur détermine le schéma de vos données. Vous pouvez écrire un classificateur personnalisé et pointer dessus à partir d’ AWS Glue.

Création de classifieurs

Pour ajouter un classifieur dans la console AWS Glue, choisissez Add classifier (Ajouter un classifieur). Lorsque vous définissez un classifieur, vous fournissez des valeurs pour les éléments suivants :

- Classifier name (Nom du classifieur) – indiquez un nom unique pour votre classifieur.
- Classifier type (Type de classifieur) – type de classification des tables déduites par ce classifieur.
- Last updated (Dernière mise à jour) – heure de la dernière mise à jour de ce classifieur.

Nom du classifieur

Indiquez un nom unique pour votre classifieur.

Type de classificateur

Choisissez le type de classifieur à créer.

Selon le type de classificateur que vous choisissez, configurez les propriétés suivantes pour votre classificateur :

Grok

- Classement

  Décrivez le format ou le type des données classées ou fournissez une étiquette personnalisée.

- Modèle grok

  Cela est utilisé pour analyser vos données dans un schéma structuré. Le modèle grok est composé de modèles nommés qui décrivent le format de votre magasin de données. Vous écrivez ce modèle grok à l'aide des modèles intégrés nommés fournis par AWS Glue et des modèles personnalisés que vous avez écrits et inclus dans le champ Modèles personnalisés. Même si les résultats du débogueur grok peuvent ne pas correspondre exactement à ceux d'AWS Glue, nous vous suggérons d'essayer votre modèle en utilisant des exemples de données avec un débogueur grok. Vous pouvez trouver des débogueurs grok sur le Web. Les modèles intégrés nommés fournis par AWS Glue sont généralement compatibles avec les modèles grok disponibles sur le Web.

  Créez votre modèle grok en ajoutant de manière itérative des modèles nommés et vérifiez vos résultats dans un débogueur. Cela permet de vous assurer que vos données peuvent être analysées lorsque l'crawler AWS Glue exécute votre modèle grok.

- Modèles personnalisés

  Pour les classificateurs grok, il s'agit de blocs de construction facultatifs pour le Grok pattern (Modèle grok) que vous écrivez. Lorsque les modèles intégrés ne peuvent pas analyser vos données, vous pouvez avoir besoin d'écrire un modèle personnalisé. Ces modèles personnalisés sont définis dans ce champ et référencés dans le champ Grok pattern (Modèle grok). Chaque modèle personnalisé est défini sur une ligne distincte. À l'image d'un modèle

intégré, il se compose d'une définition de modèle nommé qui utilise une syntaxe d'expression régulière (regex).

L'exemple suivant utilise le nom `MESSAGEPREFIX`, suivi d'une définition d'expression régulière à appliquer à vos données afin de déterminer si elles suivent le modèle.

```
MESSAGEPREFIX .*-.*-.*-.*-.*
```

XML

- Balise de ligne

  Pour les classificateurs XML, il s'agit du nom de la balise XML qui définit une ligne de table dans le document XML. Tapez le nom sans crochets < >. Ce nom doit respecter les règles XML relatives aux balises.

  Pour de plus amples informations, veuillez consulter Écriture de classifieurs XML personnalisés.

JSON

- Chemin JSON

  Pour les classificateurs JSON, il s'agit du chemin d'accès JSON à l'objet, au tableau ou à la valeur qui définit une ligne de la table en cours de création. Tapez le nom en utilisant les opérateurs pris en charge par AWS Glue. Veillez à respecter la syntaxe JSON d'accolades ou de points.

  Pour en savoir plus, consultez la liste des opérateurs dans Écriture de classifieurs JSON personnalisés.

CSV

- Délimiteur de colonne

  Caractère ou symbole unique pour indiquer ce qui sépare chaque entrée de colonne dans la ligne. Choisissez le délimiteur dans la liste ou sélectionnez `Other` pour saisir un délimiteur personnalisé.

- Symbole de guillemets

  Caractère ou symbole unique pour indiquer ce qui combine le contenu en une seule valeur de colonne. Doit être différent du délimiteur de colonne. Choisissez le symbole de guillemet dans la liste ou sélectionnez `Other` pour saisir un caractère de guillemet personnalisé.

- En-têtes de colonnes

  Indique le comportement à suivre pour détecter les en-têtes de colonnes dans le fichier CSV. Vous pouvez choisir `Has headings`, `No headings` ou `Detect headings`. Si votre fichier CSV personnalisé a des en-têtes de colonnes, entrez une liste séparée par des virgules de ces en-têtes de colonnes.

- Autoriser les fichiers avec une seule colonne

  Pour être classé comme CSV, les données doivent avoir au moins deux colonnes et deux lignes de données. Utilisez cette option pour autoriser le traitement des fichiers qui ne contiennent qu'une seule colonne.

- Supprimer les espaces avant d'identifier les valeurs de colonne

  Cette option spécifie s'il convient de couper les valeurs avant d'identifier le type des valeurs de colonne.

- Type de données personnalisé

  (Facultatif) – Saisissez des types de données personnalisés dans une liste délimitée par des virgules. Les types de données pris en charge sont les suivants : « BINARY », « BOOLEAN », « DATE », « DECIMAL », « DOUBLE », « FLOAT », « INT », « LONG », « SHORT », « STRING », « TIMESTAMP ».

- SerDe CSV

  (Facultatif) - A SerDe pour le traitement du CSV dans le classificateur, qui sera appliqué dans le catalogue de données. Choisissez `Open CSV SerDe`, `Lazy Simple SerDe` ou `None`. Vous pouvez spécifier la valeur `None` lorsque vous souhaitez que le Crawler effectue la détection.

Pour de plus amples informations, veuillez consulter Rédaction de classificateurs personnalisés pour divers formats de données.

Affichage des classifieurs

Pour afficher la liste de tous les classifieurs que vous avez créés, ouvrez la console AWS Glue à l'adresse https://console.aws.amazon.com/glue/ et choisissez l'onglet Classifieurs.

La liste affiche les propriétés suivantes sur chaque classifieur :

- Classifier (Classifieurs) – nom du classifieur. Lorsque vous créez un classifieur, vous devez indiquer un nom pour celui-ci.
- Classification – type de classification des tables déduites par ce classifieur.
- Last updated (Dernière mise à jour) – heure de la dernière mise à jour de ce classifieur.

Gestion des classifieurs

À partir de la liste Classifieurs de la console AWS Glue, vous pouvez ajouter, modifier ou supprimer des classifieurs. Pour afficher plus de détails sur un classifieur, choisissez le nom du classifieur dans la liste. Les détails incluent les informations que vous avez définies lors de la création du classifieur.

Configuration d'un robot

Un robot accède à votre entrepôt de données, identifie des métadonnées et crée des définitions de table dans l' AWS Glue Data Catalog. Le volet Crawlers de la AWS Glue console répertorie tous les robots d'exploration que vous créez. La liste affiche le statut et les métriques de la dernière exécution de votre crawler.

Cette rubrique décrit le step-by-step processus de configuration d'un robot d'exploration, en abordant des aspects essentiels tels que la configuration des paramètres du robot, la définition des sources de données à explorer, le paramétrage de la sécurité et la gestion des données analysées.

Rubriques

- Étape 1 : Configurer les propriétés du crawler
- Étape 2 : Choisir des sources de données et des classificateurs.
- Étape 3 : Configurer les paramètres de sécurité
- Étape 4 : Configurer la sortie et la planification
- Étape 5 : vérification et création

Étape 1 : Configurer les propriétés du crawler

Pour configurer un robot

1. Connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/. Sélectionnez Crawlers dans le panneau de navigation.
2. Cliquez sur Créer un Crawler, puis suivez les instructions de l'assistant Ajouter un Crawler. L'assistant vous guidera à travers les étapes nécessaires à la création d'un robot. Si vous souhaitez ajouter des classifieurs personnalisés pour définir le schéma, consultez Définition et gestion des classificateurs.
3. Saisissez un nom pour votre Crawler et une description (facultatif). Le cas échéant, vous pouvez baliser votre crawler avec une clé de balise et avec une valeur de balise facultative. Une fois créées, les clés de balise sont en lecture seule. Utilisez des identifications sur certaines ressources pour mieux les organiser et les identifier. Pour plus d'informations, consultez la section AWS tags in AWS Glue.

   Name

   Le nom peut contenir des lettres (A à Z), des chiffres (0 à 9), des tirets (-) ou des traits de soulignement (_) et jusqu'à 255 caractères.

   Description

   Les descriptions peuvent comporter jusqu'à 2 048 caractères.

   Étiquettes

   Utilisez des balises pour identifier et organiser vos ressources. Pour plus d'informations, consultez les ressources suivantes :

   - AWS tags dans AWS Glue

Étape 2 : Choisir des sources de données et des classificateurs.

Configurez ensuite les sources de données et les classificateurs pour le robot.

Pour plus d'informations sur les sources de données prises en charge, consultez Sources de données prises en charge pour l'indexation de site web.

Configuration de source de données

Sélectionnez l'option appropriée pour Vos données sont-elles déjà mappées à des tables AWS Glue ? et choisissez « Pas encore » ou « Oui ». Par défaut, Pas encore est déjà sélectionné.

L'crawler peut accéder aux magasins de données directement en tant que source de l'analyse, ou il peut utiliser des tables de Data Catalog existantes en tant que source. Si l'crawler utilise des tables de catalogue existantes, il analyse les magasins de données qui sont spécifiés par ces tables de catalogue.

- Pas encore : Sélectionnez une ou plusieurs sources de données à analyser. Un crawler peut analyser plusieurs magasins de données de différents types (Amazon S3, JDBC, etc.).

  Vous ne pouvez configurer qu'un seul magasin de données à la fois. Après avoir fourni les informations de connexion et inclus les chemins et les modèles d'exclusion, vous avez alors la possibilité d'ajouter un autre magasin de données.
- Oui : Sélectionnez les tables existantes dans votre catalogue de données AWS Glue. Les tables de catalogue spécifient les magasins de données à analyser. L'crawler peut uniquement analyser des tables de catalogue en une seule exécution ; il ne peut pas combiner dans d'autres types de source.

  Généralement, pour spécifier une table de catalogue en tant que source, c'est que vous avez créé la table manuellement (parce que vous connaissiez déjà la structure du magasin de données) et vous voulez qu'un crawler garde la table à jour, y compris ajouter de nouvelles partitions. Pour une discussion sur d'autres raisons, consultez Mise à jour de tables Data Catalog créées manuellement à l'aide d'crawlers.

  Lorsque vous spécifiez des tables existantes en tant que type source d'crawler, les conditions suivantes s'appliquent :

  - Le nom de la base de données est facultatif.
  - Seules les tables de catalogue qui spécifient les entrepôts de données Amazon S3, Amazon DynamoDB ou Delta Lake sont autorisées.
  - Aucune nouvelle table de catalogue n'est créée lorsque l'crawler s'exécute. Les tables sont mises à jour selon les besoins, y compris en ajoutant de nouvelles partitions.
  - Les objets supprimés trouvés dans les magasins de données sont ignorés ; aucune table de catalogue n'est supprimée. Au lieu de cela, l'crawler rédige un message de journal. (`SchemaChangePolicy.DeleteBehavior=LOG`)

- L'option de configuration de l'crawler pour créer un seul schéma pour chaque chemin Amazon S3 est activée par défaut et ne peut pas être désactivée. (`TableGroupingPolicy=CombineCompatibleSchemas`) Pour plus d'informations, consultez Création d'un schéma unique pour chaque chemin d'inclusion Amazon S3.
- Vous ne pouvez pas combiner des tables de catalogue en tant que source avec d'autres types de source (par exemple, Amazon S3 ou Amazon DynamoDB).

Pour utiliser les tables Delta, créez d'abord une table Delta à l'aide d'Athena DDL ou de l'API. AWS Glue

À l'aide d'Athena, définissez l'emplacement de votre dossier Amazon S3 et le type de table sur « DELTA ».

```
CREATE EXTERNAL TABLE database_name.table_name
LOCATION 's3://bucket/folder/'
TBLPROPERTIES ('table_type' = 'DELTA')
```

À l'aide de l' AWS Glue API, spécifiez le type de table dans la carte des paramètres de table. Les paramètres de la table doivent inclure la key/value paire suivante. Pour plus d'informations sur la création d'une table, consultez la documentation Boto3 pour create_table.

```
{
    "table_type":"delta"
}
```

Sources de données

Sélectionnez ou ajoutez la liste des sources de données à analyser par le crawler.

(Facultatif) Si vous choisissez JDBC comme source de données, vous pouvez utiliser vos propres pilotes JDBC lorsque vous spécifiez l'accès à la connexion où les informations du pilote sont stockées.

Chemin à inclure

Lors de l'évaluation des éléments à inclure ou exclure dans une analyse, un crawler commence par évaluer le chemin d'inclusion requis. Pour les magasins de données Amazon S3, MongoDB, MongoDB Atlas, Amazon DocumentDB (compatible avec MongoDB) et les magasins de données relationnelles, vous devez spécifier un chemin d'inclusion.

Pour un magasin de données Amazon S3

Déterminez si vous souhaitez spécifier un chemin dans ce compte ou dans un autre compte, puis parcourez les chemins Amazon S3 pour en choisir un.

Pour les magasins de données Amazon S3, la syntaxe de chemin d'accès inclut est `bucket-name/folder-name/file-name.ext`. Pour analyser tous les objets d'un compartiment, spécifiez juste le nom du compartiment dans le chemin d'inclusion. Le modèle d'exclusion est relatif au chemin d'inclusion

Pour un magasin de données Delta Lake

Spécifiez un ou plusieurs chemins Amazon S3 vers les tables Delta sous la forme s3 ://*bucket*/*prefix*/*object*.

Pour un magasin de données Iceberg ou Hudi

Spécifiez un ou plusieurs chemins Amazon S3 contenant des dossiers contenant des métadonnées de table Iceberg ou Hudi sous la forme s3 :///*bucket*. *prefix*

Pour les banques de données Iceberg et Hudi, le Iceberg/Hudi dossier peut se trouver dans un dossier enfant du dossier racine. Le Crawler examine tous les dossiers situés sous un chemin à la recherche d'un dossier Hudi.

Pour un magasin de données JDBC

Entrez*<database>*/*<schema>*/*<table>*ou*<database>*/*<table>*, selon le produit de base de données. Oracle Database et MySQL ne prennent pas en charge le schéma dans le chemin. Vous pouvez remplacer le pourcentage (%) par *<schema>* ou*<table>*. Par exemple, pour une base de données Oracle avec l'identificateur système (SID) `orcl`, entrez `orcl/%` pour importer toutes les tables auxquelles l'utilisateur nommé dans la connexion a accès.

> **Important**
>
> Ce champ est sensible à la casse.

> **Note**
>
> Si vous choisissez d'utiliser vos propres versions de pilotes JDBC, les Crawlers AWS Glue consommeront des ressources dans les tâches AWS Glue et les compartiments Amazon S3 pour s'assurer que les pilotes que vous avez fournis sont exécutés dans

votre environnement. L'utilisation supplémentaire des ressources sera reflétée sur votre compte. Les pilotes sont limités aux propriétés décrites dans Adding an AWS Glue connection.

Pour un magasin de données MongoDB, MongoDB Atlas ou Amazon DocumentDB

Pour MongoDB, MongoDB Atlas et Amazon DocumentDB (compatible avec MongoDB), la syntaxe est `database/collection`.

Pour les magasins de données JDBC, la syntaxe est `database-name/schema-name/table-name` ou `database-name/table-name`. La syntaxe varie selon que le moteur de base de données prend en charge ou pas les schémas dans une base de données. Par exemple, pour les moteurs de base de données tels que MySQL ou Oracle, ne spécifiez pas `schema-name` dans votre chemin d'inclusion. Vous pouvez remplacer le signe pourcentage (`%`) d'un schéma ou d'une table dans le chemin d'inclusion pour représenter tous les schémas ou toutes les tables d'une base de données. Vous ne pouvez pas remplacer le signe pourcentage (`%`) pour la base de données dans le chemin d'inclusion.

Profondeur maximale d'indexation (pour les magasins de données Iceberg ou Hudi uniquement)

Définit la profondeur maximale du chemin Amazon S3 que le Crawler peut parcourir pour découvrir le dossier de métadonnées Iceberg ou Hudi dans votre chemin Amazon S3. L'objectif de ce paramètre est de limiter la durée d'exécution du Crawler. La valeur par défaut est 10 et la valeur maximale est 20.

Modèles d'exclusion

Ils vous permettent d'exclure certains fichiers ou tables de l'analyse. Le chemin à exclure est relatif par rapport au chemin à inclure. Par exemple, pour exclure une table de votre magasin de données JDBC, tapez le nom de la table dans le chemin d'exclusion.

Un crawler se connecte à un magasin de données JDBC à l'aide d'une connexion AWS Glue qui contient une chaîne de connexion d'URI JDBC. L'crawler a uniquement accès aux objets du moteur de base de données à l'aide du nom d'utilisateur et du mot de passe JDBC de la connexion AWS Glue. L'crawler peut uniquement créer des tables auxquelles il peut accéder via la connexion JDBC. Une fois que l'crawler a accédé au moteur de base de données avec l'URI JDBC, le chemin d'inclusion est utilisé pour déterminer quelles tables du moteur de base de données sont créées dans Data Catalog. Par exemple, avec MySQL, si vous spécifiez un chemin d'inclusion `MyDatabase/%`, toutes les tables de `MyDatabase` sont créées dans Data Catalog.

Lors de l'accès à Amazon Redshift, si vous spécifiez un chemin d'inclusion `MyDatabase/%`, toutes les tables de tous les schémas de la base de données `MyDatabase` sont créées dans Data Catalog. Si vous spécifiez un chemin d'inclusion `MyDatabase/MySchema/%`, toutes les tables de la base de données `MyDatabase` et du schéma `MySchema` sont créées.

Après que vous avez spécifié un chemin d'inclusion, vous pouvez exclure les objets de l'analyse qui sinon seraient inclus dans votre chemin d'inclusion en spécifiant un ou plusieurs modèles d'exclusion `glob` de style Unix. Ces modèles sont appliqués à votre chemin d'inclusion afin de déterminer les objets exclus. Ces modèles sont également stockés en tant que propriété des tables créées par le robot d'exploration. AWS Glue PySpark les extensions, telles que`create_dynamic_frame.from_catalog`, lisent les propriétés de la table et excluent les objets définis par le modèle d'exclusion.

AWS Glue prend en charge les modèles `glob` suivants dans les modèles d'exclusion.

Modèle d'exclusion	Description
`*.csv`	Correspond à un chemin Amazon S3 qui représente un nom d'objet se terminant par `.csv` dans le dossier actuel
`*.*`	Correspond à tous les noms d'objets qui contiennent un point
`*.{csv,avro}`	Correspond aux noms d'objets se terminant par `.csv` ou `.avro`
`foo.?`	Correspond aux noms d'objets commençant par `foo.` et suivis par une extension d'un seul caractère
`myfolder/*`	Correspond aux objets situés un niveau en dessous du sous-dossier `myfolder`, comme `/myfolder/mysource`
`myfolder/*/*`	Correspond aux objets situés deux niveaux en dessous du sous-dossier `myfolder`, comme `/myfolder/mysource/data`

Modèle d'exclusion	Description
`myfolder/**`	Correspond aux objets situés dans tous les sous-dossiers de `myfolder`, comme `/myfolder/mysource/mydata` et `/myfolder/mysource/data`
`myfolder**`	Correspond au sous-dossier `myfolder`, ainsi qu'aux fichiers sous `myfolder`, tels que `/myfolder` et `/myfolder/mydata.txt`
`Market*`	Correspond aux tables d'une base de données JDBC dont les noms commencent par `Market`, comme `Market_us` et `Market_fr`

AWS Glue interprète les modèles d'exclusion `glob` comme suit :

- La barre oblique (`/`) est le délimiteur qui permet de séparer les clés Amazon S3 dans une hiérarchie de dossiers.
- L'astérisque (`*`) correspond à zéro ou plusieurs caractères d'un composant de nom sans dépasser les limites d'un dossier.
- Deux astérisques (`**`) correspondent à zéro ou plusieurs caractères dépassant les limites d'un dossier ou d'un schéma.
- Le point d'interrogation (`?`) correspond exactement à un caractère d'un composant de nom.
- La barre oblique inverse (`\`) est utilisée pour échapper des caractères qui seraient sinon interprétés comme des caractères spéciaux. L'expression `\\` correspond à une barre oblique inverse, et `\{` correspond à une accolade gauche.
- Les crochets `[ ]` créent une expression entre crochets qui correspond à un caractère unique d'un composant de nom dans un ensemble de caractères. Par exemple, `[abc]` correspond à `a`, `b` ou `c`. Le trait d'union (`-`) peut être utilisé pour spécifier une plage ; `[a-z]` spécifie une plage allant de `a` à `z` (inclus). Ces formulaires peuvent être mélangés ; `[abce-g]` correspond à `a`, `b`, `c`, `e`, `f` ou `g`. Si le caractère placé après le crochet (`[`) est un point d'exclamation (`!`), l'expression entre crochets est inversée. Par exemple, `[!a-c]` correspond à un caractère quelconque, à l'exception de `a`, `b` ou `c`.

Dans une expression entre crochets, les caractères `*`, `?` et `\` correspondent à eux-mêmes. Le trait d'union (`-`) correspond à lui-même lorsqu'il est placé en première position entre les crochets, ou s'il est le premier caractère placé après le point d'exclamation `!` lors d'une inversion.

- Les accolades (`{ }`) entourent un groupe de sous-modèles, où le groupe correspond si un des sous-modèles du groupe correspond. La virgule (`,`) est utilisée pour séparer les sous-modèles. Les groupes ne peuvent pas être imbriqués.
- Une virgule ou un point placé au début d'un nom de fichier est traité comme un caractère normal dans les opérations de correspondance. Par exemple, le modèle d'exclusion `*` correspond au nom de fichier `.hidden`.

Example Modèles d'exclusion Amazon S3

Chaque modèle d'exclusion est évalué par rapport au chemin d'inclusion. Par exemple, supposons que vous disposiez de la structure de répertoires Amazon S3 suivante :

```
/mybucket/myfolder/
   departments/
      finance.json
      market-us.json
      market-emea.json
      market-ap.json
   employees/
      hr.json
      john.csv
      jane.csv
      juan.txt
```

Soit le chemin d'inclusion `s3://mybucket/myfolder/`, voici quelques exemples de résultats pou les chemins d'exclusion :

Modèle d'exclusion	Résultats
`departments/**`	Exclut tous les fichiers et dossiers sous `departments` et inclut le dossier `employees` et ses fichiers

Modèle d'exclusion	Résultats
`departments/market*`	Exclut `market-us.json` , `market-emea.json` et `market-ap.json`
`**.csv`	Exclut tous les objets sous `myfolder` dont le nom se termine par `.csv`
`employees/*.csv`	Exclut tous les fichiers `.csv` du dossier `employees`

Example Exclusion d'un sous-ensemble de partitions Amazon S3

Supposons que vos données soient partitionnées par jour, de telle sorte que chaque jour d'une année se trouve dans une partition Amazon S3 distincte. Pour le mois de janvier 2015, il y a 31 partitions. Maintenant, pour analyser uniquement les données de la première semaine de janvier, vous devez exclure toutes les partitions, à l'exception des jours 1 à 7 :

```
2015/01/{[!0],0[8-9]}**, 2015/0[2-9]/**, 2015/1[0-2]/**
```

Examinons les différentes parties de ce modèle glob. La première partie, `2015/01/{[!0],0[8-9]}**`, exclut tous les jours qui ne commencent pas par un « 0 », ainsi que le jour 08 et le jour 09 du mois 01 de l'année 2015. Notez que « ** » est utilisé comme suffixe du modèle de numéro du jour et dépasse les limites de dossier vers les dossiers de niveau inférieur. Si « * » est utilisé, les dossiers de niveau inférieur ne sont pas exclus.

La deuxième partie, `2015/0[2-9]/**`, exclut les jours des mois 02 à 09 de l'année 2015.

La troisième partie, `2015/1[0-2]/**`, exclut les jours des mois 10, 11 et 12 de l'année 2015.

Example Modèles d'exclusion JDBC

Supposons que vous analysiez une base de données JDBC avec la structure de schéma suivante :

```
MyDatabase/MySchema/
```

```
HR_us
HR_fr
Employees_Table
Finance
Market_US_Table
Market_EMEA_Table
Market_AP_Table
```

Soit le chemin d'inclusion `MyDatabase/MySchema/%`, voici quelques exemples de résultats pou les chemins d'exclusion :

Modèle d'exclusion	Résultats
`HR*`	Exclut les tables dont le nom commence par `HR`
`Market_*`	Exclut les tables dont le nom commence par `Market_`
`**_Table`	Exclut toutes les tables dont le nom se termine par `_Table`

Paramètres de source d'crawler supplémentaires

Chaque type de source requiert un jeu différent de paramètres supplémentaires.

Connexion

Sélectionnez ou ajoutez une connexion AWS Glue. Pour obtenir des informations sur les connexions, consultez Connexion aux données.

Métadonnées supplémentaires : facultatives (pour les magasins de données JDBC)

Sélectionnez des propriétés de métadonnées supplémentaires à analyser par le crawler.

- Commentaires : analysez des commentaires associés au niveau de la table et au niveau de la colonne.
- Types bruts : conservez les types de données bruts des colonnes de la table dans les métadonnées supplémentaires. Par défaut, le crawler traduit les types de données bruts en types compatibles avec Hive.

Nom de la classe de pilote JDBC – facultatif (pour les magasins de données JDBC)

Saisissez un nom de classe de pilote JDBC personnalisé pour que le Crawler se connecte à la source de données :

- Postgres : org.postgresql.Driver
- MySQL : com.mysql.jdbc.Driver, com.mysql.cj.jdbc.Driver
- Redshift : com.amazon.redshift.jdbc.Driver, com.amazon.redshift.jdbc42.Driver
- Oracle : oracle.jdbc.driver. OracleDriver
- Serveur SQL : com.microsoft.sqlserver.jdbc. SQLServerChauffeur

Chemin S3 du pilote JDBC – facultatif (pour les magasins de données JDBC)

Choisissez un chemin Amazon S3 existant vers un fichier `.jar`. C'est là que le fichier `.jar` est stocké lorsque vous utilisez un pilote JDBC personnalisé pour que le Crawler se connecte à la source de données.

Activer l'échantillonnage de données (pour les magasins de données Amazon DynamoDB, MongoDB, MongoDB Atlas et Amazon DocumentDB uniquement)

Indiquez si vous souhaitez analyser un échantillon de données uniquement. Si ce n'est pas sélectionné, la table entière est analysée. L'analyse de tous les enregistrements peut prendre beaucoup de temps lorsque la table n'est pas à haut débit.

Créer des tables pour les requêtes (uniquement pour les magasins de données Delta Lake)

Sélectionnez la façon dont vous souhaitez créer les tables Delta Lake :

- Créer des tables natives : permettre l'intégration avec les moteurs de requêtes qui prennent directement en charge l'interrogation du journal de transactions Delta.
- Créer des tables avec des liens symboliques : créer un dossier manifeste de lien symbolique avec les fichiers manifestes partitionnés par les clés de partition, en fonction des paramètres de configuration spécifiés.

Taux d'analyse : facultatif (pour les magasins de données DynamoDB uniquement)

Spécifiez le pourcentage des unités de capacité de lecture de la table DynamoDB à utiliser par le crawler. Unités de capacité de lecture est un terme défini par DynamoDB et est une valeur numérique qui sert de limiteur de vitesse pour le nombre de lectures pouvant être effectuées sur cette table par seconde. Saisissez une valeur comprise entre 0,1 et 1.5^63 - 1. Si ce n'est pas spécifié, la valeur par défaut est de 0,5 pour les tables provisionnées et de 1/4 de la capacité configurée maximale pour les tables à la demande. Notez que seul le mode de capacité provisionnée doit être utilisé avec les AWS Glue robots d'exploration.

> **Note**
>
> Pour les magasins de données DynamoDB, définissez le mode de capacité provisionnée pour traiter les lectures et écritures dans vos tables. Le AWS Glue robot ne doit pas être utilisé avec le mode de capacité à la demande.

Connexion réseau : facultatif (pour les entrepôts de données cibles Amazon S3, Delta, Iceberg, Hudi et Catalog)

Vous pouvez également inclure une connexion réseau à utiliser avec cette cible Amazon S3. Notez que chaque crawler est limité à une connexion réseau et toutes les autres cibles Amazon S3 utilisent la même connexion (ou aucune, si ce champ est laissé vide).

Pour obtenir des informations sur les connexions, consultez Connexion aux données.

Échantillon portant sur un seul sous-ensemble de fichiers et taille de l'échantillon (pour les magasins de données Amazon S3 uniquement)

Spécifiez le nombre de fichiers dans chaque dossier feuille à analyser lors de l'analyse d'échantillons de fichiers dans un jeu de données. Lorsque cette fonction est activée, au lieu d'analyser tous les fichiers de ce jeu de données, l'crawler sélectionne au hasard certains fichiers dans chaque dossier feuille à analyser.

Le crawler d'échantillonnage est le mieux adapté aux clients qui ont des connaissances préalables sur leurs formats de données et savent que les schémas de leurs dossiers ne changent pas. L'activation de cette fonction réduira considérablement le temps d'exécution de l'crawler.

Une valeur valide est un entier compris entre 1 et 249. Si ce n'est pas spécifié, tous les fichiers sont analysés.

Exécutions ultérieures du crawler

Ce champ est un champ global qui concerne toutes les sources de données Amazon S3.

- Analyser tous les sous-dossiers : analyser à nouveau tous les dossiers à chaque analyse suivante.
- Analyser uniquement les nouveaux sous-dossiers : seuls les dossiers Amazon S3 ajoutés depuis la dernière analyse sont analysés. Si les schémas sont compatibles, de nouvelles partitions sont ajoutées aux tables existantes. Pour plus d'informations, consultez the section called “Planification d’analyses incrémentielles”.

- Analyse basée sur les événements : utiliser les événements Amazon S3 pour contrôler les dossiers à analyser. Pour de plus amples informations, veuillez consulter Accélération des analyseurs à l'aide des notifications d'événements Amazon S3.

Classifieurs personnalisés - facultatif

Définissez des classifieurs personnalisés avant de définir des crawlers. Un classifieur vérifie si un fichier donné est dans un format que le crawler peut gérer. Si c'est le cas, le classifieur crée un schéma sous la forme d'un objet `StructType` correspondant à ce format de données.

Pour de plus amples informations, veuillez consulter Définition et gestion des classificateurs.

Étape 3 : Configurer les paramètres de sécurité

Rôle IAM

L'crawler endosse ce rôle. Il doit disposer d'autorisations similaires à celles de la politique AWS géréeAWSGlueServiceRole. Pour les sources Amazon S3 et DynamoDB, il doit également disposer des autorisations pour accéder au magasin de données. Si le robot lit les données Amazon S3 chiffrées avec AWS Key Management Service (AWS KMS), le rôle doit disposer d'autorisations de déchiffrement sur la AWS KMS clé.

Pour un magasin de données Amazon S3, les autorisations supplémentaires attachées au rôle seraient similaires à ce qui suit :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": [
        "arn:aws:s3:::bucket/object*"
      ]
    }
  ]
```

```
}
```

Pour un magasin de données Amazon DynamoDB, les autorisations supplémentaires attachées au rôle seraient similaires à ce qui suit :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "dynamodb:DescribeTable",
        "dynamodb:Scan"
      ],
      "Resource": [
        "arn:aws:dynamodb:*:111122223333:table/table-name*"
      ]
    }
  ]
}
```

Pour ajouter votre propre pilote JDBC, des autorisations supplémentaires doivent être ajoutées.

- Accordez des autorisations pour les actions de tâches suivantes : `CreateJob`, `DeleteJob`, `GetJob`, `GetJobRun`, `StartJobRun`.
- Accordez des autorisations pour les actions Amazon S3 : `s3:DeleteObjects`, `s3:GetObject`, `s3:ListBucket`, `s3:PutObject`.

> **Note**
>
> Le `s3:ListBucket` n'est pas nécessaire si la stratégie de compartiment Amazon S3 est désactivée.

- Accordez l'accès principal au service bucket/folder dans la politique Amazon S3.

Exemple de stratégie Amazon S3 :

JSON

```
{
```

```
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:ListBucket",
                "s3:DeleteObject"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket/driver-parent-folder/
driver.jar",
                "arn:aws:s3:::amzn-s3-demo-bucket"
            ]
        }
    ]
}
```

AWS Glue crée les dossiers suivants (`_crawler` et `_glue_job_crawler` au même niveau que le pilote JDBC) dans votre compartiment Amazon S3. Par exemple, si le chemin du pilote est `<s3-path/driver_folder/driver.jar>`, les dossiers suivants seront créés s'ils n'existent pas déjà :

- <s3-path/driver_folder/_crawler>
- <s3-path/driver_folder/_glue_job_crawler>

Le cas échéant, vous pouvez ajouter une configuration de sécurité à un crawler pour spécifier les options de chiffrement au repos.

Pour plus d'informations, consultez Étape 2 : créer un rôle IAM pour AWS Glue et Gestion des identités et des accès pour AWS Glue.

Configuration de Lake Formation - Facultatif

Autorisez le crawler à utiliser les informations d'identification de Lake Formation pour analyser la source de données.

Si vous cochez la case Utiliser les informations d'identification Lake Formation pour analyser la source de données S3, le crawler peut utiliser les informations d'identification Lake Formation pour analyser la source de données. Si la source de données appartient à un autre compte,

vous devez fournir l'ID de compte enregistré. À défaut, le crawler n'analysera que les sources de données associées au compte. Applicable uniquement aux sources de données Amazon S3 et Data Catalog.

Configuration de sécurité - Facultatif

Les paramètres incluent les configurations de sécurité. Pour plus d'informations, consultez les ressources suivantes :

- Chiffrement des données écrites par AWS Glue

> **Note**
>
> Une fois qu'une configuration de sécurité a été définie sur un robot, vous pouvez la modifier, mais vous ne pouvez pas la supprimer. Pour réduire le niveau de sécurité d'un robot, définissez explicitement la fonctionnalité de sécurité sur `DISABLED` dans votre configuration ou créez un robot.

Étape 4 : Configurer la sortie et la planification

Configuration de la sortie

Les options incluent la façon dont l'crawler doit gérer des modifications de schéma, les objets supprimés détectés dans le magasin de données, et plus encore. Pour de plus amples informations, consultez Personnalisation du comportement d'un robot.

Planificateur d'crawler

Vous pouvez exécuter un crawler à la demande ou définir une planification temporelle pour vos crawlers et vos tâches dans AWS Glue. La définition de ces planifications utilise la syntaxe de type Unix cron. Pour de plus amples informations, veuillez consulter Planification d'un crawler.

Étape 5 : vérification et création

Passez en revue les paramètres du crawler que vous avez configurés et créez le crawler.

Planification d'un crawler

Vous pouvez exécuter un crawler AWS Glue à la demande ou selon une planification régulière. Lorsque vous configurez un robot basé sur une planification, vous pouvez spécifier certaines

contraintes, telles que la fréquence, les jours de la semaine et l'heure d'exécution du robot. Vous pouvez créer ces planifications personnalisées au format cron. Pour plus d'informations, consultez cron dans Wikipedia.

Lorsque vous configurez une planification d'crawler, prenez en compte les fonctions et limitations de cron. Par exemple, si vous choisissez d'exécuter votre crawler le 31 de chaque mois, n'oubliez pas que certains mois ne comportent pas 31 jours.

Rubriques

- Créer un calendrier du robot
- Création d'un calendrier pour un robot existant

Créer un calendrier du robot

Vous pouvez créer un calendrier pour le robot d'exploration à l'aide de la AWS Glue console ou AWS CLI.

AWS Management Console

1. Connectez-vous à et ouvrez AWS Management Console la AWS Glue console à l'adresse https://console.aws.amazon.com/glue/.
2. Sélectionnez Crawlers dans le panneau de navigation.
3. Suivez les étapes 1 à 3 dans la section Configuration d'un robot.
4. Dans Étape 4 : Configurer la sortie et la planification, choisissez un Calendrier du robot pour définir la fréquence de l'exécution. Vous pouvez faire en sorte que le robot soit exécuté toutes les heures, tous les jours, toutes les semaines, tous les mois ou définir un calendrier personnalisé à l'aide d'expressions cron.

   Une expression cron est une chaîne représentant un modèle de planification, composée de six champs séparés par des espaces : * * * * * <minute> <hour> <day of month> <month> <day of week> <year>

   Par exemple, pour exécuter une tâche tous les jours à minuit, l'expression cron est : 0 0 * * ? *

   Pour plus d'informations, consultez Expressions Cron.
5. Passez en revue les paramètres du robot que vous avez configurés et créez le robot pour qu'il s'exécute selon un calendrier.

AWS CLI

```
aws glue create-crawler
 --name myCrawler \
 --role AWSGlueServiceRole-myCrawler  \
 --targets '{"S3Targets":[{Path="s3://amzn-s3-demo-bucket/"}]}' \
 --schedule cron(15 12 * * ? *)
```

Pour plus d'informations sur l'utilisation de cron pour planifier les tâches et les crawlers, reportez-vous à la section Planifications temporelles pour les tâches et les crawlers.

Création d'un calendrier pour un robot existant

Procédez comme suit pour configurer un calendrier récurrent pour un robot existant.

AWS Management Console

1. Connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/.
2. Sélectionnez Crawlers dans le panneau de navigation.
3. Choisissez un robot que vous souhaitez planifier dans la liste disponible.
4. Dans le menu Actions, choisissez Modifier.
5. Faites défiler la page jusqu'à Step 4: Set output and scheduling, puis choisissez Modifier.
6. Mettez à jour le calendrier de votre robot sous le Calendrier du robot.
7. Choisissez Mettre à jour.

AWS CLI

Utilisez la commande de CLI suivante pour mettre à jour une configuration de robot existante.

```
aws glue update-crawler-schedule
   --crawler-name myCrawler
   --schedule cron(15 12 * * ? *)
```

Affichage des résultats et des détails du crawler

Une fois que l'crawler s'exécute correctement, il crée des définitions de table dans Data Catalog. Sélectionnez Tables dans le panneau de navigation pour afficher les tables créées par votre crawler dans la base de données que vous avez spécifiée.

Vous pouvez afficher les informations relatives à l'crawler lui-même comme suit :

- La page Crawlers de la AWS Glue console affiche les propriétés suivantes d'un robot d'exploration :

Propriété	Description
Nom	Lorsque vous créez un crawler, vous devez lui attribuer un nom unique.
Statut	Un crawler peut être prêt, en cours de démarrage, en cours d'arrêt, planifié ou suspendu par le calendrier. Un crawler en cours d'exécution progresse du démarrage à l'arrêt. Vous pouvez reprendre ou suspendre un calendrier attaché à un crawler.
Schedule	Vous pouvez choisir d'exécuter votre crawler à la demande ou choisir une fréquence avec un calendrier. Pour plus d'informations sur la planification d'un crawler, consultez Planification d'un crawler.
Dernière exécution	Date et heure de la dernière exécution du crawler.
Journal	Liens vers les journaux disponibles depuis la dernière exécution de l'crawler.
Modifications apportées aux tables depuis la dernière exécution	Le nombre de tables du AWS Glue Data Catalog qui ont été mises à jour lors de la dernière exécution du robot d'exploration.

- Pour afficher l'historique d'un crawler, choisissez Crawlers dans le panneau de navigation pour voir les crawlers que vous avez créés. Choisissez un crawler dans la liste des crawlers disponibles. Vous pouvez consulter les propriétés du crawler et consulter l'historique du crawler dans l'onglet Crawler runs (Exécutions du crawler).

  L'onglet Crawler s'exécute affiche des informations sur chaque exécution du crawler, notamment l'heure de début (UTC), l'heure de fin (UTC), la durée, l'état, les heures DPU, et les modifications apportées aux tables.

  L'onglet Exécutions du Crawler affiche uniquement les analyses qui ont eu lieu depuis la date de lancement de la fonction d'historique du Crawler et ne retient que jusqu'à 12 mois d'analyse. Les anciennes analyses ne seront pas renvoyées.
- Pour voir des informations supplémentaires, choisissez un onglet sur la page de détails du crawler. Chaque onglet affiche les informations relatives au crawler.
  - Schedule (Programme) : tous les programmes créés pour le crawler seront visibles ici.
  - Data sources (Sources de données) : toutes les sources de données analysées par le crawler seront visibles ici.
  - Classifiers (Classifieurs) : tous les classifieurs assignés au crawler seront visibles ici.
  - Balises : toutes les balises créées et attribuées à une AWS ressource seront visibles ici.

Paramètres définis sur les tables du Catalogue de données par un Crawler

Ces propriétés de table sont définies par les AWS Glue robots d'exploration. Nous nous attendons à ce que les utilisateurs consomment les propriétés `classification` et `compressionType`. D'autres propriétés, dont les estimations de la taille des tables, sont utilisées pour les calculs internes, et nous ne garantissons pas leur exactitude ou leur applicabilité aux cas d'utilisation des clients. La modification de ces paramètres peut modifier le comportement du Crawler. Nous ne prenons pas en charge ce flux.

Clé de propriété	Valeur de la propriété
`UPDATED_BY_CRAWLER`	Nom du Crawler qui effectue la mise à jour.
`connectionName`	Nom de la connexion dans le catalogue de données pour le crawler utilisé pour se connecter au magasin de données.

Clé de propriété	Valeur de la propriété
`recordCount`	Estimation du nombre d'enregistrements dans la table, basée sur la taille des fichiers et des en-têtes.
`skip.header.line.count`	Lignes ignorées pour ignorer l'en-tête. Définie sur des tables classées au format CSV.
`CrawlerSchemaSerializerVersion`	Pour utilisation interne
`classification`	Format des données, déduit par le Crawler. Pour plus d'informations sur les formats de données pris en charge par les AWS Glue robots d'exploration, consultezthe section called “Classifieurs intégrés”.
`CrawlerSchemaDeserializerVersion`	Pour utilisation interne
`sizeKey`	Taille combinée des fichiers dans la table indexée.
`averageRecordSize`	Taille moyenne des lignes dans la table, en octets.
`compressionType`	Type de compression utilisé sur les données de la table. Pour plus d'informations sur les types de compression pris en charge par les AWS Glue robots d'exploration, consultezthe section called “Classifieurs intégrés”.
`typeOfData`	`file`, `table` ou `view`.
`objectCount`	Nombre d'objets sous le chemin Amazon S3 pour la table.

Ces propriétés de table supplémentaires sont définies par les robots d' AWS Glue exploration pour les magasins de données Snowflake.

Clé de propriété	Valeur de la propriété
`aws:RawTableLastAltered`	Enregistre le dernier horodatage modifié de la table Snowflake.
`ViewOriginalText`	Affichez l'Instruction SQL.
`ViewExpandedText`	Affichez l'instruction SQL codée au format Base64.
`ExternalTable:S3Location`	Emplacement Amazon S3 de la table externe Snowflake.
`ExternalTable:FileFormat`	Format de fichier Amazon S3 de la table externe Snowflake.

Ces propriétés de table supplémentaires sont définies par des robots d' AWS Glue exploration pour les magasins de données de type JDBC tels qu'Amazon Redshift, Microsoft SQL Server, MySQL, PostgreSQL et Oracle.

Clé de propriété	Valeur de la propriété
`aws:RawType`	Lorsqu'un crawler stocke les données dans le catalogue de données, il traduit les types de données en types compatibles avec Hive, ce qui entraîne souvent la perte des informations relatives au type de données natif. Le crawler génère le paramètre `aws:RawType` pour fournir le type de données de niveau natif.
`aws:RawColumnComment`	Si un commentaire est associé à une colonne de la base de données, le crawler génère le commentaire correspondant dans la table du catalogue . La chaîne de commentaire est tronquée à 255 octets. Les commentaires ne sont pas pris en charge pour Microsoft SQL Server.

Clé de propriété	Valeur de la propriété
`aws:RawTableComment`	Si un commentaire est associé à une table de la base de données, le crawler génère le commentaire correspondant dans la table du catalogue . La chaîne de commentaire est tronquée à 255 octets. Les commentaires ne sont pas pris en charge pour Microsoft SQL Server.

Personnalisation du comportement d’un robot

Lorsque vous configurez un AWS Glue crawler, vous disposez de plusieurs options pour définir le comportement de votre robot d'exploration.

- Analyses incrémentielles : vous pouvez configurer un robot pour exécuter des analyses incrémentielles afin d’ajouter uniquement de nouvelles partitions au schéma de table.
- Index de partition : un robot crée par défaut des index de partition pour les cibles Amazon S3 et Delta Lake afin de permettre une recherche efficace de partitions spécifiques.
- Accélérez le temps d’analyse à l’aide des événements Amazon S3 : vous pouvez configurer un robot pour qu’il utilise les événements Amazon S3 afin d’identifier les changements entre deux analyses. Ce processus s’effectue en répertoriant tous les fichiers du sous-dossier qui a déclenché l’événement au lieu de répertorier la cible Amazon S3 ou du catalogue de données complète.
- Gestion des modifications de schéma : vous pouvez empêcher un robot d’apporter des modifications au schéma existant. Vous pouvez utiliser l'API AWS Management Console ou l'AWS GlueAPI pour configurer la manière dont votre robot traite certains types de modifications.
- Un seul schéma pour plusieurs chemins Amazon S3 : vous pouvez configurer un robot pour créer un seul schéma pour chaque chemin S3 si les données sont compatibles.
- Emplacement des tables et niveaux de partitionnement : l’option de robot au niveau de la table vous offre la possibilité d’indiquer au robot où se trouvent les tables et comment vous souhaitez que les partitions soient créées.
- Seuil de table : vous pouvez spécifier le nombre maximum de tables que le robot est autorisé à créer en spécifiant un seuil de table.
- AWS Lake Formation informations d'identification — Vous pouvez configurer un robot d'exploration pour qu'il utilise les informations d'identification de Lake Formation pour accéder à un magasin de

données Amazon S3 ou à une table du catalogue de données avec un emplacement Amazon S3 sous-jacent au même emplacement Compte AWS ou à un autre Compte AWS.

Pour plus d'informations sur l'utilisation de la console AWS Glue pour ajouter un crawler, consultez Configuration d'un robot.

Rubriques

- Planification d'analyses incrémentielles pour ajouter de nouvelles partitions.
- Génération des index de partition
- Empêcher un robot de modifier un schéma existant
- Création d'un schéma unique pour chaque chemin d'inclusion Amazon S3
- Spécification de l'emplacement de la table et du niveau de partitionnement
- Spécification du nombre maximal de tables que le robot est autorisé à créer
- Configuration d'un robot pour utiliser les informations d'identification de Lake Formation
- Accélération des analyseurs à l'aide des notifications d'événements Amazon S3

Planification d'analyses incrémentielles pour ajouter de nouvelles partitions.

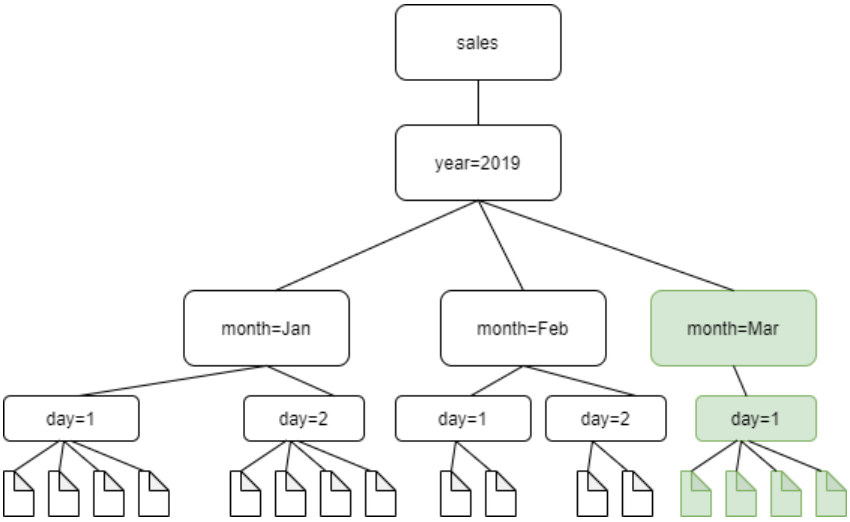
Vous pouvez configurer et AWS Glue crawler exécuter des analyses incrémentielles pour ajouter uniquement de nouvelles partitions au schéma de table. Lorsque le robot s'exécute pour la première fois, il effectue une analyse complète pour traiter l'intégralité de la source de données afin d'enregistrer le schéma complet et toutes les partitions existantes dans l' AWS Glue Data Catalog.

Les analyses qui suivent l'analyse complète initiale seront incrémentielles, le robot identifiant et ajoutant uniquement les nouvelles partitions introduites depuis l'analyse précédente. Cette approche permet d'accélérer les temps d'analyse, car le robot n'a plus besoin de traiter l'intégralité de la source de données à chaque exécution. Il peut alors se concentrer uniquement sur les nouvelles partitions.

> **Note**
>
> Les analyses incrémentielles ne détectent pas les modifications ou les suppressions de partitions existantes. Cette configuration convient parfaitement aux sources de données dotées d'un schéma stable. En cas de modification majeure ponctuelle du schéma, il est conseillé de configurer temporairement le robot pour qu'il effectue une analyse complète

afin de capturer le nouveau schéma avec précision, puis de revenir en mode d'analyse incrémentielle.

Le schéma suivant montre que lorsque le paramètre d'analyse incrémentielle est activé, le robot détecte et ajoute uniquement le dossier récemment ajouté, month=March, au catalogue.



Procédez comme suit pour mettre à jour votre robot afin d'effectuer des analyses incrémentielles :

AWS Management Console

1. Connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/.
2. Choisissez Robots dans le catalogue de données.
3. Choisissez un robot que vous souhaitez configurer pour qu'il effectuer une analyse incrémentielle.
4. Choisissez Modifier.
5. Choisissez Étape 2. Choisissez les sources de données et les classificateurs.
6. Choisissez la source de données que vous souhaitez analyser de manière incrémentielle.
7. Choisissez Modifier.
8. Choisissez Analyser les nouveaux sous-dossiers uniquement sous Exécutions ultérieures du robot.
9. Choisissez Mettre à jour.

Pour créer un calendrier pour un robot, consultez the section called "Planification d'un crawler".

AWS CLI

```
aws glue update-crawler \
 --name myCrawler \
 --recrawl-policy RecrawlBehavior=CRAWL_NEW_FOLDERS_ONLY \
 --schema-change-policy UpdateBehavior=LOG,DeleteBehavior=LOG
```

Notes et restrictions

Lorsque cette option est activée, vous ne pouvez pas modifier les magasins de données cibles Amazon S3 lors de la modification de l'crawler. Cette option affecte certains paramètres de configuration de l'crawler. Lorsqu'il est activé, il force le comportement de mise à jour et le comportement de suppression de l'crawler à `LOG`. Cela signifie que :

- S'il découvre des objets dont les schémas ne sont pas compatibles, le robot d'exploration n'ajoutera pas les objets dans le catalogue de données et ajoute ce détail sous forme de journal de connexion. CloudWatch
- Il ne met pas à jour les objets supprimés dans le catalogue de données.

Génération des index de partition

Le catalogue de données prend en charge la création d'index de partition afin de permettre une recherche efficace de partitions spécifiques. Pour en savoir plus, consultez Creating partition indexes. Le AWS Glue crawler crée des index de partition pour les cibles Amazon S3 et Delta Lake par défaut.

AWS Management Console

1. Connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/.
2. Choisissez Robots dans le catalogue de données.
3. Lorsque vous définissez un robot, l'option permettant de créer automatiquement des index de partition est activée par défaut sous Options avancées, sur la page Définir la sortie et la planification.

   Pour désactiver cette option, vous pouvez décocher la case Créer des index de partition automatiquement dans la console.
4. Complétez la configuration du robot et choisissez Créer un robot.

AWS CLI

Vous pouvez également désactiver cette option en utilisant l' AWS CLI et en définissant `CreatePartitionIndex` dans le paramètre `configuration`. La valeur par défaut est True.

```
aws glue update-crawler \
    --name myCrawler \
    --configuration '{"Version": 1.0, "CreatePartitionIndex": false }'
```

Notes d'utilisation pour les index de partition

- Les tables créées par le Crawler ne contiennent pas la variable `partition_filtering.enabled` par défaut. Pour en savoir plus, consultez AWS Glue partition indexing and filtering.
- La création d'index de partition pour les partitions chiffrées n'est pas prise en charge.

Empêcher un robot de modifier un schéma existant

Vous pouvez empêcher AWS Glue crawler s d'apporter des modifications de schéma au catalogue de données lors de son exécution. Par défaut, les robots mettent à jour le schéma du catalogue de données pour qu'il corresponde à la source de données analysée. Toutefois, dans certains cas, vous souhaiterez peut-être empêcher le robot de modifier le schéma existant, en particulier si vous avez transformé ou nettoyé les données et que vous ne souhaitez pas que le schéma d'origine remplace les modifications.

Suivez ces étapes pour configurer votre robot de manière à ne pas remplacer le schéma existant dans une définition de table.

AWS Management Console

1. Connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/.
2. Choisissez Robots dans le catalogue de données.
3. Choisissez un robot dans la liste, puis Modifier.
4. Choisissez étape 4, Définir la sortie et la planification.
5. Sous Options avancées, choisissez Ajouter de nouvelles colonnes uniquement ou Ignorer la modification et ne pas mettre à jour la table dans le catalogue de données.

6. En outre, vous pouvez définir une option de configuration sur Mettre à jour toutes les partitions nouvelles et existantes avec les métadonnées de la table. Cela définit les schémas de partition pour qu'ils héritent de la table.
7. Choisissez Mettre à jour.

AWS CLI

L'exemple suivant montre comment configurer un robot pour ne pas modifier le schéma existant, mais uniquement pour ajouter de nouvelles colonnes :

```
aws glue update-crawler \
  --name myCrawler \
  --configuration '{"Version": 1.0, "CrawlerOutput": {"Tables":
 {"AddOrUpdateBehavior": "MergeNewColumns"}}}'
```

L'exemple suivant montre comment configurer un robot pour ne pas modifier le schéma existant et ne pas ajouter de nouvelles colonnes :

```
aws glue update-crawler \
  --name myCrawler \
  --schema-change-policy UpdateBehavior=LOG \
  --configuration '{"Version": 1.0, "CrawlerOutput": {"Partitions":
 { "AddOrUpdateBehavior": "InheritFromTable" }}}'
```

API

Si vous ne voulez pas qu'un schéma de table soit modifié lorsqu'un crawler s'exécute, définissez la politique de modification du schéma sur `LOG`.

Lorsque vous configurez l'crawler à l'aide de l'API, définissez les paramètres suivants :

- Définissez le champ `UpdateBehavior` de la structure `SchemaChangePolicy` sur `LOG`.
- Définissez le champ `Configuration` avec une représentation de chaîne de l'objet JSON suivant dans l'API de l'crawler ; par exemple :

```
{
   "Version": 1.0,
   "CrawlerOutput": {
```

```
      "Partitions": { "AddOrUpdateBehavior": "InheritFromTable" }
   }
}
```

Création d'un schéma unique pour chaque chemin d'inclusion Amazon S3

Par défaut, lorsqu'un crawler définit des tables pour des données stockées dans Amazon S3, il prend en compte la compatibilité des données et la similitude de schéma. Parmi les facteurs de compatibilité des données, il prend en compte le fait que les données sont du même format (par exemple, JSON), le même type de compression (par exemple, GZIP), la structure du chemin d'accès Amazon S3 et d'autres caractéristiques de données. La similarité des schémas définit le degré de similarité des schémas d'objets Amazon S3 distincts.

Pour vous aider à illustrer cette option, supposons que vous définissez un crawler avec un chemin d'inclusion `s3://amzn-s3-demo-bucket/table1/`. Lorsque l'crawler s'exécute, il trouve deux fichiers JSON avec les caractéristiques suivantes :

- Fichier 1 – `S3://amzn-s3-demo-bucket/table1/year=2017/data1.json`
- Contenu du fichier – `{"A": 1, "B": 2}`
- Schéma – `A:int, B:int`

- Fichier 2 – `S3://amzn-s3-demo-bucket/table1/year=2018/data2.json`
- Contenu du fichier – `{"C": 3, "D": 4}`
- Schéma – `C: int, D: int`

Par défaut, l'crawler crée deux tables, nommées `year_2017` et `year_2018` car les schémas ne sont pas suffisamment similaires. Toutefois, si l'option Create a single schema for each S3 path (Créer un seul schéma pour chaque chemin S3) est sélectionnée, et si les données sont compatibles, l'crawler crée une table. La table comprend le schéma `A:int,B:int,C:int,D:int` et `partitionKey year:string`.

AWS Management Console

1. Connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/.
2. Choisissez Robots dans le catalogue de données.

3. Lorsque vous configurez un nouveau robot, sous Sortie et planification, sélectionnez l'option Créer un schéma unique pour chaque chemin S3 sous Options avancées.

AWS CLI

Vous pouvez configurer un crawler pour `CombineCompatibleSchemas` dans une définition de table courante lorsque cela est possible. Avec cette option, l'crawler considère toujours la compatibilité des données, mais ignore la similarité des schémas spécifiques lors de l'évaluation des objets Amazon S3 dans le chemin d'inclusion (include path) spécifié.

Lorsque vous configurez le robot d'exploration à l'aide du AWS CLI, définissez l'option de configuration suivante :

```
aws glue update-crawler \
   --name myCrawler \
   --configuration '{"Version": 1.0, "Grouping": {"TableGroupingPolicy":
 "CombineCompatibleSchemas" }}'
```

API

Lorsque vous configurez l'crawler à l'aide de l'API, définissez les options de configuration suivantes :

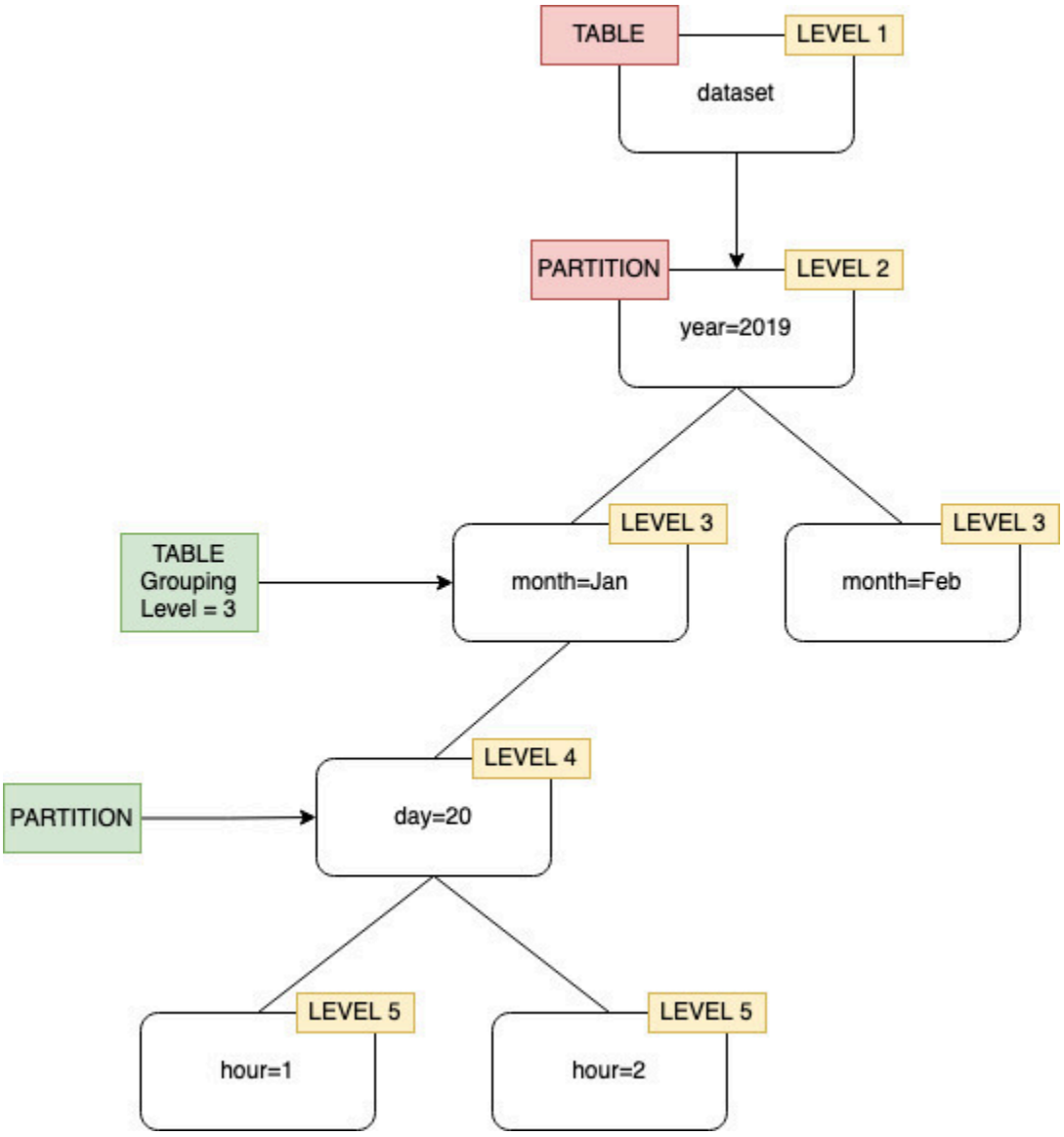
Définissez le champ `Configuration` avec une représentation de chaîne de l'objet JSON suivant dans l'API de l'crawler ; par exemple :

```
{
   "Version": 1.0,
   "Grouping": {
      "TableGroupingPolicy": "CombineCompatibleSchemas" }
}
```

Spécification de l'emplacement de la table et du niveau de partitionnement

Par défaut, lorsqu'un robot définit des tables pour les données stockées dans Amazon S3, il tente de fusionner les schémas et de créer des tables de niveau supérieur (`year=2019`). Dans certains cas, vous pouvez vous attendre à ce que l'crawler crée une table pour le dossier `month=Jan`, mais à la place, l'crawler crée une partition puisqu'un dossier frère (`month=Mar`) a été fusionné dans la même table.

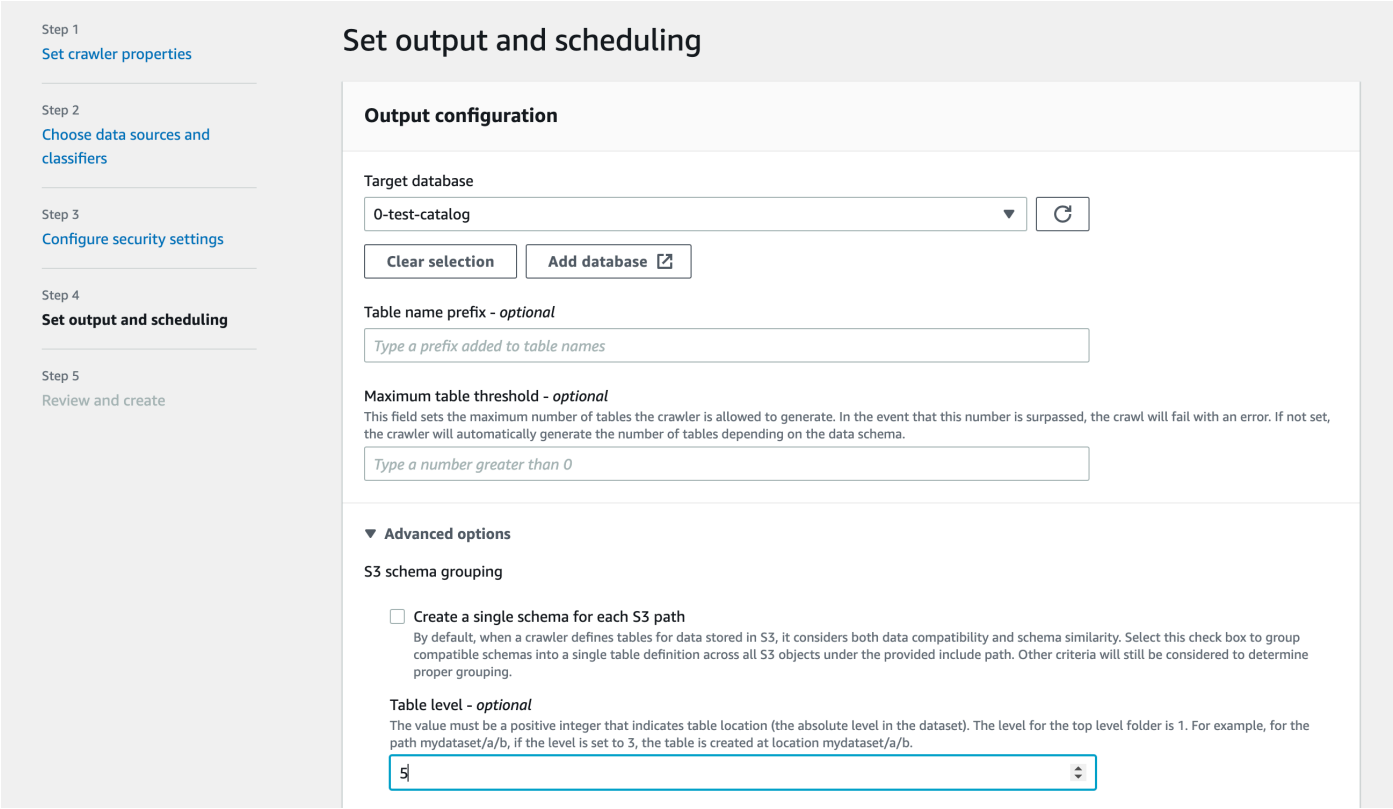
L'option d'crawler au niveau de la table vous offre la possibilité d'indiquer à l'crawler où se trouvent les tables et comment vous souhaitez que les partitions soient créées. Lorsque vous spécifiez un Table level (Niveau de la table), la table est créée à ce niveau absolu à partir du compartiment Amazon S3.



Lorsque vous configurez l'crawler sur la console, vous pouvez spécifier une valeur pour l'option d'crawler Table level (Niveau de la table). La valeur doit être un entier positif qui indique l'emplacement de la table (niveau absolu dans le jeu de données). Le niveau du dossier de niveau supérieur est 1. Par exemple, pour le chemin `mydataset/year/month/day/hour`, si le niveau est défini sur 3, la table est créée à l'emplacement `mydataset/year/month`.

AWS Management Console

1. Connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/.
2. Choisissez Robots dans le catalogue de données.
3. Lorsque vous configurez un robot, sous Sortie et planification, choisissez Niveau de table sous Options avancées.



AWS CLI

Lorsque vous configurez le robot à l'aide du AWS CLI, définissez le `configuration` paramètre comme indiqué dans l'exemple de code :

```
aws glue update-crawler \
  --name myCrawler \
  --configuration '{"Version": 1.0, "Grouping": { "TableLevelConfiguration": 2 }}'
```

API

Lorsque vous configurez l'crawler à l'aide de l'API, définissez le champ `Configuration` avec une représentation de chaînes de l'objet JSON suivant ; par exemple :

```
configuration = jsonencode(
{
   "Version": 1.0,
   "Grouping": {
            TableLevelConfiguration = 2
        }
})
```

CloudFormation

Dans cet exemple, vous définissez l'option de niveau de table disponible dans la console au sein de votre CloudFormation modèle :

```
"Configuration": "{
    \"Version\":1.0,
    \"Grouping\":{\"TableLevelConfiguration\":2}
}"
```

Spécification du nombre maximal de tables que le robot est autorisé à créer

Vous pouvez éventuellement spécifier le nombre maximum de tables que le robot d'exploration est autorisé à créer en spécifiant un `TableThreshold` via la AWS Glue console ou AWS CLI. Si les tables détectées par le crawler lors de son analyse sont supérieures à cette valeur d'entrée, l'analyse échoue et aucune donnée n'est écrite dans le catalogue de données.

Ce paramètre est utile lorsque les tables qui seraient détectées et créées par le crawler sont beaucoup plus grandes que vos prévisions. Cela peut être dû à plusieurs raisons, notamment :

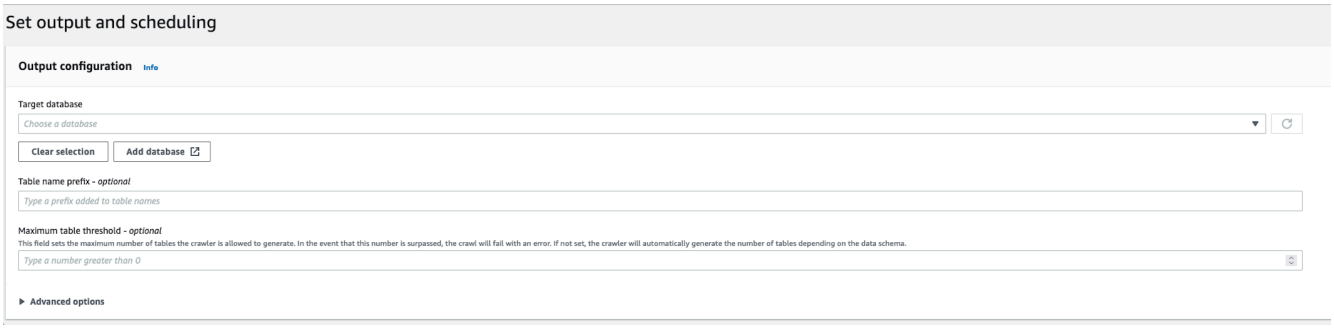
- Lorsque vous utilisez une AWS Glue tâche pour renseigner vos emplacements Amazon S3, vous pouvez vous retrouver avec des fichiers vides au même niveau qu'un dossier. Dans de tels cas, lorsque vous exécutez un crawler sur cet emplacement Amazon S3, le crawler crée plusieurs tables en raison de la présence de fichiers et de dossiers au même niveau.
- Si vous ne configurez pas `"TableGroupingPolicy": "CombineCompatibleSchemas"`, vous pourriez vous retrouver avec plus de tables que prévu.

Vous devez spécifier la propriété `TableThreshold` sous forme d'une valeur de nombre entier supérieure à 0. Cette valeur est configurée par crawler. Cela signifie qu'elle est prise en compte pour chaque analyse. Par exemple : un crawler a la valeur `TableThreshold` définie sur 5. Lors de chaque analyse, AWS Glue compare le nombre de tables détectées à cette valeur seuil de table (5) et, si le nombre de tables détectées est inférieur à 5, AWS Glue écrit les tables dans le catalogue de données. Sinon, l'analyse échoue sans écriture dans le catalogue de données.

AWS Management Console

Pour configurer à **`TableThreshold`** l'aide de AWS Management Console :

1. Connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/.
2. Lors de la configuration d'un robot, dans Sortie et planification, définissez le Seuil maximal de tables sur le nombre de tables que le robot est autorisé à générer.



AWS CLI

Pour configurer à `TableThreshold` l'aide de AWS CLI :

```
aws glue update-crawler \
    --name myCrawler \
    --configuration '{"Version": 1.0, "CrawlerOutput": {"Tables":
 { "TableThreshold": 5 }}}'
```

API

Pour définir `TableThreshold` à l'aide de l'API :

```
"{"Version":1.0,
"CrawlerOutput":
{"Tables":{"AddOrUpdateBehavior":"MergeNewColumns",
```

```
"TableThreshold":5}}}";
```

Les messages d'erreur sont consignés pour vous aider à identifier les chemins d'accès aux tables et à nettoyer vos données. Exemple de journal dans votre compte si le crawler échoue, parce que le nombre de tables était supérieur à la valeur du seuil de tables fournie :

```
Table Threshold value = 28, Tables detected - 29
```

Dans CloudWatch, nous enregistrons tous les emplacements de table détectés sous forme de message INFO. Une erreur est consignée comme motif de l'échec.

```
ERROR com.amazonaws.services.glue.customerLogs.CustomerLogService - CustomerLogService
 received CustomerFacingException with message
The number of tables detected by crawler: 29 is greater than the table threshold value
 provided: 28. Failing crawler without writing to Data Catalog.
com.amazonaws.services.glue.exceptions.CustomerFacingInternalException: The number of
 tables detected by crawler: 29 is greater than the table threshold value provided:
 28.
Failing crawler without writing to Data Catalog.
```

Configuration d'un robot pour utiliser les informations d'identification de Lake Formation

Vous pouvez configurer un robot d'exploration pour qu'il utilise des AWS Lake Formation informations d'identification pour accéder à un magasin de données Amazon S3 ou à une table de catalogue de données contenant un emplacement Amazon S3 sous-jacent au sein de la même Compte AWS ou d'une autre Compte AWS. Vous pouvez configurer une table du catalogue de données existante en tant que cible d'un crawler, si le crawler et la table du catalogue de données se trouvent dans le même compte. Actuellement, une seule cible de catalogue avec une seule table de catalogue est autorisée lors de l'utilisation d'une table du catalogue de données comme cible d'un robot.

> **Note**
>
> Lorsque vous définissez une table du catalogue de données en tant que cible du crawler, assurez-vous que l'emplacement sous-jacent de la table du catalogue de données est un emplacement Amazon S3. Les crawlers qui utilisent les informations d'identification Lake Formation ne prennent en charge que les cibles du catalogue de données avec des emplacements Amazon S3 sous-jacents

Configuration requise lorsque le crawler et l'emplacement Amazon S3 enregistré ou la table du catalogue de données se trouvent dans le même compte (indexation de site web intégrée au compte)

Pour permettre au crawler d'accéder à un magasin de données ou à une table du catalogue de données à l'aide des informations d'identification de Lake Formation, vous devez enregistrer l'emplacement des données auprès de Lake Formation. En outre, le rôle IAM du crawler doit être autorisé à lire les données depuis la destination où le compartiment Amazon S3 est enregistré.

Vous pouvez effectuer les étapes de configuration suivantes à l'aide du AWS Management Console ou AWS Command Line Interface (AWS CLI).

AWS Management Console

1. Avant de configurer un crawler pour accéder à sa source, enregistrez l'emplacement des données du magasin de données ou du catalogue de données auprès de Lake Formation. Dans la console Lake Formation (https://console.aws.amazon.com/lakeformation/), enregistrez un emplacement Amazon S3 comme emplacement racine de votre lac de données dans l' Compte AWS endroit où le robot est défini. Pour plus d'informations, consultez la rubrique Enregistrement d'un emplacement Amazon S3.
2. Accordez des autorisations Emplacement des données au rôle IAM utilisé pour l'exécution du crawler afin que celui-ci puisse lire les données depuis la destination dans Lake Formation. Pour plus d'informations, consultez la rubrique Octroi d'autorisations d'emplacement de données (même compte).
3. Accordez les autorisations d'accès au rôle de crawler (`Create`) à la base de données, qui est spécifiée comme base de données de sortie. Pour plus d'informations, consultez la rubrique Octroi d'autorisations de base de données via la console Lake Formation et la méthode de ressource nommée.
4. Dans la console IAM (https://console.aws.amazon.com/iam/), créez un rôle IAM pour le robot d'exploration. Ajoutez la stratégie `lakeformation:GetDataAccess` au rôle.
5. Dans la AWS Glue console (https://console.aws.amazon.com/glue/), lors de la configuration du robot d'exploration, sélectionnez l'option Utiliser les informations d'identification de Lake Formation pour analyser la source de données Amazon S3.

> **Note**
>
> Le champ accountId est facultatif pour l'indexation de site web intégrée au compte.

AWS CLI

```
aws glue --profile demo create-crawler --debug --cli-input-json '{
    "Name": "prod-test-crawler",
    "Role": "arn:aws:iam::111122223333:role/service-role/AWSGlueServiceRole-prod-
test-run-role",
    "DatabaseName": "prod-run-db",
    "Description": "",
    "Targets": {
    "S3Targets":[
                {
                 "Path": "s3://amzn-s3-demo-bucket"
                }
                ]
                },
   "SchemaChangePolicy": {
      "UpdateBehavior": "LOG",
      "DeleteBehavior": "LOG"
  },
  "RecrawlPolicy": {
    "RecrawlBehavior": "CRAWL_EVERYTHING"
  },
  "LineageConfiguration": {
    "CrawlerLineageSettings": "DISABLE"
  },
  "LakeFormationConfiguration": {
    "UseLakeFormationCredentials": true,
    "AccountId": "111122223333"
  },
  "Configuration": {
           "Version": 1.0,
           "CrawlerOutput": {
             "Partitions": { "AddOrUpdateBehavior": "InheritFromTable" },
             "Tables": {"AddOrUpdateBehavior": "MergeNewColumns" }
           },
           "Grouping": { "TableGroupingPolicy": "CombineCompatibleSchemas" }
         },
  "CrawlerSecurityConfiguration": "",
  "Tags": {
    "KeyName": ""
  }
}'
```

Configuration requise lorsque le crawler et l'emplacement Amazon S3 enregistré se trouvent dans des comptes différents (indexation de site web entre comptes)

Pour permettre au crawler d'accéder à un magasin de données sur un autre compte à l'aide des informations d'identification Lake Formation, vous devez d'abord enregistrer l'emplacement des données Amazon S3 auprès de Lake Formation. Ensuite, vous accordez des autorisations de localisation des données au compte du crawler en procédant comme suit.

Vous pouvez effectuer les étapes suivantes à l'aide du AWS Management Console ou AWS CLI.

AWS Management Console

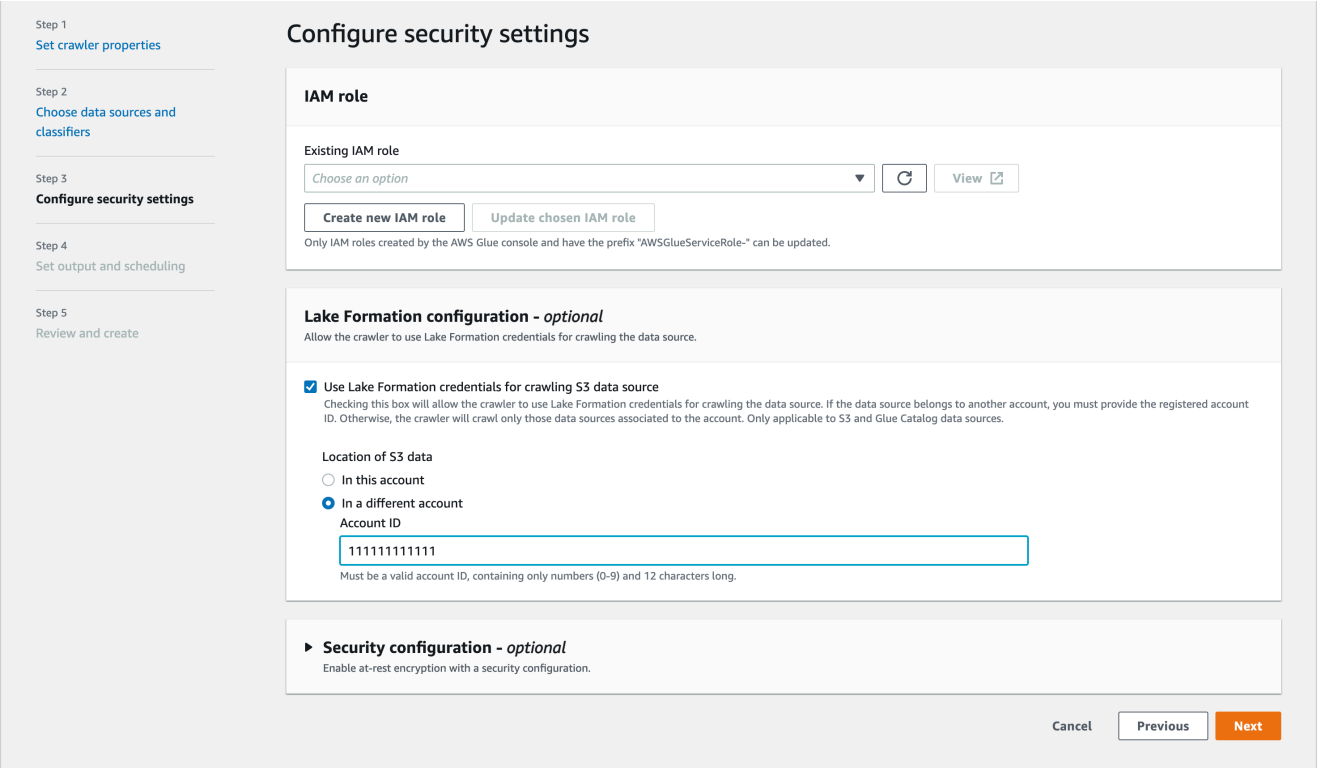
1. Dans le compte sur lequel l'emplacement Amazon S3 est enregistré (compte B) :
   a. Enregistrez un chemin Amazon S3 dans Lake Formation. Pour plus d'informations, consultez la rubrique Enregistrement d'un emplacement Amazon S3.
   b. Accordez des autorisations Data location (Emplacement des données) au compte (compte A) sur lequel le crawler sera exécuté. Pour plus d'informations, consultez la rubrique Octroi d'autorisations d'emplacement de données.
   c. Créez une base de données vide dans Lake Formation avec l'emplacement sous-jacent comme emplacement Amazon S3 cible. Pour plus d'informations, consultez la rubrique Création d'une base de données.
   d. Accordez au compte A (le compte sur lequel le crawler sera exécuté) l'accès à la base de données que vous avez créée à l'étape précédente. Pour plus d'informations, consultez la rubrique Octroi d'autorisations de base de données.
2. Dans le compte sur lequel le crawler est créé et sera exécuté (compte A) :
   a. À l'aide de la AWS RAM console, acceptez la base de données partagée depuis le compte externe (compte B). Pour plus d'informations, consultez la section Acceptation d'une invitation de partage de ressources provenant de AWS Resource Access Manager.
   b. Créez un rôle IAM pour le crawler. Ajoutez une stratégie `lakeformation:GetDataAccess` au rôle.
   c. Dans la console Lake Formation (https://console.aws.amazon.com/lakeformation/), accordez des autorisations de localisation des données sur l'emplacement Amazon S3 cible au rôle IAM utilisé pour l'exécution du robot d'exploration afin que celui-ci puisse lire les données depuis la destination dans Lake Formation. Pour plus d'informations, consultez la rubrique Octroi d'autorisations d'emplacement de données.

d. Créez un lien de ressource sur la base de données partagée. Pour plus d'informations, consultez la rubrique Création d'un lien de ressources.

e. Accordez les autorisations d'accès au rôle de crawler (`Create`) sur la base de données partagée et (`Describe`) le lien de ressource. Le lien de ressource est spécifié dans la sortie du crawler.

f. Dans la AWS Glue console (https://console.aws.amazon.com/glue/), lors de la configuration du robot d'exploration, sélectionnez l'option Utiliser les informations d'identification de Lake Formation pour analyser la source de données Amazon S3.

   Pour l'exploration entre comptes, spécifiez l' Compte AWS ID auquel l'emplacement Amazon S3 cible est enregistré auprès de Lake Formation. Le champ accountId est facultatif pour l'indexation de site web intégrée au compte.



AWS CLI

```
aws glue --profile demo create-crawler --debug --cli-input-json '{
    "Name": "prod-test-crawler",
    "Role": "arn:aws:iam::111122223333:role/service-role/AWSGlueServiceRole-prod-
test-run-role",
    "DatabaseName": "prod-run-db",
```

```
    "Description": "",
    "Targets": {
    "S3Targets":[
                {
                 "Path": "s3://amzn-s3-demo-bucket"
                }
                ]
                },
   "SchemaChangePolicy": {
      "UpdateBehavior": "LOG",
      "DeleteBehavior": "LOG"
  },
  "RecrawlPolicy": {
    "RecrawlBehavior": "CRAWL_EVERYTHING"
  },
  "LineageConfiguration": {
    "CrawlerLineageSettings": "DISABLE"
  },
  "LakeFormationConfiguration": {
    "UseLakeFormationCredentials": true,
    "AccountId": "111111111111"
  },
  "Configuration": {
          "Version": 1.0,
          "CrawlerOutput": {
            "Partitions": { "AddOrUpdateBehavior": "InheritFromTable" },
            "Tables": {"AddOrUpdateBehavior": "MergeNewColumns" }
          },
          "Grouping": { "TableGroupingPolicy": "CombineCompatibleSchemas" }
        },
  "CrawlerSecurityConfiguration": "",
  "Tags": {
    "KeyName": ""
  }
}'
```

> **Note**
>
> - Un crawler utilisant les informations d'identification Lake Formation n'est pris en charge que pour les cibles Amazon S3 et du catalogue de données.

- Pour les cibles utilisant le distributeur d'informations d'identification Lake Formation, les emplacements Amazon S3 sous-jacents doivent appartenir au même compartiment. Par exemple, les clients peuvent utiliser plusieurs cibles (s3://amzn-s3-demo-bucket1/folder1, s3://amzn-s3-demo-bucket1/folder2) à condition que tous les emplacements cibles se trouvent dans le même compartiment (amzn-s3-demo-bucket1). La spécification de différents compartiments (s3://amzn-s3-demo-bucket1/folder1, s3://amzn-s3-demo-bucket2/folder2) n'est pas autorisée.
- Actuellement, pour les crawlers de cible du catalogue de données, une seule cible de catalogue avec une seule table de catalogue est autorisée.

Accélération des analyseurs à l'aide des notifications d'événements Amazon S3

Au lieu de répertorier les objets d'une cible Amazon S3 ou de catalogue de données, vous pouvez configurer le Crawler pour qu'il utilise les événements Amazon S3 pour trouver les modifications éventuelles. Cette fonction améliore le temps de nouvelle analyse en utilisant les événements Amazon S3 pour identifier les changements entre deux analyses. Ce processus s'effectue en répertoriant tous les fichiers du sous-dossier qui a déclenché l'événement au lieu de répertorier la cible Amazon S3 ou du catalogue de données complète.

La première analyse répertorie tous les objets Amazon S3 de la cible. Après la première analyse réussie, vous pouvez choisir de faire une nouvelle analyse manuellement ou selon un calendrier défini. L'crawler indiquera uniquement les objets de ces événements au lieu de répertorier tous les objets.

Lorsque la cible est une table de catalogue de données, le robot met à jour les tables existantes du catalogue de données avec les modifications (par exemple, des partitions supplémentaires dans une table).

Passer à un crawler basé sur les événements Amazon S3 a les avantages suivants :

- Une nouvelle analyse plus rapide, car la liste de tous les objets de la cible n'est pas nécessaire ; toutefois, la liste de dossiers spécifiques est effectuée là où des objets sont ajoutés ou supprimés.
- Une réduction du coût global d'analyse étant donné que la liste de dossiers spécifiques est effectuée lorsque des objets sont ajoutés ou supprimés.

L'analyse des événements Amazon S3 s'exécute en consommant des événements Amazon S3 depuis la file d'attente SQS en fonction de la planification de l'crawler. Il n'y aura aucun

coût s'il n'y a pas d'événements dans la file d'attente. Les événements Amazon S3 peuvent être configurés pour accéder directement à la file d'attente SQS ou, dans les cas où plusieurs consommateurs ont besoin du même événement, une combinaison de SNS et de SQS. Pour de plus amples informations, veuillez consulter the section called "Configuration de votre compte pour les notifications d'événements Amazon S3".

Après avoir créé et configuré le Crawler en mode événement, la première analyse s'exécute en mode de liste en répertoriant la liste de la cible Amazon S3 ou du catalogue de données complète. Le journal suivant confirme l'opération de l'analyse en consommant des événements Amazon S3 après la première analyse réussie : « L'analyse est exécutée en consommant des événements Amazon S3. »

Après avoir créé l'analyse des événements Amazon S3 et mis à jour les propriétés de l'crawler susceptibles d'avoir un impact sur l'analyse, l'analyse fonctionne en mode liste et le journal suivant est ajouté : « L'analyse n'est pas exécutée en mode événement S3 ».

> **Note**
>
> Le nombre maximum de messages à consommer est de 100 000 par analyse.

Considérations et restrictions

Les considérations et restrictions suivantes s'appliquent lorsque vous configurez un robot pour qu'il utilise les notifications d'événements Amazon S3 afin de trouver les modifications éventuelles.

- Comportement important avec les partitions supprimées

  Lorsque vous utilisez les robots d'événements Amazon S3 avec les tables du catalogue de données :

  - Si vous supprimez une partition à l'aide de l'appel d'API `DeletePartition`, vous devez également supprimer tous les objets S3 sous cette partition et sélectionner Tous les événements de suppression d'objets lorsque vous configurez vos notifications d'événements S3. Si les événements de suppression ne sont pas configurés, le robot recrée la partition supprimée lors de sa prochaine exécution.
- Seule une cible unique est prise en charge par le Crawler, que ce soit pour les cibles Amazon S3 ou du Catalogue de données.
- Un SQS sur un VPC privé n'est pas pris en charge.

- L'échantillonnage Amazon S3 n'est pas pris en charge.
- La cible du Crawler doit être un dossier pour une cible Amazon S3, ou une ou plusieurs tables de Catalogue de données AWS Glue pour une cible de Catalogue de données.
- Le caractère générique du chemin « tout » n'est pas pris en charge : s3://%
- Pour une cible de Catalogue de données, toutes les tables du catalogue doivent pointer vers le même compartiment Amazon S3 pour le mode événement Amazon S3.
- Pour une cible de Catalogue de données, une table de catalogue ne doit pas pointer vers un emplacement Amazon S3 au format Delta Lake (contenant des dossiers _symlink ou vérifiant le `InputFormat` de la table du catalogue).

Rubriques

- Configuration de votre compte pour les notifications d'événements Amazon S3
- Configuration d'un robot pour les notifications d'événements Amazon S3 pour une cible Amazon S3
- Configuration d'un robot pour les notifications d'événements Amazon S3 à l'aide de la table du catalogue de données

Configuration de votre compte pour les notifications d'événements Amazon S3

Réalisez les tâches de configuration suivantes. Notez que les valeurs entre parenthèses font référence aux paramètres configurables du script.

1. Vous devez configurer les notifications d'événements pour votre compartiment Amazon S3.

   Pour plus d'informations, consultez Amazon S3 Event Notifications.
2. Pour utiliser le robot basé sur les événements Amazon S3, vous devez activer la notification d'événements sur le compartiment Amazon S3 avec des événements filtrés à partir du préfixe identique à la cible et au magasin S3 dans SQS. Vous pouvez configurer SQS et la notification d'événements via la console en suivant les étapes de Walkthrough: Configuring a bucket for notifications.
3. Ajoutez la politique SQS suivante au rôle utilisé par le robot.

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
```

```
    {
      "Sid": "VisualEditor0",
      "Effect": "Allow",
      "Action": [
        "sqs:DeleteMessage",
        "sqs:GetQueueUrl",
        "sqs:ListDeadLetterSourceQueues",
        "sqs:ReceiveMessage",
        "sqs:GetQueueAttributes",
        "sqs:ListQueueTags",
        "sqs:SetQueueAttributes",
        "sqs:PurgeQueue"
      ],
      "Resource": "arn:aws:sqs:us-east-1:111122223333:cfn-sqs-queue"
    }
  ]
}
```

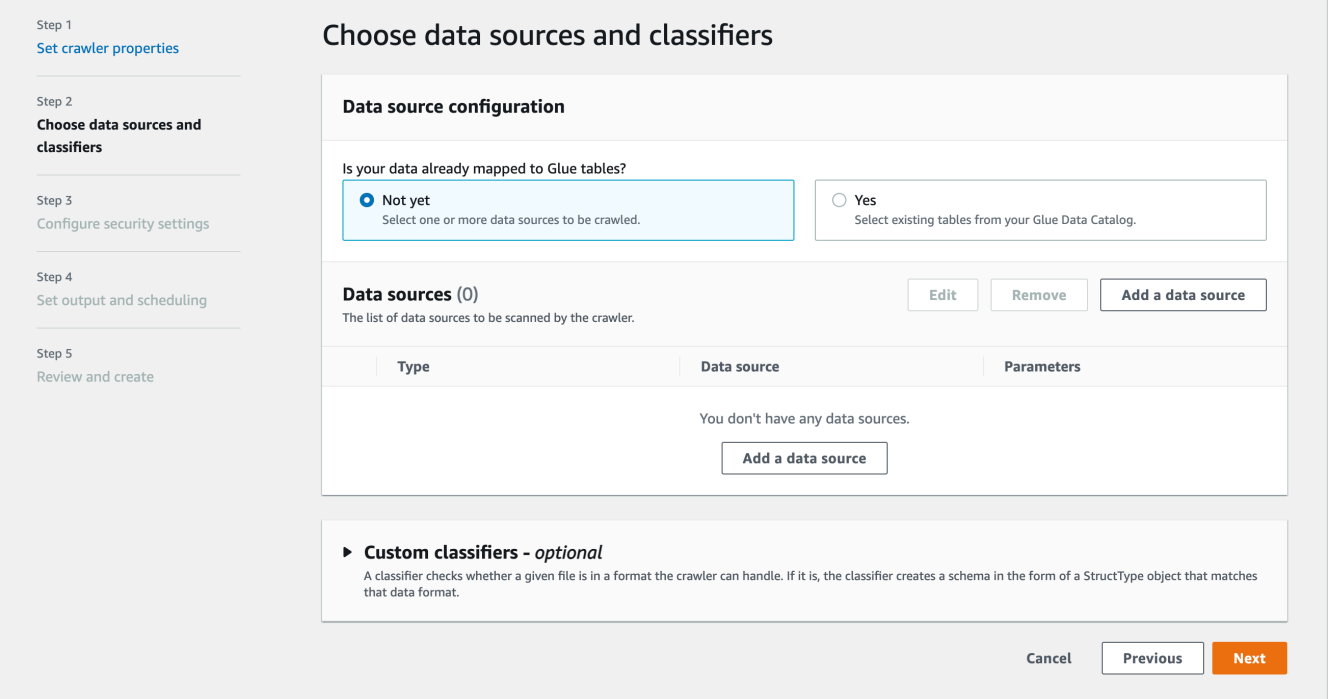
Configuration d'un robot pour les notifications d'événements Amazon S3 pour une cible Amazon S3

Suivez ces étapes pour configurer un robot d'exploration pour les notifications d'événements Amazon S3 pour une cible Amazon S3 à l'aide du AWS Management Console ou AWS CLI.

AWS Management Console

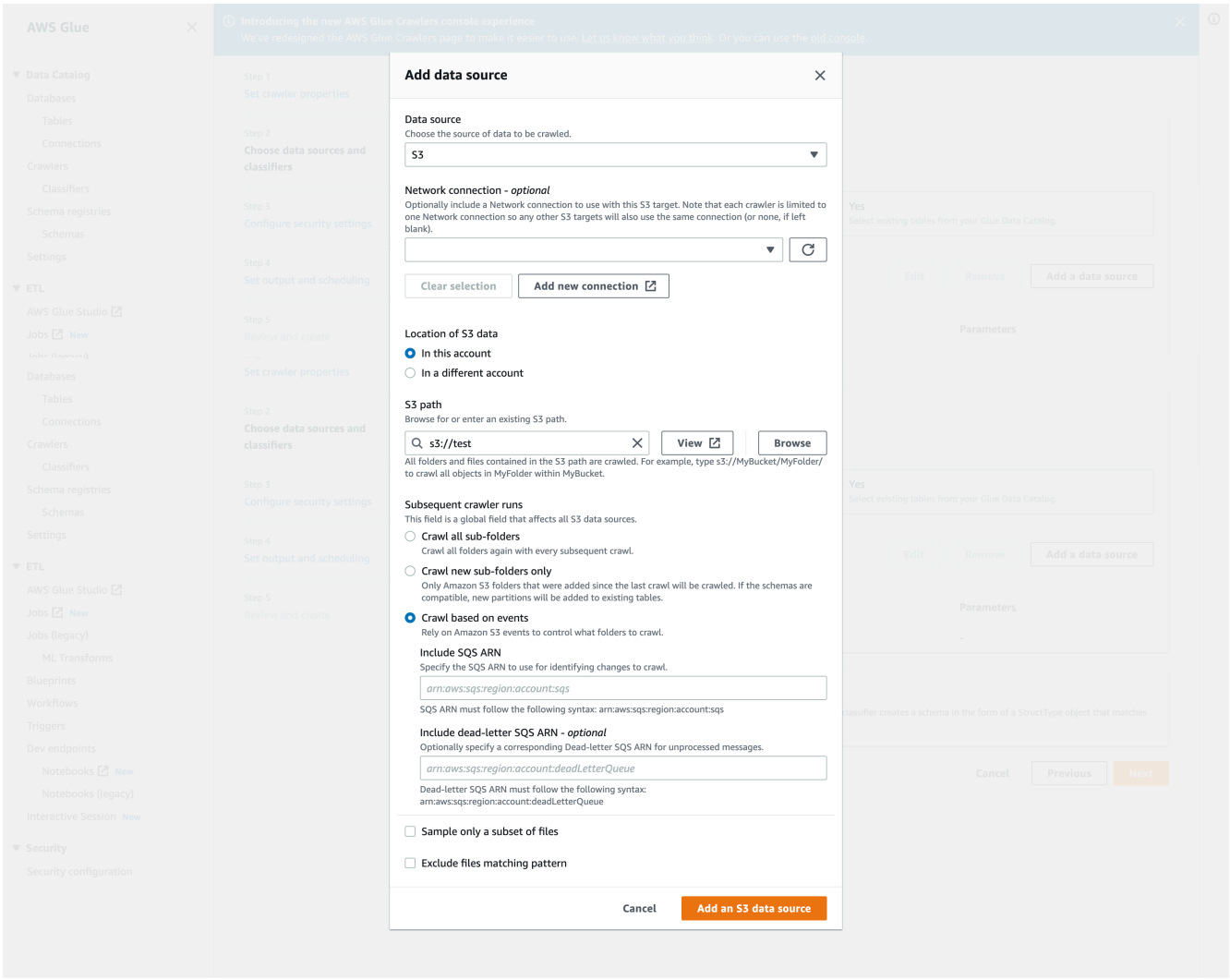
1. Connectez-vous à la GuardDuty console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/guardduty/.
2. Définissez les propriétés de votre crawler. Pour plus d'informations, consultez la rubrique Configuration des options de configuration du crawler dans la console AWS Glue.
3. Dans la section Configuration de source de données, la question Is your data already mapped to AWS Glue tables? (Vos données sont-elles déjà mappées aux tables ?) vous est posée.

   Par défaut, la réponse Not yet (Pas encore) est déjà sélectionnée. Laissez cette réponse par défaut, car vous utilisez une source de données Amazon S3 et les données ne sont pas encore mappées aux tables AWS Glue.
4. Dans la section Data sources (Sources de données), choisissez Add a data source (Ajouter une source de données).



5. Dans le modal Add data source (Ajouter une source de données), configurez la source de données Amazon S3 :

   - Data source (Source de données) : Amazon S3 est sélectionné par défaut.
   - Network connection (Connexion réseau) (Facultatif) : sélectionnez Add new connection (Ajouter une nouvelle connexion).
   - Location of Amazon S3 data (Emplacement des données Amazon S3) : par défaut, l'option In this account (Dans ce compte) est sélectionnée.
   - Amazon S3 path (Chemin d'accès Amazon S3) : spécifiez le chemin d'accès Amazon S3 où les dossiers et les fichiers sont analysés.
   - Subsequent crawler runs (Exécutions ultérieures du crawler) : choisissez Crawl based on events (Analyse en fonction des événements) pour utiliser les notifications d'événements Amazon S3 pour votre crawler.
   - Include SQS ARN (Inclure un ARN SQS) : spécifiez les paramètres du magasin de données, y compris l'ARN SQS valide. (Par exemple, `arn:aws:sqs:region:account:sqs`.)
   - Include dead-letter SQS ARN (Inclure un ARN SQS de lettres mortes) (Facultatif) : spécifiez un ARN SQS de lettres mortes Amazon valide. (Par exemple, `arn:aws:sqs:region:account:deadLetterQueue`.)
   - Choisissez Add an Amazon S3 data source (Ajouter une source de données Amazon S3).



AWS CLI

Voici un exemple d' AWS CLI appel Amazon S3 pour configurer un robot d'exploration afin qu'il utilise les notifications d'événements pour explorer un compartiment cible Amazon S3.

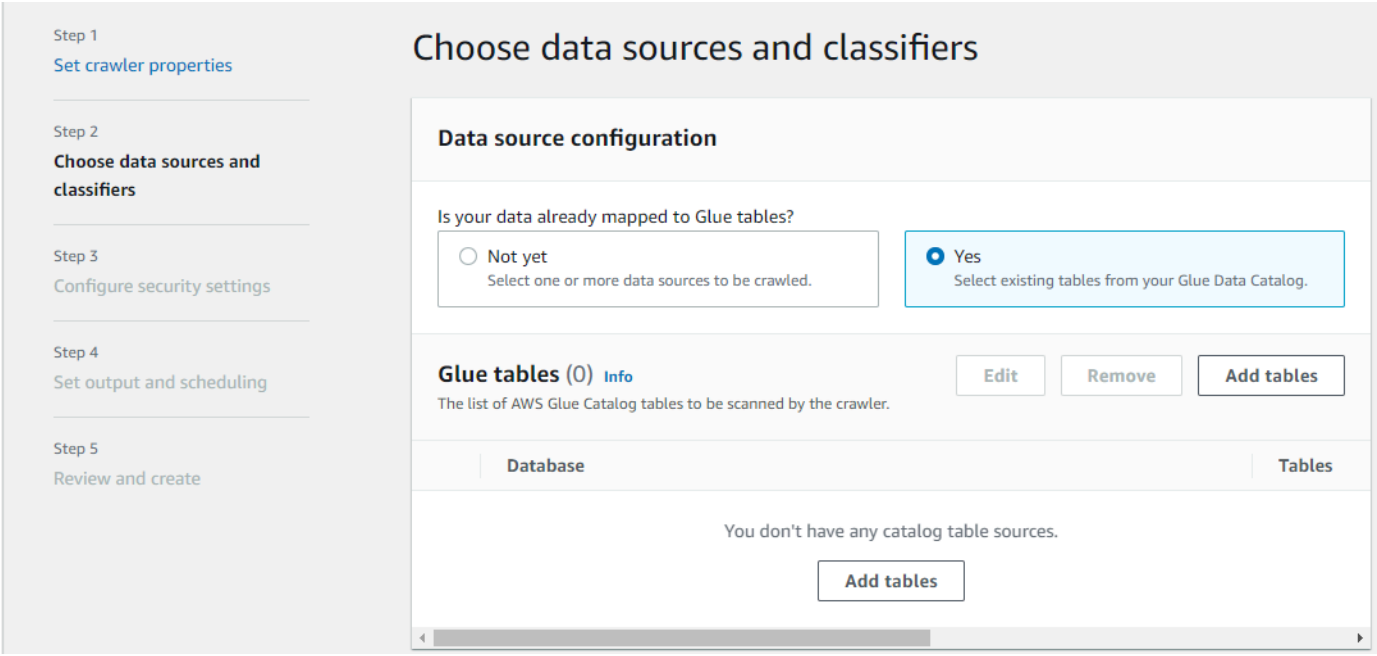
```
Create Crawler:
aws glue update-crawler \
    --name myCrawler \
    --recrawl-policy RecrawlBehavior=CRAWL_EVENT_MODE \
    --schema-change-policy UpdateBehavior=UPDATE_IN_DATABASE,DeleteBehavior=LOG
    --targets '{"S3Targets":[{"Path":"s3://amzn-s3-demo-bucket/", "EventQueueArn":
 "arn:aws:sqs:us-east-1:012345678910:MyQueue"}]}'
```

Configuration d'un robot pour les notifications d'événements Amazon S3 à l'aide de la table du catalogue de données

Lorsque vous avez une table du catalogue de données, configurez un robot pour les notifications d'événements Amazon S3 à l'aide de la console AWS Glue :

1. Définissez les propriétés de votre crawler. Pour plus d'informations, consultez la rubrique Configuration des options de configuration du crawler dans la console AWS Glue.
2. Dans la section Configuration de source de données, la question Is your data already mapped to AWS Glue tables? (Vos données sont-elles déjà mappées aux tables ?) vous est posée.

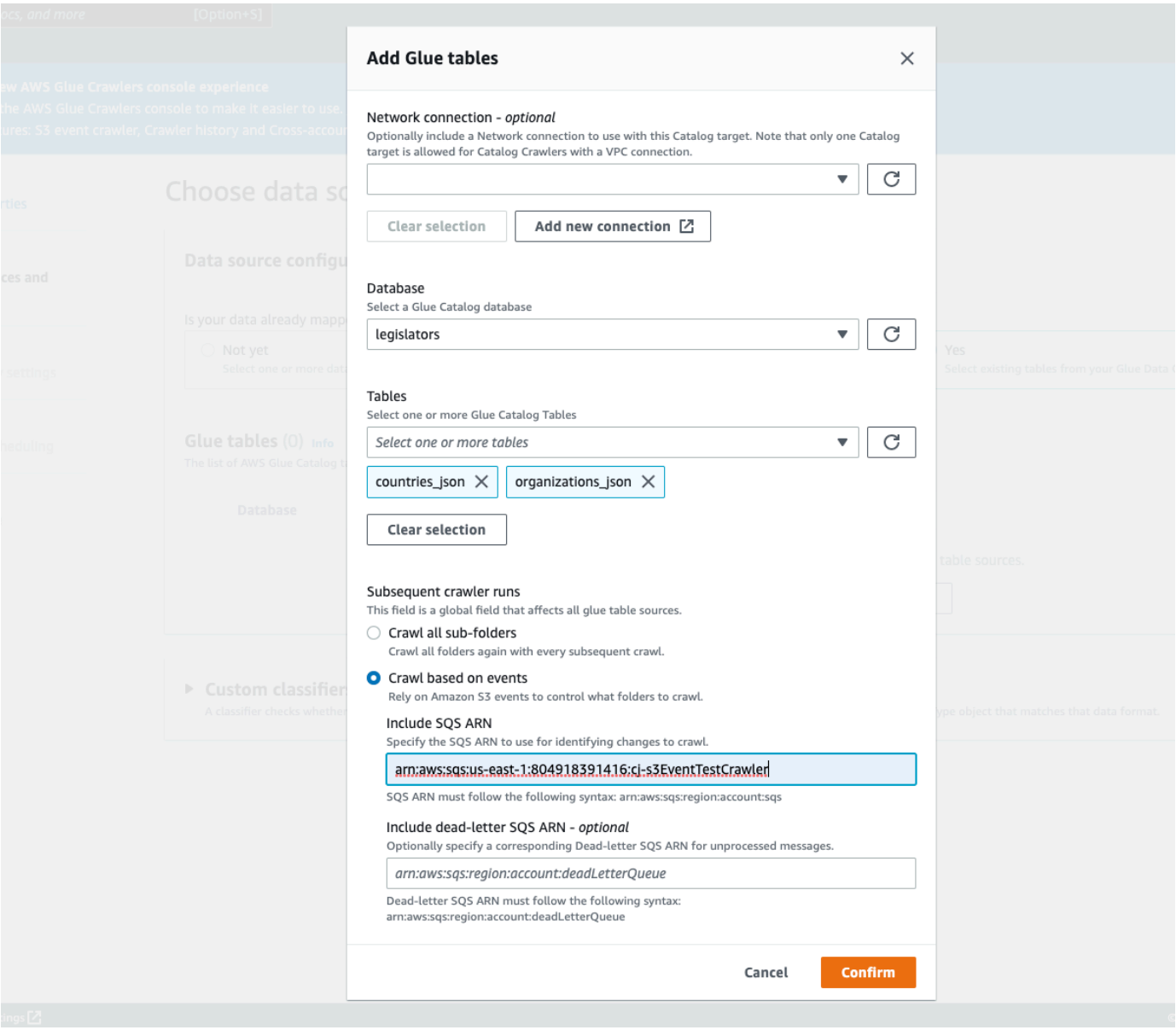
   Sélectionnez Yes (Oui) pour choisir les tables existantes de votre Catalogue de données comme source de données.
3. Dans la section Glue tables (Tables Glue), choisissez Add tables (Ajouter des tables).



4. Dans le modal Add table (Ajouter une table), configurez la base de données et les tables :

   - Network connection (Connexion réseau) (Facultatif) : sélectionnez Add new connection (Ajouter une nouvelle connexion).
   - Database (Base de données) : sélectionnez une base de données dans le Catalogue de données.

- Tables : sélectionnez une ou plusieurs tables de cette base de données dans le Catalogue de données.
- Subsequent crawler runs (Exécutions ultérieures du crawler) : choisissez Crawl based on events (Analyse en fonction des événements) pour utiliser les notifications d'événements Amazon S3 pour votre crawler.
- Include SQS ARN (Inclure un ARN SQS) : spécifiez les paramètres du magasin de données, y compris l'ARN SQS valide. (Par exemple, `arn:aws:sqs:region:account:sqs`.)
- Include dead-letter SQS ARN (Inclure un ARN SQS de lettres mortes) (Facultatif) : spécifiez un ARN SQS de lettres mortes Amazon valide. (Par exemple, `arn:aws:sqs:region:account:deadLetterQueue`.)
- Choisissez Confirmer.



Didacticiel : Ajout d'un crawler AWS Glue

Pour ce AWS Glue, il vous est demandé d'analyser les données d'arrivée des principaux transporteurs aériens afin de calculer la popularité des aéroports de départ d'un mois à l'autre. Vous disposez de données de vols pour l'année 2016 au format CSV stockées dans Amazon S3. Avant de transformer et d'analyser vos données, vous cataloguez ses métadonnées dans AWS Glue Data Catalog.

Dans ce didacticiel, ajoutons un crawler qui déduit les métadonnées de ces journaux de vol dans Amazon S3 et crée une table dans votre catalogue de données.

Rubriques

- Conditions préalables
- Étape 1 : Ajouter un crawler
- Étape 2 : Exécuter le crawler
- Étape 3 : Afficher les objets AWS Glue Data Catalog

Conditions préalables

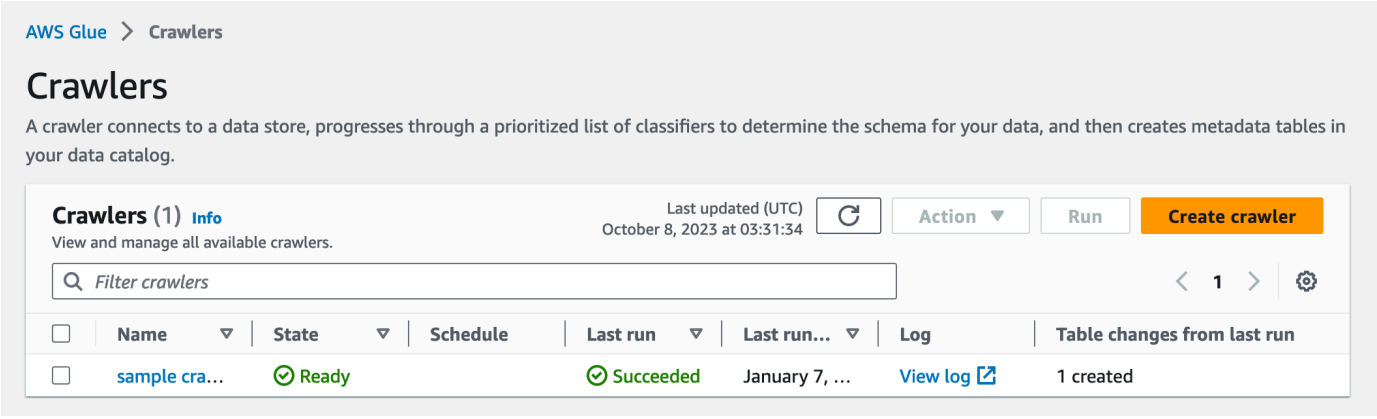
Ce didacticiel part du principe que vous disposez d'un AWS compte et que vous y avez accèsAWS Glue.

Étape 1 : Ajouter un crawler

Procédez comme suit pour configurer et exécuter un crawler qui extrait les métadonnées d'un fichier CSV stocké dans Amazon S3.

Pour créer un crawler qui lit les fichiers stockés sur Amazon S3

1. Sur la console AWS Glue de service, dans le menu de gauche, choisissez Crawlers.
2. Sur la page Crawlers, sélectionnez Créer un Crawler. Cela démarre une série de pages qui vous invitent à saisir les détails du crawler.



3. Dans le champ Nom de l'analyseur, saisissez **`Flights Data Crawler`**, et sélectionnez Suivant.

   Les crawlers invoquent les classifieurs pour déduire le schéma de vos données. Ce didacticiel utilise le classifieur intégré pour CSV par défaut.

4. Pour le type de source du crawler, sélectionnez Data stores (Magasins de données), puis Next (Suivant).
5. Maintenant, faisons pointer le crawler vers vos données. Sur la page Add a data store (Ajouter un magasin de données), sélectionnez le magasin de données Amazon S3. Ce tutoriel n'utilise pas de connexion, alors laissez le champ Connection (Connexion) vide s'il est visible.

   Pour l'option Crawl data in (Analyser les données dans), sélectionnez Specified path in another account (Chemin d'accès spécifié dans un autre compte). Ensuite, pour Inclure le chemin, saisissez le chemin où le crawler peut trouver les données de vol, qui est **`s3://crawler-public-us-east-1/flight/2016/csv`**. Après avoir saisi le chemin d'accès, le titre de ce champ devient Include path (Inclure le chemin). Choisissez Suivant.
6. Vous pouvez analyser plusieurs magasins de données avec un seul crawler. Cependant, dans ce tutoriel, nous n'utilisons qu'un seul magasin de données, alors sélectionnez No (Non), puis Next (Suivant).
7. Le crawler a besoin d'autorisations pour accéder au magasin de données et créer des objets dans le répertoire AWS Glue Data Catalog. Pour configurer ces autorisations, sélectionnez Create an IAM role (Créer un rôle IAM). Le nom du rôle IAM commence par `AWSGlueServiceRole-`, et dans le champ, vous saisissez la dernière partie du nom du rôle. Saisissez **`CrawlerTutorial`**, puis sélectionnez Save (Enregistrer).

   > **Note**
   >
   > Pour créer un rôle IAM, votre utilisateur AWS doit avoir les autorisations `CreateRole`, `CreatePolicy` et `AttachRolePolicy`.

   L'assistant crée un rôle IAM nomméAWSGlueServiceRole-CrawlerTutorial, associe la politique AWS gérée `AWSGlueServiceRole` à ce rôle et ajoute une politique en ligne qui autorise l'accès en lecture à l'emplacement Amazon S3. `s3://crawler-public-us-east-1/flight/2016/csv`
8. Créez une planification pour le crawler. Pour Frequency (Fréquence), sélectionnez Run on demand (Exécuter à la demande), puis Next (Suivant).
9. Les crawlers créent des tables dans votre catalogue de données. Les tables sont contenues dans une base de données dans le catalogue de données. Tout d'abord, sélectionnez Add database (Ajouter une base de données) pour créer une base de données. Dans la

fenêtre contextuelle, saisissez **`test-flights-db`** pour le nom de la base de données, puis sélectionnez Create (Créer).

Ensuite, saisissez **`flights`** pour Prefix added to tables (Préfixe ajouté aux tables). Utilisez les valeurs par défaut pour le reste des champs et sélectionnez Next (Suivant).

10. Vérifiez les choix que vous avez fait dans le panneau Add crawler (Ajouter un analyseur) de l'assistant. Si vous voyez des erreurs, vous pouvez choisir Back (Retour) pour revenir aux pages précédentes et apporter des modifications.

    Après avoir examiné les informations, sélectionnez Terminer pour créer le crawler.

Étape 2 : Exécuter le crawler

Après avoir créé un crawler, l'assistant vous envoie à la page d'affichage des Crawlers. Étant donné que vous créez le crawler avec une planification à la demande, vous avez la possibilité d'exécuter le crawler.

Pour exécuter un crawler

1. La bannière située en haut de cette page vous indique que le crawler a été créé et vous demande si vous souhaitez l'exécuter maintenant. Sélectionnez Lancez-le maintenant ? pour exécuter le crawler.

   La bannière change pour afficher les messages « Tentative d'exécution » et « Exécution » pour votre crawler. Une fois que le crawler démarre, la bannière disparaît et l'affichage du crawler est mis à jour pour afficher l'état Starting (Démarrage) de celui-ci. Après une minute, vous pouvez cliquer sur l'icône Actualiser pour mettre à jour l'état du crawler qui s'affiche dans le tableau.
2. Une fois que le crawler a terminé sa tâche, une nouvelle bannière s'affiche qui décrit les modifications apportées par le crawler. Vous pouvez choisir le test-flights-dblien pour afficher les objets du catalogue de données.

Étape 3 : Afficher les objets AWS Glue Data Catalog

Le crawler lit les données à l'emplacement source et crée des tables dans le catalogue de données. Une table est une définition de métadonnées qui représente vos données. Les tables du catalogue de données ne contiennent pas de données. Au lieu de cela, vous utilisez ces tables comme source ou cible dans une définition de tâche.

Pour afficher les objets du catalogue de données créés par le crawler.

1. Dans le panneau de navigation de gauche, sous Data catalog (Catalogue de données), sélectionnez Databases (Bases de données). Ici, vous pouvez voir la base de données `flights-db` qui est créée par le crawler.
2. Dans le panneau de navigation de gauche, sous Data catalog (Catalogue de données), et sous Databases (Bases de données), sélectionnez Tables. Ici, vous pouvez voir la table `flightscsv` qui est créée par le crawler. Si vous sélectionnez le nom de la table, vous pouvez afficher la configuration, les paramètres et les propriétés de la table. En faisant défiler cette vue vers le bas, vous pouvez afficher le schéma, qui est une information sur les colonnes et les types de données de la table.
3. Si vous sélectionnez View partitions (Afficher les partitions) sur la page d'affichage de la table, vous pouvez voir les partitions créées pour les données. La première colonne est la clé de partition.

Définition manuelle des métadonnées

Le catalogue de AWS Glue données est un référentiel central qui stocke les métadonnées relatives à vos sources de données et à vos ensembles de données. Bien qu'un robot puisse automatiquement explorer et renseigner les métadonnées des sources de données prises en charge, il existe certains scénarios dans lesquels vous devrez peut-être définir les métadonnées manuellement dans le catalogue de données :

- Formats de données non pris en charge : si certaines de vos sources de données ne sont pas prises en charge par le robot, vous devez définir manuellement les métadonnées de ces sources de données dans le catalogue de données.
- Exigences relatives aux métadonnées personnalisées — Les métadonnées AWS Glue crawler sont déduites en fonction de règles et de conventions prédéfinies. Si vous avez des exigences spécifiques en matière de métadonnées qui ne sont pas couvertes par les métadonnées AWS Glue crawler déduites, vous pouvez définir manuellement les métadonnées pour répondre à vos besoins
- Gouvernance des données et normalisation : dans certains cas, vous souhaiterez peut-être mieux contrôler les définitions des métadonnées pour des raisons de gouvernance, de conformité ou de sécurité des données. La définition manuelle des métadonnées vous permet de vous assurer que celles-ci sont conformes aux normes et aux politiques de votre organisation.
- Espace réservé pour une future ingestion de données : si certaines de vos sources de données ne sont pas immédiatement disponibles ou accessibles, vous pouvez créer des tables de schéma

vides en tant qu'espaces réservés. Une fois les sources de données disponibles, vous pouvez remplir les tables avec les données réelles, tout en conservant la structure prédéfinie.

Pour définir les métadonnées manuellement, vous pouvez utiliser la AWS Glue console, la console Lake Formation, AWS Glue l'API ou le AWS Command Line Interface (AWS CLI). Vous pouvez créer des bases de données, des tables et des partitions, et spécifier des propriétés de métadonnées telles que des noms de colonnes, des types de données, des descriptions et d'autres attributs.

Création de bases de données

Les bases de données sont utilisées pour organiser les tables de métadonnées dans AWS Glue. Lorsque vous définissez une table dans le AWS Glue Data Catalog, vous l'ajoutez à une base de données. Une table ne peut se trouver que dans une seule base de données à la fois.

Votre base de données peut contenir des tables qui définissent les données de différents magasins de données. Ces données peuvent inclure des objets dans Amazon Simple Storage Service (Amazon S3) et des tables relationnelles dans Amazon Relational Database Service.

> **Note**
>
> Lorsque vous supprimez une base de données AWS Glue, toutes les tables que celle-ci contient sont également supprimées.

Pour consulter la liste des bases de données, connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/. Choisissez Bases de données, puis choisissez un nom de base de données dans la liste pour afficher les détails.

Sous l'onglet Bases de données de la console AWS Glue, vous pouvez ajouter, modifier ou supprimer des bases de données :

- Pour créer une base de données, choisissez Ajouter une base de données et fournissez un nom et une description. Pour la compatibilité avec d'autres magasins de métadonnées, tels qu'Apache Hive, le nom est en lettres minuscules.

> **Note**
>
> Si vous prévoyez d'accéder à la base de données depuis Amazon Athena, fournissez un nom contenant uniquement des caractères alphanumériques et des traits de soulignement. Pour plus d'informations, consultez la rubrique Noms Athena.

- Pour modifier la description d'une base de données, cochez la case située en regard du nom de base de données et choisissez Edit (Modifier).
- Pour supprimer une base de données, cochez la case située en regard du nom de base de données et choisissez Remove (Supprimer).
- Pour afficher la liste des tables contenues dans la base de données, choisissez le nom de la base de données et les propriétés de la base de données afficheront alors toutes les tables de la base de données.

Pour modifier la base de données sur laquelle un crawler écrit, vous devez modifier la définition de l'crawler. Pour de plus amples informations, veuillez consulter Utilisation de robots pour alimenter le catalogue de données.

Liens de ressources de base de données

La console AWS Glue a été récemment mise à jour. La version actuelle de la console ne prend pas en charge les liens de ressources de base de données.

Le catalogue de données peut également contenir des liens de ressources vers des bases de données. Un lien de ressource de base de données est un lien vers une base de données locale ou partagée. Actuellement, vous ne pouvez créer des liens de ressources que dans AWS Lake Formation. Après avoir créé un lien de ressource vers une base de données, vous pouvez utiliser le nom du lien de ressource partout où vous utiliseriez le nom de la base de données. Avec les bases de données que vous possédez ou qui sont partagées avec vous, les liens de ressources de base de données sont renvoyés par `glue:GetDatabases()` et apparaissent sous la forme d'entrées sur la page des Bases de données de la console AWS Glue.

Le catalogue de données peut également contenir des liens de ressources de table.

Pour plus d'informations sur les liens de ressources, veuillez consulter la rubrique Création de liens de ressources dans le Guide du développeur AWS Lake Formation .

Création de tables

Même si l'exécution d'un robot est la méthode recommandée pour inventorier les données de vos magasins de données, vous pouvez y ajouter des tables de métadonnées AWS Glue Data Catalog manuellement. Cette approche vous permet de mieux contrôler les définitions des métadonnées et de les personnaliser en fonction de vos besoins spécifiques.

Vous pouvez également ajouter manuellement des tables au catalogue de données de la manière suivante :

- Utilisez la console AWS Glue pour créer manuellement une table dans l' AWS Glue Data Catalog. Pour de plus amples informations, veuillez consulter Création de tables à l'aide de la console.
- Utilisez l'opération `CreateTable` de l'AWS Glue API pour créer une table dans le AWS Glue Data Catalog. Pour de plus amples informations, veuillez consulter CreateTable action (Python : créer_table).
- Utilisez CloudFormation des modèles. Pour de plus amples informations, veuillez consulter AWS CloudFormation pour AWS Glue.

Lorsque vous définissez une table manuellement à l'aide de la console ou d'une API, vous spécifiez le schéma de table et la valeur d'un champ de classification qui indique le type et le format des données de la source de données. Si un crawler crée la table, le format et le schéma des données sont déterminés par un classifieur intégré ou un classifieur personnalisé. Pour plus d'informations sur la création d'une table à l'aide de la console AWS Glue, consultez Création de tables à l'aide de la console.

Rubriques

- Partitions de table
- Liens de ressources de table
- Création de tables à l'aide de la console
- Création d'index de partition
- Mise à jour de tables Data Catalog créées manuellement à l'aide d'crawlers
- Propriétés de la table Catalogue de données

Partitions de table

Une définition de table AWS Glue d'un dossier Amazon Simple Storage Service (Amazon S3) peut décrire une table partitionnée. Par exemple, pour améliorer la performance des requêtes, une table partitionnée peut séparer les données mensuelles dans différents fichiers en utilisant le nom du mois en tant que clé. Dans AWS Glue, les définitions de table incluent la clé de partitionnement d'une table. Lorsque AWS Glue évalue les données dans les dossiers Amazon S3 pour faire l'inventaire d'une table, il détermine si une table individuelle ou une table partitionnée est ajoutée.

Vous pouvez créer des index de partition sur une table pour récupérer un sous-ensemble des partitions au lieu de charger toutes les partitions de la table. Pour en savoir plus sur l'utilisation des index de partition, consultez Création d'index de partition.

Toutes les conditions suivantes doivent être remplies pour que AWS Glue crée une table partitionnée pour un dossier Amazon S3 :

- Les schémas des fichiers sont similaires, comme déterminé par AWS Glue.
- Le format de données des fichiers est le même.
- Le format de compression des fichiers est le même.

Par exemple, imaginons que vous possédez un compartiment Amazon S3 nommé `my-app-bucket`, où vous stockez des données de vente d'applications iOS et Android. Les données sont partitionnées par année, mois et jour. Les fichiers de données pour les ventes iOS et Android ont le même schéma, format de données et format de compression. Dans le AWS Glue Data Catalog, le AWS Glue robot crée une définition de table avec des clés de partitionnement pour l'année, le mois et le jour.

La liste Amazon S3 `my-app-bucket` suivante présente certaines partitions. Le symbole = est utilisé pour attribuer des valeurs de clé de partition.

```
my-app-bucket/Sales/year=2010/month=feb/day=1/iOS.csv
my-app-bucket/Sales/year=2010/month=feb/day=1/Android.csv
my-app-bucket/Sales/year=2010/month=feb/day=2/iOS.csv
my-app-bucket/Sales/year=2010/month=feb/day=2/Android.csv
...
my-app-bucket/Sales/year=2017/month=feb/day=4/iOS.csv
my-app-bucket/Sales/year=2017/month=feb/day=4/Android.csv
```

Liens de ressources de table

La console AWS Glue a été récemment mise à jour. La version actuelle de la console ne prend pas en charge les liens de ressources de table.

Le catalogue de données peut également contenir des liens de ressources vers des tables. Un lien de ressource de table est un lien vers une table locale ou partagée. Actuellement, vous ne pouvez créer des liens de ressources que dans AWS Lake Formation. Après avoir créé un lien de ressource vers une table, vous pouvez utiliser le nom du lien de ressource partout où vous utiliseriez le nom de la table. Avec les tables que vous possédez ou qui sont partagées avec vous, les liens de ressources de table sont renvoyés par `glue:GetTables()` et apparaissent sous la forme d'entrées sur la page Tables de la console AWS Glue.

Le catalogue de données peut également contenir des liens de ressources de base de données.

Pour plus d'informations sur les liens de ressources, veuillez consulter la rubrique Création de liens de ressources dans le Guide du développeur AWS Lake Formation .

Création de tables à l'aide de la console

Un tableau figurant dans le AWS Glue Data Catalog est la définition des métadonnées qui représente les données d'un magasin de données. Vous créez des tables lorsque vous exécutez un crawler ; vous pouvez aussi créer une table manuellement dans la console AWS Glue . La liste Tables de la console AWS Glue affiche les valeurs des métadonnées de votre table. Les définitions de table vous permettent de spécifier des sources et des cibles lors de la création de tâches ETL (extraction, transformation et chargement).

> **Note**
>
> Compte tenu des récentes modifications apportées à la console de AWS gestion, il se peut que vous deviez modifier vos rôles IAM existants pour `SearchTables`obtenir l'autorisation. Pour la création de nouveaux rôles, l'autorisation d'API `SearchTables` a déjà été ajoutée par défaut.

Pour commencer, connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/. Choisissez l'onglet Tables et utilisez le bouton Add

tables (Ajouter des tables) pour créer des tables avec un crawler ou en saisissant manuellement les attributs.

Ajout de tableaux sur la console

Pour utiliser un crawler afin d'ajouter des tables, choisissez Add tables (Ajouter des tables), puis Add tables using a crawler (Ajouter des tables à l'aide d'un crawler). Ensuite, suivez les instructions fournies dans l'assistant Add crawler (Ajouter un crawler). Une fois que l'crawler s'exécute, les tables sont ajoutées à l' AWS Glue Data Catalog. Pour de plus amples informations, veuillez consulter Utilisation de robots pour alimenter le catalogue de données.

Si vous connaissez les attributs requis pour créer une définition de table Amazon Simple Storage Service (Amazon S3) dans votre Data Catalog, vous pouvez la créer avec l'assistant de table. Choisissez Add tables (Ajouter des tables), Add table manually (Ajouter une table manuellement), et suivez les instructions fournies dans l'assistant Add table (Ajouter une table).

Lors de l'ajout manuel d'une table à l'aide de la console, tenez compte des points suivants :

- Si vous prévoyez d'accéder à la table depuis Amazon Athena, fournissez un nom contenant uniquement des caractères alphanumériques et des traits de soulignement. Pour plus d'informations, consultez Noms Athena.
- L'emplacement de vos données sources doit être un chemin d'accès Amazon S3.
- Le format des données doit correspondre à l'un des formats répertoriés dans l'assistant. La classification correspondante et SerDe les autres propriétés du tableau sont automatiquement renseignées en fonction du format choisi. Vous pouvez définir des tables aux formats suivants :

  Avro

  Format binaire JSON Apache Avro.

  CSV

  Valeurs séparées par des caractères. Vous spécifiez également le délimiteur de virgule, de tube, de point-virgule, de tabulation ou de Ctrl-A.

  JSON

  JavaScript Notation d'objets.

  xml

  Format XML (Extensible Markup Language). Spécifiez la balise XML qui définit une ligne dans les données. Les colonnes sont définies dans des balises de ligne.

Parquet

Stockage en colonnes Apache Parquet.

ORC

Format de fichier Optimized Row Columnar (ORC). Un format conçu pour stocker efficacement les données Hive.

- Vous pouvez définir une clé de partition pour la table.
- Actuellement, les tables partitionnées que vous créez avec la console ne peuvent pas être utilisées dans les tâches ETL.

Attributs des tables

Les attributs suivants font partie des plus importants de votre table :

Nom

Le nom est déterminé lorsque la table est créée, et vous ne pouvez pas le modifier. Vous vous référez à un nom de table dans de nombreuses opérations AWS Glue.

Base de données

Objet conteneur dans lequel se trouve votre table. Cet objet contient une organisation de vos tables qui existe au sein de votre banque de données AWS Glue Data Catalog et qui peut être différente de celle d'une organisation dans votre magasin de données. Lorsque vous supprimez une base de données, toutes les tables que celle-ci contient sont également supprimées de Data Catalog.

Description

Description de la table. Vous pouvez écrire une description vous aidant à comprendre le contenu de la table.

Format de tableau

Spécifiez la création d'une AWS Glue table standard ou d'une table au format Apache Iceberg.

Le catalogue de données fournit les options d'optimisation des tables suivantes pour gérer le stockage des tables et améliorer les performances des requêtes pour les tables Iceberg.

- Compactage : les fichiers de données sont fusionnés et réécrits pour supprimer les données obsolètes et regrouper les données fragmentées dans des fichiers plus volumineux et plus efficaces.
- Conservation des instantanés : les instantanés sont des versions horodatées d'une table Iceberg. Les configurations de conservation des instantanés permettent aux clients de définir la durée de conservation des instantanés et le nombre d'instantanés à conserver. La configuration d'un optimiseur de conservation des instantanés peut aider à gérer la charge de stockage en supprimant les instantanés plus anciens et inutiles, ainsi que leurs fichiers sous-jacents associés.
- Suppression de fichiers orphelins : les fichiers orphelins sont des fichiers qui ne sont plus référencés par les métadonnées de la table Iceberg. Ces fichiers peuvent s'accumuler au fil du temps, en particulier après des opérations telles que la suppression de tables ou l'échec de tâches ETL. L'activation de la suppression des fichiers orphelins permet AWS Glue d'identifier et de supprimer périodiquement ces fichiers inutiles, libérant ainsi de l'espace de stockage.

Pour de plus amples informations, veuillez consulter Optimisation des tables Iceberg.

Configuration de l'optimisation

Vous pouvez utiliser les paramètres par défaut ou personnaliser les paramètres pour activer les optimiseurs de table.

Rôle IAM

Pour exécuter les optimiseurs de table, le service endosse un rôle IAM en votre nom. Vous pouvez choisir un rôle IAM à l'aide de la liste déroulante. Assurez-vous que le rôle dispose des autorisations requises pour activer le compactage.

Pour en savoir plus sur les autorisations requises pour le rôle IAM, consultez Conditions préalables requises pour l'optimisation des tables.

Location

Pointeur de l'emplacement des données dans un magasin de données que cette définition de table représente.

Classification

Valeur de catégorie fournie lors de la création de la table. En général, elle est écrite lors de l'exécution d'un crawler et spécifie le format de la source des données.

Dernière mise à jour

Date et heure (UTC) auxquelles cette table a été mise à jour dans Data Catalog.

Date ajoutée

Date et heure (UTC) auxquelles cette table a été ajoutée dans Data Catalog.

Obsolète

Si AWS Glue détecte qu'une table du Data Catalog n'existe plus dans son magasin de données d'origine, il la marque comme obsolète dans le catalogue de données. Si vous exécutez une tâche qui fait référence à une table obsolète, la tâche peut échouer. Modifiez les tâches qui font référence à des tables obsolètes pour les supprimer en tant que sources et cibles. Nous vous recommandons de supprimer les tables obsolètes lorsqu'ils ne sont plus nécessaires.

Connexion

Si AWS Glue nécessite une connexion à votre magasin de données, le nom de la connexion est associé à la table.

Affichage et gestion des détails d’une table

Pour afficher les détails d'une table existante, choisissez le nom de la table dans la liste, puis Action, View details (Action, Afficher les détails).

Les détails de la table comprennent les propriétés de votre table et son schéma. Cette vue affiche le schéma de la table, y compris les noms de colonnes dans l'ordre défini pour la table, les types de données et les colonnes de clés pour les partitions. Si une colonne est de type complexe, vous pouvez choisir View properties (Afficher les propriétés) pour afficher les détails de la structure de ce champ, comme illustré dans l'exemple suivant :

```
{
"StorageDescriptor":
    {
      "cols": {
         "FieldSchema": [
           {
             "name": "primary-1",
             "type": "CHAR",
             "comment": ""
           },
```

```
            {
              "name": "second ",
              "type": "STRING",
              "comment": ""
            }
          ]
      },
      "location": "s3://aws-logs-111122223333-us-east-1",
      "inputFormat": "",
      "outputFormat": "org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat",
      "compressed": "false",
      "numBuckets": "0",
      "SerDeInfo": {
           "name": "",
           "serializationLib": "org.apache.hadoop.hive.serde2.OpenCSVSerde",
           "parameters": {
               "separatorChar": "|"
            }
      },
      "bucketCols": [],
      "sortCols": [],
      "parameters": {},
      "SkewedInfo": {},
      "storedAsSubDirectories": "false"
    },
    "parameters": {
       "classification": "csv"
    }
}
```

Pour plus d'informations sur les propriétés d'une table, comme `StorageDescriptor`, consultez StorageDescriptor structure.

Pour modifier le schéma d'une table, choisissez Edit schema (Modifier le schéma) pour ajouter et supprimer des colonnes, modifier les noms de colonnes et modifier les types de données.

Pour comparer différentes versions d'une table, y compris son schéma, choisissez Comparer les versions pour voir une side-by-side comparaison des deux versions du schéma d'une table. Pour de plus amples informations, veuillez consulter Comparaison des versions de schéma de table.

Pour afficher les fichiers qui constituent une partition Amazon S3, sélectionnez View partition (Afficher la partition). Pour les tables Amazon S3, la colonne Key (Clé) affiche les clés de partition qui sont utilisées pour partitionner la table dans le magasin de données source. Le partitionnement permet de

diviser un tableau en parties connexes en fonction des valeurs d'une colonne clé, telle que la date, le lieu ou le département. Pour plus d'informations sur les partitions, effectuez une recherche sur Internet pour en savoir plus sur « le partitionnement Hive ».

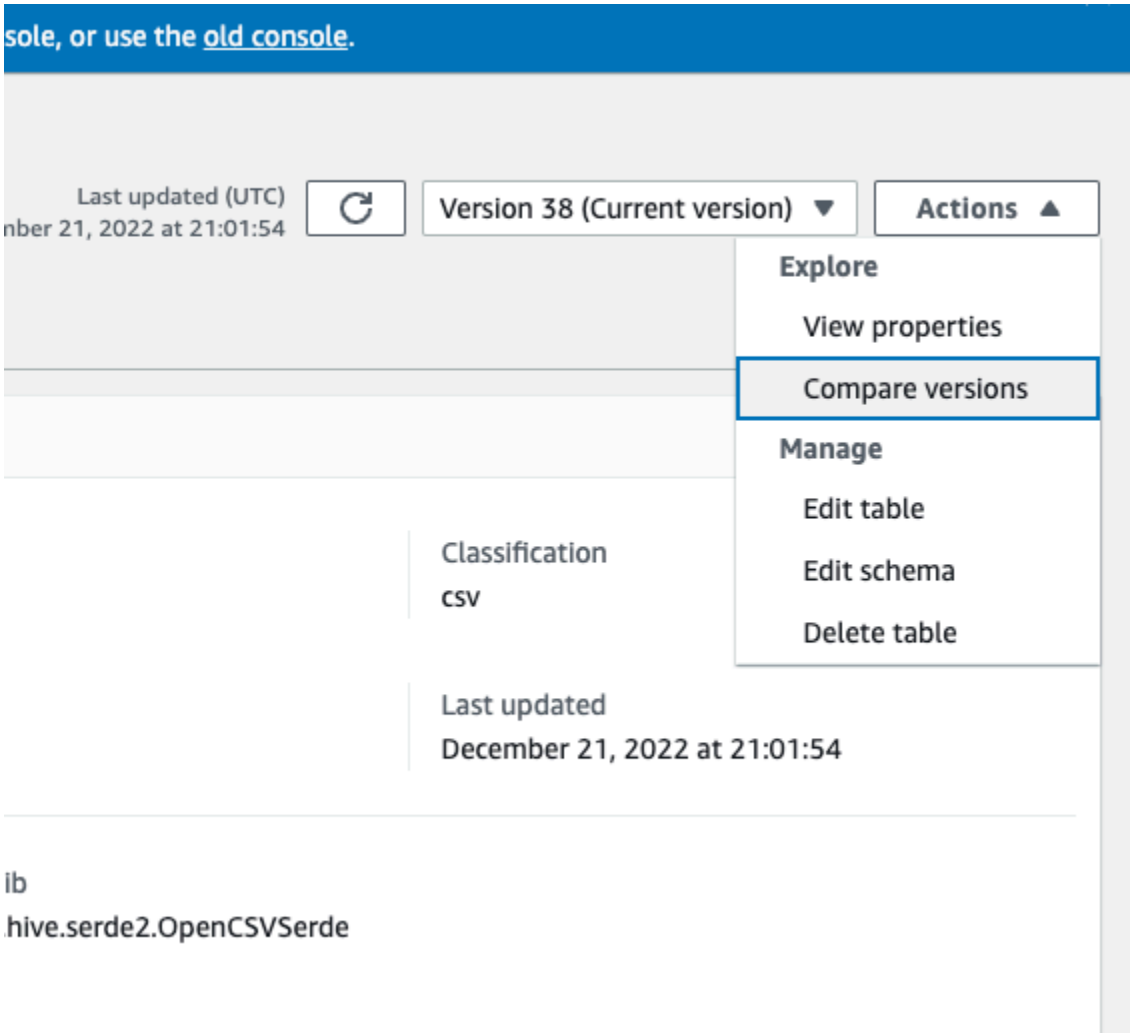
> **Note**
>
> Pour obtenir step-by-step des conseils sur l'affichage des détails d'un tableau, consultez le didacticiel Explore le tableau dans la console.

Comparaison des versions de schéma de table

Lorsque vous comparez deux versions de schémas de table, vous pouvez comparer les modifications des lignes imbriquées en développant et en réduisant les lignes imbriquées, comparer les schémas de deux versions et afficher les side-by-side propriétés des tables. side-by-side

Pour comparer les versions

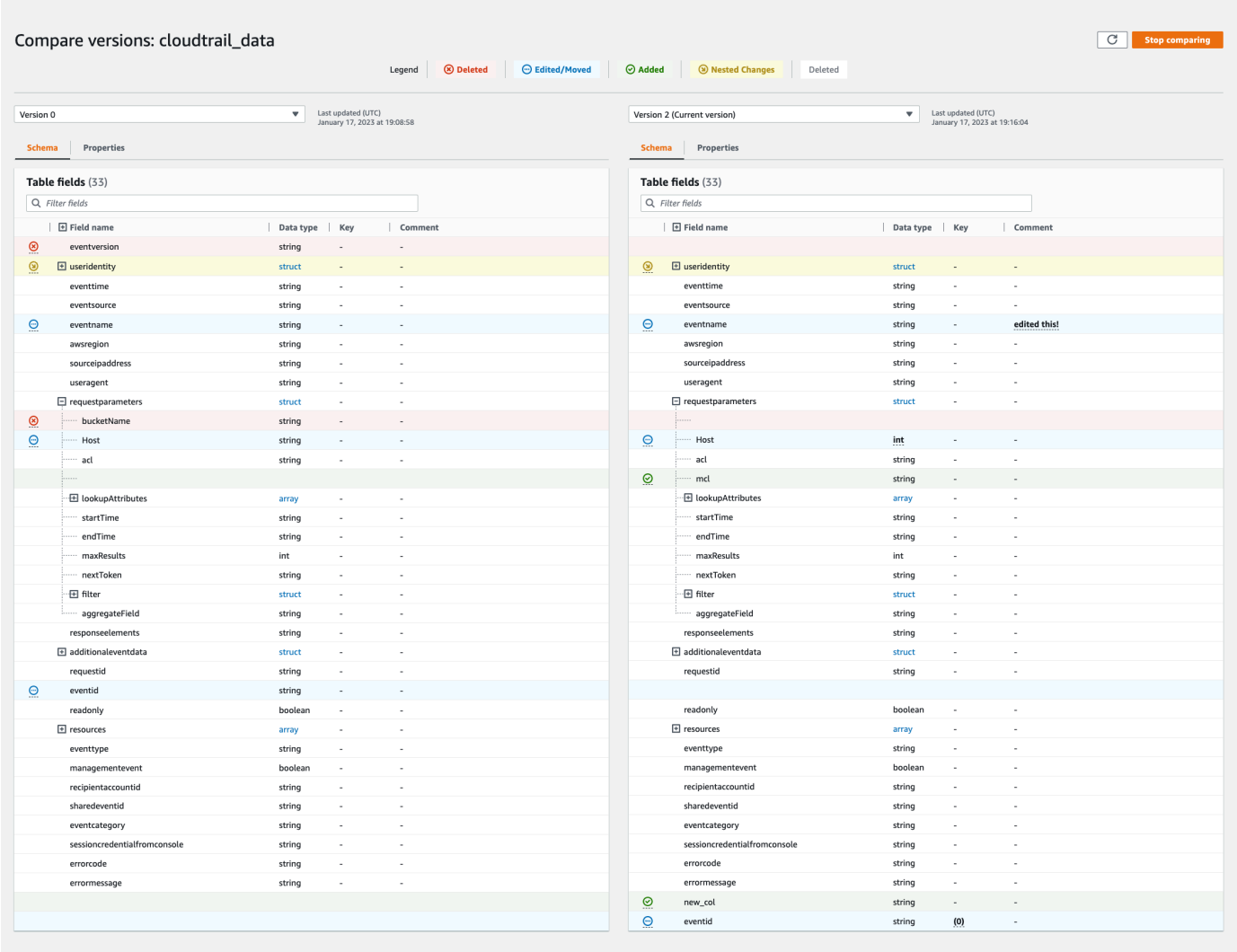
1. Dans la console AWS Glue, choisissez Tables, puis Actions et choisissez Comparer les versions.



2. Choisissez une version à comparer en sélectionnant le menu déroulant des versions. Lorsque vous comparez des schémas, l'onglet Schéma est surligné en orange.
3. Lorsque vous comparez des tables entre deux versions, les schémas des tables s'affichent à gauche et à droite de l'écran. Cela vous permet de déterminer visuellement les modifications en comparant le nom de colonne, le type de données, la clé et les champs de commentaire side-by-side. En cas de modification, une icône colorée indique le type de modification apportée.

   - Supprimé : représenté par une icône rouge, indique l'endroit où la colonne a été supprimée d'une version précédente du schéma de table.
   - Modifié ou déplacé : représenté par une icône bleue, indique l'endroit où la colonne a été modifiée ou déplacée dans une version plus récente du schéma de table.
   - Ajouté : représenté par une icône verte, indique l'endroit où la colonne a été ajoutée à une version plus récente du schéma de table.

- Modifications imbriquées : représentées par une icône jaune, indiquent où se trouvent les modifications dans la colonne imbriquée. Choisissez la colonne à développer et affichez les colonnes qui ont été supprimées, modifiées, déplacées ou ajoutées.



4. Utilisez la barre de recherche des champs de filtre pour afficher les champs en fonction des caractères que vous saisissez ici. Si vous saisissez un nom de colonne dans l'une ou l'autre des versions de table, les champs filtrés s'affichent dans les deux versions de la table pour vous indiquer l'emplacement des modifications.
5. Pour comparer les propriétés, cliquez sur l'onglet Propriétés.
6. Pour arrêter la comparaison des versions, choisissez Arrêter la comparaison pour revenir à la liste des tables.

Création d'index de partition

Au fil du temps, des centaines de milliers de partitions sont ajoutées à une table. L'GetPartitions API est utilisée pour récupérer les partitions de la table. L'API renvoie des partitions qui correspondent à l'expression fournie dans la demande.

Prenons comme exemple une table sales_data qui est partitionnée par les clés Country, Category, Year, Month et creationDate. Si vous souhaitez obtenir des données de ventes pour tous les articles vendus pour la catégorie Livres en 2020, après le 2020-08-15, vous devez faire une demande `GetPartitions` avec l'expression « Category = 'Books' and creationDate > '2020-08-15' » au catalogue de données.

Si aucun index de partition n'est présent sur la table, AWS Glue charge toutes les partitions de la table, puis filtre les partitions chargées à l'aide de l'expression de requête fournie par l'utilisateur dans la demande `GetPartitions`. L'exécution de la requête prend plus de temps à mesure que le nombre de partitions augmente sur une table sans index. Avec un index, la requête `GetPartitions` essaiera de récupérer un sous-ensemble des partitions au lieu de charger toutes les partitions de la table.

Rubriques

- À propos des index de partition
- Création d'un tableau avec index de partition
- Ajout d'un index de partition à une table existante
- Description des indexes de partition sur une table
- Limitations de l'utilisation des index de partition
- Utilisation d'index pour un appel optimisé GetPartitions
- Intégration avec les moteurs

À propos des index de partition

Lorsque vous créez un index de partition, vous spécifiez une liste de clés de partition qui existent déjà sur une table donnée. L'index de partition est une sous-liste des clés de partition définies dans la table. Un index de partition peut être créé sur n'importe quelle permutation de clés de partition définie sur la table. Pour la table sales_data ci-dessus, les index possibles sont (country, category, creationDate), (country, category, year), (country, category), (country), (category, country, year, month), etc.

Data Catalog concaténera les valeurs de partition dans l'ordre fourni au moment de la création de l'index. L'index est créé de manière cohérente à mesure que des partitions sont ajoutées à la table. Des index peuvent être créés pour les types de colonnes String (string, char et varchar) Numeric (int, bigint, long, tinyint, and smallint) et Date (yyyy-MM-dd).

Types de données pris en charge

- Date : date au format ISO, par exemple `YYYY-MM-DD`. Par exemple, date `2020-08-15`. Le format utilise des tirets (-) pour séparer l'année, le mois et le jour. La plage de dates autorisée pour l'indexation s'étend de `0000-01-01` à `9999-12-31`.
- String : libellé chaîne entre guillemets simples ou doubles.
- Char : données en caractères de longueur fixe, avec une longueur spécifiée entre 1 et 255, par exemple char(10).
- Varchar : données en caractères de longueur variable, avec une longueur spécifiée entre 1 et 65535, par exemple varchar(10).
- Numeric : int, bigint, long, tinyint, and smallint

Les index sur les types de données Numeric, String et Date prennent en charge les opérateurs **=**, **>**, **>=**, **<**, **<=** et BETWEEN. La solution d'indexation ne prend actuellement en charge que l'opérateur logique `AND`. Les sous-expressions avec les opérateurs « LIKE », « IN », « OR » et « NOT » sont ignorées dans l'expression pour le filtrage à l'aide d'un index. Le filtrage de la sous-expression ignorée est effectué sur les partitions récupérées après application du filtrage d'index.

Pour chaque partition ajoutée à une table, un élément d'index correspondant est créé. Pour une table avec 'n' partitions, 1 index de partition entraînera 'n' éléments d'index de partition. L'index de partition 'm' sur la même table se traduira par des éléments d'index de partition 'm*n'. Chaque élément d'index de partition sera facturé selon la politique de tarification AWS Glue en vigueur pour le stockage du catalogue de données. Pour plus de détails sur la tarification des objets de stockage, consultez la tarification AWS Glue.

Création d'un tableau avec index de partition

Vous pouvez créer un index de partition lors de la création d'une table. La demande `CreateTable` prend une liste d'objets `PartitionIndex` comme entrée. 3 index de partition au maximum peuvent être créés sur une table donnée. Chaque index de partition nécessite un nom et une liste de `partitionKeys` définie pour la table. Les index créés sur une table peuvent être récupérés à l'aide de l'API `GetPartitionIndexes`

Ajout d'un index de partition à une table existante

Pour ajouter un index de partition à une table existante, utilisez l'opération `CreatePartitionIndex`. Vous pouvez créer un `PartitionIndex` par opération `CreatePartitionIndex`. L'ajout d'un index n'affecte pas la disponibilité d'une table, car la table reste disponible pendant la création des index.

L'état d'index d'une partition ajoutée est défini sur CREATING et la création des données d'index est lancée. Si le processus de création des index réussit, l'indexStatus est mis à jour sur ACTIVE et pour un processus infructueux, l'état de l'index est mis à jour sur FAILED. La création d'index peut échouer pour plusieurs raisons et vous pouvez utiliser l'opération `GetPartitionIndexes` pour récupérer les détails de l'échec. Les échecs possibles sont :

- ENCRYPTED_PARTITION_ERROR – la création d'index sur une table avec des partitions chiffrées n'est pas prise en charge.
- INVALID_PARTITION_TYPE_DATA_ERROR – observé lorsque la valeur `partitionKey` n'est pas une valeur valide pour le type de données `partitionKey` correspondant. Par exemple : un `partitionKey` avec le type de données 'int' a une valeur 'foo'.
- MISSING_PARTITION_VALUE_ERROR – observé lorsque le for `partitionValue` pour un `indexedKey` est absent. Cela peut se produire lorsqu'une table n'est pas partitionnée de manière cohérente.
- UNSUPPORTED_PARTITION_CHARACTER_ERROR – observé lorsque la valeur d'une clé de partition indexée contient les caractères \u0000, \u0001 ou \u0002
- INTERNAL_ERROR – une erreur interne s'est produite lors de la création des index.

Description des indexes de partition sur une table

Pour extraire les index de partition créés sur une table, utilisez l'opération `GetPartitionIndexes`. La réponse renvoie tous les index de la table, ainsi que le statut actuel de chaque index (`IndexStatus`).

Le `IndexStatus` pour un index de partition sera l'un des suivants :

- `CREATING` – l'index est en cours de création et n'est pas encore disponible pour utilisation.
- `ACTIVE` – l'index est prêt à l'emploi. Les requêtes peuvent utiliser l'index pour effectuer une requête optimisée.

- `DELETING` – l'index est actuellement en cours de suppression et ne peut plus être utilisé. Un index à l'état actif peut être supprimé à l'aide de la demande `DeletePartitionIndex`, qui fait passer l'état de ACTIVE à DELETING.
- `FAILED` – échec de la création d'index sur une table existante. Chaque table stocke les 10 derniers index ayant échoué.

Les transitions d'état possibles pour les index créés sur une table existante sont :

- CREATING → ACTIVE → DELETING
- CREATING → FAILED

Limitations de l'utilisation des index de partition

Une fois que vous avez créé un index de partition, notez ces modifications apportées aux fonctionnalités de table et de partition :

Création d'une partition (après ajout d'index)

Une fois qu'un index de partition est créé sur une table, toutes les nouvelles partitions ajoutées à la table seront validées pour les vérifications de type de données pour les clés indexées. La valeur de partition des clés indexées sera validée pour le format de type de données. Si la vérification du type de données échoue, l'opération de création de partition échouera. Pour la table sales_data, si un index est créé pour les clés (category, year) où la catégorie est de type `string` et l'année de type `int`, la création de la nouvelle partition avec une valeur de YEAR telle que "foo" échouera.

Une fois les index activés, l'ajout de partitions avec des valeurs de clé indexées ayant les caractères U+0000, U+00001 et U+0002 commencera à échouer.

Mises à jour des tables

Une fois qu'un index de partition est créé sur une table, vous ne pouvez pas modifier les noms de clé de partition pour les clés de partition existantes, et vous ne pouvez pas changer le type ou l'ordre des clés qui sont enregistrées avec l'index.

Utilisation d'index pour un appel optimisé GetPartitions

Lorsque vous appelez `GetPartitions` une table avec un index, vous pouvez inclure une expression et, le cas échéant, Data Catalog utilisera un index si possible. La première clé de

l'index doit être passée dans l'expression pour les index à utiliser dans le filtrage. L'optimisation de l'index dans le filtrage est appliquée au mieux. Data Catalog essaie d'utiliser l'optimisation d'index autant que possible, mais en cas d'index manquant ou d'opérateur non pris en charge, il revient à l'implémentation existante de chargement de toutes les partitions.

Pour la table sales_data ci-dessus, ajoutons l'index [Country, Category, Year]. Si « Country » (Pays) n'est pas passé dans l'expression, l'index enregistré ne pourra pas filtrer les partitions à l'aide d'index. Vous pouvez ajouter jusqu'à 3 index pour prendre en charge divers modèles de requête.

Prenons quelques exemples d'expressions et voyons comment les index fonctionnent dessus :

Expressions	Comment l'index sera utilisé
Country = 'US'	Index sera utilisé pour filtrer les partitions.
Country = 'US' and Category = 'Shoes'	Index sera utilisé pour filtrer les partitions.
Category = 'Shoes'	Les index ne seront pas utilisés, car la valeur « country » (pays) n'est pas fournie dans l'expression. Toutes les partitions seront chargées pour renvoyer une réponse.
Country = 'US' and Category = 'Shoes' and Year > '2018'	Index sera utilisé pour filtrer les partitions.
Country = 'US' and Category = 'Shoes' and Year > '2018' and month = 2	L'index sera utilisé pour récupérer toutes les partitions avec pays = "US" and category = "shoes" and year > 2018. Ensuite, le filtrage sur l'expression du mois sera effectué.
Country = 'US' AND Category = 'Shoes' OR Year > '2018'	Les index ne seront pas utilisés, car un opérateur 0R est présent dans l'expression.
Country = 'US' AND Category = 'Shoes' AND (Year = 2017 OR Year = '2018')	Index sera utilisé pour récupérer toutes les partitions avec country = "US" and category = "shoes", puis un filtrage sur l'expression de l'année sera effectué.

Expressions	Comment l'index sera utilisé
Country in ('US', 'UK') AND Category = 'Shoes'	Les index ne seront pas utilisés pour le filtrage, car l'opérateur `IN` n'est pas pris en charge actuellement.
Country = 'US' AND Category in ('Shoes', 'Books')	L'index sera utilisé pour récupérer toutes les partitions avec country = "US", puis un filtrage sur l'expression Category sera effectué.
Country = 'US' AND Category in ('Shoes', 'Books') AND (creationDate > '2023-9-01'	L'index sera utilisé pour récupérer toutes les partitions avec country = "US", avec creationDate > '2023-9-01', puis un filtrage sur l'expression Category sera effectué.

Intégration avec les moteurs

Redshift Spectrum, Amazon EMR et AWS Glue ETL Spark peuvent utiliser des index pour récupérer des partitions une fois que les index DataFrames sont à l'état ACTIF dans. AWS Glue Athena et les trames dynamiques AWS Glue d'extraction, de transfert et de chargement vous obligent à suivre des étapes supplémentaires pour utiliser les index à des fins d'amélioration des requêtes.

Activation du filtrage de partition

Pour activer le filtrage de partition dans Athena, vous devez mettre à jour les propriétés de la table comme suit :

1. Dans la AWS Glue console, sous Catalogue de données, sélectionnez Tables.
2. Choisissez une table .
3. Sous Actions, choisissez Modifier la table.
4. Sous Propriétés de la table, ajoutez ce qui suit :
   - Clé : `partition_filtering.enabled`
   - Valeur : `true`
5. Cliquez sur Appliquer.

Vous pouvez également définir ce paramètre en exécutant une requête ALTER TABLE SET PROPERTIES dans Athena.

```
ALTER TABLE partition_index.table_with_index
SET TBLPROPERTIES ('partition_filtering.enabled' = 'true')
```

Mise à jour de tables Data Catalog créées manuellement à l'aide d'crawlers

Vous souhaiterez peut-être créer des AWS Glue Data Catalog tables manuellement, puis les mettre à jour à l'aide de AWS Glue robots d'exploration. Les crawlers respectant un calendrier peuvent ajouter de nouvelles partitions et mettre à jour les tables avec des modifications de schéma. Cela s'applique également aux tables migrées depuis un métastore Apache Hive.

Pour ce faire, lorsque vous définissez un crawler, au lieu de spécifier un ou plusieurs magasins de données en tant que source d'une analyse, vous spécifiez une ou plusieurs tables Data Catalog existantes. L'crawler analyse ensuite les magasins de données spécifiés par les tables du catalogue. Dans ce cas, aucune nouvelle table n'est créée ; au lieu de cela, vos tables créées manuellement sont mises à jour.

Voici d'autres raisons qui peuvent vous amener à vouloir créer manuellement des tables de catalogue et spécifier les tables de catalogue en tant qu'crawler source :

- Vous voulez choisir le nom de la table de catalogue et de ne pas vous fier à l'algorithme d'attribution de noms de la table de catalogue.
- Vous souhaitez empêcher de nouvelles tables d'être créées au cas où des fichiers dont le format pourrait perturber la détection de partition soient enregistrés par erreur dans le chemin de la source de données.

Pour de plus amples informations, veuillez consulter Étape 2 : Choisir des sources de données et des classificateurs..

Propriétés de la table Catalogue de données

Les propriétés de table, ou paramètres, comme on les appelle dans la AWS CLI, sont des chaînes de clés et de valeurs non validées. Vous pouvez définir vos propres propriétés sur la table pour prendre en charge les utilisations du Catalogue de données en dehors d' AWS Glue. D'autres services utilisant le catalogue de données peuvent également le faire. AWS Glue définit certaines propriétés de table lors de l'exécution de jobs ou de robots d'exploration. Sauf indication contraire, ces propriétés sont destinées à un usage interne. Nous ne garantissons pas le fait qu'elles continueront d'exister sous leur forme actuelle, et nous ne garantissons pas le comportement du produit si ces propriétés sont modifiées manuellement.

Pour plus d'informations sur les propriétés de table définies par AWS Glue les robots d'exploration, consultezthe section called "Paramètres définis sur les tables du Catalogue de données par un Crawler".

Intégration aux tables Amazon S3

AWS Glue L'intégration du catalogue de données à Amazon S3 Tables vous permet de découvrir, d'interroger et de joindre des tables S3 aux données des lacs de données Amazon S3 à l'aide d'un seul catalogue. Lorsque vous intégrez des tables S3 au catalogue de données, le service crée une structure de catalogue fédérée qui mappe les ressources des tables S3 aux objets du AWS Glue catalogue :

- Un bucket de table S3 devient un catalogue dans le catalogue de données
- Un espace de noms S3 devient une AWS Glue base de données
- Une table S3 devient une AWS Glue table

Contrôles d'accès

Le catalogue de données prend en charge deux modes de contrôle d'accès pour l'intégration des tables S3 :

- Contrôle d'accès IAM — Utilise les politiques IAM pour contrôler l'accès aux tables S3 et au catalogue de données. Dans cette approche, vous devez disposer d'autorisations IAM sur les ressources des tables S3 et sur les objets du catalogue de données pour accéder aux ressources.
- AWS Lake Formation contrôle d'accès — Utilise AWS Lake Formation des autorisations en plus des autorisations AWS Glue IAM pour contrôler l'accès aux tables S3 via le catalogue de données. Dans ce mode, les principaux ont besoin d'autorisations IAM pour interagir avec le catalogue de données, et les AWS Lake Formation autorisations déterminent les ressources du catalogue (bases de données, tables, colonnes, lignes) auxquelles le principal peut accéder. Ce mode prend en charge à la fois un contrôle d'accès grossier (autorisations au niveau de la base de données et au niveau de la table) et un contrôle d'accès détaillé (sécurité au niveau des colonnes et au niveau des lignes). Lorsqu'un rôle enregistré est configuré et que la vente d'informations d'identification est activée, les autorisations IAM de S3 Tables ne sont pas requises pour le principal, car il vend des informations AWS Lake Formation d'identification au nom du principal en utilisant le rôle enregistré. AWS Lake Formation le contrôle d'accès prend également en charge la vente d'informations d'identification pour les moteurs d'analyse tiers. Pour plus d'informations, consultez la section Création d'un catalogue de tables S3 dans le manuel du AWS Lake Formation développeur.

Vous pouvez passer d'un mode de contrôle d'accès à un autre au fur et à mesure de l'évolution de vos besoins.

Hiérarchie du catalogue pour le montage automatique

Lorsque vous intégrez des tables S3 au catalogue de données à l'aide de la console de gestion Amazon S3, la console crée un catalogue fédéré appelé `s3tablescatalog` dans le catalogue de données de votre compte dans cette AWS région. Ce catalogue fédéré sert de catalogue parent pour tous les compartiments de tables S3 existants et futurs de ce compte et de cette région. L'intégration mappe les ressources du bucket de tables Amazon S3 dans la hiérarchie suivante :

- Catalogue fédéré — `s3tablescatalog` (créé automatiquement)
- Catalogues enfants : chaque compartiment de table S3 devient un catalogue enfant sous `s3tablescatalog`
- Bases de données : chaque espace de noms S3 dans un compartiment de table devient une base de données
- Tables — Chaque table S3 au sein d'un espace de noms devient une table

Par exemple, si vous avez un compartiment de table S3 nommé « analytics-bucket » avec un espace de noms « sales » contenant une table « transactions », le chemin complet du catalogue de données serait le suivant : `s3tablescatalog/analytics-bucket/sales/transactions`

Cette hiérarchie en quatre parties s'applique aux scénarios de même compte dans lesquels les tables S3 et le catalogue de données se trouvent dans le même compte. AWS Pour les scénarios entre comptes, vous montez manuellement des compartiments de table S3 individuels dans le catalogue de données, ce qui crée une hiérarchie en trois parties.

Régions prises en charge

L'intégration des tables S3 au catalogue de données est disponible dans les AWS régions suivantes :

Code région	Nom de la région
us-east-1	USA Est (Virginie du Nord)
us-east-2	USA Est (Ohio)
us-west-1	USA Ouest (Californie du Nord)

Code région	Nom de la région
us-west-2	USA Ouest (Oregon)
af-south-1	Afrique (Le Cap)
ap-east-1	Asie-Pacifique (Hong Kong)
ap-east-2	Asie-Pacifique (Taipei)
ap-northeast-1	Asie-Pacifique (Tokyo)
ap-northeast-2	Asie-Pacifique (Séoul)
ap-northeast-3	Asie-Pacifique (Osaka)
ap-south-1	Asie-Pacifique (Mumbai)
ap-south-2	Asie-Pacifique (Hyderabad)
ap-southeast-1	Asie-Pacifique (Singapour)
ap-southeast-2	Asie-Pacifique (Sydney)
ap-southeast-3	Asie-Pacifique (Jakarta)
ap-southeast-4	Asie-Pacifique (Melbourne)
ap-southeast-5	Asie-Pacifique (Malaisie)
ap-southeast-6	Asie-Pacifique (Nouvelle Zélande)
ap-southeast-7	Asie-Pacifique (Thaïlande)
ca-central-1	Canada (Centre)
ca-west-1	Canada-Ouest (Calgary)
eu-central-1	Europe (Francfort)
eu-central-2	Europe (Zurich)

Code région	Nom de la région
eu-north-1	Europe (Stockholm)
eu-south-1	Europe (Milan)
eu-south-2	Europe (Espagne)
eu-west-1	Europe (Irlande)
eu-west-2	Europe (Londres)
eu-west-3	Europe (Paris)
il-central-1	Israël (Tel Aviv)
mx-central-1	Mexique (Centre)
sa-east-1	Amérique du Sud (Sao Paulo)

Rubriques

- Conditions préalables
- Activation de l'intégration des tables S3 au catalogue de données
- Ajout de bases de données et de tables au catalogue de tables S3
- Partage d'objets du catalogue S3 Tables
- Gestion de l'intégration des tables S3

Conditions préalables

Avant de créer un catalogue fédéré pour les tables S3 dans le catalogue de AWS Glue données, assurez-vous que votre principal IAM (utilisateur ou rôle) dispose des autorisations requises.

Autorisations IAM requises

Votre principal IAM a besoin des autorisations suivantes pour activer l'intégration des tables S3 :

AWS Glue autorisations :

- `glue:CreateCatalog`— Nécessaire pour créer le `s3tablescatalog` catalogue fédéré

- `glue:GetCatalog`— Nécessaire pour consulter les détails du catalogue
- `glue:GetDatabase`— Nécessaire pour afficher les espaces de noms S3 sous forme de bases de données
- `glue:GetTable`— Nécessaire pour afficher les tables S3
- `glue:passConnection`— Accorde au principal appelant le droit de déléguer la `aws:s3tables` connexion au AWS Glue service

Autorisations S3 Tables (pour le contrôle d'accès IAM) :

- `s3tables:CreateTableBucket`
- `s3tables:GetTableBucket`
- `s3tables:CreateNamespace`
- `s3tables:GetNamespace`
- `s3tables:ListNamespaces`
- `s3tables:CreateTable`
- `s3tables:GetTable`
- `s3tables:ListTables`
- `s3tables:UpdateTableMetadataLocation`
- `s3tables:GetTableMetadataLocation`
- `s3tables:GetTableData`
- `s3tables:PutTableData`

Exemple de politique IAM

La politique IAM suivante fournit les autorisations minimales requises pour permettre l'intégration des tables S3 au catalogue de données en mode IAM :

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "GlueDataCatalogPermissions",
      "Effect": "Allow",
```

```
      "Action": [
        "glue:CreateCatalog",
        "glue:GetCatalog",
        "glue:GetDatabase",
        "glue:GetTable"
      ],
      "Resource": [
        "arn:aws:glue:region:account-id:catalog/s3tablescatalog",
        "arn:aws:glue:region:account-id:database/s3tablescatalog/*/*",
        "arn:aws:glue:region:account-id:table/s3tablescatalog/*/*/*"
      ]
    },
    {
      "Sid": "S3TablesDataAccessPermissions",
      "Effect": "Allow",
      "Action": [
        "s3tables:GetTableBucket",
        "s3tables:GetNamespace",
        "s3tables:GetTable",
        "s3tables:GetTableMetadataLocation",
        "s3tables:GetTableData"
      ],
      "Resource": [
        "arn:aws:s3tables:region:account-id:bucket/*",
        "arn:aws:s3tables:region:account-id:bucket/*/table/*"
      ]
    }
  ]
}
```

Activation de l'intégration des tables S3 au catalogue de données

Vous pouvez activer l'intégration des tables S3 au catalogue de AWS Glue données à l'aide de la console de gestion Amazon S3 ou AWS CLI. Lorsque vous activez l'intégration à l'aide de la console, vous AWS créez un catalogue fédéré nommé `s3tablescatalog` qui découvre et monte automatiquement tous les compartiments de table S3 de votre AWS compte et de votre région.

Activer l'intégration des tables S3 à l'aide de la console de gestion Amazon S3

1. Ouvrez la console Amazon S3 à l'adresse https://console.aws.amazon.com/s3/.
2. Dans le volet de navigation de gauche, choisissez Compartiments de tables.
3. Choisissez Créer un compartiment de tables.

4. Saisissez un Nom du compartiment de table et veillez à ce que la case Activer l'intégration soit cochée.
5. Choisissez Créer un compartiment de tables.

Amazon S3 intègre automatiquement vos compartiments de table dans cette région. La première fois que vous intégrez des compartiments de table dans une région, Amazon S3 les crée `s3tablescatalog` dans le catalogue de données de cette région.

Une fois le catalogue créé, tous les compartiments de tables S3 de votre compte et de votre région sont automatiquement montés en tant que catalogues enfants. Vous pouvez consulter les bases de données (espaces de noms) et les tables en accédant au catalogue dans le catalogue de données.

Activez l'intégration des tables S3 à l'aide de AWS CLI

Utilisez la `glue create-catalog` commande pour créer le `s3tablescatalog` catalogue.

```
aws glue create-catalog \
  --name "s3tablescatalog" \
  --catalog-input '{
    "Description": "Federated catalog for S3 Tables",
    "FederatedCatalog": {
      "Identifier": "arn:aws:s3tables:region:account-id:bucket/*",
      "ConnectionName": "aws:s3tables"
    },
    "CreateDatabaseDefaultPermissions": [{
      "Principal": {
        "DataLakePrincipalIdentifier": "IAM_ALLOWED_PRINCIPALS"
      },
      "Permissions": ["ALL"]
    }],
    "CreateTableDefaultPermissions": [{
      "Principal": {
        "DataLakePrincipalIdentifier": "IAM_ALLOWED_PRINCIPALS"
      },
      "Permissions": ["ALL"]
    }]
  }'
```

Remplacez *`region`* par votre AWS région et *`account-id`* par votre numéro de AWS compte.

Vérification de l'intégration

Après avoir créé le catalogue, vous pouvez vérifier que les compartiments de table S3 sont montés en répertoriant les catalogues enfants :

```
aws glue get-catalogs \
  --parent-catalog-id s3tablescatalog
```

Ajout de bases de données et de tables au catalogue de tables S3

Assurez-vous de disposer des autorisations nécessaires pour répertorier et créer des catalogues, des bases de données et des tables dans le catalogue de données de votre région. Assurez-vous que l'intégration de S3 Tables est activée dans votre AWS compte et dans votre région.

Ajouter une base de données au catalogue de tables S3

Ajouter une base de données (console)

1. Ouvrez la AWS Glue console à la https://console.aws.amazon.com/glue/maison.
2. Dans le panneau de navigation de gauche, sélectionnez Bases de données.
3. Choisissez Ajouter une base de données.
4. Choisissez Glue Database dans S3 Tables Federated Catalog.
5. Entrez un nom unique pour la base de données.
6. Sélectionnez le catalogue cible qui correspond à un compartiment de tables dans S3 Tables.
7. Choisissez Create database (Créer une base de données).

Ajouter une base de données (AWS CLI)

```
aws glue create-database \
  --region region \
  --catalog-id "account-id:s3tablescatalog/my-catalog" \
  --database-input '{"Name": "my-database"}'
```

Ajouter une table au catalogue de tables S3

Ajouter une table (console)

1. Ouvrez la AWS Glue console à la https://console.aws.amazon.com/glue/maison.

2. Dans le volet de navigation de gauche, choisissez Tables.
3. Sélectionnez le catalogue S3 Tables approprié dans la liste déroulante du catalogue.
4. Choisissez Ajouter un tableau.
5. Entrez un nom unique pour votre table.
6. Vérifiez que le catalogue S3 Tables correct est sélectionné dans la liste déroulante du catalogue.
7. Sélectionnez la base de données dans la liste déroulante des bases de données.
8. Entrez le schéma de table en saisissant un code JSON ou en ajoutant chaque colonne individuellement.
9. Choisissez Créer un tableau.

Ajouter un tableau (AWS CLI)

```
aws glue create-table \
  --region region \
  --catalog-id "account-id:s3tablescatalog/my-catalog" \
  --database-name "my-database" \
  --table-input '{
    "Name": "my-table",
    "Parameters": {
      "classification": "",
      "format": "ICEBERG"
    },
    "StorageDescriptor": {
      "Columns": [
        {"Name": "id", "Type": "int", "Parameters": {}},
        {"Name": "val", "Type": "string", "Parameters": {}}
      ]
    }
  }'
```

Partage d'objets du catalogue S3 Tables

Lorsque vous utilisez le contrôle d'accès IAM, vous pouvez partager des objets du catalogue S3 Tables avec d'autres utilisateurs à l'aide de liens de AWS Glue ressources pour le partage entre les mêmes comptes. Pour le partage entre comptes, vous pouvez partager des compartiments de table S3 avec un autre AWS compte et le rôle ou l'utilisateur IAM du compte destinataire peut créer un objet de AWS Glue catalogue à l'aide du compartiment de table partagé.

Partage au sein d'un même compte à l'aide de liens vers des ressources

Les liens vers les ressources vous permettent de créer des références aux AWS Glue bases de données et aux `s3tablescatalog` tables figurant dans votre catalogue AWS Glue par défaut. Cela est utile pour organiser l'accès aux données ou créer des groupements logiques de tables.

Création d'un lien vers une ressource (console)

1. Ouvrez la AWS Glue console à l'adresse https://console.aws.amazon.com/glue/.
2. Dans le volet de navigation, sélectionnez Catalogues.
3. Dans la liste Catalogue, sélectionnez s3tablescatalog.
4. Sélectionnez le tableau que vous souhaitez partager dans les`s3tablescatalog`.
5. Choisissez Actions, puis sélectionnez Créer un lien vers une ressource.
6. Dans Nom du lien de ressource, entrez un nom pour le lien de ressource.
7. Pour la base de données cible, sélectionnez la base de données dans laquelle vous souhaitez créer le lien de ressource.
8. (Facultatif) Sous Description, entrez une description.
9. Choisissez Créer.

Le lien vers la ressource apparaît dans la base de données cible et pointe vers la table d'origine dans`s3tablescatalog`.

Créer des liens vers des ressources (AWS CLI)

Créez un lien vers une ressource de base de données :

```
aws glue create-database \
  --database-name "my-database-resource-link" \
  --database-input '{
    "Name": "sales_data_link",
    "TargetDatabase": {
      "CatalogId": "account-id:s3tablescatalog/analytics-bucket",
      "DatabaseName": "sales"
    }
  }'
```

Créez un lien vers une ressource de table :

```
aws glue create-table \
```

```
  --table-name "my-table-resource-link" \
  --table-input '{
    "Name": "sales_data_link",
    "TargetTable": {
      "CatalogId": "account-id:s3tablescatalog/analytics-bucket",
      "DatabaseName": "sales",
      "Name": "transactions"
    }
  }'
```

Gestion de l'intégration des tables S3

Activer AWS Lake Formation

Vous pouvez activer votre catalogue AWS Lake Formation de tables S3 lorsque vous souhaitez adapter vos exigences en matière de gouvernance des données. AWS Lake Formation fournit des autorisations de type base de données pour gérer un accès précis, dimensionner les autorisations à l'aide d'un accès basé sur des balises et octroyer des autorisations en fonction d'attributs utilisateur tels que les associations de groupes à vos tables dans S3 Tables.

Accédez à la console AWS Lake Formation de gestion AWS Lake Formation pour activer votre catalogue de tables S3 dans AWS Glue. Pour plus d'informations, consultez la section Création d'un catalogue de tables S3 dans le manuel du AWS Lake Formation développeur.

Supprimer l'intégration des tables S3

Vous pouvez supprimer l'intégration des tables S3 en supprimant l'intégration du catalogue dans le catalogue de données. Cette opération supprime uniquement les métadonnées du catalogue de données et non les ressources des tables S3.

Assurez-vous de disposer des autorisations nécessaires pour répertorier, modifier et supprimer des objets du catalogue AWS Glue.

Supprimer l'intégration (console)

1. Ouvrez la AWS Glue console à la https://console.aws.amazon.com/glue/maison.
2. Dans le volet de navigation, sélectionnez Catalogues.
3. Dans la liste Catalogue, sélectionnez s3tablescatalog.
4. Sélectionnez Delete (Supprimer).
5. Vérifiez que la suppression du catalogue entraîne également la suppression de tous les objets de catalogue associés dans le catalogue de données.

6. Sélectionnez Delete (Supprimer).

Supprimer l'intégration (AWS CLI)

```
aws glue delete-catalog \
  --region region \
  --catalog-id "s3tablescatalog"
```

Intégration à d'autres AWS services

Bien que vous puissiez utiliser AWS Glue crawler s pour remplir le catalogue AWS Glue Data Catalog, plusieurs AWS services peuvent automatiquement s'intégrer au catalogue et le remplir pour vous. Les sections suivantes fournissent des informations supplémentaires sur les cas d'utilisation spécifiques pris en charge par AWS les services qui peuvent alimenter le catalogue de données.

Rubriques

- AWS Lake Formation
- Amazon Athena

AWS Lake Formation

AWS Lake Formation est un service qui facilite la configuration d'un lac de données sécurisé dans AWS. Lake Formation est construite sur AWS Glue, et Lake Formation et AWS Glue partage la même chose AWS Glue Data Catalog. Vous pouvez enregistrer l'emplacement de vos données Amazon S3 auprès de Lake Formation et utiliser la console Lake Formation pour créer des bases de données et des tables dans le catalogue de AWS Glue données, définir des politiques d'accès aux données et auditer l'accès aux données dans votre lac de données à partir d'un emplacement central. Vous pouvez utiliser le contrôle précis des accès de Lake Formation pour gérer les ressources de votre catalogue de données existantes et les emplacements de données Amazon S3.

Avec les données enregistrées auprès de Lake Formation, vous pouvez partager en toute sécurité les ressources du catalogue de données entre les principaux responsables, les AWS comptes, les AWS organisations et les unités organisationnelles d'IAM.

Pour plus d'informations sur la création de ressources de catalogue de données à l'aide de Lake Formation, consultez la section Création de tables et de bases de données de catalogue de données dans le guide du AWS Lake Formation développeur.

Amazon Athena

Amazon Athena utilise le catalogue de données pour stocker et récupérer les métadonnées des tables correspondant aux données Amazon S3 de votre AWS compte. Les métadonnées de la table permettent au moteur de requête Athena de savoir comment trouver, lire et traiter les données que vous souhaitez interroger.

Vous pouvez remplir le AWS Glue Data Catalog en utilisant directement les instructions `CREATE TABLE` Athena. Vous pouvez définir et remplir manuellement le schéma et les métadonnées de partition dans le catalogue de données sans avoir à exécuter un robot.

1. Dans la console Athena, créez une base de données qui stockera les métadonnées de la table dans le catalogue de données.
2. Utilisez l'instruction `CREATE EXTERNAL TABLE` pour définir le schéma de votre source de données.
3. Utilisez la clause `PARTITIONED BY` pour définir les clés de partition si vos données sont partitionnées.
4. Utilisez la clause `LOCATION` pour spécifier le chemin d'accès à Amazon S3 où vos fichiers de données réels sont stockés.
5. Exécutez l'instruction `CREATE TABLE`.

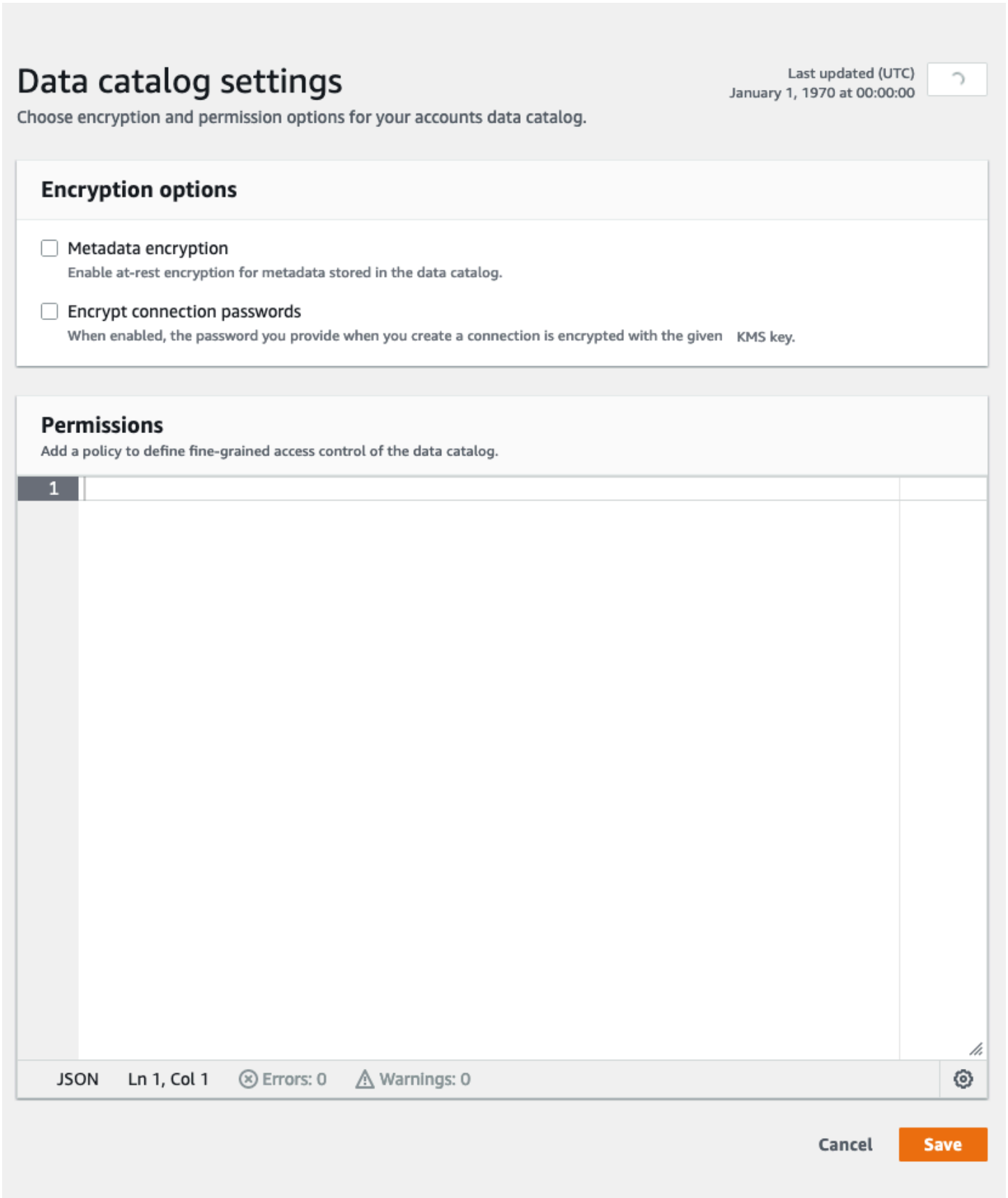
   Cette requête crée les métadonnées de table dans le catalogue de données en fonction du schéma et des partitions que vous avez définis, sans réellement analyser les données.

Vous pouvez interroger la table dans Athena, qui utilisera les métadonnées du catalogue de données pour accéder à vos fichiers de données et les interroger dans Amazon S3.

Pour plus d'informations, consultez Creating databases and tables dans le Guide de l'utilisateur Amazon Athena.

Paramètres du catalogue de données

Les paramètres du catalogue de données contiennent des options permettant de définir les options de chiffrement et d'autorisations pour le catalogue de données de votre compte.


Data catalog settings
Choose encryption and permission options for your accounts data catalog.
Last updated (UTC)
January 1, 1970 at 00:00:00
Encryption options
Metadata encryption
Enable at-rest encryption for metadata stored in the data catalog.
Encrypt connection passwords
When enabled, the password you provide when you create a connection is encrypted with the given KMS key.
Permissions
Add a policy to define fine-grained access control of the data catalog.
1
JSON
Ln 1, Col 1
Errors: 0
Warnings: 0
Cancel
Save

Pour modifier le contrôle d'accès détaillé du catalogue de données.

1. Connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/.
2. Choisissez une option de chiffrement.

   - Chiffrement des métadonnées : cochez cette case pour chiffrer les métadonnées dans votre catalogue de données. Les métadonnées sont chiffrées au repos à l'aide de la clé AWS Key Management Service (AWS KMS) que vous spécifiez. Pour de plus amples informations, veuillez consulter Chiffrement de votre catalogue de données.
   - Chiffrer les mots de passe de connexion : cochez cette case pour chiffrer les mots de passe dans l'objet de connexion AWS Glue lorsque la connexion est créée ou mise à jour. Les mots de passe sont chiffrés à l'aide de la AWS KMS clé que vous spécifiez. Lorsque les mots de passe sont renvoyés, ils sont chiffrés. Cette option dispose d'un paramètre global pour toutes les connexions AWS Glue dans le catalogue de données. Si vous décochez cette case, les mots de passe précédemment chiffrés restent chiffrés à l'aide de la clé utilisée lors de leur création ou de leur mise à jour. Pour plus d'informations sur les connexions AWS Glue, consultez Connexion aux données.

     Lorsque vous activez cette option, choisissez une AWS KMS clé ou choisissez Enter a key ARN et fournissez le Amazon Resource Name (ARN) pour la clé. Entrez l'ARN sous la forme `arn:aws:kms:`*`region`*`:`*`account-id`*`:key/`*`key-id`* . Vous pouvez également fournir l'ARN sous la forme d'un alias de clé, (par exemple, `arn:aws:kms:`*`region`*`:`*`account-id`*`:alias/`*`alias-name`*).

     > **Important**
     >
     > Si cette option est sélectionnée, tout utilisateur ou rôle qui crée ou met à jour une connexion doit dispose d'une autorisation `kms:Encrypt` sur la clé KMS spécifiée.

     Pour de plus amples informations, veuillez consulter Chiffrement des mots de passe de connexion.
3. Sélectionnez Settings (Paramètres), puis dans l'éditeur Permissions (Autorisations), ajoutez l'instruction de politique pour modifier le contrôle d'accès détaillé du catalogue de données pour votre compte. Une seule politique peut être attachée à un catalogue de données à la fois. Vous

pouvez coller une politique ressource JSON dans ce contrôle. Pour de plus amples informations, veuillez consulter Politiques basées sur les ressources au sein de Glue AWS.

4. Sélectionnez Enregistrer pour mettre à jour le catalogue de données avec les modifications que vous avez apportées.

Vous pouvez également utiliser les opérations d'API AWS Glue pour placer, obtenir et supprimer des politiques de ressources. Pour de plus amples informations, veuillez consulter Sécurité APIs dans AWS Glue.

Remplissage et gestion des tables transactionnelles

Apache Iceberg, Apache Hudi et Linux Foundation Delta Lake sont des formats de table open source conçus pour gérer l'analytique des données à grande échelle et les charges de travail des lacs de données dans Apache Spark.

Vous pouvez remplir les tables Iceberg, Hudi et Delta Lake à l' AWS Glue Data Catalog aide des méthodes suivantes :

- AWS Glue crawler; — AWS Glue crawler s peut automatiquement découvrir et renseigner les métadonnées des tables Iceberg, Hudi et Delta Lake dans le catalogue de données. Pour de plus amples informations, veuillez consulter Utilisation de robots pour alimenter le catalogue de données.
- AWS Glue Tâches ETL : vous pouvez créer des tâches ETL pour écrire des données dans les tables Iceberg, Hudi et Delta Lake et renseigner leurs métadonnées dans le catalogue de données. Pour plus d'informations, consultez la section Utilisation de frameworks de lacs de données avec des tâches AWS Glue ETL.
- AWS Glue console, AWS Lake Formation console AWS CLI ou API : vous pouvez utiliser la AWS Glue console, la console Lake Formation ou l'API pour créer et gérer les définitions de tables Iceberg dans le catalogue de données.

Rubriques

- Création de tables Apache Iceberg
- Optimisation des tables Iceberg
- Optimisation des performances des requêtes pour les tables Iceberg

Création de tables Apache Iceberg

Vous pouvez créer des tables Apache Iceberg qui utilisent le format de données Apache Parquet dans l' AWS Glue Data Catalog avec des données résidant dans Amazon S3. Une table dans le catalogue de données correspond à la définition des métadonnées qui représente les données dans un entrepôt de données. Par défaut, AWS Glue crée des tables Iceberg v2. Pour connaître la différence entre les tables v1 et v2, consultez la section Modifications de version de format dans la documentation Apache Iceberg.

Apache Iceberg est un format de table ouvert pour les jeux de données analytiques très volumineux. Iceberg permet de modifier facilement votre schéma, également ce que l'on appelle l'évolution du schéma, ce qui signifie que les utilisateurs peuvent ajouter, renommer ou supprimer des colonnes d'une table de données sans perturber les données sous-jacentes. Iceberg fournit également une prise en charge pour la gestion des versions des données, ce qui permet aux utilisateurs de suivre les modifications apportées aux données au fil du temps. Cela active la fonctionnalité Time Travel, qui permet aux utilisateurs d'accéder aux versions historiques des données, de les interroger, mais aussi d'analyser les modifications apportées aux données entre les mises à jour et les suppressions.

Vous pouvez utiliser AWS Glue la console Lake Formation ou l'`CreateTable`opération de l' AWS Glue API pour créer une table Iceberg dans le catalogue de données. Pour plus d'informations, consultez CreateTable action (Python : create_table).

Lorsque vous créez une table Iceberg dans le catalogue de données, vous devez spécifier le format de la table et le chemin d'accès au fichier de métadonnées dans Amazon S3 pour pouvoir effectuer des lectures et des écritures.

Vous pouvez utiliser Lake Formation pour sécuriser votre table Iceberg à l'aide d'autorisations de contrôle d'accès précises lorsque vous enregistrez l'emplacement des données Amazon S3 auprès de celui-ci. AWS Lake Formation Pour les données source dans Amazon S3 et les métadonnées qui ne sont pas enregistrées auprès de Lake Formation, l'accès est déterminé par les politiques d'autorisation IAM pour Amazon S3 et AWS Glue les actions. Pour plus d'informations, consultez Managing permissions.

> **Note**
>
> Le catalogue de données ne prend pas en charge la création de partitions ni l'ajout de propriétés de table Iceberg.

Conditions préalables

Pour créer des tables Iceberg dans le catalogue de données et configurer les autorisations d'accès aux données de Lake Formation, vous devez remplir les conditions suivantes :

1. Autorisations requises pour créer des tables Iceberg sans les données enregistrées auprès de Lake Formation :

   Outre les autorisations requises pour créer une table dans le catalogue de données, le créateur de table exige les autorisations suivantes :

   - `s3:PutObject` sur la ressource arn:aws:s3:::{bucketName}
   - `s3:GetObject` sur la ressource arn:aws:s3:::{bucketName}
   - `s3:DeleteObject` sur la ressource arn:aws:s3:::{bucketName}
2. Autorisations requises pour créer des tables Iceberg avec des données enregistrées auprès de Lake Formation :

   Pour utiliser Lake Formation afin de gérer et de sécuriser les données de votre lac de données, enregistrez votre site Amazon S3 contenant les données pour les tables auprès de Lake Formation. Cela permet à Lake Formation de vendre des informations d'identification à AWS des services d'analyse tels qu'Athena, Redshift Spectrum et Amazon EMR pour accéder aux données. Pour plus d'informations sur l'enregistrement d'un emplacement Amazon S3, consultez Adding an Amazon S3 location to your data lake.

   Un principal qui lit et écrit les données sous-jacentes enregistrées auprès de Lake Formation doit disposer des autorisations suivantes :

   - `lakeformation:GetDataAccess`
   - `DATA_LOCATION_ACCESS`

     Un principal qui possède des autorisations de localisation des données sur un emplacement possède également des autorisations d'emplacement sur tous les sites enfant.

     Pour plus d'informations sur les autorisations d'emplacement des données, consultez l'ulink Underlying data access control.

Pour activer le compactage, le service doit endosser un rôle IAM autorisé à mettre à jour les tables dans le catalogue de données. Pour plus d'informations, consultez Conditions préalables requises pour l'optimisation des tables.

Création d'une table Iceberg

Vous pouvez créer des tables Iceberg v1 et v2 à l'aide AWS Glue de la console Lake Formation ou AWS Command Line Interface comme indiqué sur cette page. Vous pouvez également créer des tables Iceberg à l'aide d' AWS Glue crawler. Pour plus d'informations, consultez Data Catalog and Crawlers dans le Guide du développeur AWS Glue .

Pour créer une table Iceberg

Console

1. Connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/.
2. Sous catalogue de données, choisissez Tables, puis utilisez le bouton Créer une table pour spécifier les attributs suivants :
   - Nom de table : saisissez un nom pour la table. Si vous utilisez Athena pour accéder à des tables, utilisez ces conseils de dénomination figurant dans le Guide de l'utilisateur Amazon Athena.
   - Base de données : choisissez une base de données existante ou créez-en une.
   - Description : description de la table. Vous pouvez écrire une description vous aidant à comprendre le contenu de la table.
   - Format de tableau : pour Format de tableau, choisissez Apache Iceberg.
   - Activer le compactage : choisissez Activer le compactage pour compacter les petits objets Amazon S3 de la table en objets plus grands.
   - Rôle IAM : pour exécuter le compactage, le service endosse un rôle IAM en votre nom. Vous pouvez choisir un rôle IAM à l'aide de la liste déroulante. Assurez-vous que le rôle dispose des autorisations requises pour activer le compactage.

     Pour en savoir plus sur les autorisations nécessaires, consultez Conditions préalables requises pour l'optimisation des tables.
   - Emplacement : indiquez le chemin d'accès au dossier dans Amazon S3 qui stocke la table de métadonnées. Iceberg a besoin d'un fichier de métadonnées et d'un emplacement dans le catalogue de données pour pouvoir effectuer des lectures et des écritures.

- Schéma : choisissez Ajouter des colonnes pour ajouter des colonnes et les types de données des colonnes. Vous avez la possibilité de créer une table vide et de mettre à jour le schéma ultérieurement. Le catalogue de données prend en charge les types de données Hive. Pour plus d'informations, consultez Hive data types.

  Iceberg vous permet de faire évoluer le schéma et la partition après avoir créé la table. Vous pouvez utiliser des requêtes Athena pour mettre à jour le schéma de table et des requêtes Spark pour mettre à jour les partitions.

AWS CLI

```
aws glue create-table \
    --database-name iceberg-db \
    --region us-west-2 \
    --open-table-format-input '{
      "IcebergInput": {
           "MetadataOperation": "CREATE",
           "Version": "2"
        }
      }' \
    --table-input '{"Name":"test-iceberg-input-demo",
           "TableType": "EXTERNAL_TABLE",
           "StorageDescriptor":{
              "Columns":[
                  {"Name":"col1", "Type":"int"},
                  {"Name":"col2", "Type":"int"},
                  {"Name":"col3", "Type":"string"}
               ],
              "Location":"s3://DOC_EXAMPLE_BUCKET_ICEBERG/"
           }
       }'
```

Optimisation des tables Iceberg

AWS Glue prend en charge plusieurs options d'optimisation des tables afin d'améliorer la gestion et les performances des tables Apache Iceberg utilisées par les moteurs AWS d'analyse et les tâches ETL. Ces optimiseurs permettent d'utiliser efficacement le stockage, d'améliorer les performances

des requêtes et de gérer efficacement les données. Il existe trois types d'optimiseurs de table disponibles dans AWS Glue :

- Compactage : le compactage des données compacte les petits fichiers de données afin de réduire l'utilisation du stockage et d'améliorer les performances de lecture. Les fichiers de données sont fusionnés et réécrits pour supprimer les données obsolètes et regrouper les données fragmentées dans des fichiers plus volumineux et plus efficaces. Vous pouvez configurer le compactage pour qu'il s'exécute automatiquement.

  Binpack est la stratégie de compactage par défaut d'Apache Iceberg. Elle combine des fichiers de données plus petits dans des fichiers plus volumineux pour des performances optimales. Le compactage prend également en charge les stratégies de tri et d'ordre Z qui regroupent des données similaires. Le tri organise les données en fonction de colonnes spécifiées, améliorant ainsi les performances des requêtes pour les opérations filtrées. L'ordre Z crée des jeux de données triés qui améliorent les performances des requêtes lorsque plusieurs colonnes sont interrogées simultanément. Les trois stratégies de compactage (Binpak, Tri et Ordre Z) réduisent la quantité de données numérisées par les moteurs de requêtes, réduisant ainsi les coûts de traitement des requêtes.
- Conservation des instantanés : les instantanés sont des versions horodatées d'une table Iceberg. Les configurations de conservation des instantanés permettent aux clients de définir la durée de conservation des instantanés et le nombre d'instantanés à conserver. La configuration d'un optimiseur de conservation des instantanés peut aider à gérer la charge de stockage en supprimant les instantanés plus anciens et inutiles, ainsi que leurs fichiers sous-jacents associés.
- Suppression de fichiers orphelins : les fichiers orphelins sont des fichiers qui ne sont plus référencés par les métadonnées de la table Iceberg. Ces fichiers peuvent s'accumuler au fil du temps, en particulier après des opérations telles que la suppression de tables ou l'échec de tâches ETL. L'activation de la suppression des fichiers orphelins permet AWS Glue d'identifier et de supprimer périodiquement ces fichiers inutiles, libérant ainsi de l'espace de stockage.

La configuration d'optimisation au niveau du catalogue est disponible via la console Lake Formation et à l'aide de l'opération AWS Glue `UpdateCatalog` API. Vous pouvez activer ou désactiver le compactage, la conservation des instantanés et les optimiseurs de suppression de fichiers orphelins pour les tables Iceberg individuelles du catalogue de données à l'aide de la AWS Glue console ou des opérations de l'API AWS CLI. AWS Glue

Rubriques

- Conditions préalables requises pour l'optimisation des tables

- Optimiseurs de tables au niveau du catalogue
- Optimisation du compactage
- Optimisation de la conservation des instantanés
- Suppression des fichiers orphelins
- Affichage des détails de l'optimisation
- Afficher Amazon CloudWatch les métriques
- Suppression d'un optimiseur
- Considérations et restrictions
- Régions prises en charge pour les optimiseurs de tables

Conditions préalables requises pour l'optimisation des tables

L'optimiseur de table utilise les autorisations du rôle Gestion des identités et des accès AWS (IAM) que vous spécifiez lorsque vous activez les options d'optimisation (compactage, conservation des instantanés et suppression de fichiers orphelins) pour une table. Vous pouvez soit créer un rôle unique pour tous les optimiseurs, soit créer des rôles distincts pour chaque optimiseur.

> **Note**
>
> L'optimiseur de suppression de fichiers orphelins ne nécessite pas les autorisations `glue:updateTable` ou `s3:putObject`. Les optimiseurs d'expiration et de compactage des instantanés nécessitent le même ensemble d'autorisations.

Le rôle IAM doit être autorisé à lire les données et à mettre à jour les métadonnées dans le catalogue de données. Vous pouvez créer un rôle IAM et y attacher les stratégies en ligne suivantes :

- Ajoutez la politique intégrée suivante qui accorde à Amazon S3 des read/write autorisations sur l'emplacement pour les données qui ne sont pas enregistrées auprès AWS Lake Formation de celui-ci. Cette politique inclut également des autorisations pour mettre à jour le tableau dans le catalogue de données, ainsi que AWS Glue pour autoriser l'ajout de journaux dans les Amazon CloudWatch journaux et la publication de métriques. Pour les données sources dans Amazon S3 qui ne sont pas enregistrées auprès de Lake Formation, l'accès est déterminé par les stratégies d'autorisation IAM pour Amazon S3 et les actions AWS Glue .

Dans les stratégies en ligne suivantes, remplacez le `bucket-name` par le nom de votre compartiment Amazon S3, `aws-account-id` et `region` par un numéro valide du compte AWS et une région du catalogue de données, `database_name` par le nom de votre base de données et `table_name` par le nom de la table.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:DeleteObject"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "glue:UpdateTable",
                "glue:GetTable"
            ],
            "Resource": [
                "arn:aws:glue:us-east-1:111122223333:table/<database-
name>/<table-name>",
                "arn:aws:glue:us-east-1:111122223333:database/<database-name>",
                "arn:aws:glue:us-east-1:111122223333:catalog"
            ]
```

```
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": [
                "arn:aws:logs:us-east-1:111122223333:log-group:/aws-glue/
iceberg-compaction/logs:*",
                "arn:aws:logs:us-east-1:111122223333:log-group:/aws-glue/
iceberg-retention/logs:*",
                "arn:aws:logs:us-east-1:111122223333:log-group:/aws-glue/
iceberg-orphan-file-deletion/logs:*"
            ]
        }
    ]
}
```

- Utilisez la stratégie suivante pour activer le compactage des données enregistrées auprès de Lake Formation.

  Si le rôle d'optimisation ne dispose pas d'autorisations de groupe `IAM_ALLOWED_PRINCIPALS` accordées sur la table, le rôle nécessite des autorisations Lake Formation `ALTER`, `DESCRIBE`, `INSERT` et `DELETE` sur la table.

  Pour plus d'informations sur l'enregistrement d'un compartiment Amazon S3 auprès de Lake Formation, consultez Adding an Amazon S3 location to your data lake.

  JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "lakeformation:GetDataAccess"
      ],
      "Resource": "*"
    },
    {
```

```
      "Effect": "Allow",
      "Action": [
        "glue:UpdateTable",
        "glue:GetTable"
      ],
      "Resource": [
        "arn:aws:glue:us-east-1:111122223333:table/databaseName/tableName",
        "arn:aws:glue:us-east-1:111122223333:database/databaseName",
        "arn:aws:glue:us-east-1:111122223333:catalog"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents"
      ],
      "Resource": [
        "arn:aws:logs:us-east-1:111122223333:log-group:/aws-glue/iceberg-
compaction/logs:*",
        "arn:aws:logs:us-east-1:111122223333:log-group:/aws-glue/iceberg-
retention/logs:*",
        "arn:aws:logs:us-east-1:111122223333:log-group:/aws-glue/iceberg-
orphan-file-deletion/logs:*"
      ]
    }
  ]
}
```

- (Facultatif) Pour optimiser des tables Iceberg avec des données contenues dans des compartiments Amazon S3 chiffrés à l'aide du chiffrement côté serveur, le rôle de compactage nécessite des autorisations pour déchiffrer les objets Amazon S3 et générer une nouvelle clé de données pour écrire des objets dans les compartiments chiffrés. Ajoutez la politique suivante à la AWS KMS clé souhaitée. Nous prenons uniquement en charge le chiffrement au niveau du compartiment.

```
{
    "Effect": "Allow",
    "Principal": {
        "AWS": "arn:aws:iam::<aws-account-id>:role/<optimizer-role-name>"
    },
    "Action": [
```

```
        "kms:Decrypt",
        "kms:GenerateDataKey"
    ],
    "Resource": "*"
}
```

- (Facultatif) Pour l'emplacement des données enregistré auprès de Lake Formation, le rôle utilisé pour enregistrer l'emplacement nécessite des autorisations pour déchiffrer les objets Amazon S3 et générer une nouvelle clé de données pour écrire des objets dans les compartiments chiffrés. Pour plus d'informations, consultez la rubrique Enregistrement d'un emplacement Amazon S3 chiffré.
- (Facultatif) Si la AWS KMS clé est stockée dans un autre AWS compte, vous devez inclure les autorisations suivantes pour le rôle de compactage.

  JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "kms:Decrypt",
        "kms:GenerateDataKey"
      ],
      "Resource": [
        "arn:aws:kms:us-east-1:111122223333:key/key-id"
      ]
    }
  ]
}
```

- Le rôle que vous utilisez pour exécuter le compactage doit disposer de l'autorisation `iam:PassRole` correspondante.

  JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iam:PassRole"
```

```
      ],
      "Resource": [
        "arn:aws:iam::111122223333:role/<optimizer-role-name>"
      ]
    }
  ]
}
```

- Ajoutez la politique de confiance suivante au rôle afin que le AWS Glue service assume le rôle IAM pour exécuter le processus de compactage.

  JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": "glue.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

- (Facultatif) Pour mettre à jour les paramètres du catalogue de données afin de permettre l'optimisation des tables au niveau du catalogue, le rôle IAM utilisé doit disposer de l'`glue:UpdateCatalog`autorisation ou AWS Lake Formation `ALTER CATALOG` de l'autorisation sur le catalogue racine. Vous pouvez utiliser l'API `GetCatalog` pour vérifier les propriétés du catalogue.

Optimiseurs de tables au niveau du catalogue

Avec une configuration de catalogue unique, vous pouvez configurer des optimiseurs automatiques tels que le compactage, la conservation des instantanés et la suppression des fichiers orphelins pour toutes les tables Apache Iceberg nouvelles et mises à jour dans l' AWS Glue Data Catalog. Les configurations d'optimiseur au niveau du catalogue vous permettent d'appliquer des paramètres

d'optimisation cohérents à toutes les tables d'un catalogue, éliminant ainsi le besoin de configurer les optimiseurs individuellement pour chaque table.

Les administrateurs de lac de données peuvent configurer les optimiseurs de table en sélectionnant le catalogue par défaut dans la console Lake Formation et en activant les optimiseurs à l'aide de l'option `Table optimization`. Lorsque vous créez des tables ou que vous mettez à jour des tables existantes dans le catalogue de données, ce dernier exécute automatiquement les optimisations des tables afin de réduire la charge opérationnelle.

Si vous avez configuré l'optimisation au niveau de la table ou si vous avez précédemment supprimé les paramètres d'optimisation des tables pour une table, ces paramètres propres à la table ont priorité sur les paramètres du catalogue par défaut pour l'optimisation des tables. Si aucun paramètre de configuration n'est défini au niveau de la table ou du catalogue, la valeur de propriété de la table Iceberg sera appliquée. Ce paramètre s'applique à la conservation des instantanés et à l'optimiseur de suppression de fichiers orphelins.

Tenez compte des éléments suivants lorsque vous activez les optimisateurs au niveau du catalogue :

- Lorsque vous configurez les paramètres d'optimisation au moment de la création du catalogue, puis que vous désactivez les optimisations par le biais d'une demande de mise à jour du catalogue, l'opération se répercute sur toutes les tables du catalogue.
- Si vous avez déjà configuré des optimiseurs pour une table donnée, l'opération de désactivation au niveau du catalogue n'aura aucun impact sur cette table.
- Lorsque vous désactivez les optimiseurs au niveau du catalogue, les tables présentant des configurations d'optimiseur existantes conservent leurs paramètres spécifiques et ne sont pas affectées par les modifications apportées au niveau du catalogue. Toutefois, les tables dépourvues de leur propre configuration d'optimiseur hériteront de l'état désactivé au niveau du catalogue.
- Étant donné que les optimiseurs de conservation des instantanés et de suppression de fichiers orphelins peuvent être basés sur un calendrier, les mises à jour introduiront un retard aléatoire dans le début de leur calendrier. Cela entraînera le démarrage de chaque optimiseur à des heures légèrement différentes, répartissant ainsi la charge et réduisant le risque de dépassement des limites de service.
- Les paramètres de l'optimiseur au niveau du catalogue ne sont pas automatiquement hérités par les tables lorsque le AWS Glue Data Catalog chiffrement est activé. Si le chiffrement des métadonnées est activé dans votre catalogue, vous devez configurer les optimiseurs de table individuellement pour chaque table. Pour utiliser l'héritage de l'optimiseur au niveau du catalogue, le chiffrement des métadonnées doit être désactivé dans le catalogue.

Rubriques

- Activation de l'optimisation automatique des tables au niveau du catalogue
- Affichage des optimisations au niveau du catalogue
- Désactivation de l'optimisation des tables au niveau du catalogue

Activation de l'optimisation automatique des tables au niveau du catalogue

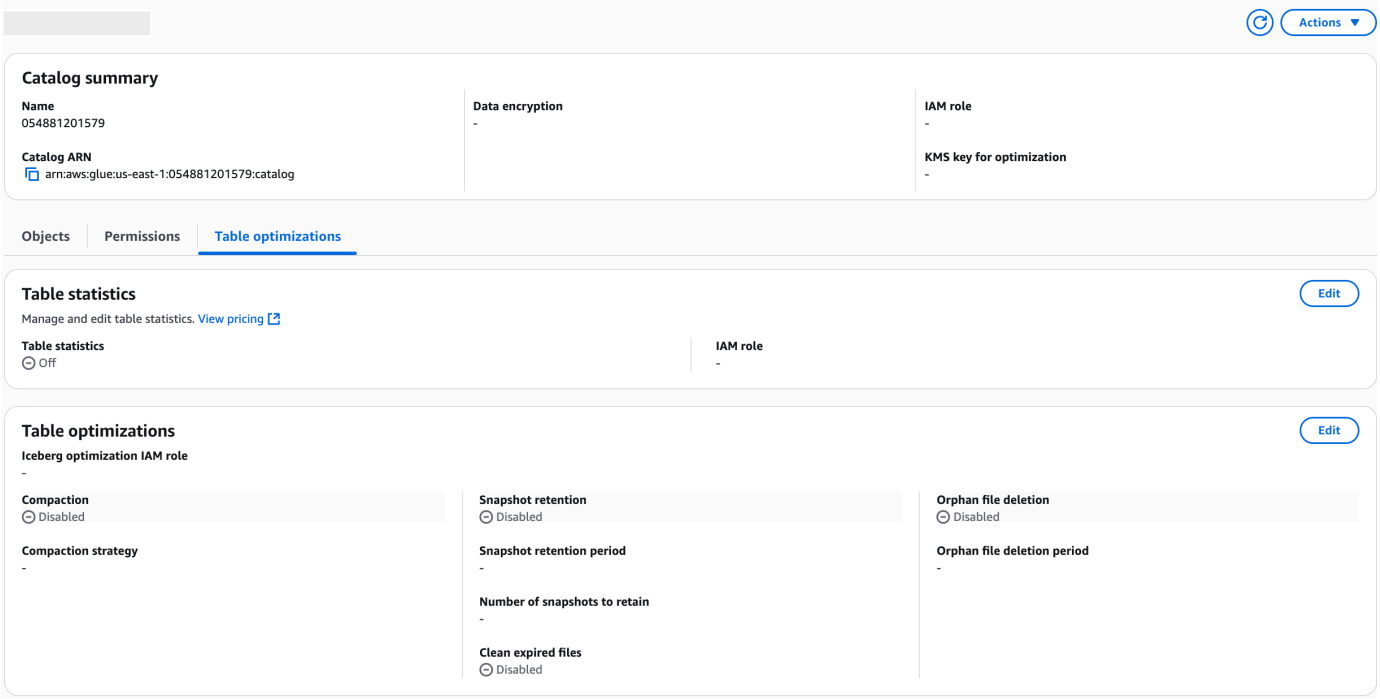
Vous pouvez activer l'optimisation automatique des tables pour toutes les nouvelles tables Apache Iceberg du catalogue de données. Après avoir créé la table, vous pouvez également mettre à jour manuellement de manière explicite les paramètres d'optimisation des tables.

Pour mettre à jour les paramètres du catalogue de données afin de permettre l'optimisation des tables au niveau du catalogue, le rôle IAM utilisé doit disposer de l'autorisation `glue:UpdateCatalog` sur le catalogue racine. Vous pouvez utiliser l'API `GetCatalog` pour vérifier les propriétés du catalogue.

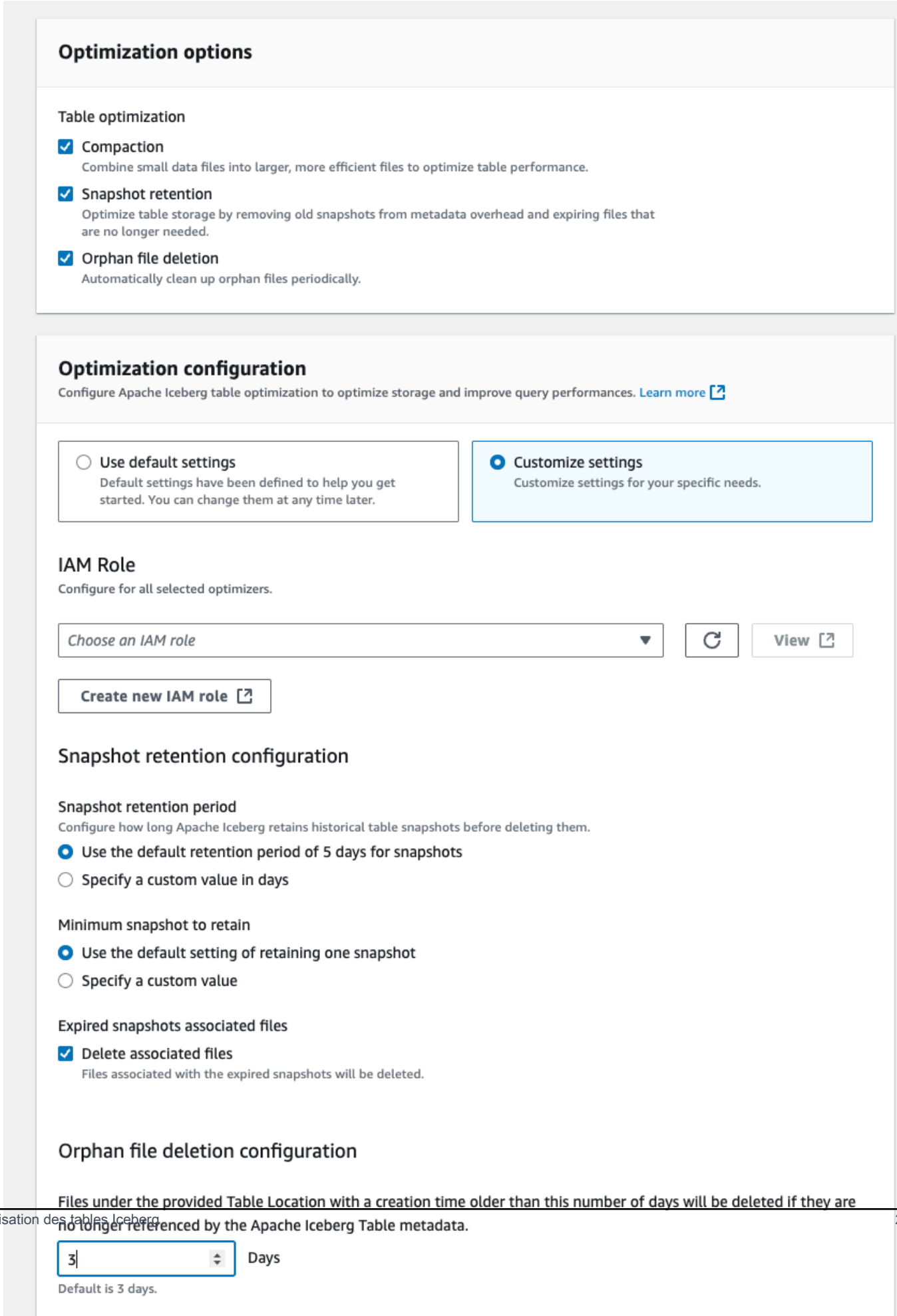
Pour les tables gérées par Lake Formation, le rôle IAM sélectionné lors de la configuration d'optimisation du catalogue a besoin des autorisations Lake Formation `ALTER`, `DESCRIBE`, `INSERT` et `DELETE` pour toute nouvelle table ou pour les tables mises à jour.

Pour activer les optimiseurs au niveau du catalogue (console)

1. Ouvrez la console Lake Formation à l'adresse https://console.aws.amazon.com/lakeformation/.
2. Dans le panneau de navigation, choisissez catalogue de données.
3. Sélectionnez l'onglet Catalogues.
4. Choisissez le catalogue au niveau du compte.
5. Choisissez Optimisations des tables, puis Modifier sous l'onglet Optimisations des tables. Vous pouvez également choisir Modifier les optimisations dans Actions.



6. Sur la page Optimisation des tables, configurez les options suivantes :


Optimization options
Table optimization
Compaction
Combine small data files into larger, more efficient files to optimize table performance.
Snapshot retention
Optimize table storage by removing old snapshots from metadata overhead and expiring files that are no longer needed.
Orphan file deletion
Automatically clean up orphan files periodically.
Optimization configuration
Configure Apache Iceberg table optimization to optimize storage and improve query performances. Learn more
Use default settings
Default settings have been defined to help you get started. You can change them at any time later.
Customize settings
Customize settings for your specific needs.
IAM Role
Configure for all selected optimizers.
Choose an IAM role
View
Create new IAM role
Snapshot retention configuration
Snapshot retention period
Configure how long Apache Iceberg retains historical table snapshots before deleting them.
Use the default retention period of 5 days for snapshots
Specify a custom value in days
Minimum snapshot to retain
Use the default setting of retaining one snapshot
Specify a custom value
Expired snapshots associated files
Delete associated files
Files associated with the expired snapshots will be deleted.
Orphan file deletion configuration
Files under the provided Table Location with a creation time older than this number of days will be deleted if they are no longer referenced by the Apache Iceberg Table metadata.
3
Days
Default is 3 days.

a. Configurez les paramètres de compactage :

- Activez/désactivez le compactage.
- Choisissez le rôle IAM disposant des autorisations nécessaires pour exécuter les optimiseurs.

  Pour en savoir plus sur les autorisations requises pour le rôle IAM, consultez Conditions préalables requises pour l'optimisation des tables.

b. Configurez les paramètres de conservation des instantanés :

- Activez/désactivez la conservation.
- Définissez la période de conservation des instantanés en jours. La valeur par défaut est de cinq jours.
- Définissez le nombre d'instantanés à retenir. La valeur par défaut est d'un instantané.
- Activez/désactivez le nettoyage des fichiers expirés.

c. Configurez les paramètres de suppression des fichiers orphelins :

- Activez/désactivez la suppression des fichiers orphelins.
- Définissez la période de conservation des fichiers orphelins en jours. La valeur par défaut est de trois jours.

7. Choisissez Enregistrer.

Activation des optimiseurs au niveau du catalogue via AWS CLI

Utilisez la commande CLI suivante pour mettre à jour un catalogue existant avec les paramètres de l'optimiseur :

Example Mise à jour du catalogue avec les paramètres de l'optimiseur

```
aws glue update-catalog \
   --name catalog-id \
  --catalog-input \
  '{
    "CatalogId": "111122223333",
    "CatalogInput": {
        "CatalogProperties": {
            "CustomProperties": {
                "ColumnStatistics.Enabled": "false",
```

```
                "ColumnStatistics.RoleArn": "arn:aws:iam::111122223333:role/service-
role/stats-role-name"
            },
            "IcebergOptimizationProperties": {
                "RoleArn": "arn:aws:iam::111122223333:role/optimizer-role-name",
                "Compaction": {
                    "enabled": "true"
                },
                "Retention": {
                    "enabled": "true",
                    "snapshotRetentionPeriodInDays": "10",
                    "numberOfSnapshotsToRetain": "5",
                    "cleanExpiredFiles": "true"
                },
                "OrphanFileDeletion": {
                    "enabled": "true",
                    "orphanFileRetentionPeriodInDays": "3"
                }
            }
        }
    }
}'
```

Si vous rencontrez des problèmes avec les optimiseurs au niveau du catalogue, vérifiez les points suivants :

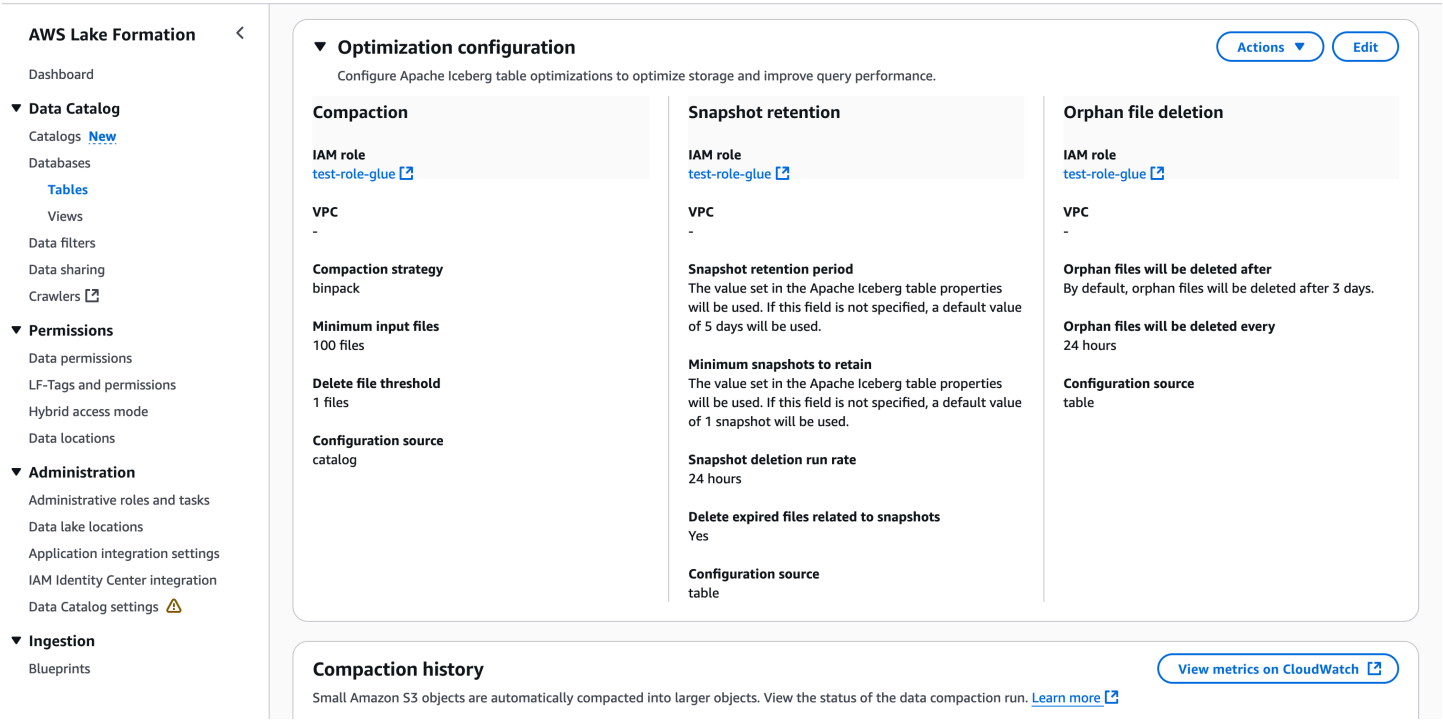
- Assurez-vous que le rôle IAM dispose des autorisations appropriées, comme indiqué dans la section Conditions préalables.
- Consultez les CloudWatch journaux pour détecter tout message d'erreur lié aux opérations de l'optimiseur.

  Pour de plus amples informations, consultez Affichage des métriques disponibles dans le Guide de l'utilisateur Amazon CloudWatch .
- Vérifiez que les paramètres du catalogue ont été correctement appliqués en vérifiant la configuration du catalogue.
- En cas d'échec d'accès aux tables, consultez les CloudWatch journaux et EventBridge les notifications pour obtenir des informations détaillées sur les erreurs.

Affichage des optimisations au niveau du catalogue

Lorsque l'optimisation des tables au niveau du catalogue est activée, chaque fois qu'une table Apache Iceberg est créée ou mise à jour via le `CreateTable` SDK ou AWS Glue crawler qu'un paramètre de niveau de table équivalent est créé pour cette table. `UpdateTable` APIs AWS Management Console

Après avoir créé ou mis à jour une table, vous pouvez vérifier ses détails pour confirmer son optimisation. `Table optimization` affiche la propriété `Configuration source` définie comme `Catalog`.



Désactivation de l'optimisation des tables au niveau du catalogue

Vous pouvez désactiver l'optimisation des tables pour les nouvelles tables à l'aide de la AWS Lake Formation console, l'`glue:UpdateCatalog`API.

Pour désactiver les optimisations des tables au niveau du catalogue

1. Ouvrez la console Lake Formation à l'adresse https://console.aws.amazon.com/lakeformation/.
2. Dans la barre de navigation de gauche, choisissez Catalogues.
3. Sur la page Résumé du catalogue, choisissez Modifier sous Optimisations des tables.
4. Sur la page Modifier l'optimisation, désélectionnez les options d'optimisation.

5. Choisissez Enregistrer.

Optimisation du compactage

Les lacs de données Amazon S3 utilisant des formats de table ouverts comme Apache Iceberg stockent les données sous forme d'objets S3. La présence de milliers de petits objets Amazon S3 dans une table de lac de données augmente la surcharge de métadonnées et affecte les performances de lecture. AWS Glue Data Catalog fournit un compactage géré pour les tables Iceberg, en compactant de petits objets en de plus grands objets pour améliorer les performances de lecture grâce à des services AWS d'analyse tels qu' Amazon Athena Amazon EMR et à des tâches ETL. AWS Glue Le catalogue de données effectue le compactage sans interférer avec les requêtes simultanées et prend en charge le compactage uniquement pour les tables au format Parquet.

L'optimiseur de table surveille en permanence les partitions des tables et lance le processus de compactage lorsque le seuil du nombre et de la taille des fichiers est dépassé.

Dans le catalogue de données, le processus de compactage démarre lorsqu'une table ou l'une de ses partitions contient plus de 100 fichiers. Chaque fichier doit être inférieur à 75 % de la taille du fichier cible. La taille du fichier cible est définie par la propriété de la table `write.target-file-size-bytes`, qui est par défaut de 512 Mo si elle n'est pas définie explicitement.

Pour connaître les limitations, veuillez consulter Formats pris en charge et restrictions pour le compactage de données géré.

Rubriques

- Activation de l'optimiseur de compactage
- Désactivation de l'optimiseur de compactage

Activation de l'optimiseur de compactage

Vous pouvez utiliser AWS Glue la console ou l' AWS API pour activer le compactage de vos tables Apache Iceberg dans le catalogue de AWS Glue données. AWS CLI Pour les nouvelles tables, vous pouvez choisir Apache Iceberg comme format de table et activer le compactage lors de la création de la table. Le compactage est désactivé par défaut pour les nouvelles tables.

Console

Pour activer le compactage

1. Ouvrez la AWS Glue console https://console.aws.amazon.com/glue/et connectez-vous en tant qu'administrateur du lac de données, créateur de table ou utilisateur ayant obtenu les `lakeformation:GetDataAccess` autorisations `glue:UpdateTable` et sur la table.
2. Dans le panneau de navigation, sous Catalogue de données, choisissez Tables.
3. Sur la page Tables, choisissez une table au format ouvert pour laquelle vous souhaitez activer le compactage, puis dans le menu Actions, choisissez Optimisation, puis Activer.

   Vous pouvez également activer le compactage en sélectionnant l'onglet Optimisation des tables sur la page Détails de la table. Choisissez l'onglet Optimisation des tables dans la partie inférieure de la page, puis sélectionnez Activer le compactage.

   L'option Activer l'optimisation est également disponible lorsque vous créez une table Iceberg dans le catalogue de données.
4. Sur la page Activer l'optimisation, sélectionnez Compactage sous Options d'optimisation.

Optimization configuration

Configure Apache Iceberg table optimization to optimize storage and improve query performances. Learn more

- Use default settings
  Default settings have been defined to help you get started. You can change them at any time later.
- Customize settings
  Customize settings for your specific needs.

IAM Role

Configure for all selected optimizers.

- [x] Apply the selected IAM role to all selected optimizers. Update individual roles later as needed.
  Create a single role with necessary permissions to configure all table optimizers. Learn more

IAM role

Choose an IAM role | View

Create new IAM role

Virtual private cloud (VPC) - *optional*

Choose a network connection to access a data store in your VPC.

Choose a Glue connection | Create Glue network connection

Compaction configuration

Compaction strategy

Choose a compaction strategy Learn more

- Binpack
  Combines small files into larger files, typically targeting sizes over 100MB, while applying any pending deletes. This is the default and recommended compaction strategy for most use cases.
- Sort
  Organizes data based on specified columns which are sorted hierarchically during compaction, improving query performance for filtered operations. This strategy is recommended when your queries frequently filter on specific columns. To use this strategy, you must first define a sort order in your Iceberg table properties using the 'sort_order' table property.
- Z-Order
  Optimizes data organization by blending multiple attributes into a single scalar value that can be used for sorting, allowing efficient querying across multiple dimensions. This strategy is recommended when you need to query data across multiple dimensions simultaneously. To use this strategy, you must first define a sort order in your Apache Iceberg table properties using the 'sort_order' table property.

Minimum input files

Minimum number of data-files to be present in a partition before compaction will actually compact files.

100 files

Default is 100 files.

Delete file threshold

Minimum number of deletes in a data-file to be eligible for compaction.

1 deletes

Default is 1 delete.

5. Sélectionnez ensuite un rôle IAM dans le menu déroulant avec les autorisations indiquées dans la section Conditions préalables requises pour l'optimisation des tables.

   Vous pouvez aussi choisir l'option Créer un rôle IAM pour créer un rôle personnalisé avec des autorisations requises pour exécuter le compactage.

   Suivez les étapes ci-dessous pour mettre à jour un rôle IAM existant :

   a. Pour mettre à jour la stratégie d'autorisation pour le rôle IAM, dans la console IAM, accédez au rôle IAM utilisé pour exécuter le compactage.

   b. Dans la section Ajouter des autorisations, choisissez Créer une politique. Dans la fenêtre du navigateur nouvellement ouverte, créez une nouvelle stratégie à utiliser avec votre rôle.

   c. Sur la page Créer une politique, choisissez l'onglet `JSON`. Copiez le code JSON affiché dans les prérequis dans le champ de l'éditeur de politique.

6. Si vous avez des configurations de politique de sécurité dans lesquelles l'optimiseur de table Iceberg doit accéder aux compartiments Amazon S3 à partir d'un Virtual Private Cloud (VPC) spécifique, créez AWS Glue une connexion réseau ou utilisez une connexion réseau existante.

   Si aucune connexion AWS Glue VPC n'est déjà configurée, créez-en une nouvelle en suivant les étapes de la section Création de connexions pour les connecteurs à l'aide de la AWS Glue console ou du /SDK. AWS CLI

7. Choisissez une politique de compactage. Les options disponibles sont les suivantes :

   - Binpack : Binpack est la stratégie de compactage par défaut d'Apache Iceberg. Elle combine des fichiers de données plus petits dans des fichiers plus volumineux pour des performances optimales.
   - Tri : le tri dans Apache Iceberg est une technique d'organisation des données qui regroupe les informations au sein de fichiers en fonction de colonnes spécifiées, ce qui améliore considérablement les performances des requêtes en réduisant le nombre de fichiers à traiter. Vous définissez l'ordre de tri dans les métadonnées d'Iceberg à l'aide du champ d'ordre de tri, et lorsque plusieurs colonnes sont spécifiées, les données sont triées dans l'ordre dans lequel les colonnes apparaissent dans l'ordre de tri, ce qui garantit que les enregistrements présentant des valeurs similaires sont stockés ensemble dans des fichiers. La stratégie de tri du compactage va plus loin dans l'optimisation en triant les données dans tous les fichiers d'une partition.
   - Ordre Z : l'ordre Z est un moyen d'organiser les données lorsque vous devez trier selon plusieurs colonnes de la même importance. Contrairement au tri traditionnel qui donne la priorité à une colonne par rapport aux autres, l'ordre Z donne une pondération équilibrée à

chaque colonne, ce qui permet à votre moteur de requête de lire moins de fichiers lors de la recherche de données.

La technique fonctionne en combinant les chiffres binaires de valeurs issues de différentes colonnes. Par exemple, si vous avez les chiffres 3 et 4 sur deux colonnes, l'ordre Z les convertit d'abord en binaire (3 devient 011 et 4 devient 100), puis entrelace ces chiffres pour créer une valeur : 011010. Cet entrelacement crée un modèle qui maintient les données associées physiquement proches les unes des autres.

L'ordre Z est particulièrement efficace pour les requêtes multidimensionnelles. Par exemple, une table client triée en Z par revenu, État et code postal peut offrir des performances supérieures à celles d'un tri hiérarchique lors d'une interrogation portant sur plusieurs dimensions. Cette organisation permet aux requêtes ciblant des combinaisons spécifiques de revenus et de situation géographique de localiser rapidement les données pertinentes tout en minimisant les analyses de fichiers inutiles.

8. Minimum de fichiers d'entrée : nombre de fichiers de données requis dans une partition avant le déclenchement du compactage.
9. Seuil de suppression de fichiers : nombre minimum d'opérations de suppression requises dans un fichier de données avant qu'il ne soit éligible au compactage.
10. Choisissez Activer l'optimisation.

AWS CLI

L'exemple suivant montre comment activer le compactage. Remplacez l'identifiant de compte par un identifiant de AWS compte valide. Remplacez le nom de la base de données et le nom de la table par un nom réel de la table Iceberg et le nom de la base de données. Remplacez le `roleArn` par l'ARN (Amazon Resource Name) AWS du rôle IAM et le nom du rôle IAM qui dispose des autorisations requises pour exécuter le compactage. Vous pouvez remplacer la stratégie de compactage `sort` par d'autres stratégies prises en charge, telles que `z-order` ou `binpack`.

commander en fonction de vos besoins.

```
aws glue create-table-optimizer \
  --catalog-id 123456789012 \
  --database-name iceberg_db \
  --table-name iceberg_table \
  --table-optimizer-configuration '{
```

```
    "roleArn": "arn:aws:iam::123456789012:role/optimizer_role",
    "enabled": true,
    "vpcConfiguration": {"glueConnectionName": "glue_connection_name"},
    "compactionConfiguration": {
      "icebergConfiguration": {"strategy": "sort"}
    }
  }'\
--type compaction
```

AWS API

Appelez une opération CreateTableOptimizer pour activer le compactage d'une table.

Après avoir activé le compactage, l'onglet Optimisation des tables affiche les détails de compactage suivants une fois le compactage terminé :

L'heure de début

Heure à laquelle le processus de compactage a commencé dans le catalogue de données. La valeur est un horodatage selon le fuseau UTC.

L'heure de fin

Heure à laquelle le processus de compactage a pris fin dans le catalogue de données. La valeur est un horodatage selon le fuseau UTC.

Statut

L'état d'exécution du compactage. Les valeurs sont la réussite ou l'échec.

Fichiers compactés

Nombre total de fichiers compactés.

Octets compactés

Nombre total d'octets compactés.

Désactivation de l'optimiseur de compactage

Vous pouvez désactiver le compactage automatique pour une table Apache Iceberg particulière à l'aide de AWS Glue la console ou. AWS CLI

Console

1. Connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/.
2. Dans le panneau de navigation de gauche, sous Catalogue de données, choisissez Tables.
3. Dans la liste des tables, choisissez la table Iceberg dont vous souhaitez désactiver le compactage.
4. Choisissez l'onglet Optimisation des tables dans la section inférieure de la page Détails des tables.
5. Dans Actions, choisissez Désactiver, puis sélectionnez Compactage.
6. Choisissez Désactiver le compactage dans le message de confirmation. Vous pouvez réactiver le compactage ultérieurement.

   Une fois que vous avez confirmé, le compactage est désactivé et l'état de compactage de la table revient à `Disabled`.

AWS CLI

Dans l'exemple suivant, remplacez l'ID de compte par un ID de AWS compte valide. Remplacez le nom de la base de données et le nom de la table par un nom réel de la table Iceberg et le nom de la base de données. Remplacez le `roleArn` par le nom de AWS ressource (ARN) du rôle IAM et le nom réel du rôle IAM disposant des autorisations requises pour exécuter le compactage.

```
aws glue update-table-optimizer \
  --catalog-id 123456789012 \
  --database-name iceberg_db \
  --table-name iceberg_table \
  --table-optimizer-configuration
 '{"roleArn":"arn:aws:iam::123456789012:role/optimizer_role", "enabled":'false',
 "vpcConfiguration":{"glueConnectionName":"glue_connection_name"}}'\
  --type compaction
```

AWS API

Appelez l'opération UpdateTableOptimizer pour désactiver le compactage d'une table spécifique.

Optimisation de la conservation des instantanés

La fonctionnalité de conservation des instantanés Apache Iceberg permet aux utilisateurs d'interroger les données historiques à des moments précis et d'annuler les modifications indésirables apportées à leurs tables. Dans le catalogue de AWS Glue données, la configuration de conservation des instantanés contrôle la durée de conservation de ces instantanés (versions des données de table) avant leur expiration et leur suppression. Cela permet de gérer les coûts de stockage et la surcharge de métadonnées en supprimant automatiquement les anciens instantanés en fonction d'une période de conservation configurée ou du nombre maximal d'instantanés à conserver.

Vous pouvez configurer la période de conservation en jours et le nombre maximal de clichés à conserver pour une table. AWS Glue supprime les instantanés antérieurs à la période de rétention spécifiée des métadonnées de la table, tout en préservant les instantanés les plus récents dans les limites configurées. Après avoir supprimé les anciens instantanés des métadonnées, AWS Glue supprime les données et les fichiers de métadonnées correspondants qui ne sont plus référencés et uniques aux instantanés expirés. Cela permet d'autoriser les requêtes Time Travel uniquement jusqu'aux instantanés conservés restants, tout en récupérant l'espace de stockage utilisé par les données d'instantanés expirés.

Rubriques

- Activation de l'optimiseur de conservation des instantanés
- Mise à jour de l'optimiseur de conservation des instantanés
- Désactivation de l'optimiseur de conservation des instantanés

Activation de l'optimiseur de conservation des instantanés

Vous pouvez utiliser AWS Glue la console ou l' AWS API pour activer les optimiseurs de rétention des instantanés pour vos tables Apache Iceberg dans le catalogue de données. AWS CLI Pour les nouvelles tables, vous pouvez choisir Apache Iceberg comme format de table et activer l'optimiseur de conservation des instantanés lors de la création de la table. La conservation des instantanés est désactivée par défaut pour les nouvelles tables.

Console

Pour activer l'optimiseur de conservation des instantanés

1. Ouvrez la AWS Glue console https://console.aws.amazon.com/glue/et connectez-vous en tant qu'administrateur du lac de données, créateur de table ou utilisateur ayant obtenu les `lakeformation:GetDataAccess` autorisations `glue:UpdateTable` et sur la table.
2. Dans le panneau de navigation, sous Catalogue de données, choisissez Tables.
3. Sur la page Tables, choisissez une table Iceberg pour laquelle vous souhaitez activer l'optimiseur de conservation des instantanés, puis dans le menu Actions, choisissez Activer sous Optimisation.

   Vous pouvez également activer l'optimisation en sélectionnant la table et en ouvrant la page Détails de la table. Choisissez l'onglet Optimisation des tables dans la partie inférieure de la page, puis sélectionnez Activer la conservation des instantanés.
4. Sur la page Activer l'optimisation, sous Configuration de l'optimisation, vous avez deux options : Utiliser les paramètres par défaut ou Personnaliser les paramètres. Si vous choisissez d'utiliser les paramètres par défaut, AWS Glue utilise les propriétés définies dans la configuration de la table Iceberg pour déterminer la période de conservation des instantanés et le nombre de clichés à conserver. En l'absence de cette configuration, AWS Glue conserve un instantané pendant cinq jours et supprime les fichiers associés aux instantanés expirés.
5. Ensuite, choisissez un rôle IAM qui AWS Glue peut assumer en votre nom l'exécution de l'optimiseur. Pour en savoir plus sur les autorisations requises pour le rôle IAM, consultez la section Conditions préalables requises pour l'optimisation des tables.

   Suivez les étapes ci-dessous pour mettre à jour un rôle IAM existant :

   a. Pour mettre à jour la stratégie d'autorisation pour le rôle IAM, dans la console IAM, accédez au rôle IAM utilisé pour exécuter le compactage.

   b. Dans la section Ajouter des autorisations, choisissez Créer une politique. Dans la fenêtre du navigateur nouvellement ouverte, créez une nouvelle stratégie à utiliser avec votre rôle.

   c. Sur la page Créer une politique, choisissez l'onglet JSON. Copiez le code JSON affiché dans les prérequis dans le champ de l'éditeur de politique.
6. Si vous préférez définir les valeurs de la configuration de conservation des instantanés manuellement, choisissez Personnaliser les paramètres.

Optimization configuration

Configure Apache Iceberg table optimization to optimize storage and improve query performances. Learn more

- Use default settings
  Default settings have been defined to help you get started. You can change them at any time later.
- Customize settings
  Customize settings for your specific needs.

IAM Role

Configure for all selected optimizers.

- [x] Apply the selected IAM role to all selected optimizers. Update individual roles later as needed.
  Create a single role with necessary permissions to configure all table optimizers. Learn more

IAM role

Admin

View

Create new IAM role

Virtual private cloud (VPC) - *optional*

Choose a network connection to access a data store in your VPC.

Choose a Glue connection

Create Glue network connection

Snapshot retention configuration

Snapshot retention period

Configure how long Apache Iceberg retains historical table snapshots before deleting them.

- Use the value specified in Apache Iceberg table configuration
  If no value is provided, use the default retention period of 5 days for snapshots.
- Specify a custom value in days

Minimum snapshot to retain

- Use the value specified in Apache Iceberg table configuration
  If no value is provided, use the default setting of retaining one snapshot.
- Specify a custom value

Snapshot deletion run rate

Configure the interval in hours between two deletion job runs.

- Use the value specified in Apache Iceberg table configuration
  If no value is provided, use the default setting of 24 hours between deletion jobs.
- Specify a custom value in hours

Expired snapshots associated files

- [x] Delete associated files
  Files associated with the expired snapshots will be deleted.

7. Cochez la case Appliquer le rôle IAM sélectionné aux optimiseurs sélectionnés pour utiliser un seul rôle IAM pour tout en activant tous les optimiseurs.
8. Si vous avez des configurations de politique de sécurité dans lesquelles l'optimiseur de table Iceberg doit accéder aux compartiments Amazon S3 à partir d'un Virtual Private Cloud (VPC) spécifique, créez AWS Glue une connexion réseau ou utilisez une connexion réseau existante.

   Si aucune connexion AWS Glue VPC n'est déjà configurée, créez-en une nouvelle en suivant les étapes de la section Création de connexions pour les connecteurs à l'aide de la AWS Glue console ou du /SDK. AWS CLI
9. Ensuite, sous Configuration de conservation des instantanés, choisissez d'utiliser les valeurs spécifiées dans la configuration de la table Iceberg ou de spécifier des valeurs personnalisées pour la période de conservation des instantanés (history.expire). max-snapshot-age-ms), nombre minimum de clichés (history.expire. min-snapshots-to-keep) à conserver, ainsi que le délai en heures entre deux exécutions consécutives de suppression d'instantanés.
10. Choisissez Supprimer les fichiers associés pour supprimer les fichiers sous-jacents lorsque l'optimiseur de table supprime les anciens instantanés des métadonnées de la table.

    Si vous ne choisissez pas cette option, lorsque les anciens instantanés sont supprimés des métadonnées de la table, les fichiers associés resteront dans le stockage en tant que fichiers orphelins.
11. Ensuite, lisez la mise en garde, puis choisissez Je confirme pour continuer.

    > **Note**
    >
    > Dans le catalogue de données, l'optimiseur de conservation des instantanés respecte le cycle de vie contrôlé par les politiques de conservation au niveau des branches et des balises. Pour plus d'informations, consultez la section Branching and tagging dans la documentation Iceberg.
12. Examinez la configuration et choisissez Activer l'optimisation.

    Attendez quelques minutes que l'optimiseur de conservation s'exécute et que les anciens instantanés expirent en fonction de la configuration.

AWS CLI

Pour activer la conservation des instantanés pour les nouvelles tables Iceberg dans AWS Glue, vous devez créer un optimiseur de table de type `retention` et définir le `enabled` champ sur `true` dans le. `table-optimizer-configuration` Vous pouvez le faire à l'aide de la AWS CLI commande `create-table-optimizer` ou`update-table-optimizer`. En outre, vous devez spécifier les champs de configuration de conservation tels que `snapshotRetentionPeriodInDays` et `numberOfSnapshotsToRetain` en fonction de vos besoins.

L'exemple suivant montre comment activer l'optimiseur de conservation des instantanés. Remplacez l'identifiant de compte par un identifiant de AWS compte valide. Remplacez le nom de la base de données et le nom de la table par un nom réel de la table Iceberg et le nom de la base de données. Remplacez le `roleArn` par le nom de AWS ressource (ARN) du rôle IAM et le nom du rôle IAM disposant des autorisations requises pour exécuter l'optimiseur de rétention des instantanés.

```
aws glue create-table-optimizer \
  --catalog-id 123456789012 \
  --database-name iceberg_db \
  --table-name iceberg_table \
  --table-optimizer-configuration
 '{"roleArn":"arn:aws:iam::123456789012:role/optimizer_role","enabled":'true',
 "vpcConfiguration":{
"glueConnectionName":"glue_connection_name"}, "retentionConfiguration":
{"icebergConfiguration":
{"snapshotRetentionPeriodInDays":7,"numberOfSnapshotsToRetain":3,"cleanExpiredFiles":'true'}
  --type retention
```

Cette commande crée un optimiseur de conservation pour la table Iceberg spécifiée dans le catalogue, la base de données et la région donnés. table-optimizer-configurationSpécifie l'ARN du rôle IAM à utiliser, active l'optimiseur et définit la configuration de rétention. Dans cet exemple, il conserve les instantanés pendant sept jours, garde au moins trois instantanés et nettoie les fichiers expirés.

- snapshotRetentionPeriodInDays : le nombre de jours pendant lesquels les instantanés sont conservés avant leur expiration. La valeur par défaut est 5.
- numberOfSnapshotsToRetain — Le nombre minimum de clichés à conserver, même s'ils sont antérieurs à la période de conservation. La valeur par défaut est 1.

- cleanExpiredFiles — Un booléen indiquant s'il faut supprimer les fichiers de données expirés après l'expiration des instantanés. La valeur par défaut est `true`.

  Lorsque ce paramètre est défini sur true, les anciens instantanés sont supprimés des métadonnées de la table et leurs fichiers sous-jacents sont supprimés. Si ce paramètre est défini sur false, les anciens instantanés sont supprimés des métadonnées de la table, mais leurs fichiers sous-jacents restent dans le stockage en tant que fichiers orphelins..

AWS API

CreateTableOptimizerOpération d'appel pour activer l'optimiseur de rétention des instantanés pour une table.

Après avoir activé le compactage, l'onglet Optimisation des tables affiche les détails de compactage suivants (après environ 15 à 20 minutes) :

L'heure de début

L'heure à laquelle l'optimiseur de conservation des instantanés a démarré. La valeur est un horodatage selon le fuseau UTC.

Durée de l'exécution

Le temps indique la durée nécessaire à l'optimiseur pour terminer la tâche. La valeur est un horodatage selon le fuseau UTC.

Statut

L'état d'exécution de l'optimiseur. Les valeurs sont la réussite ou l'échec.

Fichiers de données supprimés

Nombre total de fichiers supprimés.

Fichiers manifestes supprimés

Nombre total de fichiers manifestes supprimés.

Listes de manifestes supprimées

Nombre total de listes de manifestes supprimées.

Mise à jour de l'optimiseur de conservation des instantanés

Vous pouvez mettre à jour la configuration existante d'un optimiseur de rétention des instantanés pour une table Apache Iceberg particulière à l'aide de la AWS Glue console ou de l' UpdateTableOptimizerAPI. AWS CLI

Console

Pour mettre à jour la configuration de la conservation des instantanés

1. Connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/.
2. Choisissez Catalogue de données, puis choisissez Tables. Dans la liste des tables, choisissez la table Iceberg dont vous souhaitez mettre à jour la configuration de l'optimiseur de conservation des instantanés.
3. Dans la section inférieure de la page Détails des tables, sélectionnez l'onglet Optimisation des tables, puis Modifier. Vous pouvez également choisir Modifier sous Optimisation dans le menu Actions situé dans le coin supérieur droit de la page.
4. Sur la page Modifier l'optimisation, apportez les modifications souhaitées.
5. Choisissez Enregistrer.

AWS CLI

Pour mettre à jour un optimiseur de conservation des instantanés à l'aide de AWS CLI, vous pouvez utiliser la commande suivante :

```
aws glue update-table-optimizer \
 --catalog-id 123456789012 \
 --database-name iceberg_db \
 --table-name iceberg_table \
 --table-optimizer-configuration
 '{"roleArn":"arn:aws:iam::123456789012:role/optimizer_role"","enabled":'true',
 "vpcConfiguration":
{"glueConnectionName":"glue_connection_name"},"retentionConfiguration":
{"icebergConfiguration":
{"snapshotRetentionPeriodInDays":7,"numberOfSnapshotsToRetain":3,"cleanExpiredFiles":'true'}
 \
 --type retention
```

Cette commande met à jour la configuration de rétention pour la table spécifiée dans le catalogue, la base de données et la région donnés. Les principaux paramètres sont les suivants :

- snapshotRetentionPeriodInDays : le nombre de jours pendant lesquels les instantanés sont conservés avant leur expiration. La valeur par défaut est `1`.
- numberOfSnapshotsToRetain — Le nombre minimum de clichés à conserver, même s'ils sont antérieurs à la période de conservation. La valeur par défaut est `5`.
- cleanExpiredFiles — Un booléen indiquant s'il faut supprimer les fichiers de données expirés après l'expiration des instantanés. La valeur par défaut est `true`.

  Lorsque ce paramètre est défini sur true, les anciens instantanés sont supprimés des métadonnées de la table et leurs fichiers sous-jacents sont supprimés. Si ce paramètre est défini sur false, les anciens instantanés sont supprimés des métadonnées de la table, mais leurs fichiers sous-jacents restent dans le stockage en tant que fichiers orphelins..

API

Pour mettre à jour un optimiseur de table, vous pouvez utiliser l'API `UpdateTableOptimizer`. Cette API vous permet de mettre à jour la configuration d'un optimiseur de table existant pour le compactage, la conservation ou la suppression de fichiers orphelins. Les paramètres de demande comprennent :

- catalogId (obligatoire) : l'ID du catalogue contenant la table
- databaseName (facultatif) : le nom de la base de données contenant la table
- tableName (facultatif) : le nom de la table
- type (obligatoire) : le type d'optimiseur de table (compaction, retention ou orphan_file_delete)
- retentionConfiguration (obligatoire) : configuration mise à jour de l'optimiseur de table, y compris l'ARN du rôle, l'état activé, la configuration de la conservation et la configuration de suppression des fichiers orphelins

Désactivation de l'optimiseur de conservation des instantanés

Vous pouvez désactiver l'optimiseur de rétention des instantanés pour une table Apache Iceberg particulière à l'aide de AWS Glue la console ou. AWS CLI

Console

Pour désactiver la conservation des instantanés

1. Connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/.
2. Choisissez Catalogue de données, puis choisissez Tables. Dans la liste des tables, choisissez la table Iceberg dont vous souhaitez désactiver l'optimiseur pour la conservation des instantanés.
3. Dans la section inférieure de la page des Détails des tables, choisissez Optimisation des tables et Désactiver, Conservation des instantanés sous Actions.

   Vous pouvez également choisir Désactiver sous Optimisation dans le menu Actions situé dans le coin supérieur droit de la page.
4. Choisissez Désactiver dans le message de confirmation. Vous pouvez réactiver l'optimiseur de conservation des instantanés ultérieurement.

   Une fois que vous avez confirmé, l'optimiseur de conservation des instantanés est désactivé et l'état de conservation des instantanés revient à `Not enabled`.

AWS CLI

Dans l'exemple suivant, remplacez l'ID de compte par un ID de AWS compte valide. Remplacez le nom de la base de données et le nom de la table par un nom réel de la table Iceberg et le nom de la base de données. Remplacez le `roleArn` par le nom de AWS ressource (ARN) du rôle IAM et le nom réel du rôle IAM disposant des autorisations requises pour exécuter l'optimiseur de rétention.

```
aws glue update-table-optimizer \
  --catalog-id 123456789012 \
  --database-name iceberg_db \
  --table-name iceberg_table \
  --table-optimizer-configuration
 '{"roleArn":"arn:aws:iam::123456789012:role/optimizer_role", "vpcConfiguration":
{"glueConnectionName":"glue_connection_name"}, "enabled":'false'}'\
  --type retention
```

AWS API

UpdateTableOptimizerOpération d'appel pour désactiver l'optimiseur de rétention des instantanés pour une table spécifique.

Suppression des fichiers orphelins

AWS Glue Data Catalog vous permet de supprimer les fichiers orphelins de vos tables Iceberg. Les fichiers orphelins sont des fichiers non référencés qui existent dans votre source de données Amazon S3 à l'emplacement de la table spécifié, qui ne sont pas suivis par les métadonnées de la table Iceberg et qui sont plus anciens que la limite d'âge que vous avez configurée. Ces fichiers orphelins peuvent s'accumuler au fil du temps en cas d'échec d'opérations telles que le compactage, la suppression de partitions ou la réécriture de tables, et occuper un espace de stockage inutile.

L'optimiseur de suppression des fichiers orphelins AWS Glue analyse les métadonnées de la table et les fichiers de données réels, identifie les fichiers orphelins et les supprime pour récupérer de l'espace de stockage. L'optimiseur supprime uniquement les fichiers créés après la date de création de l'optimiseur qui répondent également aux critères de suppression configurés. Les fichiers créés avant ou à la date de création de l'optimiseur ne sont jamais supprimés.

Logique de suppression de fichiers orphelins

1. Vérification de la date : compare la date de création du fichier avec la date de création de l'optimiseur. Si le fichier est antérieur ou égal à la date de création de l'optimiseur, il est ignoré.
2. Vérification de la configuration de l'optimiseur : si le fichier est plus récent que la date de création de l'optimiseur, il évalue le fichier par rapport à la limite d'âge configurée. L'optimiseur supprime le fichier s'il répond aux critères de suppression. Ignore le fichier s'il ne correspond pas aux critères.

Vous pouvez initier la suppression des fichiers orphelins en créant un optimiseur de table de suppression de fichiers orphelins dans le catalogue de données.

> **Important**
>
> Par défaut, la suppression des fichiers orphelins évalue les fichiers situés à l'emplacement de votre AWS Glue table. Bien que vous puissiez configurer un sous-préfixe pour limiter la portée de l'évaluation à l'aide d'un paramètre d'API, vous devez vous assurer que l'emplacement de votre table ne contient pas de fichiers provenant d'autres sources de

données ou tables. Si l'emplacement de votre table chevauche celui d'autres sources de données, le service peut identifier et supprimer des fichiers non liés en tant qu'orphelins.

Rubriques

- Activation de la suppression des fichiers orphelins
- Mise à jour de l'optimiseur de suppression de fichiers orphelins
- Désactivation de la suppression des fichiers orphelins

Activation de la suppression des fichiers orphelins

Vous pouvez utiliser AWS Glue la console ou l' AWS API pour activer la suppression des fichiers orphelins pour vos tables Apache Iceberg dans le catalogue de données. AWS CLI Pour les nouvelles tables, vous pouvez choisir Apache Iceberg comme format de table et activer l'optimiseur de suppression de fichiers orphelins lors de la création de la table. La conservation des instantanés est désactivée par défaut pour les nouvelles tables.

Console

Pour activer la suppression des fichiers orphelins

1. Ouvrez la AWS Glue console https://console.aws.amazon.com/glue/et connectez-vous en tant qu'administrateur du lac de données, créateur de table ou utilisateur ayant obtenu les `lakeformation:GetDataAccess` autorisations `glue:UpdateTable` et sur la table.
2. Dans le panneau de navigation, sous Catalogue de données, choisissez Tables.
3. Sur la page Tables, choisissez une table Iceberg dans laquelle vous souhaitez activer la suppression des fichiers orphelins.

   Choisissez l'onglet Optimisation des tables dans la partie inférieure de la page, puis sélectionnez Activer, Suppression des fichiers orphelins dans Actions.

   Vous pouvez également choisir Activer sous Optimisation dans le menu Actions situé dans le coin supérieur droit de la page.
4. Sur la page Activer l'optimisation, sélectionnez Suppression des fichiers orphelins sous Options d'optimisation.

5. Si vous choisissez d'utiliser les paramètres par défaut, tous les fichiers orphelins seront supprimés au bout de trois jours. Si vous souhaitez conserver les fichiers orphelins pendant un certain nombre de jours, choisissez Personnaliser les paramètres.
6. Choisissez ensuite un rôle IAM avec les autorisations requises pour supprimer les fichiers orphelins.
7. Si vous avez des configurations de politique de sécurité dans lesquelles l'optimiseur de table Iceberg doit accéder aux compartiments Amazon S3 à partir d'un Virtual Private Cloud (VPC) spécifique, créez AWS Glue une connexion réseau ou utilisez une connexion réseau existante.

   Si aucune connexion AWS Glue VPC n'est déjà configurée, créez-en une nouvelle en suivant les étapes de la section Création de connexions pour les connecteurs à l'aide de la AWS Glue console ou du /SDK. AWS CLI
8. Si vous choisissez Personnaliser les paramètres, entrez le nombre de jours pendant lesquels les fichiers doivent être conservés avant leur suppression dans Configuration de la suppression des fichiers orphelins. Vous pouvez également indiquer l'intervalle entre deux exécutions d'optimiseur consécutives. La valeur par défaut est 24 heures.
9. Choisissez Activer l'optimisation.

AWS CLI

Pour activer la suppression des fichiers orphelins pour une table Iceberg dans AWS Glue, vous devez créer un optimiseur de table de type `orphan_file_deletion` et définir le `enabled` champ sur true. Pour créer un optimiseur de suppression de fichiers orphelins pour une table Iceberg à l'aide de AWS CLI, vous pouvez utiliser la commande suivante :

```
aws glue create-table-optimizer \
 --catalog-id 123456789012 \
 --database-name iceberg_db \
 --table-name iceberg_table \
 --table-optimizer-configuration
 '{"roleArn":"arn:aws:iam::123456789012:role/optimizer_role","enabled":true,
 "vpcConfiguration":{
"glueConnectionName":"glue_connection_name"}, "orphanFileDeletionConfiguration":
{"icebergConfiguration":{"orphanFileRetentionPeriodInDays":3, "location":'S3
 location'}}}'\
 --type orphan_file_deletion
```

Cette commande crée un optimiseur de suppression de fichiers orphelins pour la table Iceberg spécifiée. Les principaux paramètres sont les suivants :

- roleArn : l'ARN du rôle IAM autorisé à accéder au compartiment S3 et aux ressources Glue.
- enabled : définissez cette valeur sur true pour activer l'optimiseur.
- orphanFileRetentionPeriodInDays — Le nombre de jours pendant lesquels les fichiers orphelins sont conservés avant de les supprimer (minimum 1 jour).
- type : définissez la valeur sur orphan_file_deletion pour créer un optimiseur de suppression de fichiers orphelins.

Après avoir créé l'optimiseur de table, les fichiers orphelins seront supprimés périodiquement (une fois par jour si cette option reste activée). Vous pouvez vérifier les exécutions à l'aide de l'API `list-table-optimizer-runs`. La tâche de suppression des fichiers orphelins identifiera et supprimera les fichiers qui ne sont pas suivis dans les métadonnées Iceberg de la table.

API

CreateTableOptimizerOpération d'appel pour créer l'optimiseur de suppression de fichiers orphelins pour une table spécifique.

Mise à jour de l'optimiseur de suppression de fichiers orphelins

Vous pouvez modifier la configuration de l'optimiseur de suppression de fichiers orphelins, par exemple en modifiant la période de conservation des fichiers orphelins ou le rôle IAM utilisé par l'optimiseur à l'aide de la AWS Glue console ou de l'opération. AWS CLI`UpdateTableOptimizer`

AWS Management Console

Pour mettre à jour l'optimiseur de suppression de fichiers orphelins

1. Choisissez Catalogue de données, puis choisissez Tables. Dans la liste des tables, choisissez la table dont vous souhaitez mettre à jour la configuration de l'optimiseur de suppression des fichiers orphelins.
2. Dans la section inférieure de la page Détails des tables, choisissez Optimisation des tables, puis sélectionnez Modifier.
3. Sur la page Modifier l'optimisation, apportez les modifications souhaitées.
4. Choisissez Enregistrer.

AWS CLI

Vous pouvez utiliser l'`update-table-optimizer`appel pour mettre à jour l'optimiseur de suppression de fichiers orphelins dans AWS Glue, vous pouvez utiliser l'appel. Cela vous permet de modifier la `OrphanFileDeletionConfiguration` dans le champ `icebergConfiguration` dans lequel vous pouvez spécifier la `OrphanFileRetentionPeriodInDays` mise à jour pour définir le nombre de jours pendant lesquels les fichiers orphelins seront conservés, afin de spécifier l'emplacement de la table Iceberg dans laquelle les fichiers orphelins seront supprimés.

```
aws glue update-table-optimizer \
 --catalog-id 123456789012 \
 --database-name iceberg_db \
 --table-name Iceberg_table \
 --table-optimizer-configuration
 '{"roleArn":"arn:aws:iam::123456789012:role/optimizer_role","enabled":true,
 "vpcConfiguration":
{"glueConnectionName":"glue_connection_name"},"orphanFileDeletionConfiguration":
{"icebergConfiguration":{"orphanFileRetentionPeriodInDays":5}}}' \
 --type orphan_file_deletion
```

API

Appelez l'UpdateTableOptimizeropération pour mettre à jour l'optimiseur de suppression de fichiers orphelins pour une table.

Désactivation de la suppression des fichiers orphelins

Vous pouvez désactiver l'optimiseur de suppression de fichiers orphelins pour une table Apache Iceberg particulière à l'aide AWS Glue de la console ou. AWS CLI

Console

Pour désactiver la suppression des fichiers orphelins

1. Choisissez Catalogue de données, puis choisissez Tables. Dans la liste des tables, choisissez la table Iceberg dont vous souhaitez désactiver l'optimiseur pour la suppression des fichiers orphelins.
2. Dans la section inférieure de la page des Détails des tables, choisissez l'onglet Optimisation des tables.

3. Choisissez Actions, puis sélectionnez Désactiver, Suppression des fichiers orphelins.

   Vous pouvez également choisir Désactiver sous Optimisation depuis le menu Actions.

4. Choisissez Désactiver dans le message de confirmation. Vous pouvez réactiver l'optimiseur de suppression des fichiers orphelins ultérieurement.

   Une fois que vous avez confirmé, l'optimiseur de suppression des fichiers orphelins est désactivé et l'état de suppression des fichiers orphelins revient à `Not enabled`.

AWS CLI

Dans l'exemple suivant, remplacez l'ID de compte par un ID de AWS compte valide. Remplacez le nom de la base de données et le nom de la table par un nom réel de la table Iceberg et le nom de la base de données. Remplacez le `roleArn` par le nom de AWS ressource (ARN) du rôle IAM et le nom réel du rôle IAM disposant des autorisations requises pour désactiver l'optimiseur.

```
aws glue update-table-optimizer \
  --catalog-id 123456789012 \
  --database-name iceberg_db \
  --table-name iceberg_table \
  --table-optimizer-configuration
 '{"roleArn":"arn:aws:iam::123456789012:role/optimizer_role", "enabled":'false'}'\
  --type orphan_file_deletion
```

API

Appelez l'UpdateTableOptimizeropération pour désactiver l'optimiseur de rétention des instantanés pour une table spécifique.

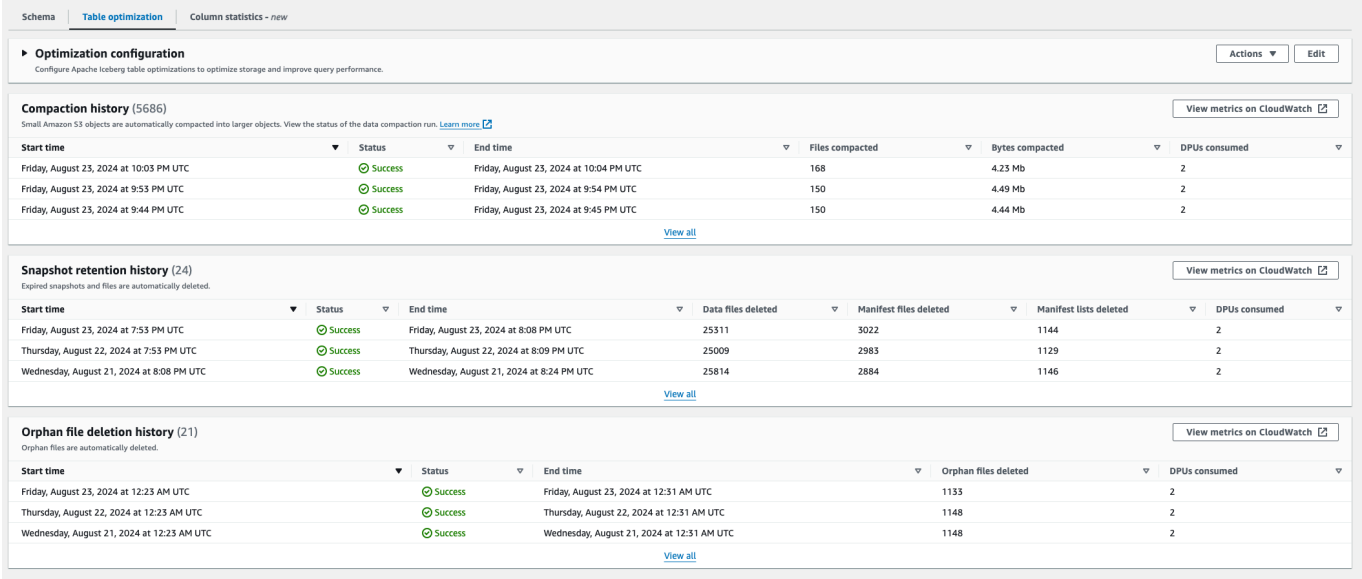
Affichage des détails de l'optimisation

Vous pouvez consulter l'état d'optimisation des tables Apache Iceberg dans la AWS Glue console ou à l'aide des opérations d' AWS API. AWS CLI

Console

Pour afficher l'état d'optimisation des tables Iceberg (console)

- Vous pouvez consulter l'état d'optimisation des tables Iceberg sur la AWS Glue console en choisissant une table Iceberg dans la liste des tables sous Catalogue de données. Sous Optimisation des tables. Choisissez l'option Afficher tout.



AWS CLI

Vous pouvez consulter les détails de l'optimisation à l'aide de AWS CLI.

Dans les exemples suivants, remplacez l'ID de compte par un ID de AWS compte valide, le nom de la base de données et le nom de la table par le nom réel de la table Iceberg. Pour `type`, indiquez un type d'optimisation. Les valeurs acceptables sont `compaction`, `retention` et `orphan-file-deletion`.

- Pour obtenir les détails de la dernière exécution du compactage d'une table

```
aws get-table-optimizer \
  --catalog-id 123456789012 \
  --database-name iceberg_db \
  --table-name iceberg_table \
  --type compaction
```

- Utilisez l'exemple suivant pour récupérer l'historique d'un optimiseur pour une table spécifique.

```
aws list-table-optimizer-runs \
  --catalog-id 123456789012 \
  --database-name iceberg_db \
  --table-name iceberg_table \
  --type compaction
```

- L'exemple suivant montre comment récupérer les détails de l'exécution de l'optimisation et de la configuration de plusieurs optimiseurs. Vous pouvez spécifier un maximum de 20 optimiseurs.

```
aws glue batch-get-table-optimizer \
--entries '[{"catalogId":"123456789012", "databaseName":"iceberg_db",
 "tableName":"iceberg_table", "type":"compaction"}]'
```

API

- Utilisez l'opération `GetTableOptimizer` pour récupérer les détails de la dernière exécution d'un optimiseur.
- Utilisez l'opération `ListTableOptimizerRuns` pour récupérer l'historique d'un optimiseur donné sur une table spécifique. Vous pouvez spécifier 20 optimiseurs en un seul appel d'API.
- Utilisez cette BatchGetTableOptimizeropération pour récupérer les détails de configuration de plusieurs optimiseurs de votre compte.

Afficher Amazon CloudWatch les métriques

Après avoir exécuté les optimisateurs de tables avec succès, le service crée des métriques Amazon CloudWatch sur les performances de la tâche d'optimisation. Vous pouvez accéder aux CloudWatch métriques et choisir Metrics, All metrics. Vous pouvez filtrer les métriques en fonction de l'espace de noms spécifique (par exemple AWS Glue), du nom de la table ou du nom de la base de données.

Pour de plus amples informations, consultez Affichage des métriques disponibles dans le Guide de l'utilisateur Amazon CloudWatch .

Compactage

- Nombre d'octets compactés
- Nombre de fichiers compactés

- Nombre de DPU allouées à la tâche
- Durée de la tâche (heures)

Conservation des instantanés

- Nombre de fichiers de données supprimés
- Nombre de fichiers manifestes supprimés
- Nombre de listes de manifestes supprimées
- Durée de la tâche (heures)

Suppression de fichiers orphelins

- Nombre de fichiers orphelins supprimés
- Durée de la tâche (heures)

Suppression d'un optimiseur

Vous pouvez supprimer un optimiseur et les métadonnées associées à la table à l'aide d' AWS CLI une opération d' AWS API.

Exécutez la AWS CLI commande suivante pour supprimer l'historique d'optimisation d'une table. Vous devez spécifier l'optimiseur `type`, ainsi que l'ID du catalogue, le nom de la base de données et le nom de la table. Les valeurs acceptables sont `compaction`, `retention` et `orphan_file_deletion`.

```
aws glue delete-table-optimizer \
  --catalog-id 123456789012 \
  --database-name iceberg_db \
  --table-name iceberg_table \
  --type compaction
```

Utilisez l'opération `DeleteTableOptimizer` pour supprimer un optimiseur pour une table.

Considérations et restrictions

Cette section inclut les éléments à prendre en compte lors de l'utilisation d'optimiseurs de tables dans AWS Glue Data Catalog.

Durabilité et exactitude

Emplacements des tables S3 :

Lorsque plusieurs AWS Glue Data Catalog tables partagent le même emplacement Amazon S3 et que les optimiseurs sont activés, l'optimiseur de conservation des instantanés ou de suppression de fichiers orphelins pour une table peut supprimer les fichiers qui sont toujours référencés par l'autre table. Assurez-vous que chaque table sur laquelle les optimiseurs sont activés possède un emplacement Amazon S3 unique qui n'est partagé avec aucune autre table, y compris les tables de différentes bases de données.

Expiration du cycle de vie S3 :

Les règles d'expiration du cycle de vie d'Amazon S3 qui s'appliquent aux emplacements de stockage des tables Iceberg peuvent supprimer le manifeste et les fichiers de données qui sont toujours référencés par des instantanés actifs. Si votre bucket comporte des règles d'expiration du cycle de vie, assurez-vous qu'elles excluent le chemin de stockage des tables Iceberg.

Formats pris en charge et restrictions pour le compactage de données géré

Le compactage des données prend en charge divers types de fichiers et formats de compression pour la lecture et l'écriture de données, y compris la lecture de données provenant de tables chiffrées.

Contrôle de la simultanéité :

Apache Iceberg prend en charge un contrôle de simultanéité optimiste, permettant à plusieurs rédacteurs d'effectuer des opérations simultanément. Les conflits sont détectés et résolus au moment de la validation. Lorsque vous travaillez avec des pipelines de streaming, configurez les paramètres de nouvelle tentative appropriés via les propriétés des tables et les paramètres de compactage afin de gérer efficacement les écritures simultanées. Pour obtenir des conseils détaillés, consultez le blog AWS Big Data sur la gestion des écritures simultanées dans les tables Iceberg.

Nouvelles tentatives de compactage :

Lorsque les opérations de compactage échouent quatre fois de suite, l'optimisation de la table de AWS Glue catalogue suspend automatiquement l'optimiseur pour éviter une consommation inutile de ressources informatiques. Examinez d'abord les journaux et essayez de comprendre pourquoi le compactage échoue à plusieurs reprises. Pour reprendre l'optimisation du compactage, vous pouvez réactiver l'optimiseur via la AWS Glue console ou l'API.

Le compactage des données prend en charge :

- Chiffrement : le compactage des données prend uniquement en charge le chiffrement Amazon S3 (SSE-S3) et le chiffrement KMS côté serveur (SSE-KMS).
- Stratégies de compactage : binpack, tri et tri par ordre Z
- Vous pouvez exécuter le compactage depuis le compte où réside le catalogue de données lorsque le compartiment Amazon S3 qui stocke les données sous-jacentes se trouve dans un autre compte. Pour ce faire, le rôle de compactage nécessite l'accès au compartiment Amazon S3.

Le compactage des données ne prend pas en charge actuellement :

- Compactage sur des tables entre comptes : vous ne pouvez pas exécuter le compactage sur des tables entre comptes.
- Compactage sur des tables entre régions : vous ne pouvez pas exécuter le compactage sur des tables entre régions.
- Activation du compactage sur des liens de ressources
- Tables de la classe de stockage Amazon S3 Express One Zone : vous ne pouvez pas exécuter le compactage sur les tables Amazon S3 Express One Zone Iceberg.
- La stratégie de compactage par ordre Z ne prend pas en charge les types de données suivants :
  - Décimal
  - TimestampWithoutZone

Considérations relatives à la conservation des instantanés et aux optimiseurs de suppression de fichiers orphelins

Les considérations suivantes s'appliquent à la conservation des instantanés et aux optimiseurs de suppression des fichiers orphelins.

- Les processus de conservation des instantanés et de suppression de fichiers orphelins ont une limite maximale de suppression de 1 000 000 fichiers par exécution. Lorsque vous supprimez des instantanés expirés, si le nombre de fichiers éligibles à la suppression dépasse 1 000 000, tous les fichiers restants au-delà de ce seuil continueront d'exister dans le stockage de la table en tant que fichiers orphelins.
- Les instantanés sont conservés par l'optimiseur de conservation d'instantanés seulement lorsque les deux critères suivants sont remplis : le nombre minimum d'instantanés à conserver et la période de conservation spécifiée.

- L'optimiseur de conservation d'instantanés supprime les métadonnées des instantanés expirés d'Apache Iceberg, empêchant ainsi les requêtes d'historique d'instantanés expirés et supprimant éventuellement les fichiers de données associés.
- L'optimiseur de suppression des fichiers orphelins supprime les fichiers de données et de métadonnées qui ne sont plus référencés par les métadonnées Iceberg si leur date de création est antérieure à la période de conservation des fichiers orphelins à compter de l'exécution de l'optimiseur.
- Apache Iceberg facilite le contrôle des versions grâce à des branches et des balises, qui sont des pointeurs nommés vers des états d'instantanés spécifiques. Chaque branche et balise suit son propre cycle de vie indépendant, régi par des politiques de conservation définies à leurs niveaux respectifs. Les AWS Glue Data Catalog optimiseurs tiennent compte de ces politiques de cycle de vie, garantissant ainsi le respect des règles de conservation spécifiées. Les politiques de conservation au niveau des branches et des balises ont priorité sur les configurations de l'optimiseur.

  Pour plus d'informations, consultez Branching and Tagging dans la documentation Apache Iceberg.
- Les optimiseurs de conservation des instantanés et de suppression de fichiers orphelins supprimeront les fichiers éligibles au nettoyage conformément aux paramètres configurés. Améliorez votre contrôle sur la suppression de fichiers en mettant en œuvre des politiques de gestion des versions et de cycle de vie S3 sur les compartiments appropriés.

  Pour obtenir des instructions détaillées sur la configuration de la gestion des versions et la création de règles de cycle de vie, consultez https://docs.aws.amazon.com/AmazonS3/latest/userguide/Versioning.html.
- Pour déterminer correctement les fichiers orphelins, assurez-vous que l'emplacement de la table fourni et les éventuels sous-chemins ne se chevauchent pas ou ne contiennent pas de données provenant d'autres tables ou sources de données. Si les chemins se chevauchent, vous risquez une perte de données irrécupérable en cas de suppression involontaire de fichiers.

Exception de débogage OversizedAllocationException

Pour résoudre une exception `OversizedAllocationException` :

- Réduisez la taille du lot du lecteur vectorisé et vérifiez. La taille du lot par défaut est de 5 000. Ceci est contrôlé dans `read.parquet.vectorization.batch-size`.
  - Si cela ne fonctionne pas même après plusieurs variations, désactivez la vectorisation. Ceci est contrôlé dans `read.parquet.vectorization.enabled`.

Régions prises en charge pour les optimiseurs de tables

Les fonctionnalités d'optimisation des tables (compactage, conservation des instantanés et suppression de fichiers orphelins) pour AWS Glue Data Catalog sont disponibles dans les versions suivantes : Régions AWS

- Asie-Pacifique (Tokyo)
- Asie-Pacifique (Séoul)
- Asie-Pacifique (Mumbai)
- Asie-Pacifique (Singapour)
- Asie-Pacifique (Sydney)
- Asie-Pacifique (Jakarta)
- Canada (Centre)
- Europe (Irlande)
- Europe (Londres)
- Europe (Francfort)
- Europe (Stockholm)
- USA Est (Virginie du Nord)
- USA Est (Ohio)
- USA Ouest (Oregon)
- Amérique du Sud (São Paulo)

Optimisation des performances des requêtes pour les tables Iceberg

Apache Iceberg est un format de table ouverte très performant pour de grands ensembles de données analytiques. AWS Glue prend en charge le calcul et la mise à jour du nombre de valeurs distinctes (NDVs) pour chaque colonne des tables Iceberg. Ces statistiques peuvent améliorer l'optimisation des requêtes, la gestion des données et l'efficacité des performances pour les ingénieurs et les scientifiques de données travaillant avec des jeux de données à grande échelle.

AWS Glue estime le nombre de valeurs distinctes dans chaque colonne de la table Iceberg et les stocke dans des fichiers Puffin sur Amazon S3 associés à des instantanés de table Iceberg. Puffin est un format de fichier Iceberg conçu pour stocker des métadonnées telles que des index, des

statistiques et des croquis. Le stockage de croquis dans des fichiers Puffin liés à des instantanés garantit la cohérence transactionnelle et l'actualité des statistiques NDV.

Vous pouvez configurer pour exécuter la tâche de génération de statistiques de colonne à l'aide de AWS Glue la console ou AWS CLI. Lorsque vous lancez le processus, AWS Glue démarre une tâche Spark en arrière-plan et met à jour les métadonnées de la AWS Glue table dans le catalogue de données. Vous pouvez consulter les statistiques des colonnes à l'aide de la AWS Glue console AWS CLI ou en appelant l'opération GetColumnStatisticsForTableAPI.

> **Note**
>
> Si vous utilisez AWS Lake Formation des autorisations pour contrôler l'accès à la table, le rôle assumé par la tâche de statistiques sur les colonnes nécessite un accès complet à la table pour générer des statistiques.

Rubriques

- Conditions préalables à la génération de statistiques de colonne
- Génération de statistiques de colonne pour les tables Iceberg
- Consultez aussi

Conditions préalables à la génération de statistiques de colonne

Pour générer ou mettre à jour des statistiques de colonne pour les tables Iceberg, la tâche de génération de statistiques assume un rôle Gestion des identités et des accès AWS (IAM) en votre nom. Sur la base des autorisations accordées au rôle, la tâche de génération de statistiques de colonne peut lire les données à partir du magasin de données Amazon S3.

Lorsque vous configurez la tâche de génération de statistiques de AWS Glue colonne, vous pouvez créer un rôle qui inclut la politique `AWSGlueServiceRole` AWS gérée ainsi que la politique en ligne requise pour la source de données spécifiée.

Si vous spécifiez un rôle existant pour la génération de statistiques de colonne, assurez-vous qu'il inclut la politique `AWSGlueServiceRole` ou l'équivalent (ou une version limitée de cette politique), ainsi que les politiques en ligne requises.

Pour plus d'informations sur les autorisations requises, consultez Conditions préalables à la génération de statistiques de colonne.

Génération de statistiques de colonne pour les tables Iceberg

Procédez comme suit pour configurer un calendrier de génération de statistiques dans le catalogue de données à l'aide de la AWS Glue console AWS CLI ou de l'opération ou exécutez l'StartColumnStatisticsTaskRunopération.

Pour générer des statistiques de colonne

1. Connectez-vous à la AWS Glue console à l'adresse https://console.aws.amazon.com/glue/.
2. Sélectionnez Tables sous Catalogue de données.
3. Choisissez une table Iceberg dans la liste.
4. Choisissez Statistiques de colonne, Générer à la demande dans le menu Actions.

   Vous pouvez également choisir le bouton Générer des statistiques sous l'onglet Statistiques de colonne dans la section inférieure de la page Tables.
5. Sur la page Générer des statistiques, fournissez les détails relatifs à la génération des statistiques. Suivez les étapes 6 à 11 de la section Génération de statistiques de colonne selon un calendrier pour configurer un calendrier de génération de statistiques pour les tables Iceberg.

   Vous pouvez également choisir de générer des statistiques de colonnes à la demande en suivant les instructions de Génération de statistiques de colonne à la demande.

   > **Note**
   >
   > L'option d'échantillonnage n'est pas disponible pour les tables Iceberg.

   AWS Glue calcule le nombre de valeurs distinctes pour chaque colonne de la table Iceberg dans un nouveau fichier Puffin validé avec l'ID de snapshot spécifié dans votre emplacement Amazon S3.

Consultez aussi

- Affichage des statistiques de colonne
- Affichage des exécutions de tâches de statistiques de colonne
- Arrêt d'exécution de la tâche de statistiques de colonne
- Supprimer les statistiques de colonne

Gestion du catalogue de données

AWS Glue Data Catalog Il s'agit d'un référentiel de métadonnées central qui stocke les métadonnées structurelles et opérationnelles de vos ensembles de données Amazon S3. La gestion efficace du catalogue de données est essentielle au maintien de la qualité, des performances, de la sécurité et de la gouvernance des données.

En comprenant et en appliquant ces pratiques de gestion du catalogue de données, vous pouvez vous assurer que vos métadonnées restent précises, performantes, sécurisées et bien gouvernées au fil de l'évolution de votre environnement de données.

Cette section couvre les aspects suivants de la gestion du catalogue de données :

- Mise à jour du schéma de table et des partitions : à mesure de l'évolution de vos données, vous devrez peut-être mettre à jour le schéma de table ou la structure de partition définis dans le catalogue de données. Pour plus d'informations sur la façon d'effectuer ces mises à jour par programmation à l'aide de l' AWS Glue ETL, consultez. Mise à jour du schéma et ajout de nouvelles partitions dans le catalogue de données à l'aide des tâches AWS Glue ETL
- Gestion des statistiques de colonne : des statistiques précises sur les colonnes permettent d'optimiser les plans de requêtes et d'améliorer les performances. Pour plus d'informations sur la façon de générer, de mettre à jour et de gérer les statistiques de colonne, consultez Optimisation des performances des requêtes à l'aide des statistiques de colonne.
- Chiffrement du catalogue de données Pour protéger les métadonnées sensibles, vous pouvez chiffrer votre catalogue de données à l'aide de AWS Key Management Service ()AWS KMS. Cette section explique comment activer et gérer le chiffrement de votre catalogue de données.
- Sécurisation du catalogue de données avec AWS Lake Formation : Lake Formation fournit une approche globale de la sécurité des lacs de données et du contrôle d'accès. Vous pouvez utiliser Lake Formation pour sécuriser et régir l'accès à votre catalogue de données et aux données sous-jacentes.

Rubriques

- Mise à jour du schéma et ajout de nouvelles partitions dans le catalogue de données à l'aide des tâches AWS Glue ETL
- Optimisation des performances des requêtes à l'aide des statistiques de colonne
- Chiffrement de votre catalogue de données
- Sécurisation de votre catalogue de données à l'aide de Lake Formation

- Utilisation des AWS Glue Data Catalog vues dans AWS Glue

Mise à jour du schéma et ajout de nouvelles partitions dans le catalogue de données à l'aide des tâches AWS Glue ETL

Votre travail d'extraction, de transformation et de chargement (ETL) peut créer de nouvelles partitions de table dans le magasin de données cible. Votre schéma de jeu de données peut évoluer et diverger du schéma de catalogue de données AWS Glue avec le temps. AWS Glue Les tâches ETL fournissent désormais plusieurs fonctions que vous pouvez utiliser dans votre script ETL pour mettre à jour votre schéma et vos partitions dans le catalogue de données. Ces fonctions vous permettent de voir les résultats de votre tâche ETL dans le catalogue de données, sans avoir à exécuter à nouveau l'crawler.

Nouvelles partitions

Si vous souhaitez afficher les nouvelles partitions dans le AWS Glue Data Catalog, vous pouvez effectuer l'une des opérations suivantes :

- Une fois la tâche terminée, réexécutez l'crawler pour afficher les nouvelles partitions sur la console.
- Une fois la tâche terminée, affichez immédiatement les nouvelles partitions sur la console, sans avoir à réexécuter l'crawler. Pour activer cette fonctionnalité, ajoutez quelques lignes de code au script ETL, comme illustré dans les exemples suivants. Le code utilise l'argument `enableUpdateCatalog` pour indiquer que le catalogue de données doit être mis à jour pendant l'exécution de la tâche lors de la création des partitions.

Méthode 1 :

Transmettez `enableUpdateCatalog` et `partitionKeys` dans un argument d'options.

Python

```
additionalOptions = {"enableUpdateCatalog": True}
additionalOptions["partitionKeys"] = ["region", "year", "month", "day"]


sink = glueContext.write_dynamic_frame_from_catalog(frame=last_transform,
 database=<target_db_name>,

 table_name=<target_table_name>, transformation_ctx="write_sink",
```

```
 additional_options=additionalOptions)
```

Scala

```
val options = JsonOptions(Map(
    "path" -> <S3_output_path>,
    "partitionKeys" -> Seq("region", "year", "month", "day"),
    "enableUpdateCatalog" -> true))
val sink = glueContext.getCatalogSink(
    database = <target_db_name>,
    tableName = <target_table_name>,
    additionalOptions = options)sink.writeDynamicFrame(df)
```

Méthode 2 :

Transmettez `enableUpdateCatalog` et `partitionKeys` dans `getSink( )` et appelez `setCatalogInfo( )` au niveau de l'objet `DataSink`.

Python

```
sink = glueContext.getSink(
    connection_type="s3",
    path="<S3_output_path>",
    enableUpdateCatalog=True,
    partitionKeys=["region", "year", "month", "day"])
sink.setFormat("json")
sink.setCatalogInfo(catalogDatabase=<target_db_name>,
 catalogTableName=<target_table_name>)
sink.writeFrame(last_transform)
```

Scala

```
val options = JsonOptions(
   Map("path" -> <S3_output_path>,
       "partitionKeys" -> Seq("region", "year", "month", "day"),
       "enableUpdateCatalog" -> true))
val sink = glueContext.getSink("s3", options).withFormat("json")
sink.setCatalogInfo(<target_db_name>, <target_table_name>)
sink.writeDynamicFrame(df)
```

Maintenant, vous pouvez créer des tables de catalogue, mettre à jour les tables existantes avec le schéma modifié et ajouter de nouvelles partitions de table au catalogue de données à l'aide d'une tâche ETL AWS Glue, sans avoir besoin de relancer les crawlers.

Mise à jour du schéma de table

Si vous souhaitez remplacer le schéma de la table du catalogue de données, vous pouvez effectuer l'une des opérations suivantes :

- Une fois la tâche terminée, exécutez à nouveau l'crawler et assurez-vous que celui-ci est configuré pour mettre à jour également la définition de la table. Affichez les nouvelles partitions sur la console ainsi que les mises à jour de schéma lorsque l'crawler se termine. Pour plus d'informations, consultez Configuration d'un crawler utilisant l'API.
- Une fois la tâche terminée, affichez immédiatement le schéma modifié sur la console, sans avoir à exécuter de nouveau l'crawler. Pour activer cette fonctionnalité, ajoutez quelques lignes de code au script ETL, comme illustré dans les exemples suivants. Le code utilise `enableUpdateCatalog` défini sur true et `updateBehavior` défini sur `UPDATE_IN_DATABASE`, ce qui indique que le schéma doit être remplacé et que de nouvelles partitions doivent être ajoutées au catalogue de données pendant l'exécution de la tâche.

Python

```
additionalOptions = {
    "enableUpdateCatalog": True,
    "updateBehavior": "UPDATE_IN_DATABASE"}
additionalOptions["partitionKeys"] = ["partition_key0", "partition_key1"]

sink = glueContext.write_dynamic_frame_from_catalog(frame=last_transform,
 database=<dst_db_name>,
    table_name=<dst_tbl_name>, transformation_ctx="write_sink",
    additional_options=additionalOptions)
job.commit()
```

Scala

```
val options = JsonOptions(Map(
    "path" -> outputPath,
    "partitionKeys" -> Seq("partition_0", "partition_1"),
    "enableUpdateCatalog" -> true))
```

```
val sink = glueContext.getCatalogSink(database = nameSpace, tableName = tableName,
 additionalOptions = options)
sink.writeDynamicFrame(df)
```

Vous pouvez également définir la valeur de `updateBehavior` sur `LOG` si vous souhaitez empêcher le remplacement du schéma de table, mais que vous souhaitez toujours ajouter les nouvelles partitions. Comme la valeur par défaut de `updateBehavior` est `UPDATE_IN_DATABASE`, si vous ne le définissez pas explicitement, le schéma de table est remplacé.

Si `enableUpdateCatalog` n'a pas la valeur true, quelle que soit l'option sélectionnée pour `updateBehavior`, la tâche ETL ne mettra pas à jour la table dans le catalogue de données.

Création de tables

Vous pouvez également utiliser les mêmes options pour créer une table dans le catalogue de données. Vous pouvez spécifier la base de données et le nouveau nom de table avec `setCatalogInfo`.

Python

```
sink = glueContext.getSink(connection_type="s3", path="s3://path/to/data",
    enableUpdateCatalog=True, updateBehavior="UPDATE_IN_DATABASE",
    partitionKeys=["partition_key0", "partition_key1"])
sink.setFormat("<format>")
sink.setCatalogInfo(catalogDatabase=<dst_db_name>, catalogTableName=<dst_tbl_name>)
sink.writeFrame(last_transform)
```

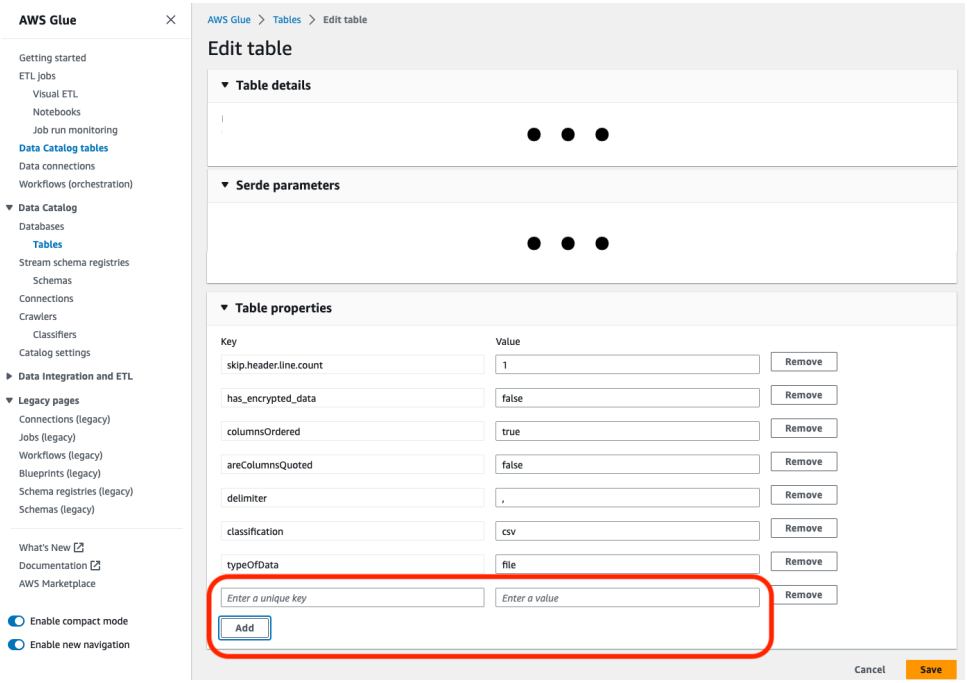
Scala

```
val options = JsonOptions(Map(
    "path" -> outputPath,
    "partitionKeys" -> Seq("<partition_1>", "<partition_2>"),
    "enableUpdateCatalog" -> true,
    "updateBehavior" -> "UPDATE_IN_DATABASE"))
val sink = glueContext.getSink(connectionType = "s3", connectionOptions =
 options).withFormat("<format>")
sink.setCatalogInfo(catalogDatabase = "<dst_db_name>", catalogTableName =
 "<dst_tbl_name>")
sink.writeDynamicFrame(df)
```

Restrictions

Prenez note des restrictions suivantes :

- Seules des cibles Amazon Simple Storage Service (Amazon S3) sont prises en charge.
- La fonctionnalité `enableUpdateCatalog` n'est pas prise en charge pour les tables gouvernées.
- Seuls les formats suivants sont pris en charge : `json`, `csv`, `avro` et `parquet`.
- Pour créer ou mettre à jour des tableaux avec la `parquet` classification, vous devez utiliser le rédacteur de parquet AWS Glue optimisé pour DynamicFrames. Cela peut être réalisé à l'aide d'un des moyens suivants :
  - Si vous mettez à jour une table existante dans le catalogue avec la classification `parquet`, la propriété de la table `"useGlueParquetWriter"` doit être définie sur `true` avant que vous ne la mettiez à jour. Vous pouvez définir cette propriété via le AWS Glue APIs /SDK, via la console ou via une instruction Athena DDL.



Une fois la propriété de la table de catalogue définie, vous pouvez utiliser l'extrait de code suivant pour mettre à jour la table de catalogue avec les nouvelles données :

```
glueContext.write_dynamic_frame.from_catalog(
    frame=frameToWrite,
    database="dbName",
    table_name="tableName",
    additional_options={
```

```
        "enableUpdateCatalog": True,
        "updateBehavior": "UPDATE_IN_DATABASE"
    }
)
```

- Si la table n'existe pas déjà dans le catalogue, vous pouvez utiliser la méthode `getSink()` de votre script avec `connection_type="s3"` pour ajouter la table et ses partitions au catalogue, ainsi que pour écrire les données sur Amazon S3. Fournissez les `partitionKeys` et la `compression` appropriés pour votre flux de travail.

```
s3sink = glueContext.getSink(
    path="s3://bucket/folder/",
    connection_type="s3",
    updateBehavior="UPDATE_IN_DATABASE",
    partitionKeys=[],
    compression="snappy",
    enableUpdateCatalog=True
)

s3sink.setCatalogInfo(
    catalogDatabase="dbName", catalogTableName="tableName"
)

s3sink.setFormat("parquet", useGlueParquetWriter=True)
s3sink.writeFrame(frameToWrite)
```

  - La valeur de `glueparquet` format est une ancienne méthode permettant d'activer le rédacteur de AWS Glue parquet.

- Quand `updateBehavior` a la valeur `LOG`, de nouvelles partitions sont ajoutées uniquement si le schéma `DynamicFrame` est équivalent ou qu'il contient un sous-ensemble des colonnes définies dans le schéma de la table du catalogue de données.
- Les mises à jour du schéma ne sont pas prises en charge pour les tables non partitionnées (qui n'utilisent pas l'option « partitionKeys »).
- Vos clés de partition (partitionKeys) doivent être équivalentes, et dans le même ordre, entre votre paramètre passé dans votre script ETL et les clés de partition (partitionKeys) du schéma de votre table du catalogue de données.
- Cette fonctionnalité ne prend actuellement pas en charge updating/creating les tables dans lesquelles les schémas de mise à jour sont imbriqués (par exemple, les tableaux à l'intérieur des structures).

Pour de plus amples informations, veuillez consulter the section called "AWS Glue pour Spark".

Utilisation des connexions MongoDB dans les tâches ETL

Vous pouvez créer une connexion pour MongoDB, puis utiliser cette connexion dans votre tâche AWS Glue. Pour plus d'informations, consultez the section called "Connexions MongoDB" le guide AWS Glue de programmation. Les connexions `url`, `username` et `password` sont stockées dans la connexion MongoDB. D'autres options peuvent être spécifiées dans votre script de tâche ETL à l'aide du paramètre `additionalOptions` de `glueContext.getCatalogSource`. Les autres options peuvent inclure :

- `database`: (Obligatoire) La base de données MongoDB à lire.
- `collection` : (Obligatoire) La collection MongoDB à lire.

En plaçant les informations `database` et `collection` dans le script de tâche ETL, vous pouvez utiliser la même connexion pour plusieurs tâches.

1. Créez une AWS Glue Data Catalog connexion pour la source de données MongoDB. Consultez "connectionType": "mongodb" pour obtenir une description des paramètres de connexion. Vous pouvez créer la connexion à l'aide de la console APIs ou de la CLI.
2. Créez une base de données dans le AWS Glue Data Catalog pour stocker les définitions de table pour vos données MongoDB. Pour plus d'informations, consultez Création de bases de données.
3. Créez un crawler qui explore les données dans MongoDB en utilisant les informations de la connexion pour se connecter à MongoDB. Le robot crée les tables AWS Glue Data Catalog qui décrivent les tables de la base de données MongoDB que vous utilisez dans le cadre de votre travail. Pour plus d'informations, consultez Utilisation de robots pour alimenter le catalogue de données.
4. Créez une tâche avec un script personnalisé. Vous pouvez créer la tâche à l'aide de la console APIs ou de la CLI. Pour plus d'informations, consultez Ajout de tâches dans AWS Glue.
5. Choisissez les cibles de données de votre tâche. Les tables qui représentent les données cible peuvent être définies dans votre catalogue de données ou votre tâche peut créer les tables cible lors son exécution. Vous choisissez un emplacement cible lorsque vous créez la tâche. Si la cible nécessite une connexion, la connexion est également référencée dans votre tâche. Si votre tâche nécessite plusieurs cibles de données, vous pouvez les ajouter ensuite en modifiant le script.

6. Personnalisez l'environnement de traitement de tâche en fournissant des arguments pour votre tâche et le script généré.

   Voici un exemple de création de `DynamicFrame` à partir de la base de données MongoDB basée sur la structure de table définie dans le catalogue de données. Le code utilise `additionalOptions` pour fournir les informations supplémentaires sur la source de données :

   Scala

   ```
   val resultFrame: DynamicFrame = glueContext.getCatalogSource(
           database = catalogDB,
           tableName = catalogTable,
           additionalOptions = JsonOptions(Map("database" -> DATABASE_NAME,
                   "collection" -> COLLECTION_NAME))
         ).getDynamicFrame()
   ```

   Python

   ```
   glue_context.create_dynamic_frame_from_catalog(
           database = catalogDB,
           table_name = catalogTable,
           additional_options = {"database":"database_name",
               "collection":"collection_name"})
   ```

7. Exécutez la tâche, à la demande ou via un déclencheur..

Optimisation des performances des requêtes à l'aide des statistiques de colonne

Vous pouvez calculer des statistiques au niveau des colonnes pour AWS Glue Data Catalog des tables dans des formats de données tels que Parquet, ORC, JSON, ION, CSV et XML sans configurer de pipelines de données supplémentaires. Les statistiques de colonne vous aident à comprendre les profils de données en obtenant des informations sur les valeurs d'une colonne.

Le catalogue de données prend en charge la génération de statistiques pour les valeurs de colonne telles que la valeur minimale, la valeur maximale, le total des valeurs nulles, le total des valeurs distinctes, la longueur moyenne des valeurs et le nombre total d'occurrences de valeurs vraies. AWS des services analytiques tels qu'Amazon Redshift et Amazon Athena peuvent utiliser ces statistiques

de colonne pour générer des plans d'exécution des requêtes et choisir le plan optimal qui améliore les performances des requêtes.

Il existe trois scénarios pour générer des statistiques de colonne :

Automatique

AWS Glue prend en charge la génération automatique de statistiques sur les colonnes au niveau du catalogue afin de pouvoir générer automatiquement des statistiques pour les nouvelles tables du. AWS Glue Data Catalog

Programmé

AWS Glue prend en charge la génération de statistiques de colonne de planification afin qu'elle puisse être exécutée automatiquement selon un calendrier récurrent.

Avec le calcul planifié des statistiques, la tâche de statistiques sur les colonnes met à jour les statistiques globales au niveau de la table, telles que min, max et avg, avec les nouvelles statistiques, fournissant aux moteurs de requêtes des statistiques précises et up-to-date permettant d'optimiser l'exécution des requêtes.

À la demande

Utilisez cette option pour générer des statistiques de colonnes à la demande chaque fois que cela s'avère nécessaire. Cela est utile pour les analyses ad hoc ou lorsque les statistiques doivent être calculées immédiatement.

Vous pouvez configurer pour exécuter une tâche de génération de statistiques de colonne à l'aide des AWS CLI opérations de AWS Glue console et AWS Glue d'API. Lorsque vous lancez le processus, AWS Glue démarre une tâche Spark en arrière-plan et met à jour les métadonnées de la AWS Glue table dans le catalogue de données. Vous pouvez consulter les statistiques des colonnes à l'aide de la AWS Glue console AWS CLI ou en appelant l'opération GetColumnStatisticsForTableAPI.

> **Note**
>
> Si vous utilisez les autorisations de Lake Formation pour contrôler l'accès à la table, le rôle assumé par la tâche de statistiques de colonne nécessite un accès complet à la table pour générer des statistiques.

Rubriques

- Conditions préalables à la génération de statistiques de colonne
- Génération automatique de statistiques de colonne
- Génération de statistiques de colonne selon un calendrier
- Génération de statistiques de colonne à la demande
- Affichage des statistiques de colonne
- Affichage des exécutions de tâches de statistiques de colonne
- Arrêt d'exécution de la tâche de statistiques de colonne
- Supprimer les statistiques de colonne
- Considérations et restrictions

Conditions préalables à la génération de statistiques de colonne

Pour générer ou mettre à jour les statistiques de colonne, la tâche de génération de statistiques assume un rôle (IAM) Gestion des identités et des accès AWS en votre nom. Sur la base des autorisations accordées au rôle, la tâche de génération de statistiques de colonne peut lire les données à partir du magasin de données Amazon S3.

Lorsque vous configurez la tâche de génération de statistiques de AWS Glue colonne, vous pouvez créer un rôle qui inclut la politique `AWSGlueServiceRole` AWS gérée ainsi que la politique en ligne requise pour la source de données spécifiée.

Si vous spécifiez un rôle existant pour la génération de statistiques de colonne, assurez-vous qu'il inclut la politique `AWSGlueServiceRole` ou un équivalent (ou une version limitée de cette politique), en plus des politiques en ligne requises. Pour créer un rôle IAM, procédez comme suit :

> **Note**
>
> Pour générer des statistiques pour les tables gérées par Lake Formation, le rôle IAM utilisé pour générer des statistiques nécessite un accès complet à la table.

Lorsque vous configurez la tâche de génération de statistiques de AWS Glue colonne, vous pouvez créer un rôle qui inclut la politique `AWSGlueServiceRole` AWS gérée ainsi que la politique en ligne requise pour la source de données spécifiée. Vous pouvez également créer un rôle et associer les

autorisations répertoriées dans la politique ci-dessous, et ajouter ce rôle à la tâche de génération de statistiques de colonne.

Pour créer un rôle IAM pour générer des statistiques de colonne

1. Pour créer un rôle IAM, consultez Création d'un rôle IAM pour AWS Glue.
2. Pour mettre à jour un rôle existant, dans la console IAM, accédez au rôle IAM utilisé par le processus de génération de statistiques de colonne.
3. Dans la section Ajouter des autorisations, choisissez Attacher des stratégies. Dans la fenêtre du navigateur qui vient d'être ouverte, choisissez la politique `AWSGlueServiceRole` AWS gérée.
4. Vous devez également inclure les autorisations pour lire les données à partir de l'emplacement de données Amazon S3.

   Dans la section Ajouter des autorisations, choisissez Créer une stratégie. Dans la fenêtre du navigateur nouvellement ouverte, créez une nouvelle stratégie à utiliser avec votre rôle.
5. Sur la page Créer une stratégie, choisissez l'onglet JSON. Copiez le code `JSON` suivant dans le champ de l'éditeur de stratégie.

> **Note**
>
> Dans les politiques suivantes, remplacez l'ID de compte par un identifiant valide Compte AWS, puis `region` remplacez-le par la région du tableau et `bucket-name` par le nom du compartiment Amazon S3.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "S3BucketAccess",
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket",
                "s3:GetObject"
            ],
            "Resource": [
```

```
            "arn:aws:s3:::amzn-s3-demo-bucket/*",
        "arn:aws:s3:::amzn-s3-demo-bucket"
            ]
        }
    ]
}
```

6. (Facultatif) Si vous utilisez les autorisations Lake Formation pour accéder à vos données, le rôle IAM nécessite des autorisations `lakeformation:GetDataAccess`.

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "LakeFormationDataAccess",
      "Effect": "Allow",
      "Action": "lakeformation:GetDataAccess",
      "Resource": [
        "*"
      ]
    }
  ]
}
```

   Si l'emplacement des données Amazon S3 est enregistré auprès de Lake Formation et que le rôle IAM assumé par la tâche de génération de statistiques de colonne ne dispose pas d'autorisations de groupe `IAM_ALLOWED_PRINCIPALS` accordées sur la table, le rôle nécessite Lake Formation `ALTER` et des autorisations `DESCRIBE` sur la table. Le rôle utilisé pour enregistrer le compartiment Amazon S3 nécessite Lake Formation `INSERT` et des autorisations `DELETE` sur la table.

   Si l'emplacement des données Amazon S3 n'est pas enregistré auprès de Lake Formation et que le rôle IAM ne dispose pas d'autorisations de groupe `IAM_ALLOWED_PRINCIPALS` accordées sur la table, le rôle nécessite les autorisations Lake Formation `ALTER`, `DESCRIBE`, `INSERT` et `DELETE` sur la table.

7. Si vous avez activé l'option `Automatic statistics generation` au niveau du catalogue, le rôle IAM doit disposer de l'autorisation `glue:UpdateCatalog` ou de l'autorisation `ALTER`

`CATALOG` Lake Formation sur le catalogue de données par défaut. Vous pouvez utiliser l'opération `GetCatalog` pour vérifier les propriétés du catalogue.

8. (Facultatif) La tâche de génération de statistiques de colonne qui écrit Amazon CloudWatch Logs chiffré nécessite les autorisations suivantes dans la stratégie de clé.

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "CWLogsKmsPermissions",
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents",
        "logs:AssociateKmsKey"
      ],
      "Resource": [
        "arn:aws:logs:us-east-1:111122223333:log-group:/aws-glue:*"
      ]
    },
    {
      "Sid": "KmsPermissions",
      "Effect": "Allow",
      "Action": [
        "kms:GenerateDataKey",
        "kms:Decrypt",
        "kms:Encrypt"
      ],
      "Resource": [
        "arn:aws:kms:us-east-1:111122223333:key/arn of key used for ETL
 cloudwatch encryption"
      ],
      "Condition": {
        "StringEquals": {
          "kms:ViaService": [
            "glue.us-east-1.amazonaws.com"
          ]
        }
      }
```

```
    }
  ]
}
```

9. Le rôle que vous utilisez pour exécuter les statistiques de colonne doit disposer de l'autorisation `iam:PassRole` correspondante.

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iam:PassRole"
      ],
      "Resource": [
        "arn:aws:iam::111122223333:role/columnstats-role-name"
      ]
    }
  ]
}
```

10. Lorsque vous créez un rôle IAM pour générer des statistiques de colonne, ce rôle doit également avoir la stratégie d'approbation suivante qui permet au service d'assumer le rôle.

    JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "TrustPolicy",
      "Effect": "Allow",
      "Principal": {
        "Service": "glue.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
```

```
}
```

Génération automatique de statistiques de colonne

La génération automatique de statistiques de colonne vous permet de planifier et de calculer automatiquement les statistiques sur les nouvelles tables de l' AWS Glue Data Catalog. Lorsque vous activez la génération automatique de statistiques, le catalogue de données découvre de nouvelles tables avec des formats de données spécifiques tels que Parquet, JSON, CSV, XML, ORC, ION et Apache Iceberg, ainsi que leurs chemins d'accès aux compartiments individuels. Avec une configuration de catalogue unique, le catalogue de données génère des statistiques pour ces tables.

Les administrateurs de lac de données peuvent configurer la génération de statistiques en sélectionnant le catalogue par défaut dans la console Lake Formation et en activant les statistiques des tables à l'aide de l'option `Optimization configuration`. Lorsque vous créez de nouvelles tables ou que vous mettez à jour des tables existantes dans le catalogue de données, le catalogue de données collecte le nombre de valeurs distinctes (NDVs) pour les tables Apache Iceberg, ainsi que des statistiques supplémentaires telles que le nombre de valeurs nulles, la longueur maximale, minimale et moyenne pour les autres formats de fichiers pris en charge sur une base hebdomadaire.

Si vous avez configuré la génération de statistiques au niveau de la table ou si vous avez précédemment supprimé les paramètres de génération de statistiques d'une table, ces paramètres propres à la table ont priorité sur les paramètres de catalogue par défaut pour la génération automatique de statistiques de colonne.

La tâche de génération automatique de statistiques analyse 50 % des enregistrements des tables pour calculer les statistiques. La génération automatique de statistiques de colonne garantit que le catalogue de données conserve des métriques hebdomadaires qui peuvent être utilisées par les moteurs de requêtes tels qu'Amazon Athena et Amazon Redshift Spectrum pour améliorer les performances des requêtes et réaliser des économies potentielles. Il permet de planifier la génération de statistiques à l'aide AWS Glue APIs de la console, fournissant ainsi un processus automatisé sans intervention manuelle.

Rubriques

- Activation de la génération automatique de statistiques au niveau du catalogue
- Affichage des paramètres au niveau de la table automatisés
- Désactivation de la génération de statistiques de colonne au niveau du catalogue

Activation de la génération automatique de statistiques au niveau du catalogue

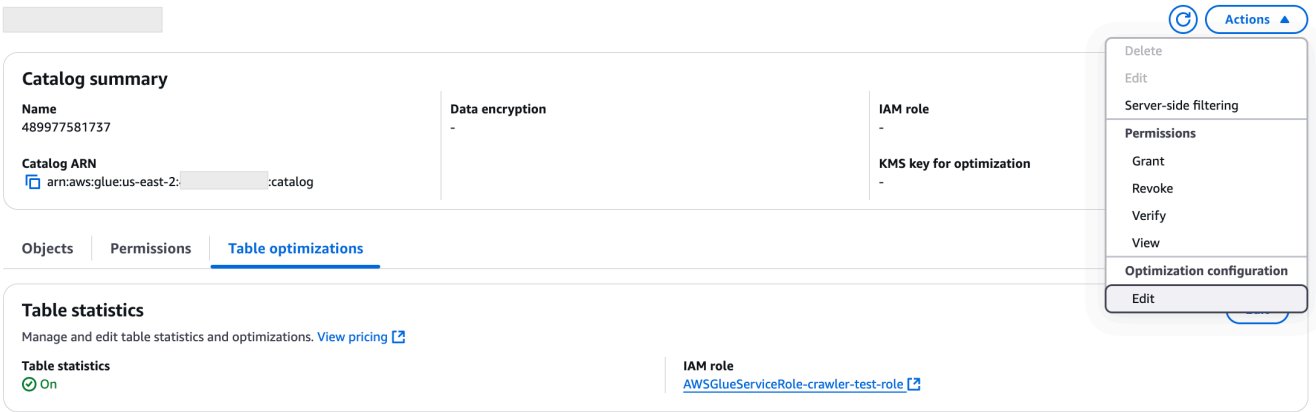
Vous pouvez activer la génération automatique de statistiques de colonne pour toutes les nouvelles tables Apache Iceberg et les tables dans des formats de table autres que OTF (Parquet, JSON, CSV, XML, ORC, ION) dans le catalogue de données. Après avoir créé la table, vous pouvez également mettre à jour manuellement de manière explicite les paramètres des statistiques de colonne.

Pour mettre à jour les paramètres du catalogue de données afin de les activer au niveau du catalogue, le rôle IAM utilisé doit disposer de l'`glue:UpdateCatalog`autorisation ou de l' AWS Lake Formation `ALTER CATALOG`autorisation sur le catalogue racine. Vous pouvez utiliser l'API `GetCatalog` pour vérifier les propriétés du catalogue.

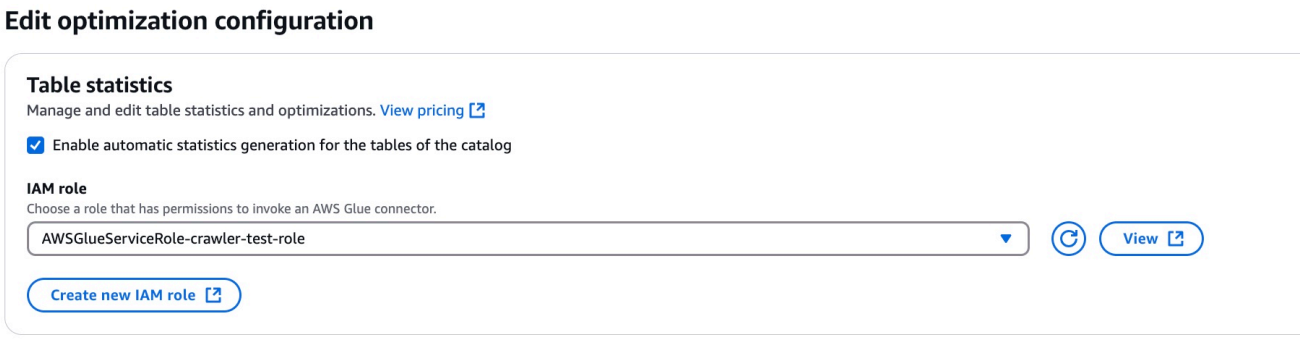
AWS Management Console

Pour activer la génération automatique de statistiques de colonne au niveau du compte

1. Ouvrez la console Lake Formation à l'adresse https://console.aws.amazon.com/lakeformation/.
2. Dans la barre de navigation de gauche, choisissez Catalogues.
3. Sur la page Résumé du catalogue, choisissez Modifier sous Configuration de l'optimisation.



4. Sur la page de Configuration de l'optimisation des tables, choisissez l'option Activer la génération automatique de statistiques pour les tables du catalogue.



5. Choisissez un rôle IAM existant ou créez-en un autre avec les autorisations nécessaires pour exécuter la tâche de statistiques de colonne.
6. Sélectionnez Soumettre.

AWS CLI

Vous pouvez également activer la collecte de statistiques au niveau du catalogue via l' AWS CLI. Pour configurer la collecte de statistiques au niveau des tables à l'aide de AWS CLI, exécutez la commande suivante :

```
aws glue update-catalog --cli-input-json '{
    "name": "123456789012",
    "catalogInput": {
        "description": "Updating root catalog with role arn",
        "catalogProperties": {
            "customProperties": {
                "ColumnStatistics.RoleArn": "arn:aws:iam::"123456789012":role/
service-role/AWSGlueServiceRole",
                "ColumnStatistics.Enabled": "true"
            }
        }
    }
}'
```

La commande ci-dessus appelle AWS Glue l'`UpdateCatalog`opération, qui prend en compte une `CatalogProperties` structure avec les paires clé-valeur suivantes pour la génération de statistiques au niveau du catalogue :

- ColumnStatistics. RoleArn — L'ARN du rôle IAM doit être utilisé pour toutes les tâches déclenchées pour la génération de statistiques au niveau du catalogue
- ColumnStatistics.Enabled — Booléen indiquant si les paramètres au niveau du catalogue sont activés ou désactivés

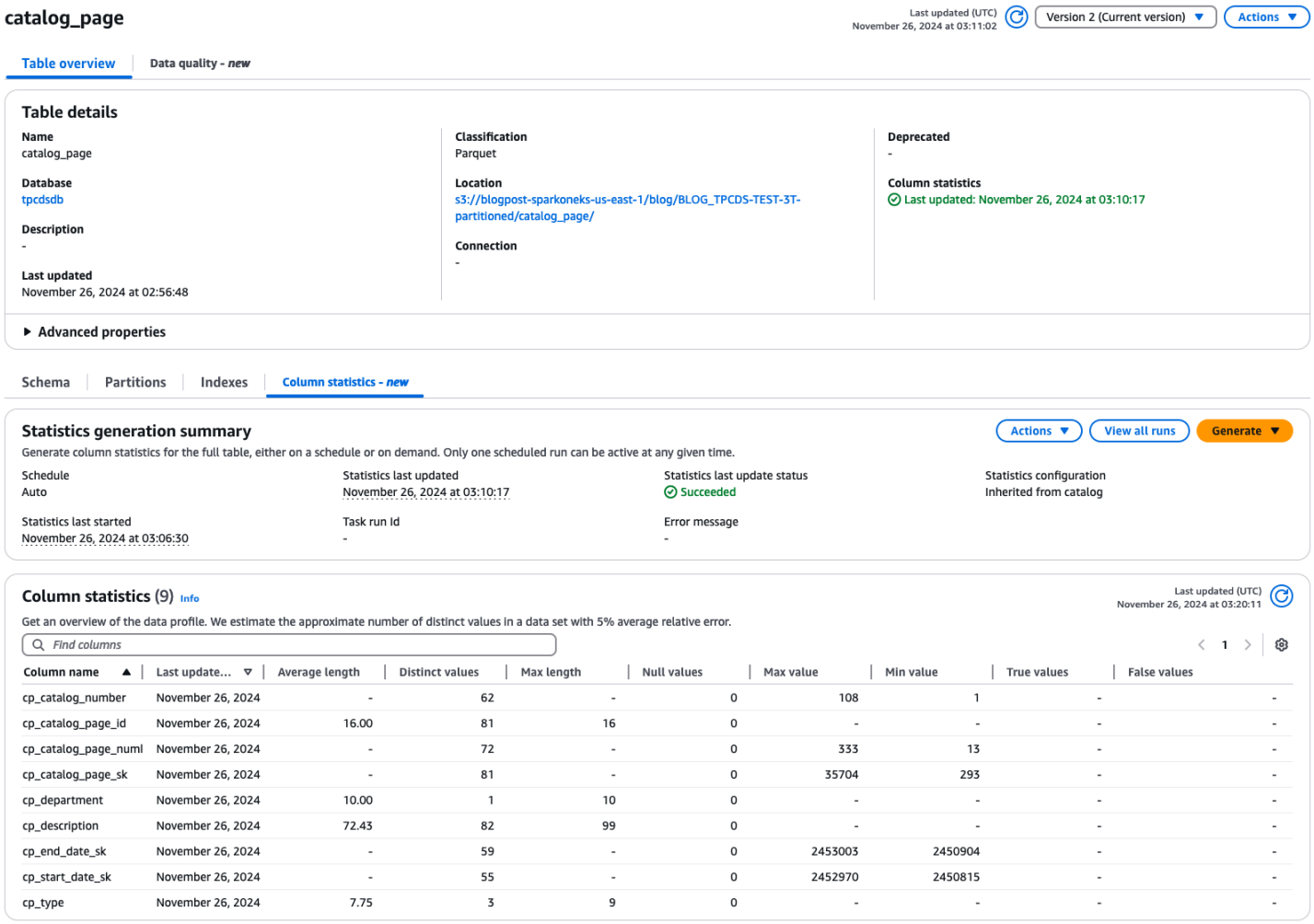
Affichage des paramètres au niveau de la table automatisés

Lorsque la collecte de statistiques au niveau du catalogue est activée, chaque fois qu'une table Apache Hive ou Apache Iceberg est créée ou mise à jour via le SDK `CreateTable` ou `UpdateTable` APIs qu'un paramètre de niveau de table équivalent est créé pour cette table. AWS Management Console AWS Glue crawler

Les tables pour lesquelles la génération automatique de statistiques est activée doivent respecter l'une des propriétés suivantes :

- Utiliser une `InputSerdeLibrary` qui commence par org.apache.hadoop et `TableType` égal à `EXTERNAL_TABLE`
- Utiliser une `InputSerdeLibrary` qui commence par `com.amazon.ion` et `TableType` égal à `EXTERNAL_TABLE`
- Contenir table_type : « ICEBERG » dans sa structure de paramètres

Après avoir créé ou mis à jour une table, vous pouvez vérifier ses détails pour confirmer la génération de statistiques. `Statistics generation summary` affiche la propriété `Schedule` définie comme `AUTO` et la valeur `Statistics configuration` est `Inherited from catalog`. Tout paramètre de table avec le paramètre suivant serait automatiquement déclenché par Glue en interne.



Désactivation de la génération de statistiques de colonne au niveau du catalogue

Vous pouvez désactiver la génération automatique de statistiques de colonne pour les nouvelles tables à l'aide de la AWS Lake Formation console, de l'`glue:UpdateCatalogSettings`API ou de l'`glue:DeleteColumnStatisticsTaskSettings`API.

Pour désactiver la génération automatique de statistiques de colonne au niveau du compte

1. Ouvrez la console Lake Formation à l'adresse https://console.aws.amazon.com/lakeformation/.
2. Dans la barre de navigation de gauche, choisissez Catalogues.
3. Sur la page Résumé du catalogue, choisissez Modifier sous Configuration de l'optimisation.
4. Sur la page de Configuration de l'optimisation des tables, désélectionnez l'option Activer la génération automatique de statistiques pour les tables du catalogue.
5. Sélectionnez Soumettre.

Génération de statistiques de colonne selon un calendrier

Suivez ces étapes pour configurer un calendrier de génération de statistiques de colonne à l' AWS Glue Data Catalog aide de la AWS Glue console, de l' AWS CLI opération ou de l'CreateColumnStatisticsTaskSettingsopération.

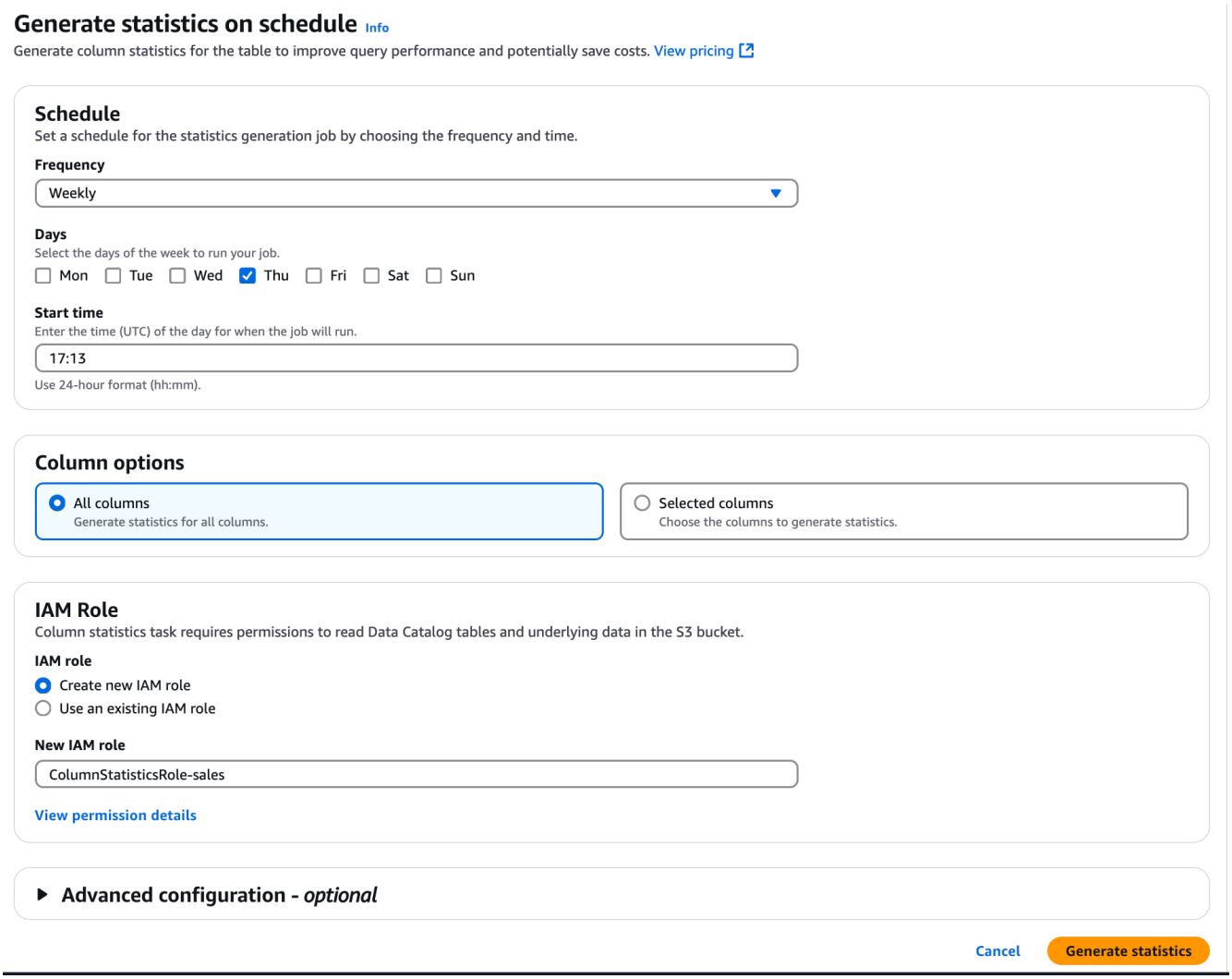
Console

Pour générer des statistiques de colonne à l'aide de la console

1. Connectez-vous à la AWS Glue console à l'adresse https://console.aws.amazon.com/glue/.
2. Choisissez des tables du catalogue de donnée.
3. Choisissez une table dans la liste.
4. Dans la partie inférieure de la page Tables, choisissez Statistiques de colonne.
5. Vous pouvez également sélectionner Générer selon le calendrier sous Statistiques de colonne depuis Actions.
6. Sur la page Générer des statistiques selon le calendrier configurez un calendrier récurrent pour exécuter la tâche de statistiques de colonne en choisissant la fréquence et l'heure de début. Vous pouvez choisir une fréquence horaire, quotidienne, hebdomadaire ou définir une expression cron pour spécifier le calendrier.

   Une expression cron est une chaîne représentant un modèle de planification, composée de 6 champs séparés par des espaces :* * * * * * <minute><hour><day of month><month><day of week><year>Par exemple, pour exécuter une tâche tous les jours à minuit, l'expression cron serait : 0 0 * * ? *

   Pour plus d'informations, consultez Expressions Cron.



7. Ensuite, choisissez l'option de colonne pour générer des statistiques.

   - Toutes les colonnes : choisissez cette option pour générer des statistiques pour toutes les colonnes de la table.
   - Colonnes sélectionnées : choisissez cette option pour générer des statistiques pour des colonnes spécifiques. Vous pouvez sélectionner les colonnes dans la liste déroulante.

8. Choisissez un rôle IAM ou créez un rôle existant autorisé à générer des statistiques. AWS Glue assume ce rôle pour générer des statistiques de colonne.

   Une approche plus rapide consiste à laisser la AWS Glue console créer un rôle pour vous. Le rôle qu'il crée est spécifiquement destiné à générer des statistiques sur les colonnes et inclut la politique `AWSGlueServiceRole` AWS gérée ainsi que la politique en ligne requise pour la source de données spécifiée.

Si vous spécifiez un rôle existant pour la génération de statistiques de colonne, assurez-vous qu'il inclut la politique `AWSGlueServiceRole` ou un équivalent (ou une version limitée de cette politique), en plus des politiques en ligne requises.

9. (Facultatif) Choisissez ensuite une configuration de sécurité pour activer le chiffrement au repos des journaux.
10. (Facultatif) Vous pouvez choisir une taille d'échantillon en indiquant uniquement un pourcentage spécifique de lignes dans la table pour générer des statistiques. La valeur par défaut est toutes les lignes. Utilisez les flèches haut et bas pour augmenter ou diminuer la valeur en pourcentage.

    Nous vous recommandons d'inclure toutes les lignes de la table pour calculer des statistiques précises. Utilisez des exemples de lignes pour générer des statistiques de colonne uniquement lorsque des valeurs approximatives sont acceptables.
11. Choisissez Générer des statistiques pour exécuter la tâche de génération des statistiques de colonne.

AWS CLI

Vous pouvez utiliser l' AWS CLI exemple suivant pour créer un calendrier de génération de statistiques de colonnes. Le nom de la base de données, le nom de la table et le rôle sont des paramètres obligatoires, tandis que les paramètres facultatifs sont le planning, l'identifiant du catalogue column-name-list, la taille de l'échantillon et la configuration de sécurité.

```
aws glue create-column-statistics-task-settings \
 --database-name 'database_name' \
 --table-name table_name \
 --role 'arn:aws:iam::123456789012:role/stats-role' \
 --schedule 'cron(0 0-5 14 * * ?)' \
 --column-name-list 'col-1' \
 --catalog-id '123456789012' \
 --sample-size '10.0 ' \
 --security-configuration 'test-security'
```

Vous pouvez également générer des statistiques de colonne en appelant l'StartColumnStatisticsTaskRunopération.

Gestion du calendrier de génération des statistiques de colonne

Vous pouvez gérer les opérations de planification telles que la mise à jour, le démarrage, l'arrêt et la suppression des planifications pour les statistiques de colonne générées dans AWS Glue. Vous pouvez utiliser les opérations de l'API de statistiques de AWS Glue console ou de AWS Glue colonne pour effectuer ces tâches. AWS CLI

Rubriques

- Mise à jour du calendrier de génération des statistiques de colonne
- Arrêt du calendrier de génération des statistiques de colonne
- Reprise du calendrier pour la génération des statistiques de colonne
- Suppression du calendrier de génération de statistiques de colonne

Mise à jour du calendrier de génération des statistiques de colonne

Vous pouvez mettre à jour le calendrier pour déclencher la tâche de génération des statistiques de colonne après sa création. Vous pouvez utiliser la AWS Glue console ou exécuter l'UpdateColumnStatisticsTaskSettingsopération pour mettre à jour le calendrier d'une table. AWS CLI Vous pouvez modifier les paramètres d'un calendrier existant, tels que le type de calendrier (à la demande ou planifié) et d'autres paramètres facultatifs.

AWS Management Console

Pour mettre à jour les paramètres d'une tâche de génération de statistiques de colonne

1. Connectez-vous à la AWS Glue console à l'adresse https://console.aws.amazon.com/glue/.
2. Choisissez la table que vous voulez mettre à jour dans la liste.
3. Dans la partie inférieure de la page de détails de la table, choisissez Statistiques de colonne.
4. Sous Actions, choisissez Modifier pour mettre à jour le calendrier.
5. Apportez les modifications souhaitées au calendrier, puis cliquez sur Enregistrer.

AWS CLI

Si vous n'utilisez pas la fonction AWS Glue de génération de statistiques de la console, vous pouvez mettre à jour le calendrier manuellement à l'aide de la `update-column-statistics-task-settings` commande. L'exemple suivant montre comment mettre à jour les statistiques de colonne à l'aide de la AWS CLI.

```
aws glue update-column-statistics-task-settings \
 --database-name 'database_name' \
 --table-name 'table_name' \
 --role arn:aws:iam::123456789012:role/stats_role \
 --schedule 'cron(0 0-5 16 * * ?)' \
 --column-name-list 'col-1' \
 --sample-size '20.0' \
 --catalog-id '123456789012'\
 --security-configuration 'test-security'
```

Arrêt du calendrier de génération des statistiques de colonne

Si vous n'avez plus besoin des statistiques incrémentielles, vous pouvez arrêter la génération planifiée pour économiser des ressources et des coûts. La suspension du calendrier n'a aucune incidence sur les statistiques générées précédemment. Vous pouvez reprendre le calendrier à votre convenance.

AWS Management Console

Pour arrêter le calendrier pour une tâche de génération des statistiques de colonne

1. Sur AWS Glue la console, sélectionnez Tables sous Catalogue de données.
2. Sélectionnez une table avec des statistiques de colonne.
3. Sur la page Détails de la table, choisissez Statistiques de colonne.
4. Sous Actions, choisissez Génération planifiée, puis Pause.
5. Choisissez Pause pour confirmer.

AWS CLI

Pour arrêter le calendrier d'exécution d'une tâche de statistiques de colonne à l'aide de AWS CLI, vous pouvez utiliser la commande suivante :

```
aws glue stop-column-statistics-task-run-schedule \
 --database-name ''database_name' \
 --table-name 'table_name'
```

Remplacez les valeurs `database_name` et `table_name` par les noms réels de la base de données et de la table pour lesquelles vous souhaitez arrêter le calendrier d'exécution de la tâche de statistiques de colonne.

Reprise du calendrier pour la génération des statistiques de colonne

Si vous avez suspendu le calendrier de génération des statistiques, vous AWS Glue permet de le reprendre à votre convenance. Vous pouvez reprendre le planning à l'aide de la AWS Glue AWS CLI console ou de l'StartColumnStatisticsTaskRunScheduleopération.

AWS Management Console

Pour reprendre le calendrier de génération des statistiques de colonne

1. Sur AWS Glue la console, sélectionnez Tables sous Catalogue de données.
2. Sélectionnez une table avec des statistiques de colonne.
3. Sur la page Détails de la table, choisissez Statistiques de colonne.
4. Sous Actions, choisissez Génération planifiée, puis Reprendre.
5. Choisissez Reprendre pour confirmer.

AWS CLI

Remplacez les valeurs `database_name` et `table_name` par les noms réels de la base de données et de la table pour lesquelles vous souhaitez arrêter le calendrier d'exécution de la tâche de statistiques de colonne.

```
aws glue start-column-statistics-task-run-schedule \
 --database-name 'database_name' \
 --table-name 'table_name'
```

Suppression du calendrier de génération de statistiques de colonne

Bien que la gestion up-to-date des statistiques soit généralement recommandée pour optimiser les performances des requêtes, il existe des cas d'utilisation spécifiques dans lesquels la suppression du calendrier de génération automatique peut être bénéfique.

- Si les données restent relativement statiques, les statistiques des colonnes existantes peuvent rester précises pendant une période prolongée, ce qui réduit le besoin de mises à jour fréquentes.

La suppression du calendrier permet d'éviter la consommation inutile de ressources et les frais généraux associés à la régénération des statistiques sur des données immuables.

- Lorsque le contrôle manuel de la génération de statistiques est préférable. En supprimant le calendrier automatique, les administrateurs peuvent mettre à jour de manière sélective les statistiques de colonne à des intervalles spécifiques ou après des modifications importantes des données, en alignant le processus sur leurs stratégies de maintenance et leurs besoins d'allocation de ressources.

AWS Management Console

Pour supprimer le calendrier de génération des statistiques de colonne

1. Sur AWS Glue la console, sélectionnez Tables sous Catalogue de données.
2. Sélectionnez une table avec des statistiques de colonne.
3. Sur la page Détails de la table, choisissez Statistiques de colonne.
4. Sous Actions, choisissez Génération planifiée, puis Supprimer.
5. Choisissez Supprimer pour confirmer.

AWS CLI

Remplacez les valeurs `database_name` et `table_name` par les noms réels de la base de données et de la table pour lesquelles vous souhaitez arrêter le calendrier d'exécution de la tâche de statistiques de colonne.

Vous pouvez supprimer le calendrier des statistiques des colonnes à l'aide de l'opération DeleteColumnStatisticsTaskSettingsAPI ou AWS CLI. L'exemple suivant montre comment supprimer le calendrier de génération de statistiques de colonne à l'aide de AWS Command Line Interface (AWS CLI).

```
aws glue delete-column-statistics-task-settings \
    --database-name 'database_name' \
    --table-name 'table_name'
```

Génération de statistiques de colonne à la demande

Vous pouvez exécuter la tâche de statistiques de colonnes pour la tâche de AWS Glue Data Catalog tables à la demande sans planification définie. Cette option est utile pour les analyses ad hoc ou lorsque les statistiques doivent être calculées immédiatement.

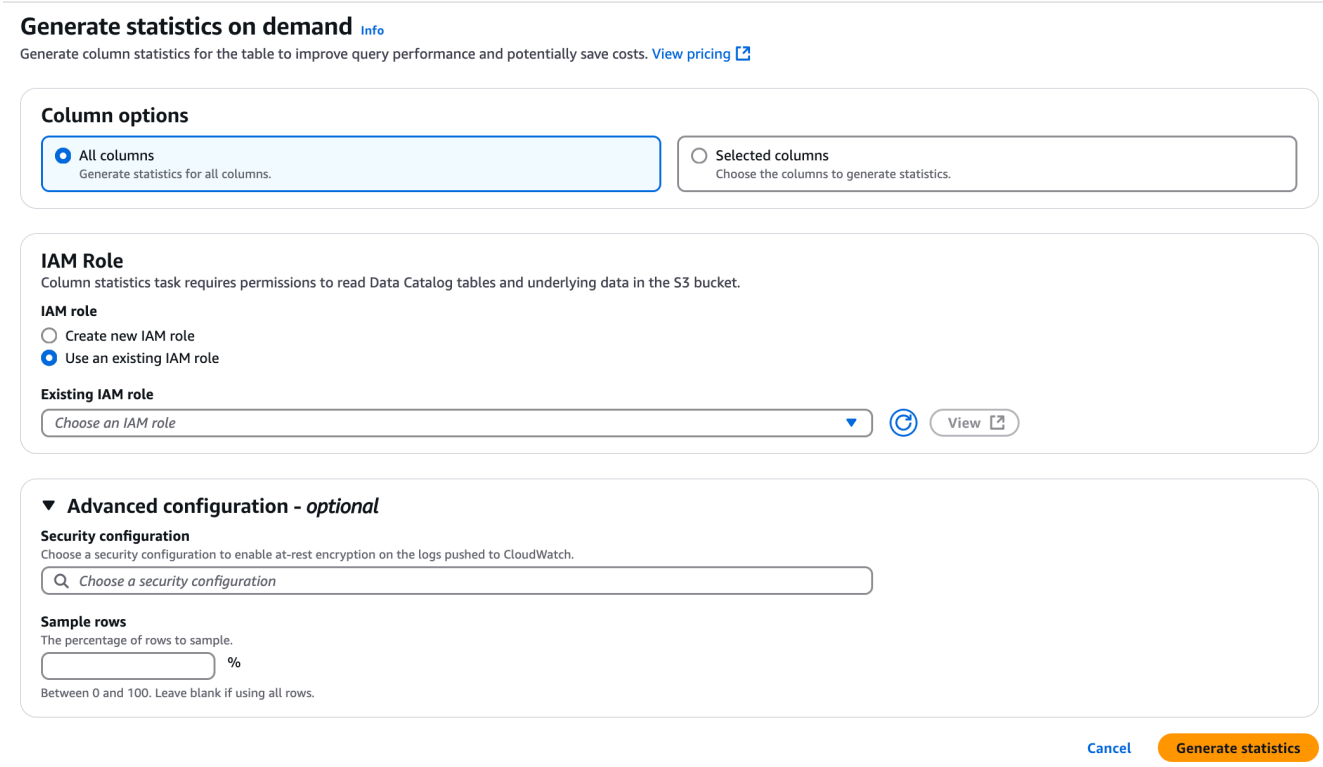
Procédez comme suit pour générer des statistiques de colonne à la demande pour les tables du catalogue de données à l'aide de AWS Glue la console ou AWS CLI.

AWS Management Console

Pour générer des statistiques de colonne à l'aide de la console

1. Connectez-vous à la AWS Glue console à l'adresse https://console.aws.amazon.com/glue/.
2. Choisissez des tables du catalogue de donnée.
3. Choisissez une table dans la liste.
4. Choisissez Générer des statistiques dans le menu Actions.

   Vous pouvez également choisir l'option Générer, Générer à la demande sous l'onglet Statistiques de colonne dans la section inférieure de la page Table.
5. Suivez les étapes 7 à 11 dans Génération de statistiques de colonne selon un calendrier pour générer des statistiques de colonne pour la table.
6. Sur la page Générer des statistiques, spécifiez les options suivantes :



- Toutes les colonnes : choisissez cette option pour générer des statistiques pour toutes les colonnes de la table.
- Colonnes sélectionnées : choisissez cette option pour générer des statistiques pour des colonnes spécifiques. Vous pouvez sélectionner les colonnes dans la liste déroulante.
- Rôle IAM : choisissez Créer un rôle IAM doté des politiques d'autorisation requises pour exécuter la tâche de génération de statistiques de colonne. Choisissez Afficher les détails des autorisations pour examiner la déclaration de politique. Vous pouvez également sélectionner un rôle IAM dans la liste. Pour plus d'informations sur les autorisations requises, consultez Conditions préalables à la génération de statistiques de colonne.

  AWS Glue assume les autorisations du rôle que vous spécifiez pour générer des statistiques.

  Pour plus d'informations sur la fourniture de rôles pour AWS Glue, consultez la section Politiques basées sur l'identité pour. AWS Glue.
- (Facultatif) Choisissez ensuite une configuration de sécurité pour activer le chiffrement au repos des journaux.

- Exemples de lignes : choisissez uniquement un pourcentage spécifique de lignes dans la table pour générer des statistiques. La valeur par défaut est toutes les lignes. Utilisez les flèches haut et bas pour augmenter ou diminuer la valeur en pourcentage.

  > **Note**
  >
  > Nous vous recommandons d'inclure toutes les lignes de la table pour calculer des statistiques précises. Utilisez des exemples de lignes pour générer des statistiques de colonne uniquement lorsque des valeurs approximatives sont acceptables.

Choisissez Générer des statistiques pour exécuter la tâche.

AWS CLI

Cette commande déclenchera l'exécution d'une tâche de statistiques de colonne pour la table spécifiée. Vous devez indiquer le nom de la base de données, le nom de la table, un rôle IAM autorisé à générer des statistiques, et éventuellement fournir des noms de colonnes et un pourcentage de taille d'échantillon pour le calcul des statistiques.

```
aws glue start-column-statistics-task-run \
    --database-name 'database_name \
    --table-name 'table_name' \
    --role 'arn:aws:iam::123456789012:role/stats-role' \
    --column-name 'col1','col2'  \
    --sample-size 10.0
```

Cette commande lancera une tâche visant à générer des statistiques de colonne pour la table spécifiée.

Mise à jour des statistiques de colonne sur demande

La gestion des statistiques des up-to-date colonnes est essentielle pour que l'optimiseur de requêtes génère des plans d'exécution efficaces, garantissant des performances de requête améliorées, une consommation de ressources réduite et de meilleures performances globales du système. Ce processus est particulièrement important après des modifications importantes des données, telles

que des chargements groupés ou des modifications majeurs, qui peuvent rendre les statistiques existantes obsolètes.

Vous devez exécuter explicitement la tâche Générer des statistiques depuis la AWS Glue console pour actualiser les statistiques des colonnes. Le catalogue de données n'actualise pas automatiquement les statistiques.

Si vous n'utilisez pas la fonction AWS Glue de génération de statistiques de la console, vous pouvez mettre à jour manuellement les statistiques des colonnes à l'aide de l'opération UpdateColumnStatisticsForTableAPI ou AWS CLI. L'exemple suivant montre comment mettre à jour les statistiques de colonne à l'aide de la AWS CLI.

```
aws glue update-column-statistics-for-table --cli-input-json:

{
    "CatalogId": "111122223333",
    "DatabaseName": "database_name",
    "TableName": "table_name",
    "ColumnStatisticsList": [
        {
            "ColumnName": "col1",
            "ColumnType": "Boolean",
            "AnalyzedTime": "1970-01-01T00:00:00",
            "StatisticsData": {
                "Type": "BOOLEAN",
                "BooleanColumnStatisticsData": {
                    "NumberOfTrues": 5,
                    "NumberOfFalses": 5,
                    "NumberOfNulls": 0
                }
            }
        }
    ]
}
```

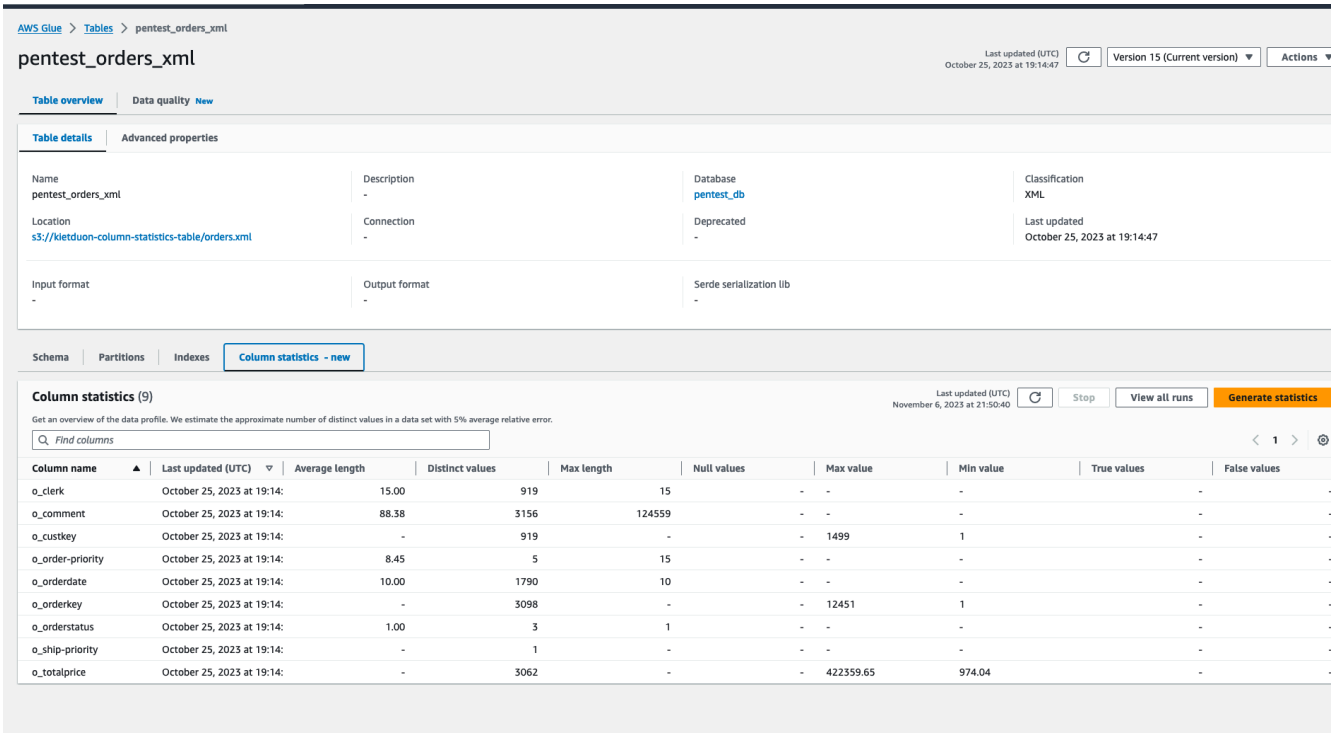
Affichage des statistiques de colonne

Une fois les statistiques générées avec succès, Data Catalog stocke ces informations pour les optimiseurs basés sur les coûts Amazon Athena et Amazon Redshift afin de faire des choix optimaux lors de l'exécution de requêtes. Les statistiques varient en fonction du type de colonne.

AWS Management Console

Pour afficher les statistiques de colonne d'une table

- Après avoir exécuté la tâche de statistiques de colonne, l'onglet Statistiques de colonne de la page Détails de la table affiche les statistiques pour la table.



Les statistiques suivantes sont disponibles :

- Nom de colonne : nom de colonne utilisé pour générer des statistiques
- Dernière mise à jour : date et heure de génération des statistiques
- Longueur moyenne : longueur moyenne des valeurs dans la colonne
- Valeurs distinctes : nombre total de valeurs distinctes dans la colonne. Nous estimons le nombre de valeurs distinctes dans une colonne avec une erreur relative de 5 %.
- Valeur maximale : la plus grande valeur de la colonne.
- Valeur minimale : la plus petite valeur de la colonne.
- Longueur maximale : longueur de la valeur la plus élevée dans la colonne.
- Valeurs nulles : nombre total de valeurs nulles dans la colonne.
- Valeurs réelles : nombre total de valeurs réelles dans la colonne.
- Valeurs fausses : nombre total de valeurs fausses dans la colonne.

- numFiles : nombre total de fichiers dans la table. Cette valeur est disponible sous l'onglet Propriétés avancées.

AWS CLI

L'exemple suivant montre comment récupérer les statistiques de colonne à l'aide de la AWS CLI.

```
aws glue get-column-statistics-for-table \
    --database-name database_name \
    --table-name table_name \
    --column-names <column_name>
```

Vous pouvez également consulter les statistiques des colonnes à l'aide de l'opération GetColumnStatisticsForTableAPI.

Affichage des exécutions de tâches de statistiques de colonne

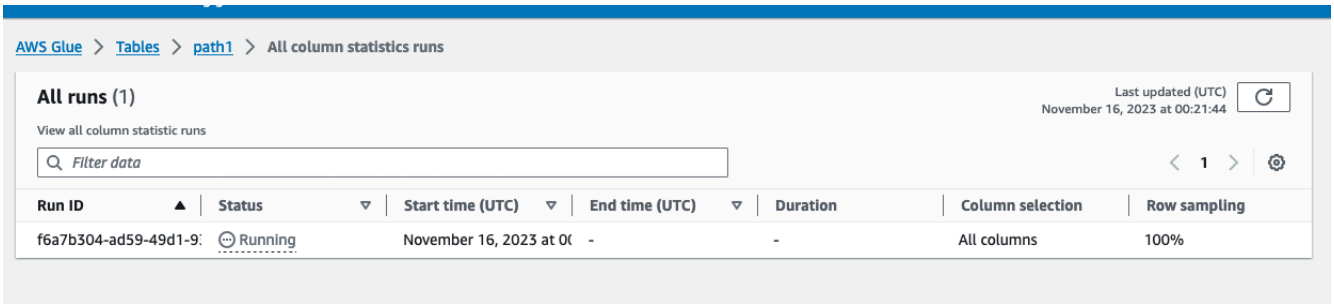
Après avoir exécuté une tâche de statistiques de colonne, vous pouvez explorer les détails de l'exécution de la tâche pour une table à l'aide de AWS Glue la console AWS CLI ou à l'aide d'une GetColumnStatisticsTaskRunsopération.

Console

Pour afficher les détails de l'exécution de la tâche de statistiques de colonne

1. Sur AWS Glue la console, sélectionnez Tables sous Catalogue de données.
2. Sélectionnez une table avec des statistiques de colonne.
3. Sur la page Détails de la table, choisissez Statistiques de colonne.
4. Choisissez Afficher les exécutions.

   Vous pouvez voir des informations sur toutes les exécutions associées à la table spécifiée.



AWS CLI

Dans l'exemple suivant, remplacez les valeurs pour `DatabaseName` et `TableName` par le nom réel de la base de données et de la table.

```
aws glue get-column-statistics-task-runs --input-cli-json file://input.json
{
    "DatabaseName": "database_name",
    "TableName": "table_name"
}
```

Arrêt d'exécution de la tâche de statistiques de colonne

Vous pouvez arrêter une tâche de statistiques de colonne exécutée pour une table à l'aide de AWS Glue la console AWS CLI ou à l'aide d'une StopColumnStatisticsTaskRunopération.

Console

Pour arrêter l'exécution d'une tâche de statistiques de colonne

1. Sur AWS Glue la console, sélectionnez Tables sous Catalogue de données.
2. Sélectionnez la table dans laquelle l'exécution de la tâche de statistiques de colonne est en cours.
3. Sur la page Détails de la table, choisissez Statistiques de colonne.
4. Choisissez Arrêter.

   Si vous arrêtez la tâche avant la fin de l'exécution, aucune statistique de colonne ne sera générée pour la table.

AWS CLI

Dans l'exemple suivant, remplacez les valeurs pour `DatabaseName` et `TableName` par le nom réel de la base de données et de la table.

```
aws glue stop-column-statistics-task-run --input-cli-json file://input.json
{
    "DatabaseName": "database_name",
    "TableName": "table_name"
```

```
}
```

Supprimer les statistiques de colonne

Vous pouvez supprimer les statistiques des colonnes à l'aide de DeleteColumnStatisticsForTablel'opération API ou AWS CLI. L'exemple suivant montre comment supprimer des statistiques de colonne à l'aide de AWS Command Line Interface (AWS CLI).

```
aws glue delete-column-statistics-for-table \
    --database-name 'database_name' \
    --table-name 'table_name' \
    --column-name 'column_name'
```

Considérations et restrictions

Les considérations et limites suivantes s'appliquent à la génération de statistiques de colonne.

Considérations

- L'utilisation de l'échantillonnage pour générer des statistiques réduit le temps d'exécution, mais peut générer des statistiques inexactes.
- Le catalogue de données ne stocke pas les différentes versions des statistiques.
- Vous ne pouvez exécuter qu'une seule tâche de génération de statistiques à la fois par table.
- Si une table est chiffrée à l'aide d'une AWS KMS clé client enregistrée dans Data Catalog, AWS Glue utilise la même clé pour chiffrer les statistiques.

La tâche de statistiques de colonne prend en charge la génération de statistiques :

- Lorsque le rôle IAM dispose d'autorisations complètes sur les tables (IAM ou Lake Formation).
- Lorsque le rôle IAM dispose d'autorisations sur la table en utilisant le mode d'accès hybride de Lake Formation.

La tâche de statistiques de colonne ne prend pas en charge la génération de statistiques pour :

- Tables avec contrôle d'accès basé sur les cellules de Lake Formation

- Lacs de données transactionnels : Linux Foundation Delta Lake, Apache Hudi.
- Tables dans des bases de données fédérées - Hive metastore, unités de partage des données d'Amazon Redshift
- Colonnes imbriquées, tableaux et types de données struct.
- Table partagée avec vous depuis un autre compte

Chiffrement de votre catalogue de données

Vous pouvez protéger vos métadonnées stockées dans l' AWS Glue Data Catalog at rest à l'aide de clés de chiffrement gérées par AWS Key Management Service (AWS KMS). Vous pouvez activer le chiffrement du catalogue de données pour le nouveau catalogue de données à l'aide des paramètres du catalogue de données. Vous pouvez activer ou désactiver le chiffrement pour le catalogue de données existant selon vos besoins. Lorsque cette option est activée, elle AWS Glue chiffre toutes les nouvelles métadonnées écrites dans le catalogue, tandis que les métadonnées existantes restent non chiffrées.

Pour obtenir des informations détaillées sur le chiffrement de votre catalogue de données, consultez Chiffrement de votre catalogue de données.

Sécurisation de votre catalogue de données à l'aide de Lake Formation

AWS Lake Formation est un service qui facilite la configuration d'un lac de données sécurisé dans AWS. Il fournit un emplacement central pour créer et gérer en toute sécurité vos lacs de données en définissant des autorisations de contrôle d'accès précises. Lake Formation utilise le catalogue de données pour stocker et extraire les métadonnées relatives à votre lac de données, telles que les définitions de tables, les informations de schéma et les paramètres de contrôle d'accès aux données.

Vous pouvez enregistrer l'emplacement de vos données Amazon S3 dans la table de métadonnées ou la base de données avec Lake Formation et l'utiliser pour définir des autorisations au niveau des métadonnées sur les ressources du catalogue de données. Vous pouvez utiliser Lake Formation pour gérer les autorisations d'accès au stockage sur les données sous-jacentes stockées dans Amazon S3 pour le compte de moteurs d'analytique intégrés.

Pour plus d'informations, voir Qu'est-ce que c'est AWS Lake Formation ? .

Utilisation des AWS Glue Data Catalog vues dans AWS Glue

Vous pouvez créer et gérer des vues dans le AWS Glue Data Catalog, communément appelé AWS Glue Data Catalog vues. Ces vues sont utiles car elles prennent en charge plusieurs moteurs de requêtes SQL, ce qui vous permet d'accéder à la même vue sur différents AWS services Amazon Athena, tels que Amazon Redshift, et AWS Glue. Vous pouvez utiliser des vues basées sur Apache Iceberg, Apache Hudi et Delta Lake.

En créant une vue dans le catalogue de données, vous pouvez utiliser des autorisations de ressources et des contrôles d'accès basés sur des balises AWS Lake Formation pour autoriser l'accès à celui-ci. Avec cette méthode de contrôle d'accès, il n'est pas nécessaire de configurer un accès supplémentaire aux tables référencées lors de la création de la vue. Cette méthode d'octroi d'autorisations est appelée sémantique du définisseur, et ces vues sont appelées vues du définisseur. Pour plus d'informations sur le contrôle d'accès dans AWS Lake Formation, consultez la section Octroi et révocation d'autorisations sur les ressources du catalogue de données dans le guide du AWS Lake Formation développeur.

Les vues du catalogue de données sont utiles dans les cas d'utilisation suivants :

- Contrôle précis des accès : vous pouvez créer une vue qui restreint l'accès aux données en fonction des autorisations dont l'utilisateur a besoin. Par exemple, vous pouvez utiliser des vues du catalogue de données pour empêcher les employés qui ne travaillent pas dans le service des ressources humaines (RH) de voir des données d'identification personnelle (PII).
- Définition complète de vue : en appliquant des filtres aux vues du catalogue de données, vous vous assurez que les enregistrements de données disponibles dans la vue sont toujours complets.
- Sécurité renforcée : la définition de requête utilisée pour créer la vue doit être complète, afin que les vues du catalogue de données soient moins vulnérables aux commandes SQL provenant d'acteurs malveillants.
- Partage de données simplifié — Partagez des données avec d'autres AWS comptes sans déplacer de données, grâce au partage de données entre comptes. AWS Lake Formation

Création d'un affichage du Catalogue de données

Vous pouvez créer des vues de catalogue de données à l'aide des scripts AWS Glue ETL AWS CLI et de Spark SQL. La syntaxe permettant de créer une vue du catalogue de données consiste à spécifier le type de vue en tant que `MULTI DIALECT` et le prédicat `SECURITY` en tant que `DEFINER`, en indiquant une vue plus précise.

Exemple d'instruction SQL permettant de créer une vue du catalogue de données :

```
CREATE PROTECTED MULTI DIALECT VIEW database_name.catalog_view SECURITY DEFINER
AS SELECT order_date, sum(totalprice) AS price
FROM source_table
GROUP BY order_date;
```

Après avoir créé une vue du catalogue de données, vous pouvez utiliser un rôle IAM doté de l'autorisation AWS Lake Formation « SELECT » sur la vue pour l'interroger à partir de services tels que des tâches ETL ou des tâches Amazon Athena Amazon Redshift ETL. AWS Glue Il n'est pas nécessaire d'accorder l'accès aux tables sous-jacentes référencées dans la vue.

Pour plus d'informations sur la création et la configuration des vues du catalogue de données, voir Création de vues du catalogue de AWS Glue données dans le guide du AWS Lake Formation développeur.

Opérations de vue prises en charge

Les fragments de commande suivants vous montrent différentes manières d'utiliser les vues du catalogue de données :

CRÉER UNE VUE

Crée une vue du catalogue de données. Voici un exemple qui montre comment créer une vue à partir d'une table existante :

```
CREATE PROTECTED MULTI DIALECT VIEW catalog_view
SECURITY DEFINER AS SELECT * FROM my_catalog.my_database.source_table
```

MODIFIER LA VUE

Syntaxe disponible :

```
ALTER VIEW view_name [FORCE] ADD DIALECT AS query
ALTER VIEW view_name [FORCE] UPDATE DIALECT AS query
ALTER VIEW view_name DROP DIALECT
```

Vous pouvez utiliser l'option `FORCE ADD DIALECT` pour forcer la mise à jour du schéma et des sous-objets conformément au nouveau dialecte du moteur. Notez que cela peut entraîner des erreurs de requête si vous n'utilisez pas également `FORCE` pour mettre à jour d'autres dialectes du moteur. Voici un exemple :

```
ALTER VIEW catalog_view FORCE ADD DIALECTAS
SELECT order_date, sum(totalprice) AS priceFROM source_tableGROUP BY orderdate;
```

Voici comment modifier une vue afin de mettre à jour le dialecte :

```
ALTER VIEW catalog_view UPDATE DIALECT AS
SELECT count(*) FROM my_catalog.my_database.source_table;
```

DÉCRIRE LA VUE

Syntaxe disponible pour décrire une vue :

`SHOW COLUMNS {FROM|IN} view_name [{FROM|IN} database_name]`— Si l'utilisateur dispose AWS Glue des AWS Lake Formation autorisations requises pour décrire la vue, il peut répertorier les colonnes. Voici quelques exemples de commandes permettant d'afficher des colonnes :

```
SHOW COLUMNS FROM my_database.source_table;
SHOW COLUMNS IN my_database.source_table;
```

`DESCRIBE view_name`— Si l'utilisateur dispose AWS Glue des AWS Lake Formation autorisations requises pour décrire la vue, il peut répertorier les colonnes de la vue ainsi que ses métadonnées.

SUPPRIMER LA VUE

Syntaxe disponible :

```
DROP VIEW [ IF EXISTS ] view_name
```

L'exemple suivant montre une instruction DROP qui teste si une vue existe avant de la supprimer :

```
DROP VIEW IF EXISTS catalog_view;
```

`SHOW CREATE VIEW view_name` : affiche l'instruction SQL qui crée la vue spécifiée. Voici un exemple qui montre comment créer une vue du catalogue de données :

```
SHOW CREATE TABLE my_database.catalog_view;CREATE PROTECTED MULTI DIALECT VIEW
 my_catalog.my_database.catalog_view (
  net_profit,
  customer_id,
```

```
  item_id,
  sold_date)
TBLPROPERTIES (
  'transient_lastDdlTime' = '1736267222')
SECURITY DEFINER AS SELECT * FROM
my_database.store_sales_partitioned_lf WHERE customer_id IN (SELECT customer_id from
 source_table limit 10)
```

AFFICHER LES VUES

Répertoriez toutes les vues du catalogue, comme les vues normales, les vues multidialectes (MDV) et les vues MDV sans le dialecte Spark. La syntaxe disponible est la suivante :

```
SHOW VIEWS [{ FROM | IN } database_name] [LIKE regex_pattern]:
```

Voici un exemple de commande permettant d'afficher les vues :

```
SHOW VIEWS IN marketing_analytics LIKE 'catalog_view*';
```

Pour plus d'informations sur la création et la configuration de vues de catalogue de données, voir Création de vues de catalogue de AWS Glue données dans le guide du AWS Lake Formation développeur.

Interrogation d'un affichage du Catalogue de données

Après avoir créé une vue du catalogue de données, vous pouvez l'interroger. Le rôle IAM configuré dans vos AWS Glue tâches doit disposer de l'autorisation Lake Formation SELECT dans la vue du catalogue de données. Il n'est pas nécessaire d'accorder l'accès aux tables sous-jacentes référencées dans la vue.

Une fois que tout est configuré, vous pouvez interroger votre vue. Par exemple, vous pouvez exécuter la requête suivante pour accéder à une vue.

```
SELECT * from my_database.catalog_view LIMIT 10;
```

Limitations

Tenez compte des restrictions suivantes lorsque vous utilisez les vues du catalogue de données.

- Vous ne pouvez créer des vues de catalogue de données qu'avec la AWS Glue version 5.0 et les versions ultérieures.

- Le définisseur de vues du catalogue de données doit disposer de l'accès `SELECT` aux tables de base sous-jacentes accessibles par la vue. La création de la vue du catalogue de données échoue si des filtres Lake Formation sont imposés au rôle de définisseur d'une table de base spécifique.
- Les tables de base ne doivent pas disposer de l'autorisation de lac de données `IAMAllowedPrincipals` dans AWS Lake Formation. Le cas échéant, l'erreur Multi Dialect views ne peut référencer que des tables sans les autorisations IAMAllowed Principaux se produit.
- L'emplacement Amazon S3 de la table doit être enregistré en tant qu'emplacement de lac de AWS Lake Formation données. Si la table n'est pas enregistrée, l'erreur `Multi Dialect views may only reference AWS Lake Formation managed tables` se produit. Pour plus d'informations sur la façon d'enregistrer des sites Amazon S3 dans AWS Lake Formation, consultez la section Enregistrement d'un site Amazon S3 dans le guide du AWS Lake Formation développeur.
- Vous pouvez uniquement créer des vues du catalogue de données `PROTECTED`. Les vues `UNPROTECTED` ne sont pas prises en charge.
- Vous ne pouvez pas référencer les tables d'un autre AWS compte dans une définition de vue du catalogue de données. Vous ne pouvez pas non plus référencer une table dans le même compte situé dans une région distincte.
- Pour partager des données entre un compte ou une région, l'affichage complet doit être partagé entre les comptes et les régions, à l'aide de liens vers des ressources AWS Lake Formation .
- Les fonctions définies par l'utilisateur (UDFs) ne sont pas prises en charge.
- Les vues du catalogue de données ne peuvent pas référencer d'autres vues.

Accès au catalogue de données

Vous pouvez utiliser le AWS Glue Data Catalog (catalogue de données) pour découvrir et comprendre vos données. Le catalogue de données fournit un moyen cohérent de gérer les définitions de schéma, les types de données, les emplacements et d'autres métadonnées. Vous pouvez accéder au catalogue de données à l'aide des méthodes suivantes :

- AWS Glue console — Vous pouvez accéder au catalogue de données et le gérer via la AWS Glue console, une interface utilisateur Web. La console vous permet de parcourir et de rechercher des bases de données, des tables et leurs métadonnées associées, ainsi que de créer, mettre à jour et supprimer des définitions de métadonnées.
- AWS Glue crawler — Les crawlers sont des programmes qui analysent automatiquement vos sources de données et alimentent le catalogue de données avec des métadonnées. Vous pouvez

créer et exécuter des robots d'exploration pour découvrir et cataloguer des données provenant de diverses sources telles qu'Amazon S3, Amazon RDS, Amazon DynamoDB Amazon CloudWatch et de bases de données relationnelles compatibles JDBC telles que MySQL et PostgreSQL, ainsi que de plusieurs sources non sources telles que Snowflake et Google.AWS BigQuery

- AWS Glue APIs — Vous pouvez accéder au catalogue de données par programmation à l'aide du. AWS Glue APIs Ils vous APIs permettent d'interagir avec le catalogue de données par programmation, ce qui permet l'automatisation et l'intégration avec d'autres applications et services.
- AWS Command Line Interface (AWS CLI) — Vous pouvez utiliser le AWS CLI pour accéder au catalogue de données et le gérer à partir de la ligne de commande. La CLI fournit des commandes pour créer, mettre à jour et supprimer des définitions de métadonnées, ainsi que pour interroger et récupérer des informations de métadonnées.
- Intégration avec d'autres AWS services — Le catalogue de données s'intègre à divers autres AWS services, vous permettant d'accéder aux métadonnées stockées dans le catalogue et de les utiliser. Par exemple, vous pouvez utiliser Amazon Athena pour interroger des sources de données à l'aide des métadonnées du catalogue de données et pour gérer l'accès aux données et la gouvernance des ressources du catalogue de données. AWS Lake Formation

Rubriques

- Connexion au catalogue de données à l'aide du point de terminaison AWS Glue Iceberg REST
- Connexion au catalogue de données à l'aide du point de terminaison de l'extension AWS Glue Iceberg REST
- AWS Glue Spécifications REST APIs pour Apache Iceberg
- Connexion au catalogue de données à partir d'une application Spark autonome
- Mappage de données entre Amazon Redshift et Apache Iceberg
- Considérations et limites relatives à l'utilisation du catalogue REST d' AWS Glue Iceberg APIs

Connexion au catalogue de données à l'aide du point de terminaison AWS Glue Iceberg REST

Le point de terminaison REST Iceberg d' AWS Glue prend en charge les opérations API spécifiées dans la spécification REST Apache Iceberg. À l'aide d'un client REST Iceberg, vous pouvez connecter votre application exécutée sur un moteur analytique au catalogue REST hébergé dans le catalogue de données.

Le point de terminaison prend en charge les deux spécifications des tables Apache Iceberg : v1 et v2, la valeur par défaut étant v2. Lorsque vous utilisez la spécification de la table Iceberg v1, vous devez spécifier v1 dans l'appel d'API. À l'aide de l'opération d'API, vous pouvez accéder aux tables Iceberg stockées à la fois dans le stockage d'objets Amazon S3 et dans le stockage de tableaux Amazon S3.

Configuration du point de terminaison

Vous pouvez accéder au catalogue REST d' AWS Glue Iceberg via le point de terminaison du service. Reportez-vous au guide de référence des points de terminaison de service AWS Glue pour le point de terminaison spécifique à la région. Par exemple, lorsque vous vous connectez à AWS Glue dans la région us-east-1, vous devez configurer la propriété d'URI du point de terminaison comme suit :

```
Endpoint : https://glue.us-east-1.amazonaws.com/iceberg
```

Propriétés de configuration supplémentaires : lorsque vous utilisez le client Iceberg pour connecter un moteur d'analytique tel que Spark au point de terminaison du service, vous devez indiquer les propriétés de configuration d'application suivantes :

```
catalog_name = "mydatacatalog"
aws_account_id = "123456789012"
aws_region = "us-east-1"
spark = SparkSession.builder \
    ... \
    .config("spark.sql.defaultCatalog", catalog_name) \
    .config(f"spark.sql.catalog.{catalog_name}",
 "org.apache.iceberg.spark.SparkCatalog") \
    .config(f"spark.sql.catalog.{catalog_name}.type", "rest") \
    .config(f"spark.sql.catalog.{catalog_name}.uri", "https://glue.
{aws_region}.amazonaws.com/iceberg") \
    .config(f"spark.sql.catalog.{catalog_name}.warehouse", "{aws_account_id}") \
    .config(f"spark.sql.catalog.{catalog_name}.rest.sigv4-enabled", "true") \
    .config(f"spark.sql.catalog.{catalog_name}.rest.signing-name", "glue") \

 .config("spark.sql.extensions","org.apache.iceberg.spark.extensions.IcebergSparkSessionExtensi
 \
    .getOrCreate()
```

AWS Glue Les supports de point de terminaison Iceberg prennent en `https://glue.us-east-1.amazonaws.com/iceberg` charge le REST Iceberg suivant : APIs

- GetConfig
- ListNamespaces
- CreateNamespace
- LoadNamespaceMetadata
- UpdateNamespaceProperties
- DeleteNamespace
- ListTables
- CreateTable
- LoadTable
- TableExists
- UpdateTable
- DeleteTable

Paramètres du chemin d'accès au préfixe et au catalogue

Le catalogue REST d'Iceberg APIs comporte un préfixe de forme libre dans sa demande. URLs Par exemple, l'appel `ListNamespaces` d'API utilise le format `GET/v1/{prefix}/namespaces` URL. AWS Glue Le préfixe suit toujours la `/catalogs/{catalog}` structure pour garantir que le chemin REST aligne la hiérarchie des catalogues AWS Glue multiples. Le paramètre de chemin `{catalog}` peut être dérivé en fonction des règles suivantes :

Modèle d'accès	Style d'ID du catalogue Glue	Style de préfixe	Exemple d'ID de catalogue par défaut	Exemple de route REST
Accéder au catalogue par défaut dans le compte actuel	facultatif	:	ne s'applique pas	GET /v1/catalogs/:/namespaces
Accéder au catalogue par défaut dans un	accountID	accountID	111122223333	GET /v1/catalogs/111122223333/namespaces

Modèle d'accès	Style d'ID du catalogue Glue	Style de préfixe	Exemple d'ID de catalogue par défaut	Exemple de route REST
compte spécifique				
Accéder à un catalogue imbriqué dans le compte actuel	catalog1/catalog2	catalog1/catalog2	rmscatalog1:db1	OBTENEZ / v1/ catalogs/ rmscatalog1:db1/ namespaces
Accéder à un catalogue imbriqué dans un compte spécifique	accountId :catalog1/ catalog2	accountId :catalog1/ catalog2	123456789012/ rmscatalog1:db1	GET /v1/catalogs/123456789012:rmscatalog1:db1/namespaces

Le mappage de cet ID de catalogue vers le préfixe n'est requis que lorsque vous appelez directement le REST APIs. Lorsque vous utilisez le catalogue REST d' AWS Glue Iceberg APIs via un moteur, vous devez spécifier l'ID du AWS Glue catalogue dans le `warehouse` paramètre de l'API de votre catalogue REST Iceberg ou dans le `glue.id` paramètre de l'API de vos AWS Glue extensions. Par exemple, découvrez comment vous pouvez l'utiliser avec EMR Spark dans Use an Iceberg cluster with Spark.

Paramètre de chemin d'accès à l'espace de noms

Les espaces de noms du APIs chemin du catalogue REST d'Iceberg peuvent comporter plusieurs niveaux. Toutefois, AWS Glue ne prend en charge que les espaces de noms à un seul niveau. Pour accéder à un espace de noms dans une hiérarchie de catalogue à plusieurs niveaux, vous pouvez vous connecter à un catalogue à plusieurs niveaux situé au-dessus de l'espace de noms pour référencer l'espace de noms. Cela permet à tout moteur de requête prenant en charge la notation en 3 parties d'accéder `catalog.namespace.table` aux objets de la hiérarchie AWS Glue du catalogue à plusieurs niveaux sans problèmes de compatibilité par rapport à l'utilisation de l'espace de noms à plusieurs niveaux.

Connexion au catalogue de données à l'aide du point de terminaison de l'extension AWS Glue Iceberg REST

AWS Glue Le point de terminaison d'extension Iceberg REST fournit des fonctionnalités supplémentaires APIs, qui ne figurent pas dans la spécification REST d'Apache Iceberg, et fournit des fonctionnalités de planification des scans côté serveur. Ces informations supplémentaires APIs sont utilisées lorsque vous accédez aux tables stockées dans le stockage géré Amazon Redshift. Le point de terminaison est accessible depuis une application utilisant les AWS Glue Data Catalog extensions Apache Iceberg.

Configuration du point de terminaison : un catalogue contenant des tables dans le stockage géré Redshift est accessible via le point de terminaison de service. Reportez-vous au guide de référence des points de terminaison de service AWS Glue pour le point de terminaison spécifique à la région. Par exemple, lorsque vous vous connectez AWS Glue à la région us-east-1, vous devez configurer la propriété URI du point de terminaison comme suit :

```
Endpoint : https://glue.us-east-1.amazonaws.com/extensions
```

```
catalog_name = "myredshiftcatalog"
aws_account_id = "123456789012"
aws_region = "us-east-1"
spark = SparkSession.builder \
    .config("spark.sql.defaultCatalog", catalog_name) \
    .config(f"spark.sql.catalog.{catalog_name}",
 "org.apache.iceberg.spark.SparkCatalog") \
    .config(f"spark.sql.catalog.{catalog_name}.type", "glue") \
    .config(f"spark.sql.catalog.{catalog_name}.glue.id",
 "{123456789012}:redshiftnamespacecatalog/redshiftdb") \

 .config("spark.sql.extensions","org.apache.iceberg.spark.extensions.IcebergSparkSessionExtensi
 \
    .getOrCreate()
```

AWS Glue Spécifications REST APIs pour Apache Iceberg

Cette section contient les spécifications relatives au catalogue et à l' AWS Glue extension AWS Glue Iceberg REST APIs, ainsi que les considérations relatives à leur utilisation. APIs

Les demandes d'API adressées aux AWS Glue Data Catalog points de terminaison sont authentifiées à l'aide de AWS Signature Version 4 (SigV4). Consultez AWS la section Signature version 4 pour les demandes d'API pour en savoir plus sur AWS SigV4.

Lors de l'accès au point de terminaison du AWS Glue service et aux AWS Glue métadonnées, l'application assume un rôle IAM qui nécessite une action `glue:getCatalog` IAM.

L'accès au catalogue de données et à ses objets peut être géré à l'aide des autorisations IAM, Lake Formation ou du mode hybride Lake Formation.

Les catalogues fédérés dans le catalogue de données contiennent des emplacements de données enregistrés pour Lake Formation. Lake Formation utilise le catalogue de données pour fournir des autorisations de type base de données afin de gérer l'accès des utilisateurs aux objets du catalogue de données.

Vous pouvez utiliser les autorisations du mode hybride IAM ou Lake Formation pour gérer l'accès au catalogue de données par défaut et à ses objets. AWS Lake Formation

Pour créer, insérer ou supprimer des données dans les objets gérés Lake Formation, vous devez configurer des autorisations spécifiques pour l'utilisateur ou le rôle IAM.

- CREATE_CATALOG : nécessaire pour créer des catalogues
- CREATE_DATABASE : nécessaire pour créer des bases de données
- CREATE_TABLE : nécessaire pour créer des tables
- DELETE : nécessaire pour supprimer des données d'une table
- DESCRIBE : nécessaire pour lire les métadonnées
- DROP — Obligatoire pour drop/delete une table ou une base de données
- INSERT : nécessaire lorsque le principal doit insérer des données dans une table
- SELECT : nécessaire lorsque le principal doit sélectionner des données d'une table

Pour plus d'informations, consultez la référence sur les autorisations de Lake Formation dans le Guide du AWS Lake Formation développeur.

GetConfig

Informations générales

Nom de l'opération	GetConfig

Type	API de catalogue REST Iceberg
Chemin REST	OBTENEZ/iceberg/v1/config
Action IAM	colle : GetCatalog
Autorisations Lake Formation	Non applicable
CloudTrail event	colle : GetCatalog
Définition Open API	https://github.com/apache/iceberg/blob/apache-iceberg-1.6.1/open-api/rest- catalog-open-api .yaml #L67

Considérations et restrictions

- Le paramètre de `warehouse` requête doit être défini sur l'ID du AWS Glue catalogue. S'il n'est pas défini, le catalogue racine du compte courant est utilisé pour renvoyer la réponse. Pour de plus amples informations, veuillez consulter the section called "Paramètres du chemin d'accès au préfixe et au catalogue".

GetCatalog

Informations générales

Nom de l'opération	GetCatalog
Type	AWS Glue API d'extension
Chemin REST	GET/extensions/v1/catalogs/{catalogue}
Action IAM	colle : GetCatalog
Autorisations Lake Formation	DESCRIBE
CloudTrail event	colle : GetCatalog
Définition Open API	https://github.com/awslabs/glue-extensions-for-iceberg/blob/main/glue-extensions-api.yaml #L40

Considérations et restrictions

- Le paramètre du chemin d'accès au catalogue doit suivre le style décrit dans la section the section called "Paramètres du chemin d'accès au préfixe et au catalogue".

ListNamespaces

Informations générales

Nom de l'opération	ListNamespaces
Type	API de catalogue REST Iceberg
Chemin REST	GET/iceberg/v1/catalogs/{catalog}/namespaces
Action IAM	colle : GetDatabase
Autorisations Lake Formation	ALL, DESCRIBE, SELECT
CloudTrail event	colle : GetDatabase
Définition Open API	https://github.com/apache/iceberg/blob/apache-iceberg-1.6.1/open-api/rest- catalog-open-api .yaml #L205

Considérations et restrictions

- Le paramètre du chemin d'accès au catalogue doit suivre le style décrit dans la section the section called "Paramètres du chemin d'accès au préfixe et au catalogue".
- Seuls les espaces de noms du niveau suivant sont affichés. Pour répertorier les espaces de noms à des niveaux plus profonds, spécifiez l'ID de catalogue imbriqué dans le paramètre de chemin d'accès au catalogue.

CreateNamespace

Informations générales

Nom de l'opération	CreateNamespace
Type	API de catalogue REST Iceberg

Chemin REST	POST/iceberg/v1/catalogs/{catalog}/namespaces
Action IAM	colle : CreateDatabase
Autorisations Lake Formation	ALL, DESCRIBE, SELECT
CloudTrail event	colle : CreateDatabase
Définition Open API	https://github.com/apache/iceberg/blob/apache-iceberg-1.6.1/open-api/rest- catalog-open-api .yaml #L256

Considérations et restrictions

- Le paramètre du chemin d'accès au catalogue doit suivre le style décrit dans la section the section called "Paramètres du chemin d'accès au préfixe et au catalogue".
- Seul un espace de noms à niveau unique peut être créé. Pour créer un espace de noms à plusieurs niveaux, vous devez créer chaque niveau de manière itérative et vous connecter au niveau à l'aide du paramètre de chemin du catalogue.

StartCreateNamespaceTransaction

Informations générales

Nom de l'opération	StartCreateNamespaceTransaction
Type	AWS Glue API d'extensions
Chemin REST	POST/extensions/v1/catalogs/{catalog}/namespaces
Action IAM	colle : CreateDatabase
Autorisations Lake Formation	ALL, DESCRIBE, SELECT
CloudTrail event	colle : CreateDatabase
Définition Open API	https://github.com/apache/iceberg/blob/apache-iceberg-1.6.1/open-api/rest- catalog-open-api .yaml #L256

Considérations et restrictions

- Le paramètre du chemin d'accès au catalogue doit suivre le style décrit dans la section the section called "Paramètres du chemin d'accès au préfixe et au catalogue".
- Vous ne pouvez créer qu'un espace de noms à niveau unique. Pour créer un espace de noms à plusieurs niveaux, vous devez créer chaque niveau de manière itérative et vous connecter au niveau à l'aide du paramètre de chemin du catalogue.
- L'API est asynchrone et renvoie un ID de transaction que vous pouvez utiliser pour le suivi à l'aide de l'appel d'API `CheckTransactionStatus`.
- Vous pouvez appeler cette API uniquement si l'appel d'API `GetCatalog` contient le paramètre `use-extensions=true` dans la réponse.

LoadNamespaceMetadata

Informations générales

Nom de l'opération	LoadNamespaceMetadata
Type	API de catalogue REST Iceberg
Chemin REST	GET/iceberg/v1/catalogs/{catalog}/namespaces/{ns}
Action IAM	colle : GetDatabase
Autorisations Lake Formation	ALL, DESCRIBE, SELECT
CloudTrail event	colle : GetDatabase
Définition Open API	https://github.com/apache/iceberg/blob/apache-iceberg-1.6.1/open-api/rest- catalog-open-api .yaml #L302

Considérations et restrictions

- Le paramètre du chemin d'accès au catalogue doit suivre le style décrit dans la section the section called "Paramètres du chemin d'accès au préfixe et au catalogue".

- Vous ne pouvez spécifier qu'un espace de noms à un seul niveau dans le paramètre de chemin d'accès REST. Pour plus d'informations, consultez la section the section called "Paramètre de chemin d'accès à l'espace de noms".

UpdateNamespaceProperties

Informations générales

Nom de l'opération	UpdateNamespaceProperties
Type	API de catalogue REST Iceberg
Chemin REST	POSTER/iceberg/v1/catalogs/{catalog}/namespaces/{ns}/properties
Action IAM	colle : UpdateDatabase
Autorisations Lake Formation	ALL, ALTER
CloudTrail event	colle : UpdateDatabase
Définition Open API	https://github.com/apache/iceberg/blob/apache-iceberg-1.6.1/open-api/rest- catalog-open-api .yaml #L400

Considérations et restrictions

- Le paramètre du chemin d'accès au catalogue doit suivre le style décrit dans la section the section called "Paramètres du chemin d'accès au préfixe et au catalogue".
- Vous ne pouvez spécifier qu'un espace de noms à un seul niveau dans le paramètre de chemin d'accès REST. Pour plus d'informations, consultez la section the section called "Paramètre de chemin d'accès à l'espace de noms".

DeleteNamespace

Informations générales

Nom de l'opération	DeleteNamespace
Type	API de catalogue REST Iceberg

Chemin REST	DELETE/iceberg/v1/catalogs/{catalog}/namespces/{ns}
Action IAM	colle : DeleteDatabase
Autorisations Lake Formation	ALL, DROP
CloudTrail event	colle : DeleteDatabase
Définition Open API	https://github.com/apache/iceberg/blob/apache-iceberg-1.6.1/open-api/rest- catalog-open-api .yaml #L365

Considérations et restrictions

- Le paramètre du chemin d'accès au catalogue doit suivre le style décrit dans la section the section called "Paramètres du chemin d'accès au préfixe et au catalogue".
- Vous ne pouvez spécifier qu'un espace de noms à un seul niveau dans le paramètre de chemin d'accès REST. Pour plus d'informations, consultez la section the section called "Paramètre de chemin d'accès à l'espace de noms".
- Si la base de données contient des objets, l'opération échouera.
- L'API est asynchrone et renvoie un ID de transaction que vous pouvez utiliser pour le suivi à l'aide de l'appel d'API `CheckTransactionStatus`.
- L'API ne peut être utilisée que si l'appel d'API `GetCatalog` indique `use-extensions=true` dans la réponse.

StartDeleteNamespaceTransaction

Informations générales

Nom de l'opération	StartDeleteNamespaceTransaction
Type	AWS Glue API d'extensions
Chemin REST	SUPPRIMER/extensions/v1/catalogs/{catalog}/namespces/{ns}
Action IAM	colle : DeleteDatabase
Autorisations Lake Formation	ALL, DROP

CloudTrail event	colle : DeleteDatabase
Définition Open API	https://github.com/awslabs/glue-extensions-for-iceberg/blob/main/glue-extensions-api.yaml #L85

Considérations et restrictions

- Le paramètre du chemin d'accès au catalogue doit suivre le style décrit dans la section the section called "Paramètres du chemin d'accès au préfixe et au catalogue".
- Vous ne pouvez spécifier qu'un espace de noms à un seul niveau dans le paramètre de chemin d'accès REST. Pour plus d'informations, consultez la section the section called "Paramètre de chemin d'accès à l'espace de noms".
- Si la base de données contient des objets, l'opération échouera.
- L'API est asynchrone et renvoie un ID de transaction que vous pouvez utiliser pour le suivi à l'aide de l'appel d'API `CheckTransactionStatus`.
- L'API ne peut être utilisée que si l'appel d'API `GetCatalog` indique `use-extensions=true` dans la réponse.

ListTables

Informations générales

Nom de l'opération	ListTables
Type	API de catalogue REST Iceberg
Chemin REST	OBTENEZ/iceberg/v1/catalogs/{catalog}/namespaces/{ns}/tables
Action IAM	colle : GetTables
Autorisations Lake Formation	ALL, SELECT, DESCRIBE
CloudTrail event	colle : GetTables
Définition Open API	https://github.com/apache/iceberg/blob/apache-iceberg-1.6.1/open-api/rest- catalog-open-api .yaml #L463

Considérations et restrictions

- Le paramètre du chemin d'accès au catalogue doit suivre le style décrit dans la section the section called "Paramètres du chemin d'accès au préfixe et au catalogue".
- Vous ne pouvez spécifier qu'un espace de noms à un seul niveau dans le paramètre de chemin d'accès REST. Pour plus d'informations, consultez la section the section called "Paramètre de chemin d'accès à l'espace de noms".
- Toutes les tables, y compris les tables non Iceberg, seront répertoriées. Pour déterminer si une table peut être chargée en tant que table Iceberg ou non, appelez l'opération `LoadTable`.

CreateTable

Informations générales

Nom de l'opération	CreateTable
Type	API de catalogue REST Iceberg
Chemin REST	OBTENEZ/iceberg/v1/catalogs/{catalog}/namespaces/{ns}/tables
Action IAM	colle : CreateTable
Autorisations Lake Formation	ALL, CREATE_TABLE
CloudTrail event	colle : CreateTable
Définition Open API	https://github.com/apache/iceberg/blob/apache-iceberg-1.6.1/open-api/rest- catalog-open-api .yaml #L497

Considérations et restrictions

- Le paramètre du chemin d'accès au catalogue doit suivre le style décrit dans la section the section called "Paramètres du chemin d'accès au préfixe et au catalogue".
- Vous ne pouvez spécifier qu'un espace de noms à un seul niveau dans le paramètre de chemin d'accès REST. Pour plus d'informations, consultez la section the section called "Paramètre de chemin d'accès à l'espace de noms".

- `CreateTable` avec copie intermédiaire n'est pas prise en charge. Si le paramètre de requête `stageCreate` est spécifié, l'opération échouera. Cela signifie que les opérations comme `CREATE TABLE AS SELECT` ne sont pas prises en charge et que vous pouvez utiliser une combinaison de `CREATE TABLE` et de `INSERT INTO` comme solution de contournement.
- L'opération d'API `CreateTable` ne prend pas en charge l'option `state-create = TRUE`.

StartCreateTableTransaction

Informations générales

Nom de l'opération	CreateTable
Type	AWS Glue API d'extensions
Chemin REST	POST/extensions/v1/catalogs/{catalog}/namespaces/{ns}/tables
Action IAM	colle : CreateTable
Autorisations Lake Formation	ALL, CREATE_TABLE
CloudTrail event	colle : CreateTable
Définition Open API	https://github.com/awslabs/glue-extensions-for-iceberg/blob/main/glue-extensions-api.yaml #L107

Considérations et restrictions

- Le paramètre du chemin d'accès au catalogue doit suivre le style décrit dans la section the section called "Paramètres du chemin d'accès au préfixe et au catalogue".
- Vous ne pouvez spécifier qu'un espace de noms à un seul niveau dans le paramètre de chemin d'accès REST. Pour plus d'informations, consultez la section the section called "Paramètre de chemin d'accès à l'espace de noms".
- `CreateTable` avec copie intermédiaire n'est pas prise en charge. Si le paramètre de requête `stageCreate` est spécifié, l'opération échouera. Cela signifie que les opérations comme `CREATE TABLE AS SELECT` ne sont pas prises en charge et que l'utilisateur doit utiliser une combinaison de `CREATE TABLE` et de `INSERT INTO` comme solution de contournement.

- L'API est asynchrone et renvoie un ID de transaction que vous pouvez utiliser pour le suivi à l'aide de l'appel d'API `CheckTransactionStatus`.
- L'API ne peut être utilisée que si l'appel d'API `GetCatalog` indique `use-extensions=true` dans la réponse.

LoadTable

Informations générales

Nom de l'opération	LoadTable
Type	API de catalogue REST Iceberg
Chemin REST	OBTENIR/iceberg/v1/catalogs/{catalog}/namespaces/{ns}/tables/{table}
Action IAM	colle : GetTable
Autorisations Lake Formation	ALL, SELECT, DESCRIBE
CloudTrail événement	colle : GetTable
Définition Open API	https://github.com/apache/iceberg/blob/apache-iceberg-1.6.1/open-api/rest- catalog-open-api .yaml #L616

Considérations

- Le paramètre du chemin d'accès au catalogue doit suivre le style décrit dans la section the section called "Paramètres du chemin d'accès au préfixe et au catalogue".
- Vous ne pouvez spécifier qu'un espace de noms à un seul niveau dans le paramètre de chemin d'accès REST. Pour plus d'informations, consultez la section the section called "Paramètre de chemin d'accès à l'espace de noms".
- `CreateTable` avec copie intermédiaire n'est pas prise en charge. Si le paramètre de requête `stageCreate` est spécifié, l'opération échouera. Cela signifie que les opérations comme `CREATE TABLE AS SELECT` ne sont pas prises en charge et que l'utilisateur doit utiliser une combinaison de `CREATE TABLE` et de `INSERT INTO` comme solution de contournement.
- L'API est asynchrone et renvoie un ID de transaction que vous pouvez utiliser pour le suivi à l'aide de l'appel d'API `CheckTransactionStatus`.

- L'API ne peut être utilisée que si l'appel d'API `GetCatalog` indique `use-extensions=true` dans la réponse.

ExtendedLoadTable

Informations générales

Nom de l'opération	LoadTable
Type	AWS Glue API d'extensions
Chemin REST	OBTENIR/extensions/v1/catalogs/{catalog}/namespaces/{ns}/tables/{table}
Action IAM	colle : GetTable
Autorisations Lake Formation	ALL, SELECT, DESCRIBE
CloudTrail événement	colle : GetTable
Définition Open API	https://github.com/awslabs/glue-extensions-for-iceberg/blob/main/glue-extensions-api.yaml #L134

Considérations

- Le paramètre du chemin d'accès au catalogue doit suivre le style décrit dans la section the section called "Paramètres du chemin d'accès au préfixe et au catalogue".
- Vous ne pouvez spécifier qu'un espace de noms à un seul niveau dans le paramètre de chemin d'accès REST. Pour plus d'informations, consultez la section the section called "Paramètre de chemin d'accès à l'espace de noms".
- Seul le mode `all` est pris en charge pour le paramètre de requête des instantanés.
- Par rapport à l'API `LoadTable`, l'API `ExtendedLoadTable` se distingue des manières suivantes :
  - Elle n'impose pas strictement que tous les champs soient disponibles.
  - Elle fournit les paramètres supplémentaires suivants dans le champ de configuration de la réponse :

Paramètres supplémentaires

Clé de configuration	Description
aws. server-side-capabilities.scan-planning	Indique si le tableau peut être scanné à l'aide du PreparePlan tableau et PlanTable APIs.
aws. server-side-capabilities.data-commit	Indique si la table peut être validée à l'aide de la StartUpdateTable transaction.
aws.glue.staging.location	Un emplacement intermédiaire géré par le service qui peut être utilisé pour le moteur afin d'écrire des fichiers de données temporaires, utilisé pour la planification de l'analyse côté serveur ou pour la validation des données.
aws.glue.staging. access-key-id	Utilisé pour la planification du scan côté serveur ou pour la validation des données, il fait partie des informations d' AWS identification temporaires permettant d'accéder à l'emplacement intermédiaire géré par le service
aws.glue.staging. secret-access-key	Utilisé pour la planification du scan côté serveur ou pour la validation des données, il fait partie des informations d' AWS identification temporaires permettant d'accéder à l'emplacement intermédiaire géré par le service.
aws.glue.staging.session-token	Utilisé pour la planification du scan côté serveur ou pour la validation des données, il fait partie des informations d' AWS identification temporaires permettant d'accéder à l'emplacement intermédiaire géré par le service.
aws.glue.staging.expiration-ms	Utilisé pour la planification de l'analyse côté serveur ou pour la validation des données, délai d'expiration des informations d'identification permettant d'accéder à l'emplacement intermédiaire géré par le service.

Clé de configuration	Description
aws.glue.staging. data-transfer-role-arn	Utilisé pour la planification de l'analyse côté serveur ou pour la validation des données, rôle IAM qui peut être endossé pour accéder à l'emplacement intermédiaire géré par le service.

PreplanTable

Informations générales

Nom de l'opération	PreplanTable
Type	AWS Glue API d'extensions
Chemin REST	POSTER/extensions/v1/catalogs/{catalog}/namespaces/{ns}/tables/{table}/preplan
Action IAM	colle : GetTable
Autorisations Lake Formation	ALL, SELECT, DESCRIBE
CloudTrail événement	colle : GetTable
Définition Open API	https://github.com/awslabs/glue-extensions-for-iceberg/blob/main/glue-extensions-api.yaml #L211

Considérations

- Le paramètre du chemin d'accès au catalogue doit suivre le style décrit dans la section the section called "Paramètres du chemin d'accès au préfixe et au catalogue".
- Vous ne pouvez spécifier qu'un espace de noms à un seul niveau dans le paramètre de chemin d'accès REST. Pour plus d'informations, consultez la section the section called "Paramètre de chemin d'accès à l'espace de noms".
- L'appelant de cette API doit toujours déterminer s'il reste des résultats à récupérer en fonction du jeton de page. Une réponse avec un élément de page vide, mais un jeton de pagination, est possible si le côté serveur est toujours en cours de traitement, mais n'est pas en mesure de produire un résultat dans le temps de réponse donné.

- Vous ne pouvez utiliser cette API que si la réponse de l'API `ExtendedLoadTable` contient `aws.server-side-capabilities.scan-planning=true`.

PlanTable

Informations générales

Nom de l'opération	PlanTable
Type	AWS Glue API d'extensions
Chemin REST	POSTER/extensions/v1/catalogs/{catalog}/namespaces/{ns}/tables/{table}/plan
Action IAM	colle : GetTable
Autorisations Lake Formation	ALL, SELECT, DESCRIBE
CloudTrail événement	colle : GetTable
Définition Open API	https://github.com/awslabs/glue-extensions-for-iceberg/blob/main/glue-extensions-api.yaml #L243

Considérations

- Le paramètre du chemin d'accès au catalogue doit suivre le style décrit dans la section the section called "Paramètres du chemin d'accès au préfixe et au catalogue".
- Vous ne pouvez spécifier qu'un espace de noms à un seul niveau dans le paramètre de chemin d'accès REST. Pour plus d'informations, consultez la section the section called "Paramètre de chemin d'accès à l'espace de noms".
- L'appelant de cette API doit toujours déterminer s'il reste des résultats à récupérer en fonction du jeton de page. Une réponse avec un élément de page vide, mais un jeton de pagination, est possible si le côté serveur est toujours en cours de traitement, mais n'est pas en mesure de produire un résultat dans le temps de réponse donné.
- Vous ne pouvez utiliser cette API que si la réponse de l'API `ExtendedLoadTable` contient `aws.server-side-capabilities.scan-planning=true`.

TableExists

Informations générales

Nom de l'opération	TableExists
Type	API de catalogue REST Iceberg
Chemin REST	HEAD/iceberg/v1/catalogs/{catalog}/namespaces/{ns}/tables/{tableau}
Action IAM	colle : GetTable
Autorisations Lake Formation	ALL, SELECT, DESCRIBE
CloudTrail événement	colle : GetTable
Définition Open API	https://github.com/apache/iceberg/blob/apache-iceberg-1.6.1/open-api/rest- catalog-open-api .yaml #L833

Considérations

- Le paramètre du chemin d'accès au catalogue doit suivre le style décrit dans la section the section called "Paramètres du chemin d'accès au préfixe et au catalogue".
- Vous ne pouvez spécifier qu'un espace de noms à un seul niveau dans le paramètre de chemin d'accès REST. Pour plus d'informations, consultez la section the section called "Paramètre de chemin d'accès à l'espace de noms".

UpdateTable

Informations générales

Nom de l'opération	UpdateTable
Type	API de catalogue REST Iceberg
Chemin REST	POST/iceberg/v1/catalogs/{catalog}/namespaces/{ns}/tables/{tableau}
Action IAM	colle : UpdateTable

Autorisations Lake Formation	ALL, ALTER
CloudTrail événement	colle : UpdateTable
Définition Open API	https://github.com/apache/iceberg/blob/apache-iceberg-1.6.1/open-api/rest- catalog-open-api .yaml #L677

Considérations

- Le paramètre du chemin d'accès au catalogue doit suivre le style décrit dans la section the section called "Paramètres du chemin d'accès au préfixe et au catalogue".
- Vous ne pouvez spécifier qu'un espace de noms à un seul niveau dans le paramètre de chemin d'accès REST. Pour plus d'informations, consultez la section the section called "Paramètre de chemin d'accès à l'espace de noms".

StartUpdateTableTransaction

Informations générales

Nom de l'opération	StartUpdateTableTransaction
Type	AWS Glue API d'extension
Chemin REST	POST/extensions/v1/catalogs/{catalog}/namespaces/{ns}/tables/{tableau}
Action IAM	colle : UpdateTable
Autorisations Lake Formation	ALL, ALTER
CloudTrail événement	colle : UpdateTable
Définition Open API	https://github.com/awslabs/glue-extensions-for-iceberg/blob/main/glue-extensions-api.yaml #L154

Considérations

- Le paramètre du chemin d'accès au catalogue doit suivre le style décrit dans la section the section called "Paramètres du chemin d'accès au préfixe et au catalogue".
- Vous ne pouvez spécifier qu'un espace de noms à un seul niveau dans le paramètre de chemin d'accès REST. Pour plus d'informations, consultez la section the section called "Paramètre de chemin d'accès à l'espace de noms".
- L'API est asynchrone et renvoie un ID de transaction que vous pouvez utiliser pour le suivi à l'aide de l'appel d'API `CheckTransactionStatus`.
- Une opération `RenamTable` peut également être effectuée par le biais de cette API. Dans ce cas, l'appelant doit également disposer de l'autorisation glue : CreateTable ou LakeFormation CREATE_TABLE pour renommer la table.
- Vous ne pouvez utiliser cette API que si la réponse de l'API `ExtendedLoadTable` contient `aws.server-side-capabilities.scan-planning=true`.

DeleteTable

Informations générales

Nom de l'opération	DeleteTable
Type	API de catalogue REST Iceberg
Chemin REST	DELETE/iceberg/v1/catalogs/{catalog}/namespaces/{ns}/tables/{tableau}
Action IAM	colle : DeleteTable
Autorisations Lake Formation	ALL, DROP
CloudTrail événement	colle : DeleteTable
Définition Open API	https://github.com/apache/iceberg/blob/apache-iceberg-1.6.1/open-api/rest- catalog-open-api .yaml #L793

Considérations

- Le paramètre du chemin d'accès au catalogue doit suivre le style décrit dans la section the section called "Paramètres du chemin d'accès au préfixe et au catalogue".
- Vous ne pouvez spécifier qu'un espace de noms à un seul niveau dans le paramètre de chemin d'accès REST. Pour plus d'informations, consultez la section the section called "Paramètre de chemin d'accès à l'espace de noms".
- Le fonctionnement de l'API `DeleteTable` prend en charge une option de purge. Lorsque la purge est définie sur `true`, les données de la table sont supprimées ; dans le cas contraire, elles ne le sont pas. Pour les tables dans Amazon S3, l'opération ne supprime pas les données des tables. L'opération échoue lorsque la table est stockée dans Amazon S3 et `purge = TRUE,`.

  Pour les tables stockées dans le stockage géré par Amazon Redshift, l'opération supprimera les données des tables, de la même manière que le comportement `DROP TABLE` dans Amazon Redshift. L'opération échoue lorsque la table est stockée dans Amazon Redshift et `purge = FALSE`.
- `purgeRequest=true` n'est pas pris en charge.

StartDeleteTableTransaction

Informations générales

Nom de l'opération	StartDeleteTableTransaction
Type	AWS Glue API d'extensions
Chemin REST	SUPPRIMER/extensions/v1/catalogs/{catalog}/namespaces/{ns}/tables/{table}
Action IAM	colle : DeleteTable
Autorisations Lake Formation	ALL, DROP
CloudTrail événement	colle : DeleteTable
Définition Open API	https://github.com/apache/iceberg/blob/apache-iceberg-1.6.1/open-api/rest- catalog-open-api .yaml #L793

Considérations

- Le paramètre du chemin d'accès au catalogue doit suivre le style décrit dans la section the section called "Paramètres du chemin d'accès au préfixe et au catalogue".
- Vous ne pouvez spécifier qu'un espace de noms à un seul niveau dans le paramètre de chemin d'accès REST. Pour plus d'informations, consultez la section the section called "Paramètre de chemin d'accès à l'espace de noms".
- `purgeRequest=false` n'est pas pris en charge.
- L'API est asynchrone et renvoie un identifiant de transaction qui peut être suivi via `CheckTransactionStatus`.

CheckTransactionStatus

Informations générales

Nom de l'opération	CheckTransactionStatus
Type	AWS Glue API d'extensions
Chemin REST	POST/extensions/v1/transactions/status
Action IAM	La même autorisation que l'action qui initie la transaction
Autorisations Lake Formation	La même autorisation que l'action qui initie la transaction
Définition Open API	https://github.com/awslabs/glue-extensions-for-iceberg/blob/main/glue-extensions-api.yaml #L273

Considérations

- Le paramètre du chemin d'accès au catalogue doit suivre le style décrit dans la section the section called "Paramètres du chemin d'accès au préfixe et au catalogue".

Connexion au catalogue de données à partir d'une application Spark autonome

Vous pouvez vous connecter au catalogue de données à partir d'une application standard à l'aide d'un connecteur Apache Iceberg.

1. Créez un rôle IAM pour l'application Spark.
2. Connectez-vous au point de terminaison AWS Glue Iceberg Rest à l'aide du connecteur Iceberg.

```
# configure your application. Refer to https://docs.aws.amazon.com/cli/latest/
userguide/cli-configure-envvars.html for best practices on configuring environment
 variables.
export AWS_ACCESS_KEY_ID=$(aws configure get appUser.aws_access_key_id)
export AWS_SECRET_ACCESS_KEY=$(aws configure get appUser.aws_secret_access_key)
export AWS_SESSION_TOKEN=$(aws configure get appUser.aws_secret_token)

export AWS_REGION=us-east-1
export REGION=us-east-1
export AWS_ACCOUNT_ID = {specify your aws account id here}

~/spark-3.5.3-bin-hadoop3/bin/spark-shell \
    --packages org.apache.iceberg:iceberg-spark-runtime-3.4_2.12:1.6.0 \
    --conf
 "spark.sql.extensions=org.apache.iceberg.spark.extensions.IcebergSparkSessionExtensions"
 \
    --conf "spark.sql.defaultCatalog=spark_catalog" \
    --conf "spark.sql.catalog.spark_catalog=org.apache.iceberg.spark.SparkCatalog" \
    --conf "spark.sql.catalog.spark_catalog.type=rest" \
    --conf "spark.sql.catalog.spark_catalog.uri=https://glue.us-east-1.amazonaws.com/
iceberg" \
    --conf "spark.sql.catalog.spark_catalog.warehouse = {AWS_ACCOUNT_ID}" \
    --conf "spark.sql.catalog.spark_catalog.rest.sigv4-enabled=true" \
    --conf "spark.sql.catalog.spark_catalog.rest.signing-name=glue" \
    --conf "spark.sql.catalog.spark_catalog.rest.signing-region=us-east-1" \
    --conf "spark.sql.catalog.spark_catalog.io-
impl=org.apache.iceberg.aws.s3.S3FileIO" \
    --conf
 "spark.hadoop.fs.s3a.aws.credentials.provider=org.apache.hadoop.fs.s3a.SimpleAWSCredentialPr
```

3. Interrogez les données du catalogue de données.

```
spark.sql("create database myicebergdb").show()
spark.sql("""CREATE TABLE myicebergdb.mytbl (name string) USING iceberg location
 's3://bucket_name/mytbl'""")
spark.sql("insert into myicebergdb.mytbl values('demo') ").show()
```

Mappage de données entre Amazon Redshift et Apache Iceberg

Redshift et Iceberg prennent en charge différents types de données. La matrice de compatibilité suivante décrit la prise en charge et les restrictions lors du mappage de données entre ces deux systèmes de données. Reportez-vous aux Types de données Amazon Redshift et aux Spécifications des tables Apache Iceberg pour plus de détails sur les types de données pris en charge par chaque système de données.

Type de données Redshift	les alias ;	Type de données Iceberg
SMALLINT	INT2	int
INTEGER	ENTIER, INT4	int
BIGINT	INT8	long
DECIMAL	NUMERIC	decimal
REAL	FLOAT4	float
REAL	FLOAT4	float
DOUBLE PRECISION	FLOAT8, FLOTTEUR	double
CHAR	CHARACTER, NCHAR	chaîne
VARCHAR	CHARACTER VARYING, NVARCHAR	chaîne
BPCHAR		chaîne
TEXT		chaîne

Type de données Redshift	les alias ;	Type de données Iceberg
DATE		date
TIME	TIME WITHOUT TIMEZONE	time
TIME	TIME WITH TIMEZONE	non pris en charge
TIMESTAMP	TIMESTAMP WITHOUT TIMEZONE	TIMESTAMP
TIMESTAMPZ	TIMESTAMP WITH TIMEZONE	TIMESTAMPZ
INTERVAL YEAR TO MONTH		Non pris en charge
INTERVAL DAY TO SECOND		Non pris en charge
BOOLEAN	BOOL	bool
HLLSKETCH		Non pris en charge
SUPER		Non pris en charge
VARBYTE	VARBINARY, BINARY VARYING	binary
GEOMETRY		Non pris en charge
GEOGRAPHY		Non pris en charge

Considérations et limites relatives à l'utilisation du catalogue REST d' AWS Glue Iceberg APIs

Voici les considérations et les restrictions liées à l’utilisation du comportement d’opération du langage de définition de données (DDL) du catalogue REST d’Apache Iceberg.

Considérations

- Comportement de l’API **`RenameTable`** : l’opération `RenameTable` est prise en charge dans les tables d’Amazon Redshift, mais pas dans Amazon S3.
- Opérations DDL pour les espaces de noms et les tables dans Amazon Redshift : les opérations de création, de mise à jour, de suppression pour les espaces de noms et les tables dans Amazon Redshift sont des opérations asynchrones, car elles dépendent du moment où le groupe de travail géré par Amazon Redshift est disponible, du fait qu’une transaction DDL et DML conflictuelle est en cours et que l’opération doit attendre le verrouillage, puis essayer de valider les modifications.

Limitations

- Les vues de APIs la spécification REST d'Apache Iceberg ne sont pas prises en charge dans le catalogue REST d' AWS Glue Iceberg.

AWS Glue Bonnes pratiques en matière de catalogue de données

Cette section présente les meilleures pratiques pour gérer et utiliser efficacement le AWS Glue Data Catalog. Il met l'accent sur des pratiques telles que l'utilisation efficace des robots d'exploration, l'organisation des métadonnées, la sécurité, l'optimisation des performances, l'automatisation, la gouvernance des données et l'intégration avec d'autres AWS services.

- Utilisez les robots d'exploration de manière efficace : exécutez régulièrement des robots d'exploration pour tenir le catalogue de données en fonction up-to-date des modifications apportées à vos sources de données. Utilisez des analyses incrémentielles pour modifier fréquemment les sources de données afin d’améliorer les performances. Configurez les robots pour ajouter automatiquement de nouvelles partitions ou mettre à jour les schémas lorsque des modifications sont détectées.
- Organiser et nommer les tables de métadonnées : établissez une convention de dénomination cohérente pour les bases de données et les tables du catalogue de données. Regroupez les sources de données associées dans des bases de données ou des dossiers logiques pour une meilleure organisation. Utilisez des noms descriptifs qui indiquent le but et le contenu de chaque table.
- Gérez les schémas de manière efficace : tirez parti des fonctionnalités d'inférence de schémas des robots d' AWS Glue exploration. Examinez et mettez à jour les modifications du schéma avant

de les appliquer afin d’éviter de perturber les applications en aval. Utilisez les fonctionnalités d’évolution du schéma pour gérer correctement les modifications de schéma.

- Sécuriser le catalogue de données : activez le chiffrement des données au repos et en transit pour le catalogue de données. Mettez en œuvre des politiques de contrôle précis des accès pour restreindre l’accès aux données sensibles. Vérifiez et examinez régulièrement les autorisations et les journaux d’activité du catalogue de données.
- Intégration à d'autres AWS services Catalogue de données Utilisez le catalogue de données comme couche de métadonnées centralisée pour des services tels qu'Amazon Athena, Redshift Spectrum et. AWS Lake Formation Tirez parti des tâches ETL AWS Glue pour transformer et charger des données dans différents entrepôts de données tout en conservant les métadonnées dans le catalogue de données.
- Surveillez et optimisez les performances Catalogue de données Surveillez les performances des robots d'exploration et des tâches ETL à l'aide de Amazon CloudWatch métriques. Partitionnez les grands jeux de données dans le catalogue de données pour améliorer les performances des requêtes. Mettez en œuvre des optimisations des performances pour les métadonnées fréquemment consultées.
- Restez à jour grâce à AWS Glue la documentation et au catalogue de données des meilleures pratiques Consultez régulièrement la AWS Glue documentation et les AWS Glue ressources pour connaître les dernières mises à jour, les meilleures pratiques et les recommandations. Participez à AWS Glue des webinaires, à des ateliers et à d'autres événements pour apprendre auprès d'experts et rester au courant des nouvelles fonctionnalités et capacités.

Surveillance des métriques d'utilisation du catalogue de données dans Amazon CloudWatch

AWS Glue Data Catalog les métriques d'utilisation sont désormais disponibles Amazon CloudWatch, ce qui simplifie la surveillance et la compréhension de l'utilisation des ressources dans votre catalogue de données. Vous avez désormais une visibilité immédiate sur l’utilisation des catalogues, des bases de données, des tables, des partitions et des connexions par l’API du catalogue Glue, ce qui facilite le suivi de votre catalogue de données.

Vue d'ensemble des métriques du catalogue de données

AWS Glue Data Catalog publie automatiquement les statistiques d'utilisation sur Amazon CloudWatch. Grâce à l'intégration CloudWatch des métriques, vous pouvez suivre les indicateurs de performance critiques à chaque minute, notamment :

- Demandes de table
- Index de partition créés
- Connexions mises à jour
- Statistiques mises à jour

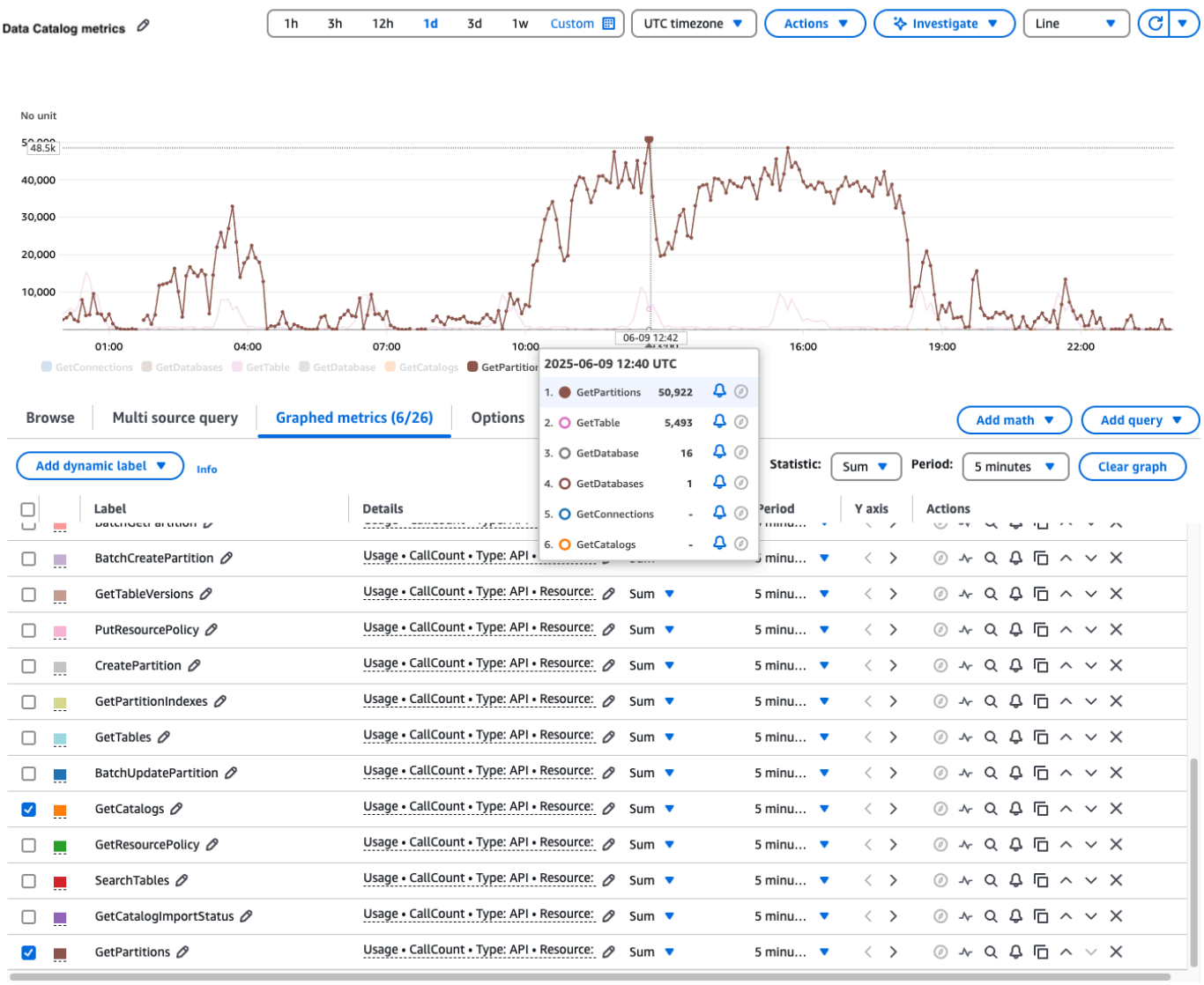
Ces métriques vous aident à identifier les goulets d'étranglement, à détecter les anomalies et à prendre des décisions basées sur les données afin d'améliorer la fiabilité globale du catalogue de données. Vous pouvez également configurer des CloudWatch alarmes pour recevoir des notifications lorsque les métriques dépassent les seuils spécifiés, ce qui permet une gestion proactive de votre déploiement.

Ajouter des métriques à votre CloudWatch tableau de bord

Vous pouvez créer des tableaux de bord personnalisés pour surveiller vos AWS Glue Data Catalog ressources et configurer des alarmes pour être averti de toute activité inhabituelle.

Vous pouvez ajouter les métriques du catalogue de données à votre CloudWatch tableau de bord en suivant ces étapes :

1. Ouvrez la CloudWatch console à l'adresse https://console.aws.amazon.com/cloudwatch/.
2. Dans le panneau de navigation, sélectionnez Métriques.
3. Sélectionnez Toutes les métriques.
4. Choisissez Utilisation>Par ressource AWS .
5. Filtrez par Glue pour voir les métriques disponibles.
6. Sélectionnez les métriques à ajouter à votre tableau de bord.
7. Ajoutez des métriques pour les catalogues, les bases de données, les tables, les partitions et les connexions à votre CloudWatch graphique.



Vous pouvez configurer des alarmes personnalisées qui se déclenchent automatiquement lorsque l'utilisation de l'API dépasse vos seuils définis afin d'identifier les anomalies dans l'utilisation de votre catalogue de données.

Pour obtenir des instructions détaillées sur la configuration des alarmes, voir Création d'une CloudWatch alarme Metrics Insights.

Registre de schémas AWS Glue

> **Note**
>
> AWS GlueLe registre des schémas n'est pas pris en charge dans les régions suivantes de la AWS Glue console : Moyen-Orient (Émirats arabes unis).

Le registre de schémas AWS Glue vous permet de découvrir, de contrôler et de faire évoluer de manière centralisée les schémas de flux de données. Un schéma définit la structure et le format d'un enregistrement de données. Avec le registre de schémas AWS Glue, vous pouvez gérer et appliquer des schémas à vos applications de streaming de données en utilisant des intégrations pratiques avec Apache Kafka, Amazon Managed Streaming pour Apache Kafka, Amazon Kinesis Data Streams, Service géré Amazon pour Apache Flink et AWS Lambda.

Le registre de schémas prend en charge le format de données AVRO (v1.11.4), le format de données JSON avec le format de schéma JSON pour le schéma (spécifications Draft-04, Draft-06 et Draft-07) avec la validation de schéma JSON, à l'aide de la bibliothèque Everit, de Protocol Buffers (Protobuf) versions proto2 et proto3, sans la prise en charge des `extensions` ou des `groups` et la prise en charge du langage Java, avec d'autres formats de données et de langages à venir. Les fonctionnalités prises en charge incluent la compatibilité, l'utilisation de schémas via des métadonnées, l'enregistrement automatique des schémas, la compatibilité IAM et la compression ZLIB facultative pour réduire le stockage et le transfert de données. Le registre de schémas est sans serveur et son utilisation est gratuite.

L'utilisation d'un schéma comme contrat de format de données entre applications producteur et consommateur conduit à une meilleure gouvernance des données, à une meilleure qualité des données, et permet aux applications consommateur de données d'être résilientes face aux changements compatibles en amont.

Le registre de schémas permet aux systèmes disparates de partager un schéma pour la sérialisation et la désérialisation. Par exemple, supposons que vous ayez un producteur et un consommateur de données. Le producteur connaît le schéma lorsqu'il publie les données. Le registre de schémas fournit un sérialiseur et un désérialiseur pour certains systèmes tels qu'Amazon MSK ou Apache Kafka.

Pour de plus amples informations, veuillez consulter Fonctionnement du registre de schémas.

Rubriques

- Schémas
- Registres
- Gestion des versions et compatibilité des schémas
- Bibliothèques Serde en open source
- Quotas du registre des schémas
- Fonctionnement du registre de schémas
- Démarrage avec le registre de schémas

Schémas

Un schéma définit la structure et le format d'un enregistrement de données. Un schéma est une spécification versionnée pour la publication, la consommation ou le stockage des données fiables.

Dans cet exemple de schéma pour Avro, le format et la structure sont définis par la disposition et les noms de champs, tandis que le format des noms de champs est défini par les types de données (par exemple, `string`, `int`).

```
{
    "type": "record",
    "namespace": "ABC_Organization",
    "name": "Employee",
    "fields": [
        {
            "name": "Name",
            "type": "string"
        },
        {
            "name": "Age",
            "type": "int"
        },
        {
            "name": "address",
            "type": {
                "type": "record",
                "name": "addressRecord",
                "fields": [
                    {
                        "name": "street",
                        "type": "string"
```

```
                    },
                    {
                        "name": "zipcode",
                        "type": "int"
                    }
                ]
            }
        }
    ]
}
```

Dans cet exemple JSON Schema Draft-07 pour JSON, le format est défini par l'organisation du schéma JSON.

```
{
 "$id": "https://example.com/person.schema.json",
 "$schema": "http://json-schema.org/draft-07/schema#",
 "title": "Person",
 "type": "object",
 "properties": {
  "firstName": {
   "type": "string",
   "description": "The person's first name."
  },
  "lastName": {
   "type": "string",
   "description": "The person's last name."
  },
  "age": {
   "description": "Age in years which must be equal to or greater than zero.",
   "type": "integer",
   "minimum": 0
  }
 }
}
```

Dans cet exemple pour Protobuf, le format est défini par la version 2 du langage Protocol Buffers (proto2).

```
syntax = "proto2";

package tutorial;
```

```
option java_multiple_files = true;
option java_package = "com.example.tutorial.protos";
option java_outer_classname = "AddressBookProtos";

message Person {
  optional string name = 1;
  optional int32 id = 2;
  optional string email = 3;

  enum PhoneType {
    MOBILE = 0;
    HOME = 1;
    WORK = 2;
  }

  message PhoneNumber {
    optional string number = 1;
    optional PhoneType type = 2 [default = HOME];
  }

  repeated PhoneNumber phones = 4;
}

message AddressBook {
  repeated Person people = 1;
}
```

Registres

Un registre est un conteneur logique de schémas. Les registres vous permettent d'organiser vos schémas, ainsi que de gérer le contrôle des accès pour vos applications. Un registre possède un Amazon Resource Name (ARN) qui vous permet d'organiser et de définir différentes autorisations d'accès aux opérations de schéma dans le registre.

Vous pouvez utiliser le registre par défaut ou créer autant de registres que nécessaire.

Hiérarchie du registre de schémas AWS Glue

- RegistryName: [chaîne]
  - RegistryArn: [AWS ARN]
  - CreatedTime: [horodatage]

    - UpdatedTime: [horodatage]

- SchemaName: [chaîne]
    -
      SchemaArn: [AWS ARN]
    - DataFormat: [Avro, Json ou Protobuf]
    - Compatibility : [par ex., BACKWARD, BACKWARD_ALL, FORWARD, FORWARD_ALL, FULL, FULL_ALL, NONE, DISABLED]
    - Status : [par ex., PENDING, AVAILABLE, DELETING]
    - SchemaCheckpoint: [entier]
    - CreatedTime: [horodatage]
    - UpdatedTime: [horodatage]
- SchemaVersion: [chaîne]
    - SchemaVersionNumber: [entier]
    - Status : [par ex., PENDING, AVAILABLE, DELETING, FAILURE]
    - SchemaDefinition: [chaîne, valeur : JSON]
    - CreatedTime: [horodatage]
- SchemaVersionMetadata: [liste]
    - MetadataKey: [chaîne]
    - MetadataInfo
    - MetadataValue: [chaîne]
    - CreatedTime: [horodatage]

Gestion des versions et compatibilité des schémas

Chaque schéma peut avoir plusieurs versions. La gestion des versions est régie par une règle de compatibilité appliquée à un schéma. Les demandes d'enregistrement de nouvelles versions de schéma sont vérifiées par rapport à cette règle par le registre de schémas avant qu'elles ne puissent aboutir.

Une version de schéma qui est marquée comme point de contrôle est utilisée pour déterminer la compatibilité de l'enregistrement de nouvelles versions d'un schéma. Lorsqu'un schéma est créé

pour la première fois, le point de contrôle par défaut sera la première version. Au fur et à mesure que le schéma évolue avec de nouvelles versions, vous pouvez utiliser le CLI/SDK pour remplacer le point de contrôle par une version d'un schéma à l'aide de l'`UpdateSchema`API qui respecte un ensemble de contraintes. Dans la console, la modification de la définition du schéma ou du mode de compatibilité modifiera le point de contrôle vers la dernière version par défaut.

Les modes de compatibilité vous permettent de contrôler la manière dont les schémas peuvent ou ne peuvent pas évoluer au fil du temps. Ces modes constituent le contrat entre les applications produisant et consommant des données. Lorsqu'une nouvelle version d'un schéma est envoyée au registre, la règle de compatibilité appliquée au nom du schéma est utilisée pour déterminer si la nouvelle version peut être acceptée. Il existe huit modes de compatibilité : NONE, DISABLED, BACKWARD, BACKWARD_ALL, FORWARD_ALL, FULL, FULL_ALL.

Au format de données Avro, les champs peuvent être facultatifs ou obligatoires. Un champ facultatif est celui dans lequel le champ `Type` inclut une valeur null. Les champs obligatoires n'ont pas de valeur nulle comme `Type`.

Dans le format de données Protobuf, les champs peuvent être facultatifs (y compris ceux répétés) ou obligatoires dans la syntaxe proto2, tandis que tous les champs sont facultatifs (y compris ceux répétés) dans la syntaxe proto3. Toutes les règles de compatibilité sont déterminées en fonction de la compréhension des spécifications de Protocol Buffers, ainsi que des directives de la Documentation Protocol Buffers de Google .

- NONE (AUCUN) : aucun mode de compatibilité ne s'applique. Vous pouvez utiliser ce choix dans les scénarios de développement ou si vous ne connaissez pas les modes de compatibilité que vous souhaitez appliquer aux schémas. Toute nouvelle version ajoutée sera acceptée sans faire l'objet d'un contrôle de compatibilité.
- DISABLED (DÉSACTIVÉ) : ce choix de compatibilité empêche la gestion des versions pour un schéma particulier. Aucune nouvelle version ne peut être ajoutée.
- BACKWARD (DESCENDANT) : ce choix de compatibilité est recommandé, car il permet aux applications consommateur de lire à la fois la version actuelle et la version précédente du schéma. Vous pouvez utiliser ce choix pour vérifier la compatibilité par rapport à la version précédente du schéma lorsque vous supprimez des champs ou ajoutez des champs facultatifs. Un cas d'utilisation typique pour BACKWARD (DESCENDANT) est lorsque votre application a été créée pour le schéma le plus récent.

  AVRO

Par exemple, supposons que vous avez un schéma défini par le prénom (obligatoire), le nom (obligatoire), l'adresse électronique (obligatoire) et le numéro de téléphone (facultatif).

Si votre prochaine version de schéma supprime le champ d'adresse électronique requis, l'enregistrement s'effectue correctement. La compatibilité BACKWARD (DESCENDANT) exige que les applications consommateur soient en mesure de lire la version actuelle et précédente du schéma. Vos applications consommateur seront en mesure de lire le nouveau schéma, car le champ d'adresse électronique supplémentaire des anciens messages est ignoré.

Si vous avez proposé une nouvelle version de schéma qui ajoute un champ obligatoire, par exemple, le code postal, elle ne s'enregistrerait pas avec la compatibilité BACKWARD (DESCENDANT). Vos applications consommateur sur la nouvelle version ne seraient pas en mesure de lire les anciens messages avant la modification du schéma, car elles ne disposent pas du champ de code postal requis. Toutefois, si le champ de code postal a été défini comme facultatif dans le nouveau schéma, la version proposée s'enregistrerait avec succès, car les applications consommateur peuvent lire l'ancien schéma sans le champ de code postal facultatif.

JSON

Par exemple, supposons que vous avez une version de schéma définie par le prénom (facultatif), le nom de famille (facultatif), l'adresse électronique (facultatif) et le numéro de téléphone (facultatif).

Si votre prochaine version de schéma ajoute la propriété de numéro de téléphone facultative, elle s'enregistrera avec succès tant que la version de schéma d'origine n'autorise aucune propriété supplémentaire en définissant le champ `additionalProperties` sur false. La compatibilité BACKWARD (DESCENDANT) exige que les applications consommateur soient en mesure de lire la version actuelle et précédente du schéma. Vos applications consommateur seront en mesure de lire les données produites avec le schéma d'origine où la propriété de numéro de téléphone n'existe pas.

Si vous avez une proposition de nouvelle version de schéma qui ajoute la propriété de numéro de téléphone facultative, elle ne s'enregistrerait pas correctement avec la compatibilité BACKWARD (DESCENDANT) lorsque la version de schéma d'origine définit le champ `additionalProperties` sur true, à savoir autoriser toute propriété supplémentaire. Vos applications consommateur sur la nouvelle version ne seraient pas en mesure de lire les anciens messages avant le changement de schéma, car elles ne peuvent pas lire les données avec la propriété de numéro de téléphone dans un type différent, par exemple une chaîne au lieu d'un numéro.

PROTOBUF

Par exemple, imaginez que vous avez une version de schéma définie par un message `Person` avec `first name` (obligatoire), `last name` (obligatoire), `email` (obligatoire) et le champ `phone number` (facultatifs) sous la syntaxe proto2.

Similaire aux scénarios AVRO, si la version de schéma suivante supprime le champ requis `email`, l'enregistrement s'effectue correctement. La compatibilité BACKWARD (DESCENDANT) exige que les applications consommateur soient en mesure de lire la version actuelle et précédente du schéma. Les consommateurs seront en mesure de lire le nouveau schéma, car le champ supplémentaire `email` des anciens messages est ignoré.

Si vous avez proposé une nouvelle version de schéma qui ajoute un champ obligatoire, par exemple, `zip code`, l'enregistrement ne s'effectue pas correctement avec la compatibilité DESCENDANTE. Les consommateurs sur la nouvelle version ne seraient pas en mesure de lire les anciens messages avant la modification du schéma, car ils ne disposent pas du champ requis `zip code`. Toutefois, si le champ `zip code` a été défini comme facultatif dans le nouveau schéma, l'enregistrement de la version proposée s'effectue correctement, car les consommateurs peuvent lire l'ancien schéma sans le champ facultatif `zip code`.

Dans le cas d'un cas d'utilisation de gRPC, l'ajout d'un nouveau service RPC ou d'une nouvelle méthode RPC est une modification de compatibilité descendante. Par exemple, imaginez que vous avez une version de schéma définie par un service RPC `MyService` avec deux méthodes RPC, `Foo` et `Bar`.

Si la version de schéma suivante ajoute une nouvelle méthode RPC appelée `Baz`, l'enregistrement s'effectue correctement. Les consommateurs seront en mesure de lire les données produites avec le schéma d'origine, en fonction de la compatibilité DESCENDANTE, car la nouvelle méthode RPC ajoutée `Baz` est facultative.

Si vous avez proposé une nouvelle version de schéma qui supprime la méthode RPC existante `Foo`, l'enregistrement ne s'effectue pas correctement avec la compatibilité DESCENDANTE. Les consommateurs sur la nouvelle version ne seraient pas en mesure de lire les anciens messages avant la modification du schéma, car ils ne peuvent pas comprendre et lire les données avec la méthode RPC inexistante `Foo` dans une application gRPC.

- BACKWARD_ALL (DESCENDANT_TOUT) : ce choix de compatibilité permet aux applications consommateur de lire à la fois la version actuelle et toutes les versions antérieures du schéma.

Vous pouvez utiliser ce choix pour vérifier la compatibilité avec toutes les versions de schéma précédentes lorsque vous supprimez des champs ou ajoutez des champs facultatifs.

- FORWARD (ASCENDANT) : ce choix de compatibilité permet aux applications consommateur de lire à la fois la version actuelle et les versions suivantes du schéma, mais pas nécessairement les versions ultérieures. Vous pouvez utiliser ce choix pour vérifier la compatibilité avec la dernière version de schéma lorsque vous ajoutez des champs ou supprimez des champs facultatifs. Un cas d'utilisation typique pour FORWARD (ASCENDANT) est lorsque votre application a été créée pour un schéma précédent et devrait être capable de traiter un schéma plus récent.

  AVRO

  Par exemple, supposons que vous avez une version de schéma définie par le prénom (obligatoire), le nom (obligatoire) et l'adresse électronique (facultatif).

  Si vous disposez d'une nouvelle version de schéma qui ajoute un champ obligatoire ; par exemple, un numéro de téléphone, l'enregistrement s'effectue correctement. La compatibilité FORWARD (ASCENDANT) exige que les applications consommateur puissent lire les données produites avec le nouveau schéma à l'aide de la version précédente.

  Si vous avez une version de schéma proposée qui supprime le champ de prénom obligatoire, elle ne s'enregistrerait pas avec la compatibilité FORWARD (ASCENDANT). Vos applications consommateur sur la version précédente ne seraient pas en mesure de lire les schémas proposés, car elles ne disposent pas du champ de prénom obligatoire. Toutefois, si le champ de prénom était initialement facultatif, le nouveau schéma proposé s'enregistrerait correctement, car les applications consommateur peuvent lire des données basées sur le nouveau schéma qui ne dispose pas du champ de prénom facultatif.

  JSON

  Par exemple, supposons que vous avez une version de schéma définie par le prénom (facultatif), le nom de famille (facultatif), l'adresse électronique (facultatif) et le numéro de téléphone (facultatif).

  Si vous disposez d'une nouvelle version de schéma qui supprime la propriété de numéro de téléphone facultative, elle s'enregistrera avec succès tant que la nouvelle version de schéma n'autorise aucune propriété supplémentaire en définissant le champ `additionalProperties` sur false. La compatibilité FORWARD (ASCENDANT) exige que les applications consommateur puissent lire les données produites avec le nouveau schéma à l'aide de la version précédente.

Si vous avez une proposition de version de schéma qui supprime la propriété de numéro de téléphone facultative, elle ne s'enregistrerait pas correctement avec la compatibilité FORWARD (ASCENDANT) lorsque la nouvelle version de schéma définit le champ `additionalProperties` sur true, à savoir autoriser toute propriété supplémentaire. Vos applications consommateur sur la version précédente ne seraient pas en mesure de lire les schémas proposés, car elles pourraient avoir la propriété de numéro de téléphone dans un type différent, par exemple une chaîne au lieu d'un numéro.

PROTOBUF

Par exemple, imaginez que vous avez une version de schéma définie par un message `Person` avec des champs `first name` (obligatoire), `last name` (obligatoire), `email` (facultatif) sous la syntaxe proto2.

Similaire aux scénarios AVRO, si vous disposez d'une nouvelle version de schéma qui ajoute un champ obligatoire ; par exemple `phone number`, l'enregistrement s'effectue correctement. La compatibilité FORWARD (ASCENDANT) exige que les applications consommateur puissent lire les données produites avec le nouveau schéma à l'aide de la version précédente.

Si vous avez une version de schéma proposée qui supprime le champ obligatoire `first name`, l'enregistrement ne s'effectue pas correctement avec la compatibilité ASCENDANTE. Les consommateurs sur la version précédente ne seraient pas en mesure de lire les schémas proposés, car le champ obligatoire `first name` est absent. Toutefois, si le champ `first name` était initialement facultatif, l'enregistrement du nouveau schéma proposé s'effectue correctement, car les consommateurs peuvent lire des données basées sur le nouveau schéma où le champ facultatif `first name` est absent.

Dans le cas d'un cas d'utilisation de gRPC, la suppression d'un service RPC ou d'une méthode RPC est une modification de compatibilité ascendante. Par exemple, imaginez que vous avez une version de schéma définie par un service RPC `MyService` avec deux méthodes RPC, `Foo` et `Bar`.

Si la version de schéma suivante supprime la méthode RPC existante nommée `Foo`, l'enregistrement s'effectue correctement en fonction de la compatibilité ASCENDANTE, car les consommateurs peuvent lire les données produites avec le nouveau schéma à l'aide de la version précédente. Si vous avez une nouvelle version de schéma proposée qui ajoute une méthode RPC `Baz`, l'enregistrement ne s'effectue pas correctement avec la compatibilité ASCENDANTE. Les consommateurs sur la version précédente ne seraient pas en mesure de lire les schémas proposés, car ils ne disposent pas de la méthode RPC `Baz`.

- FORWARD_ALL (ASCENDANT_TOUT) : ce choix de compatibilité permet aux applications consommateur de lire les données écrites par les applications producteur de tout nouveau schéma enregistré. Vous pouvez utiliser ce choix lorsque vous devez ajouter des champs ou supprimer des champs facultatifs, et vérifier la compatibilité avec à toutes les versions de schéma précédentes.
- FULL (COMPLET) : ce choix de compatibilité permet aux applications consommateur de lire les données écrites par les applications producteur en utilisant la version précédente ou suivante du schéma, mais pas les versions antérieures ou ultérieures. Vous pouvez utiliser ce choix pour vérifier la compatibilité avec la dernière version de schéma lorsque vous ajoutez ou supprimez des champs facultatifs.
- FULL_ALL (COMPLET_TOUT) : ce choix de compatibilité permet aux applications consommateur de lire les données écrites par les applications producteur utilisant toutes les versions de schéma précédentes. Vous pouvez utiliser ce choix pour vérifier la compatibilité par rapport à toutes les versions de schéma précédentes lorsque vous ajoutez ou supprimez des champs facultatifs.

Bibliothèques Serde en open source

AWS fournit des bibliothèques Serde open source comme framework pour la sérialisation et la désérialisation des données. La conception open source de ces bibliothèques permet aux applications et aux cadres open source communs de prendre en charge ces bibliothèques dans leurs projets.

Pour plus de détails sur le fonctionnement des bibliothèques Serde, veuillez consulter Fonctionnement du registre de schémas.

Quotas du registre des schémas

Les quotas, également appelés limites dans AWS, sont les valeurs maximales pour les ressources, les actions et les éléments de votre AWS compte. Voici des limites logicielles pour le registre de schémas dans AWS Glue.

Paires clé-valeur des métadonnées de version de schéma

Vous pouvez avoir jusqu'à 10 paires clé-valeur SchemaVersion par région. AWS

Vous pouvez afficher ou définir les paires de métadonnées clé-valeur à l'aide du QuerySchemaVersionMetadata action (Python : query_schema_version_metadata) ou. PutSchemaVersionMetadata action (Python : put_schema_version_metadata) APIs

Voici des limites matérielles pour le registre de schémas dans AWS Glue.

Registres

Vous pouvez avoir jusqu'à 100 registres par AWS région pour ce compte.

SchemaVersion

Vous pouvez avoir jusqu'à 10 000 versions de schéma par AWS région pour ce compte.

Chaque nouveau schéma crée une version de schéma, de sorte que vous pouvez théoriquement avoir jusqu'à 10 000 schémas par compte et par région, si chaque schéma ne dispose que d'une seule version.

Charges utiles de schéma

Il existe une limite de taille de 170 Ko pour les charges utiles de schéma.

Fonctionnement du registre de schémas

Cette section décrit le fonctionnement des processus de sérialisation et de désérialisation dans le registre de schémas.

1. Enregistrer un schéma : si le schéma n'existe pas encore dans le registre, le schéma peut être enregistré avec un nom de schéma égal au nom de la destination (par exemple, test_topic, test_stream, prod_firehose) ou le producteur peut fournir un nom personnalisé pour le schéma. Les producteurs peuvent également ajouter des paires clé-valeur au schéma sous forme de métadonnées, telles que source : MSK_Kafka_Topic_A, ou appliquer des balises aux schémas lors de la création du schéma. AWS Une fois qu'un schéma est enregistré, le registre de schémas renvoie l'ID de version de schéma au sérialiseur. Si le schéma existe, mais que le sérialiseur utilise une nouvelle version qui n'existe pas, le registre de schémas vérifie que le schéma fait référence à une règle de compatibilité pour s'assurer que la nouvelle version est compatible avant de l'enregistrer en tant que nouvelle version.

   Il existe deux méthodes d'enregistrement d'un schéma : l'enregistrement manuel et l'enregistrement automatique. Vous pouvez enregistrer un schéma manuellement via la console AWS Glue ou la CLI/le SDK.

   Lorsque l'enregistrement automatique est activé dans les paramètres du sérialiseur, l'enregistrement automatique du schéma est effectué. Si la valeur `REGISTRY_NAME` n'est pas

fournie dans les configurations du producteur, l'enregistrement automatique enregistrera alors la nouvelle version du schéma sous le registre par défaut (default-registry). Voir Installation de SerDe bibliothèques pour plus d'informations sur la spécification de la propriété d'enregistrement automatique.

2. Le sérialiseur valide les enregistrements de données par rapport au schéma : lorsque l'application produisant des données a enregistré son schéma, le sérialiseur du registre de schémas valide l'enregistrement produit par l'application structurée avec les champs et les types de données correspondant à un schéma enregistré. Si le schéma de l'enregistrement ne correspond pas à un schéma enregistré, le sérialiseur renvoie une exception et l'application ne parvient pas à livrer l'enregistrement à la destination.

   Si aucun schéma n'existe et si le nom du schéma n'est pas fourni via les configurations du producteur, le schéma est créé avec le même nom que le nom de la rubrique (s'il s'agit d'Apache Kafka ou d'Amazon MSK) ou le nom du flux (s'il s'agit de Kinesis Data Streams).

   Chaque enregistrement a une définition de schéma et des données. La définition du schéma est interrogée par rapport aux schémas et versions existants dans le registre de schémas.

   Par défaut, les producteurs mettent en cache les définitions des schémas et les versions IDs des schémas enregistrés. Si la définition de version de schéma d'un enregistrement ne correspond pas à ce qui est disponible dans le cache, le producteur tentera de valider le schéma avec le registre de schémas. Si la version du schéma est valide, son ID de version et sa définition seront alors mis en cache localement sur le producteur.

   Vous pouvez ajuster la période de mise en cache par défaut (24 heures) dans les propriétés facultatives du producteur à l'étape 3 de Installation de SerDe bibliothèques.

3. Sérialiser et livrer des enregistrements : si l'enregistrement est conforme au schéma, le sérialiseur décore chaque enregistrement avec l'ID de version du schéma, sérialise l'enregistrement en fonction du format de données sélectionné (AVRO, JSON, Protobuf ou autres formats prochainement disponibles), compresse l'enregistrement (configuration du producteur facultative) et le livre à la destination.

4. Les applications consommateur désérialisent les données : les applications consommateur qui lisent ces données utilisent la bibliothèque du désérialiseur du registre de schémas qui analyse l'ID de version du schéma à partir de la charge utile de l'enregistrement.

5. Le désérialiseur peut demander le schéma à partir du registre de schémas : si c'est la première fois que le désérialiseur a constaté des enregistrements avec un ID de version de schéma particulier, à l'aide de l'ID de version de schéma, il demandera le schéma à partir du registre de

schémas et mettra en cache le schéma localement sur l'application consommateur. Si le registre de schémas ne peut pas désérialiser l'enregistrement, l'application consommateur peut journaliser les données de l'enregistrement et continuer, mais aussi arrêter l'application.

6. Le désérialiseur utilise le schéma pour désérialiser l'enregistrement : lorsque le désérialiseur récupère l'ID de version de schéma auprès du registre de schémas, le désérialiseur décompresse l'enregistrement (si l'enregistrement envoyé par le producteur est compressé) et utilise le schéma pour désérialiser l'enregistrement. L'application traite à présent l'enregistrement.

> **Note**
>
> Chiffrement : vos clients communiquent avec le registre de schémas via des appels d'API qui chiffrent les données en transit à l'aide du chiffrement TLS sur HTTPS. Les schémas stockés dans le registre des schémas sont toujours chiffrés au repos à l'aide d'une clé gérée par le service AWS Key Management Service ()AWS KMS.

> **Note**
>
> Autorisation de l'utilisateur : le registre de schémas prend en charge les politiques IAM basées sur l'identité.

Démarrage avec le registre de schémas

Les sections suivantes fournissent une présentation et vous expliquent comment configurer et utiliser le registre de schémas. Pour plus d'informations sur les concepts et les composants du registre de schémas, consultez Registre de schémas AWS Glue.

Rubriques

- Installation de SerDe bibliothèques
- Intégration au registre de schémas AWS Glue
- Migration d'un registre de schémas tiers vers le registre de schémas AWS Glue

Installation de SerDe bibliothèques

Les SerDe bibliothèques fournissent un cadre pour la sérialisation et la désérialisation des données.

Vous allez installer le sérialiseur open source pour vos applications produisant des données (collectivement les « sérialiseurs »). Le sérialiseur gère la sérialisation, la compression et l'interaction avec le registre de schémas. Le sérialiseur extrait automatiquement le schéma d'un enregistrement en cours d'écriture vers une destination compatible avec le registre de schémas, telle qu'Amazon MSK. De même, vous allez installer le désérialiseur open source sur vos applications consommant des données.

Implémentation Java

> **Note**
>
> Prérequis : avant d'effectuer les étapes suivantes, vous devez avoir un Amazon Managed Streaming pour Apache Kafka (Amazon MSK) ou un cluster Apache Kafka en cours d'exécution. Vos applications producteur et consommateur doivent utiliser Java 8 ou une version supérieure.

Pour installer les bibliothèques sur les applications producteur et consommateur :

1. Dans les fichiers pom.xml des applications producteur et consommateur, ajoutez cette dépendance via le code ci-dessous :

   ```
   <dependency>
       <groupId>software.amazon.glue</groupId>
       <artifactId>schema-registry-serde</artifactId>
       <version>1.1.5</version>
   </dependency>
   ```

   Vous pouvez également cloner le référentiel Github du registre de schémas AWS Glue.

2. Configurez vos applications producteur avec les propriétés requises suivantes :

   ```
   props.put(ProducerConfig.KEY_SERIALIZER_CLASS_CONFIG,
    StringSerializer.class.getName()); // Can replace StringSerializer.class.getName())
    with any other key serializer that you may use
   props.put(ProducerConfig.VALUE_SERIALIZER_CLASS_CONFIG,
    GlueSchemaRegistryKafkaSerializer.class.getName());
   props.put(AWSSchemaRegistryConstants.AWS_REGION, "us-east-2");
   properties.put(AWSSchemaRegistryConstants.DATA_FORMAT, "JSON"); // OR "AVRO"
   ```

S'il n'existe aucun schéma existant, l'enregistrement automatique doit être activé (étape suivante). Si vous avez un schéma que vous souhaitez appliquer, remplacez « my-schema » par le nom de votre schéma. La valeur « registry-name » doit également être fournie si l'enregistrement automatique du schéma est désactivé. Si le schéma est créé sous la valeur « default-registry », le nom du registre peut être omis.

3. (Facultatif) Définissez l'une de ces propriétés de producteur facultatives. Pour une description détaillée des propriétés, consultez le ReadMe fichier.

```
props.put(AWSSchemaRegistryConstants.SCHEMA_AUTO_REGISTRATION_SETTING, "true"); // If
 not passed, uses "false"
props.put(AWSSchemaRegistryConstants.SCHEMA_NAME, "my-schema"); // If not passed,
 uses transport name (topic name in case of Kafka, or stream name in case of Kinesis
 Data Streams)
props.put(AWSSchemaRegistryConstants.REGISTRY_NAME, "my-registry"); // If not passed,
 uses "default-registry"
props.put(AWSSchemaRegistryConstants.CACHE_TIME_TO_LIVE_MILLIS, "86400000"); // If
 not passed, uses 86400000 (24 Hours)
props.put(AWSSchemaRegistryConstants.CACHE_SIZE, "10"); // default value is 200
props.put(AWSSchemaRegistryConstants.COMPATIBILITY_SETTING, Compatibility.FULL); //
 Pass a compatibility mode. If not passed, uses Compatibility.BACKWARD
props.put(AWSSchemaRegistryConstants.DESCRIPTION, "This registry is used for several
 purposes."); // If not passed, constructs a description
props.put(AWSSchemaRegistryConstants.COMPRESSION_TYPE,
 AWSSchemaRegistryConstants.COMPRESSION.ZLIB); // If not passed, records are sent
 uncompressed
```

L'enregistrement automatique enregistre la version du schéma sous le registre par défaut (« default-registry »). Si une valeur `SCHEMA_NAME` n'est pas spécifiée à l'étape précédente, le nom de la rubrique est alors déduit comme étant `SCHEMA_NAME`.

Pour de plus amples informations sur les modes de compatibilité, veuillez consulter Gestion des versions et compatibilité des schémas.

4. Configurez vos applications consommateur avec les propriétés requises suivantes :

```
props.put(ConsumerConfig.KEY_DESERIALIZER_CLASS_CONFIG,
 StringDeserializer.class.getName());
props.put(ConsumerConfig.VALUE_DESERIALIZER_CLASS_CONFIG,
 GlueSchemaRegistryKafkaDeserializer.class.getName());
props.put(AWSSchemaRegistryConstants.AWS_REGION, "us-east-2"); // Pass an Région AWS
```

```
props.put(AWSSchemaRegistryConstants.AVRO_RECORD_TYPE,
 AvroRecordType.GENERIC_RECORD.getName()); // Only required for AVRO data format
```

5. (Facultatif) Définissez ces propriétés d'application consommateur facultatives. Pour une description détaillée des propriétés, consultez le ReadMe fichier.

```
properties.put(AWSSchemaRegistryConstants.CACHE_TIME_TO_LIVE_MILLIS, "86400000"); //
 If not passed, uses 86400000
props.put(AWSSchemaRegistryConstants.CACHE_SIZE, "10"); // default value is 200
props.put(AWSSchemaRegistryConstants.SECONDARY_DESERIALIZER,
 "com.amazonaws.services.schemaregistry.deserializers.external.ThirdPartyDeserializer"); //
 For migration fall back scenario
```

Implémentation en C#

> **Note**
>
> Prérequis : avant d'effectuer les étapes suivantes, vous devez avoir un Amazon Managed Streaming pour Apache Kafka (Amazon MSK) ou un cluster Apache Kafka en cours d'exécution. Vos producteurs et consommateurs doivent utiliser .NET 8.0 ou une version ultérieure.

Installation

Pour les applications C#, installez le SerDe NuGet package AWS Glue Schema Registry en utilisant l'une des méthodes suivantes :

CLI .NET :

Utilisez la commande suivante pour installer le package :

```
dotnet add package Aws.Glue.SchemaRegistry --version 1.0.0-<rid>
```

où <rid> cela pourrait être`1.0.0-linux-x64`, `1.0.0-linux-musl-x64` ou `1.0.0-linux-arm64`

PackageReference (dans votre fichier .csproj) :

Ajoutez ce qui suit à votre fichier de projet :

```
<PackageReference Include="Aws.Glue.SchemaRegistry" Version="1.0.0-<rid>" />
```

où <rid> cela pourrait être`1.0.0-linux-x64`, `1.0.0-linux-musl-x64` ou `1.0.0-linux-arm64`

Configuration du fichier de configuration

Créez un fichier de propriétés de configuration (par exemple,`gsr-config.properties`) avec les paramètres requis :

Configuration minimale :

Voici un exemple de configuration minimale :

```
region=us-east-1
registry.name=default-registry
dataFormat=AVRO
schemaAutoRegistrationEnabled=true
```

Utilisation de la bibliothèque cliente C# Glue Schema pour Kafka SerDes

Exemple d'utilisation du sérialiseur :

L'exemple suivant montre comment utiliser le sérialiseur :

```
private static readonly string PROTOBUF_CONFIG_PATH = "<PATH_TO_CONFIG_FILE>";
var protobufSerializer = new GlueSchemaRegistryKafkaSerializer(PROTOBUF_CONFIG_PATH);
var serialized = protobufSerializer.Serialize(message, message.Descriptor.FullName);
// send serialized bytes to Kafka using producer.Produce(serialized)
```

Exemple d'utilisation du désérialiseur :

L'exemple suivant montre comment utiliser le désérialiseur :

```
private static readonly string PROTOBUF_CONFIG_PATH = "<PATH_TO_CONFIG_FILE>";
var dataConfig = new GlueSchemaRegistryDataFormatConfiguration(
    new Dictionary<string, dynamic>
    {
        {
            GlueSchemaRegistryConstants.ProtobufMessageDescriptor, message.Descriptor
        }
```

```
    }
);
var protobufDeserializer = new
 GlueSchemaRegistryKafkaDeserializer(PROTOBUF_CONFIG_PATH, dataConfig);

// read message from Kafka using serialized = consumer.Consume()
var deserializedObject = protobufDeserializer.Deserialize(message.Descriptor.FullName,
 serialized);
```

Utilisation de la bibliothèque cliente C# Glue Schema avec for KafkaFlow SerDes

Exemple d'utilisation du sérialiseur :

L'exemple suivant montre comment configurer KafkaFlow avec le sérialiseur :

```
services.AddKafka(kafka => kafka
    .UseConsoleLog()
    .AddCluster(cluster => cluster
        .WithBrokers(new[] { "localhost:9092" })
        .AddProducer<CustomerProducer>(producer => producer
            .DefaultTopic("customer-events")
            .AddMiddlewares(m => m

 .AddSerializer<GlueSchemaRegistryKafkaFlowProtobufSerializer<Customer>>(
                    () => new
 GlueSchemaRegistryKafkaFlowProtobufSerializer<Customer>("config/gsr-
config.properties")
                )
            )
        )
    )
);
```

Exemple d'utilisation du désérialiseur :

L'exemple suivant montre comment configurer KafkaFlow avec le désérialiseur :

```
.AddConsumer(consumer => consumer
    .Topic("customer-events")
    .WithGroupId("customer-group")
    .WithBufferSize(100)
    .WithWorkersCount(10)
    .AddMiddlewares(middlewares => middlewares
```

```
        .AddDeserializer<GlueSchemaRegistryKafkaFlowProtobufDeserializer<Customer>>(
            () => new
 GlueSchemaRegistryKafkaFlowProtobufDeserializer<Customer>("config/gsr-
config.properties")
        )
        .AddTypedHandlers(h => h.AddHandler<CustomerHandler>())
    )
)
```

Propriétés facultatives du producteur

Vous pouvez étendre votre fichier de configuration avec des propriétés facultatives supplémentaires :

```
# Auto-registration (if not passed, uses "false")
schemaAutoRegistrationEnabled=true

# Schema name (if not passed, uses topic name)
schema.name=my-schema

# Registry name (if not passed, uses "default-registry")
registry.name=my-registry

# Cache settings
cacheTimeToLiveMillis=86400000
cacheSize=200

# Compatibility mode (if not passed, uses BACKWARD)
compatibility=FULL

# Registry description
description=This registry is used for several purposes.

# Compression (if not passed, records are sent uncompressed)
compressionType=ZLIB
```

Formats de données pris en charge

Les implémentations Java et C# prennent en charge les mêmes formats de données :

- AVRO : format binaire Apache Avro
- JSON : format de schéma JSON
- PROTOBUF : format des tampons de protocole

Remarques

- Pour commencer à utiliser la bibliothèque, rendez-vous sur https://www.nuget. org/packages/AWS.Colle. SchemaRegistry
- Le code source est disponible à l'adresse suivante : https://github.com/awslabs/aws-glue-schema-registry

Création d'un registre

Vous pouvez utiliser le registre par défaut ou créer autant de nouveaux registres que nécessaire à l'aide de la AWS Glue console AWS Glue APIs or.

AWS Glue APIs

Vous pouvez suivre ces étapes pour effectuer cette tâche à l'aide du AWS Glue APIs.

Pour utiliser le AWS CLI registre des AWS Glue schémas APIs, assurez-vous de mettre à jour votre version AWS CLI à la plus récente.

Pour ajouter un nouveau registre, utilisez l'API CreateRegistry action (Python : create_registry). Spécifiez `RegistryName` comme nom du registre à créer, avec une longueur maximale de 255, contenant uniquement des lettres, des chiffres, des traits d'union, des traits de soulignement, le symbole dollar ou le dièse.

Spécifiez une valeur `Description` comme une chaîne d'une longueur maximale de 2 048 octets, correspondant au modèle de chaîne à plusieurs lignes d'adresse URI.

Vous pouvez également spécifier une ou plusieurs `Tags` pour votre registre, sous forme de tableau de mappage de paires clé-valeur.

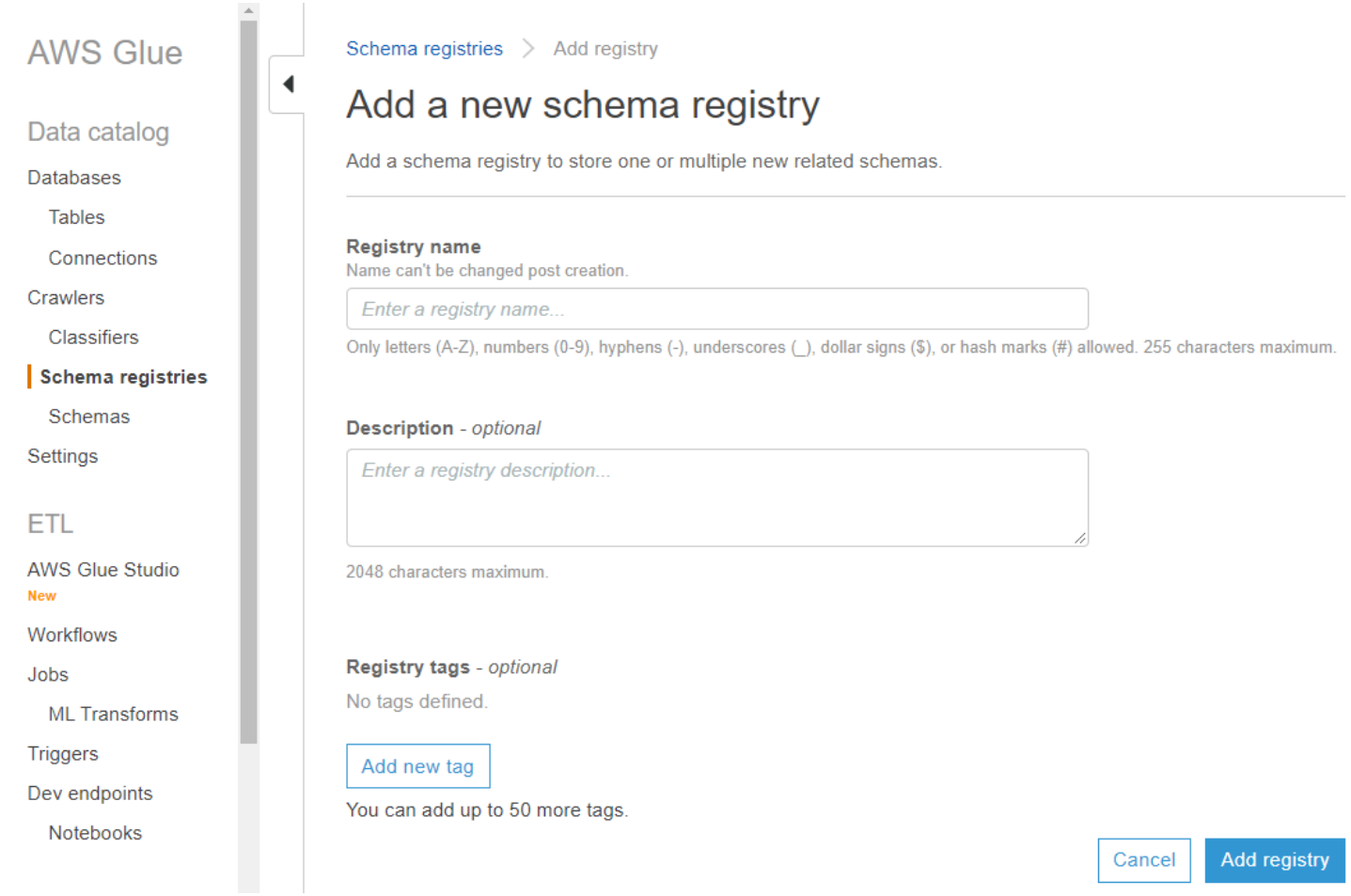
```
aws glue create-registry --registry-name registryName1 --description description
```

Lorsque votre registre est créé, il se voit attribuer un Amazon Resource Name (ARN), que vous pouvez consulter dans le `RegistryArn` de la réponse de l'API. Maintenant que vous avez créé un registre, créez un ou plusieurs schémas pour ce registre.

Console AWS Glue

Pour ajouter un nouveau registre dans la console AWS Glue :

1. Connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/.
2. Dans le panneau de navigation, sous Data catalog (Catalogue de données), choisissez Schema registries (Registres de schémas).
3. Choisissez Add registry (Ajouter un registre).
4. Saisissez un nom du registre à donner au registre, composé de lettres, de chiffres, de traits d'union et de traits de soulignement. Ce nom ne peut pas être modifié.
5. Saisissez une description (facultatif) pour le registre.
6. Si vous le souhaitez, appliquez une ou plusieurs balises à votre registre. Choisissez Add new tag (Ajouter une nouvelle balise) et spécifiez une clé de balise, et éventuellement une valeur de balise.
7. Choisissez Add registry (Ajouter un registre).



Lorsque votre registre est créé, un Amazon Resource Name (ARN) lui est attribué, que vous pouvez consulter en choisissant le registre dans la liste Schema registries (Registres de schémas). Maintenant que vous avez créé un registre, créez un ou plusieurs schémas pour ce registre.

Traiter un enregistrement spécifique (JAVA POJO) pour JSON

Vous pouvez utiliser un ancien objet Java simple (POJO) et transmettre l'objet en tant qu'enregistrement. Ceci est similaire à la notion d'enregistrement spécifique dans AVRO. Ils mbknor-jackson-jsonschemapeuvent générer un schéma JSON pour le POJO transmis. Cette bibliothèque peut également injecter des informations supplémentaires dans le schéma JSON.

La bibliothèque de registre de schémas AWS Glue utilise le champ « ClassName » injecté dans le schéma pour fournir un nom de classe entièrement classifié. Le champ « ClassName » est utilisé par le désérialiseur pour désérialiser dans un objet de cette classe.

```
 Example class :

@JsonSchemaDescription("This is a car")
@JsonSchemaTitle("Simple Car Schema")
@Builder
@AllArgsConstructor
@EqualsAndHashCode
// Fully qualified class name to be added to an additionally injected property
// called className for deserializer to determine which class to deserialize
// the bytes into
@JsonSchemaInject(
        strings = {@JsonSchemaString(path = "className",
                value =
 "com.amazonaws.services.schemaregistry.integrationtests.generators.Car")}
)
// List of annotations to help infer JSON Schema are defined by https://github.com/
mbknor/mbknor-jackson-jsonSchema
public class Car {
    @JsonProperty(required = true)
    private String make;

    @JsonProperty(required = true)
    private String model;

    @JsonSchemaDefault("true")
    @JsonProperty
    public boolean used;
```

```
    @JsonSchemaInject(ints = {@JsonSchemaInt(path = "multipleOf", value = 1000)})
    @Max(200000)
    @JsonProperty
    private int miles;

    @Min(2000)
    @JsonProperty
    private int year;

    @JsonProperty
    private Date purchaseDate;

    @JsonProperty
    @JsonFormat(shape = JsonFormat.Shape.NUMBER)
    private Date listedDate;

    @JsonProperty
    private String[] owners;

    @JsonProperty
    private Collection<Float> serviceChecks;

    // Empty constructor is required by Jackson to deserialize bytes
    // into an Object of this class
    public Car() {}
}
```

Création d'un schéma

Vous pouvez créer un schéma à l'aide de la console AWS Glue APIs ou de la AWS Glue console.

AWS Glue APIs

Vous pouvez suivre ces étapes pour effectuer cette tâche à l'aide du AWS Glue APIs.

Pour ajouter un nouveau schéma, utilisez l'API CreateSchema action (Python : créer_schéma).

Spécifiez une structure `RegistryId` pour indiquer un registre pour le schéma. Ou, omettez la valeur `RegistryId` pour utiliser le registre par défaut.

Spécifiez une valeur `SchemaName` composée de lettres, de chiffres, de traits d'union et de traits de soulignement, ainsi qu'une valeur `DataFormat` comme **`AVRO`** ou **`JSON`**. Une fois la valeur `DataFormat` définie sur un schéma, elle n'est pas modifiable.

Spécifiez un mode `Compatibility` :

- Backward (Descendant) (recommandé) — L'application consommateur peut lire à la fois la version actuelle et précédente.
- Backward all (Descendant tout) — L'application consommateur peut lire les versions actuelles et toutes les versions précédentes.
- Forward (Ascendant) — L'application consommateur peut lire à la fois la version actuelle et la version ultérieure.
- Forward all (Ascendant tout) — L'application consommateur peut lire à la fois les versions actuelles et les versions ultérieures.
- Full (Complet) — Combinaison de Forward (Ascendant) et de Backward (Descendant).
- Full all (Complet tout) — Combinaison de Backward all (Descendant tout) et de Forward all (Ascendant tout).
- None (Aucun) — Aucune vérification de compatibilité n'est effectuée.
- Disabled (Désactivé) — Empêche toute gestion des versions pour ce schéma.

Le cas échéant, spécifiez `Tags` pour votre schéma.

Spécifiez une valeur `SchemaDefinition` pour définir le schéma au format de données Avro, JSON ou Protobuf. Veuillez consulter les exemples.

Pour le format de données Avro :

```
aws glue create-schema --registry-id RegistryName="registryName1" --schema-name
 testschema --compatibility NONE --data-format AVRO --schema-definition "{\"type\":
 \"record\", \"name\": \"r1\", \"fields\": [ {\"name\": \"f1\", \"type\": \"int\"},
 {\"name\": \"f2\", \"type\": \"string\"} ]}"
```

```
aws glue create-schema --registry-id RegistryArn="arn:aws:glue:us-
east-2:901234567890:registry/registryName1" --schema-name testschema --compatibility
 NONE --data-format AVRO  --schema-definition "{\"type\": \"record\", \"name\": \"r1\",
 \"fields\": [ {\"name\": \"f1\", \"type\": \"int\"}, {\"name\": \"f2\", \"type\":
 \"string\"} ]}"
```

Pour le format de données JSON :

```
aws glue create-schema --registry-id RegistryName="registryName" --schema-name
 testSchemaJson --compatibility NONE --data-format JSON --schema-definition "{\"$schema
```

```
\": \"http://json-schema.org/draft-07/schema#\",\"type\":\"object\",\"properties\":
{\"f1\":{\"type\":\"string\"}}}"
```

```
aws glue create-schema --registry-id RegistryArn="arn:aws:glue:us-
east-2:901234567890:registry/registryName" --schema-name testSchemaJson --compatibility
 NONE --data-format JSON --schema-definition "{\"$schema\": \"http://json-schema.org/
draft-07/schema#\",\"type\":\"object\",\"properties\":{\"f1\":{\"type\":\"string\"}}}"
```

Pour le format de données Protobuf :

```
aws glue create-schema --registry-id RegistryName="registryName" --schema-name
 testSchemaProtobuf --compatibility NONE --data-format PROTOBUF --schema-definition
 "syntax = \"proto2\";package org.test;message Basic { optional int32 basic = 1;}"
```

```
aws glue create-schema --registry-id RegistryArn="arn:aws:glue:us-
east-2:901234567890:registry/registryName" --schema-name testSchemaProtobuf
 --compatibility NONE --data-format PROTOBUF --schema-definition "syntax =
 \"proto2\";package org.test;message Basic { optional int32 basic = 1;}"
```

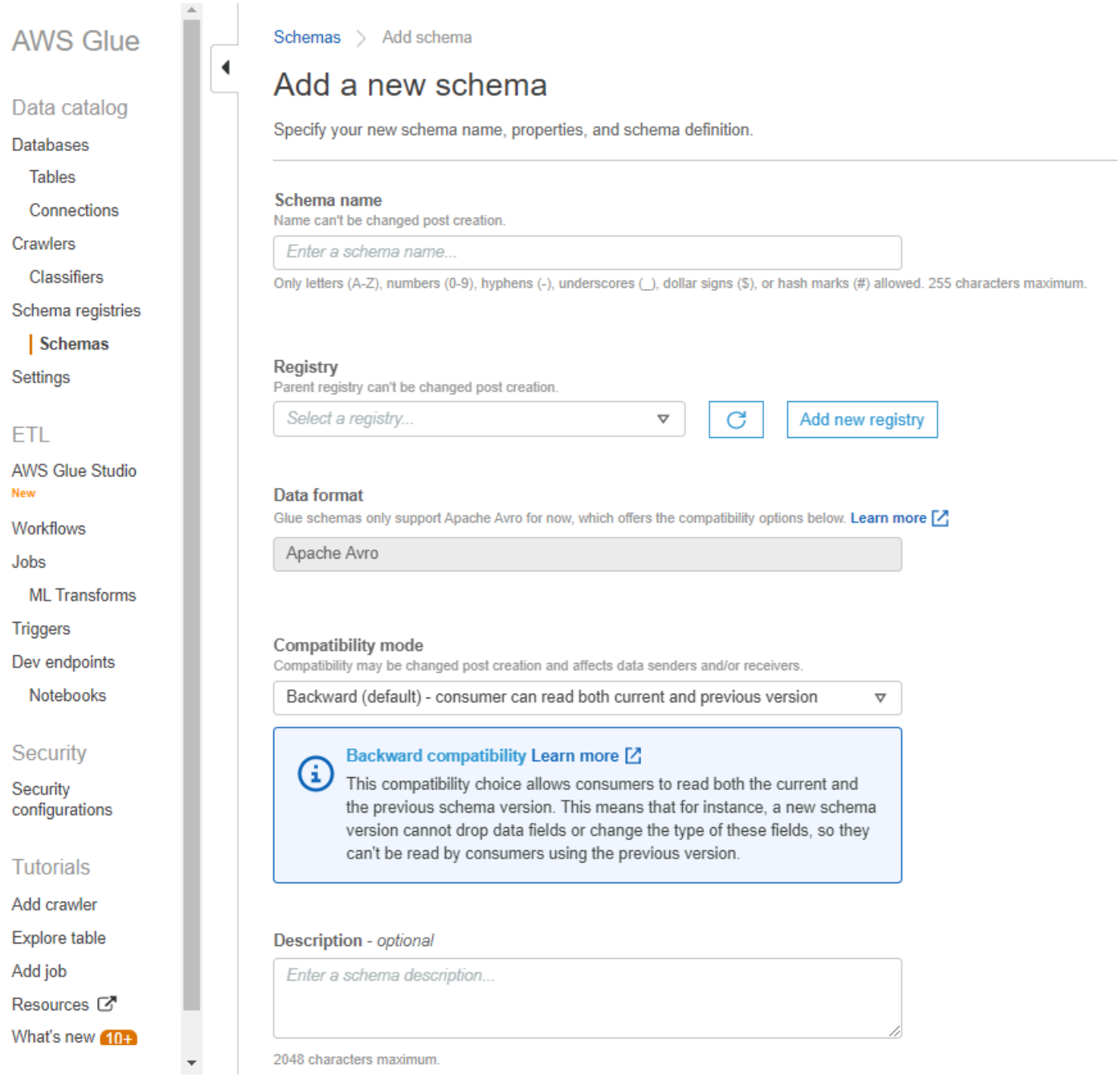
Console AWS Glue

Pour ajouter un nouveau schéma à l'aide de la console AWS Glue :

1. Connectez-vous à la console AWS de gestion et ouvrez-la à l'AWS Glueadresse https://console.aws.amazon.com/glue/.
2. Dans le panneau de navigation, sous Data catalog (Catalogue de données), choisissez Schemas (Schémas).
3. Choisissez Add schema (Ajouter un schéma).
4. Saisissez un nom de schéma, composé de lettres, de chiffres, de traits d'union, de traits de soulignement, de symboles dollar ou de dièses. Ce nom ne peut pas être modifié.
5. Cliquez sur l'onglet Registry (Registre) où le schéma sera stocké à partir du menu déroulant. Le registre parent ne peut pas être modifié après la création.
6. Quitter le format de données en tant qu'Apache Avro ou JSON. Ce format s'applique à toutes les versions de ce schéma.
7. Choisissez un mode de compatibilité.
   - Backward (Descendant) (recommandé) — Le récepteur peut lire à la fois les versions actuelles et précédentes.

- Backward All (Descendant tout) — Le récepteur peut lire les versions actuelles et toutes les versions précédentes.
- Forward (Ascendant) — L'expéditeur peut écrire à la fois les versions actuelles et précédentes.
- Forward all (Ascendant tout) — L'expéditeur peut écrire la version actuelle et toutes les versions précédentes.
- Full (Complet) — Combinaison de Forward (Ascendant) et de Backward (Descendant).
- Full All (Complet tout) — Combinaison de Backward All (Descendant tout) et de Forward All (Ascendant tout).
- None (Aucun) — Aucune vérification de compatibilité n'est effectuée.
- Disabled (Désactivé) — Empêche toute gestion des versions pour ce schéma.

8. Saisissez une description facultative pour le registre de 250 caractères maximum.

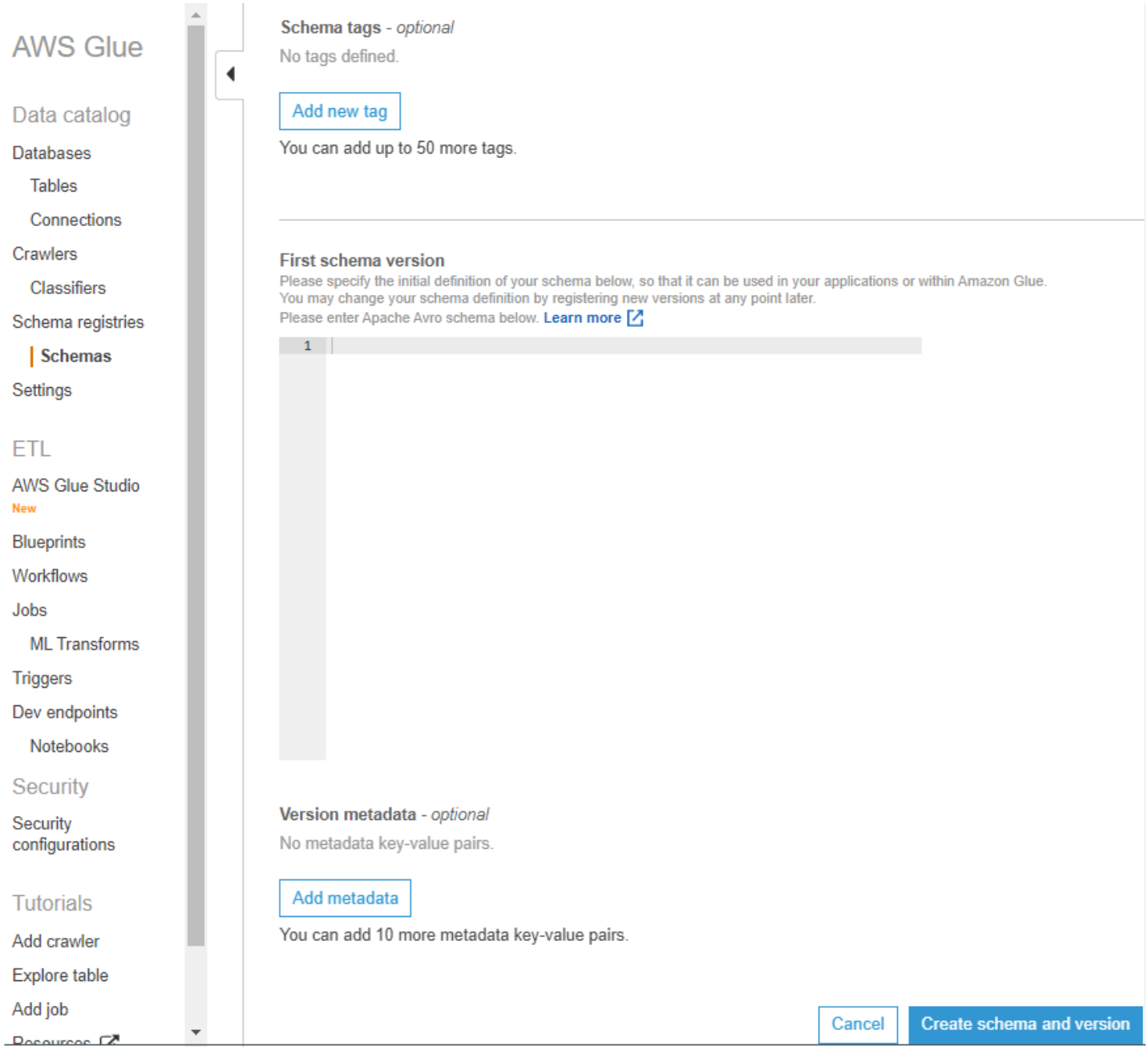


9. Si vous le souhaitez, appliquez une ou plusieurs balises à votre schéma. Choisissez Add new tag (Ajouter une nouvelle balise) et spécifiez une clé de balise, et éventuellement une valeur de balise.
10. Dans la case First schema version (Première version de schéma), saisissez ou collez votre schéma initial.

    Pour le format Avro, voir Utilisation du format de données Avro

    Pour le format JSON, voir Utilisation du format de données JSON

11. Choisissez éventuellement Add metadata (Ajouter des métadonnées) pour ajouter des métadonnées de version afin d'annoter ou de classer votre version de schéma.
12. Choisissez Create schema and version (Créer un schéma et une version).



Le schéma est créé et apparaît dans la liste sous Schemas (Schémas).

Utilisation du format de données Avro

Avro fournit des services de sérialisation et d'échange de données. Avro stocke la définition des données au format JSON, ce qui la rend facile à lire et à interpréter. Les données elles-mêmes sont stockées au format binaire.

Pour plus d'informations sur la définition d'un schéma Apache Avro, veuillez consulter la spécification d'Apache Avro.

Utilisation du format de données JSON

Les données peuvent être sérialisées au format JSON. Le format de schéma JSON définit la norme pour le format de schéma JSON.

Mise à jour d'un schéma ou d'un registre

Une fois créés, vous pouvez modifier vos schémas, vos versions de schéma ou votre registre.

Mise à jour d'un registre

Vous pouvez mettre à jour un registre à l'aide de la console AWS Glue APIs ou de la AWS Glue console. Le nom d'un registre existant ne peut pas être modifié. Vous pouvez modifier la description d'un registre.

AWS Glue APIs

Pour mettre à jour un registre existant, utilisez l'API UpdateRegistry action (Python : update_registry).

Spécifiez une structure `RegistryId` pour indiquer le registre que vous souhaitez mettre à jour. Transmettez une `Description` pour modifier la description d'un registre.

```
aws glue update-registry --description updatedDescription --registry-id
 RegistryArn="arn:aws:glue:us-east-2:901234567890:registry/registryName1"
```

Console AWS Glue

Pour mettre à jour un registre à l'aide de la console AWS Glue :

1. Connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/.

2. Dans le panneau de navigation, sous Data catalog (Catalogue de données), choisissez Schema registries (Registres de schémas).
3. Sélectionnez un registre dans la liste des registres, en cochant sa case.
4. Dans le menu Action, choisissez Edit registry (Modifier un registre).

Mise à jour d'un schéma

Vous pouvez mettre à jour le paramètre de description ou de compatibilité pour un schéma.

Pour mettre à jour un schéma existant, utilisez l'API UpdateSchema action (Python : update_schema).

Spécifiez une structure `SchemaId` pour indiquer le schéma que vous souhaitez mettre à jour. `VersionNumber` ou `Compatibility` doit être fourni.

Exemple de code 11 :

```
aws glue update-schema --description testDescription --schema-id
 SchemaName="testSchema1",RegistryName="registryName1" --schema-version-number
 LatestVersion=true --compatibility NONE
```

```
aws glue update-schema --description testDescription --schema-id
 SchemaArn="arn:aws:glue:us-east-2:901234567890:schema/registryName1/testSchema1" --
schema-version-number LatestVersion=true --compatibility NONE
```

Ajout d'une version de schéma

Lorsque vous ajoutez une version de schéma, vous devez comparer les versions pour vous assurer que le nouveau schéma sera accepté.

Pour ajouter une nouvelle version à un schéma existant, utilisez l'API RegisterSchemaVersion action (Python : register_schema_version).

Spécifiez une structure `SchemaId` pour indiquer le schéma pour lequel vous souhaitez ajouter une version, ainsi qu'une valeur `SchemaDefinition` pour définir le schéma.

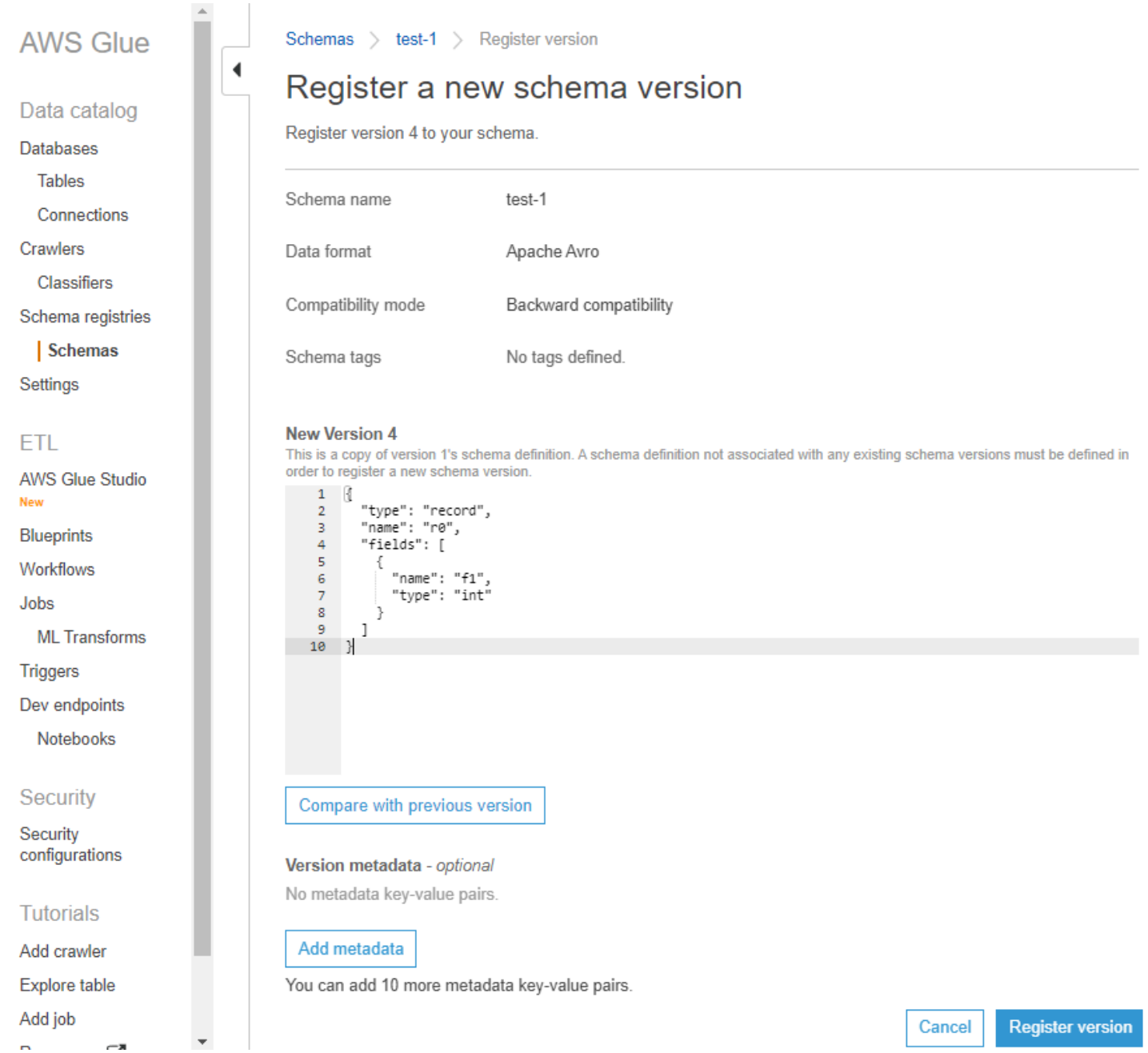
Exemple de code 12 :

```
aws glue register-schema-version --schema-definition "{\"type\": \"record\", \"name\":
 \"r1\", \"fields\": [ {\"name\": \"f1\", \"type\": \"int\"}, {\"name\": \"f2\", \"type
```

```
\": \"string\"} ]}" --schema-id SchemaArn="arn:aws:glue:us-east-1:901234567890:schema/
registryName/testschema"
```

```
aws glue register-schema-version --schema-definition "{\"type\": \"record\", \"name\":
 \"r1\", \"fields\": [ {\"name\": \"f1\", \"type\": \"int\"}, {\"name\": \"f2\", \"type
\": \"string\"} ]}" --schema-id SchemaName="testschema",RegistryName="testregistry"
```

1. Connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/.
2. Dans le panneau de navigation, sous Data catalog (Catalogue de données), choisissez Schemas (Schémas).
3. Sélectionnez le schéma dans la liste des schémas, en cochant sa case.
4. Sélectionnez un ou plusieurs schémas dans la liste, en cochant les cases.
5. Dans le menu Action, choisissez Register new version (Enregistrer une nouvelle version).
6. Dans New version (Nouvelle version), saisissez ou collez votre nouveau schéma.
7. Choisissez Compare with previous version (Comparer avec la version précédente) pour voir les différences avec la version de schéma précédente.
8. Choisissez éventuellement Add metadata (Ajouter des métadonnées) pour ajouter des métadonnées de version afin d'annoter ou de classer votre version de schéma. Saisissez une clé et une valeur facultative.
9. Choisissez Register version (Version de registre).



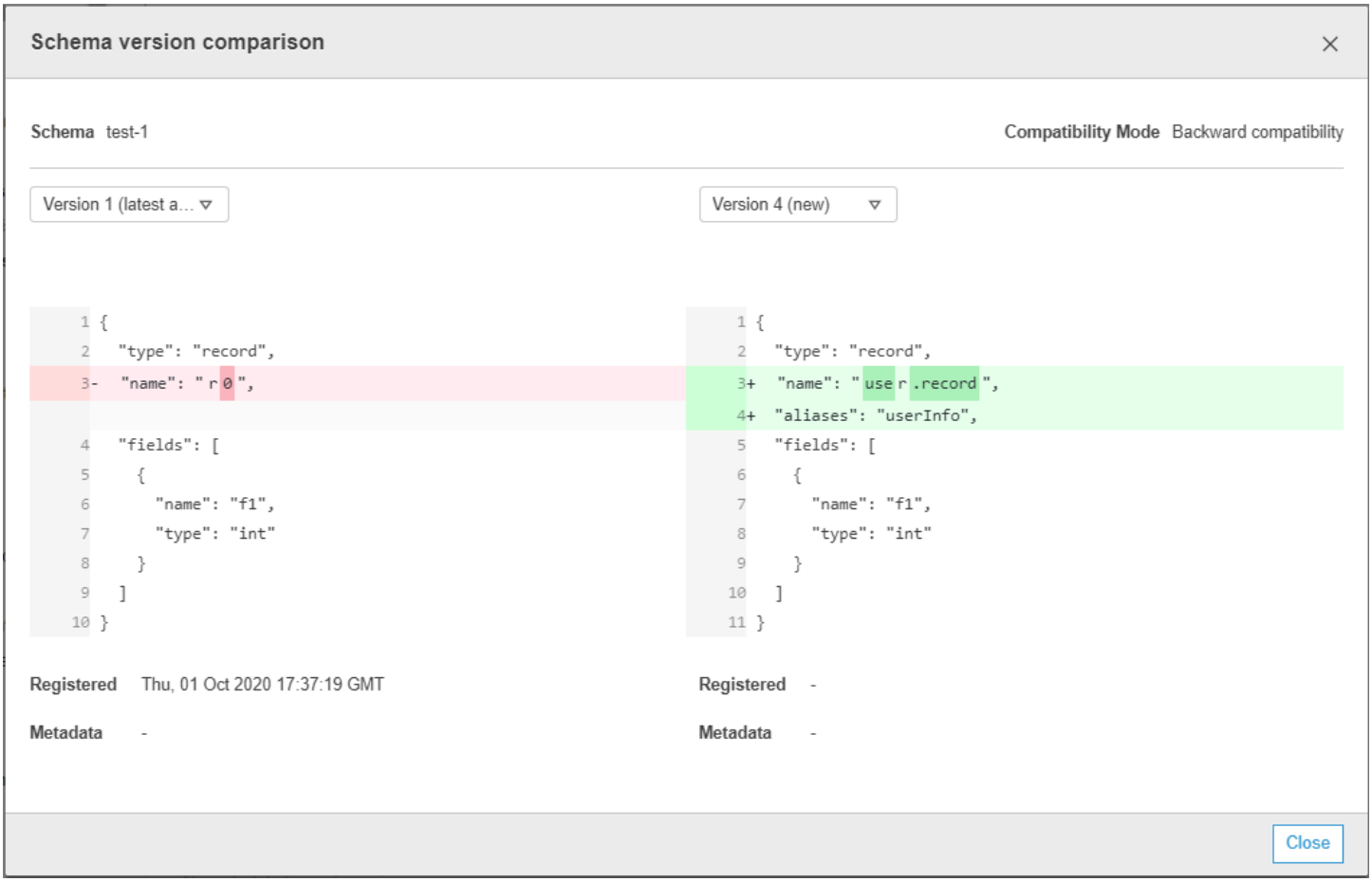
La version du ou des schémas s'affiche dans la liste des versions. Si la version a changé le mode de compatibilité, la version sera marquée comme point de contrôle.

Exemple de comparaison d'une version de schéma

Lorsque vous choisissez Compare with previous version (Comparer avec la version précédente), vous verrez les versions précédentes et les nouvelles versions affichées ensemble. Les informations modifiées seront mises en évidence comme suit :

- Jaune : indique les informations modifiées.
- Vert : indique le contenu ajouté à la dernière version.
- Rouge : indique le contenu supprimé dans la dernière version.

Vous pouvez également comparer les versions antérieures.



Suppression d'un schéma ou d'un registre

La suppression d'un schéma, d'une version de schéma ou d'un registre est une action permanente qui ne peut pas être annulée.

Suppression d'un schéma

Vous souhaiterez peut-être supprimer un schéma lorsqu'il ne sera plus utilisé dans un registre, à l'aide de l' AWS Management Console API ou de l'DeleteSchema action (Python : supprimer_schéma)API.

La suppression d'un ou plusieurs schémas est une action permanente qui ne peut pas être annulée. Assurez-vous que le ou les schémas ne sont plus nécessaires.

Pour supprimer un schéma du registre, appelez l'API DeleteSchema action (Python : supprimer_schéma), en spécifiant la structure `SchemaId` pour identifier le schéma.

Par exemple :

```
aws glue delete-schema --schema-id SchemaArn="arn:aws:glue:us-
east-2:901234567890:schema/registryName1/schemaname"
```

```
aws glue delete-schema --schema-id SchemaName="TestSchema6-
deleteschemabyname",RegistryName="default-registry"
```

Console AWS Glue

Pour supprimer un schéma de la console AWS Glue :

1. Connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/.
2. Dans le panneau de navigation, sous Data catalog (Catalogue de données), choisissez Schema registries (Registres de schémas).
3. Choisissez le registre qui contient votre schéma dans la liste des registres.
4. Sélectionnez un ou plusieurs schémas dans la liste, en cochant les cases.
5. Dans le menu Action, sélectionnez Delete schema (Supprimer un schéma).
6. Saisissez le texte **`Delete`** dans le champ pour confirmer la suppression.
7. Choisissez Delete (Supprimer).

Le ou les schémas que vous avez spécifiés sont supprimés du registre.

Suppression d'une version de schéma

Au fur et à mesure que les schémas s'accumulent dans le registre, vous souhaiterez peut-être supprimer les versions de schéma indésirables à l'aide de l' AWS Management Console API ou de l'DeleteSchemaVersions action (Python : delete_schema_versions)API. La suppression d'une ou de plusieurs versions de schéma est une action permanente qui ne peut pas être annulée. Assurez-vous que les versions de schéma ne sont plus nécessaires.

Lorsque vous supprimez des versions de schéma, veuillez tenir compte des contraintes suivantes :

- Vous ne pouvez pas supprimer une version comportant un point de contrôle.
- La gamme de versions contiguës ne peut pas dépasser 25.
- La version de schéma la plus récente ne doit pas être dans un état en attente.

Spécifiez la structure `SchemaId` pour identifier le schéma et spécifiez `Versions` comme une plage de versions à supprimer. Pour plus d'informations sur la spécification d'une version ou d'une plage de versions, veuillez consulter DeleteRegistry action (Python : supprimer_registre). Les versions de schéma que vous avez spécifiées sont supprimées du registre.

En appelant l'API ListSchemaVersions action (Python : list_schema_versions) après cet appel, vous obtiendrez la liste de l'état des versions supprimées.

Par exemple :

```
aws glue delete-schema-versions --schema-id
 SchemaName="TestSchema6",RegistryName="default-registry" --versions "1-1"
```

```
aws glue delete-schema-versions --schema-id SchemaArn="arn:aws:glue:us-
east-2:901234567890:schema/default-registry/TestSchema6-NON-Existent" --versions "1-1"
```

1. Connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/.
2. Dans le panneau de navigation, sous Data catalog (Catalogue de données), choisissez Schema registries (Registres de schémas).
3. Choisissez le registre qui contient votre schéma dans la liste des registres.
4. Sélectionnez un ou plusieurs schémas dans la liste, en cochant les cases.
5. Dans le menu Action, sélectionnez Delete schema (Supprimer un schéma).
6. Saisissez le texte **`Delete`** dans le champ pour confirmer la suppression.
7. Sélectionnez Delete (Supprimer).

Les versions de schéma que vous avez spécifiées sont supprimées du registre.

Suppression d'un registre

Vous pouvez supprimer un registre lorsque les schémas qu'il contient ne doivent plus être organisés sous ce registre. Vous devrez réaffecter ces schémas à un autre registre.

La suppression d'un ou de plusieurs registres est une action permanente qui ne peut pas être annulée. Assurez-vous que le ou les registres ne sont plus requis.

Le registre par défaut peut être supprimé à l'aide de la AWS CLI.

API AWS Glue

Pour supprimer l'intégralité du registre, y compris le schéma et toutes ses versions, appelez l'API DeleteRegistry action (Python : supprimer_registre). Spécifiez une structure `RegistryId` pour identifier le registre.

Par exemple :

```
aws glue delete-registry --registry-id RegistryArn="arn:aws:glue:us-
east-2:901234567890:registry/registryName1"
```

```
aws glue delete-registry --registry-id RegistryName="TestRegistry-deletebyname"
```

Pour obtenir le statut de l'opération de suppression, vous pouvez appeler l'API `GetRegistry` après l'appel asynchrone.

Console AWS Glue

Pour supprimer un registre de la console AWS Glue :

1. Connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/.
2. Dans le panneau de navigation, sous Data catalog (Catalogue de données), choisissez Schema registries (Registres de schémas).
3. Sélectionnez un registre dans la liste en cochant une case.
4. Dans le menu Action, choisissez Delete registry (Supprimer un registre).
5. Saisissez le texte **`Delete`** dans le champ pour confirmer la suppression.
6. Sélectionnez Delete (Supprimer).

Les registres que vous avez sélectionnés sont supprimés de AWS Glue.

Exemples d'IAM pour les sérialiseurs

> **Note**
>
> AWS les politiques gérées accordent les autorisations nécessaires pour les cas d'utilisation courants. Pour plus d'informations sur l'utilisation des politiques gérées pour gérer le registre de schéma, veuillez consulter AWS politiques gérées (prédéfinies) pour AWS Glue.

Pour les sérialiseurs, vous devez créer une politique minimale similaire à celle ci-dessous pour vous donner la possibilité de trouver le `schemaVersionId` pour une définition de schéma donnée. Notez que vous devez disposer des autorisations de lecture sur le registre afin de lire les schémas dans le registre. Vous pouvez limiter les registres qui peuvent être lus à l'aide de la clause `Resource`.

Exemple de code 13 :

```
{
    "Sid" : "GetSchemaByDefinition",
    "Effect" : "Allow",
    "Action" :
 [
        "glue:GetSchemaByDefinition"
    ],
        "Resource" : ["arn:aws:glue:us-east-2:012345678:registry/registryname-1",
                      "arn:aws:glue:us-east-2:012345678:schema/registryname-1/
schemaname-1",
                      "arn:aws:glue:us-east-2:012345678:schema/registryname-1/
schemaname-2"
                     ]
}
```

En outre, vous pouvez également autoriser les applications producteur à créer des schémas et versions en incluant les méthodes supplémentaires suivantes. Notez que vous devriez être en mesure d'inspecter le registre afin de vérifier add/remove/evolve les schémas qu'il contient. Vous pouvez limiter les registres qui peuvent être inspectés à l'aide de la clause `Resource`.

Exemple de code 14 :

```
{
```

```
    "Sid" : "RegisterSchemaWithMetadata",
    "Effect" : "Allow",
    "Action" :
 [
        "glue:GetSchemaByDefinition",
        "glue:CreateSchema",
        "glue:RegisterSchemaVersion",
        "glue:PutSchemaVersionMetadata",
    ],
    "Resource" : ["arn:aws:glue:aws-region:123456789012:registry/registryname-1",
                  "arn:aws:glue:aws-region:123456789012:schema/registryname-1/
schemaname-1",
                  "arn:aws:glue:aws-region:123456789012:schema/registryname-1/
schemaname-2"
                ]
}
```

Exemples d'IAM pour les désérialiseurs

Pour les désérialiseurs (côté application consommateur), vous devez créer une politique similaire à celle ci-dessous pour autoriser le désérialiseur à récupérer le schéma à partir du registre de schéma pour la désérialisation. Notez que vous devriez être en mesure d'inspecter le registre afin d'extraire les schémas qu'il contient.

Exemple de code 15 :

```
{
    "Sid" : "GetSchemaVersion",
    "Effect" : "Allow",
    "Action" :
 [
        "glue:GetSchemaVersion"
    ],
    "Resource" : ["*"]
}
```

Connectivité privée à l'aide de AWS PrivateLink

Vous pouvez l'utiliser AWS PrivateLink pour connecter le VPC de votre producteur de données AWS Glue en définissant un point de terminaison VPC d'interface pour. AWS Glue Lorsque vous utilisez un point de terminaison de VPC, la communication entre votre VPC et AWS Glue est gérée au sein du

réseau AWS . Pour plus d'informations, veuillez consulter Utilisation d'AWS Glue avec les points de terminaison d'un VPC.

Accès aux CloudWatch métriques d'Amazon

Les CloudWatch statistiques Amazon sont disponibles dans le cadre CloudWatch de l'offre gratuite. Vous pouvez accéder à ces métriques dans la CloudWatch console. Les indicateurs au niveau de l'API incluent CreateSchema (succès et latence) GetSchemaByDefinition, (succès et latence), GetSchemaVersion (succès et latence), RegisterSchemaVersion (succès et latence), PutSchemaVersionMetadata (succès et latence). Les métriques au niveau des ressources incluent le registre. ThrottledByLimit, SchemaVersion. ThrottledByLimit, SchemaVersion .Taille.

Exemple de CloudFormation modèle pour le registre de schémas

Voici un exemple de modèle pour créer des ressources de registre de schémas dans CloudFormation. Pour créer cette pile dans votre compte, copiez le modèle ci-dessus dans un fichier `SampleTemplate.yaml` et exécutez la commande suivante :

```
aws cloudformation create-stack --stack-name ABCSchemaRegistryStack --template-body
 "'cat SampleTemplate.yaml'"
```

Cet exemple utilise `AWS::Glue::Registry` pour créer un registre, `AWS::Glue::Schema` pour créer un schéma, `AWS::Glue::SchemaVersion` pour créer une version de schéma et `AWS::Glue::SchemaVersionMetadata` pour renseigner les métadonnées de version de schéma.

```
Description: "A sample CloudFormation template for creating Schema Registry resources."
Resources:
  ABCRegistry:
    Type: "AWS::Glue::Registry"
    Properties:
      Name: "ABCSchemaRegistry"
      Description: "ABC Corp. Schema Registry"
      Tags:
        Project: "Foo"
  ABCSchema:
    Type: "AWS::Glue::Schema"
    Properties:
      Registry:
        Arn: !Ref ABCRegistry
      Name: "TestSchema"
      Compatibility: "NONE"
      DataFormat: "AVRO"
```

```
      SchemaDefinition: >
        {"namespace":"foo.avro","type":"record","name":"user","fields":
[{"name":"name","type":"string"},{"name":"favorite_number","type":"int"}]}
      Tags:
        Project: "Foo"
  SecondSchemaVersion:
    Type: "AWS::Glue::SchemaVersion"
    Properties:
      Schema:
        SchemaArn: !Ref ABCSchema
      SchemaDefinition: >
        {"namespace":"foo.avro","type":"record","name":"user","fields":
[{"name":"status","type":"string", "default":"ON"}, {"name":"name","type":"string"},
{"name":"favorite_number","type":"int"}]}
  FirstSchemaVersionMetadata:
    Type: "AWS::Glue::SchemaVersionMetadata"
    Properties:
      SchemaVersionId: !GetAtt ABCSchema.InitialSchemaVersionId
      Key: "Application"
      Value: "Kinesis"
  SecondSchemaVersionMetadata:
    Type: "AWS::Glue::SchemaVersionMetadata"
    Properties:
      SchemaVersionId: !Ref SecondSchemaVersion
      Key: "Application"
      Value: "Kinesis"
```

Intégration au registre de schémas AWS Glue

Ces sections décrivent les intégrations au registre de schémas AWS Glue. Les exemples présentés dans cette section montrent un schéma au format de données AVRO. Pour d'autres exemples, notamment des schémas au format de données JSON, consultez les tests d'intégration et les ReadMe informations dans le référentiel open source du registre des AWS Glue schémas.

Rubriques

- Cas d'utilisation : connexion du registre de schémas à Amazon MSK ou à Apache Kafka
- Cas d'utilisation : intégration d'Amazon Kinesis Data Streams au registre de schémas AWS Glue
- Cas d'utilisation : Service géré Amazon pour Apache Flink
- Cas d'utilisation : intégration avec AWS Lambda
- Cas d'utilisation : AWS Glue Data Catalog

- Cas d'utilisation : streaming AWS Glue
- Cas d'utilisation : Apache Kafka Streams

Cas d'utilisation : connexion du registre de schémas à Amazon MSK ou à Apache Kafka

Supposons que vous écrivez des données sur une rubrique Apache Kafka, et que vous pouvez suivre ces étapes pour commencer.

1. Créez un cluster Amazon Managed Streaming pour Apache Kafka (Amazon MSK) ou Apache Kafka avec au moins un sujet. Si vous créez un cluster Amazon MSK, vous pouvez utiliser la AWS Management Console. Suivez les instructions ci-après : Mise en route avec Amazon MSK dans le Guide du développeur Amazon Managed Streaming for Apache Kafka.
2. Suivez l'étape Installation de SerDe bibliothèques ci-dessus.
3. Pour créer des registres de schéma, des schémas ou des versions de schéma, suivez les instructions sous la section Démarrage avec le registre de schémas de ce document.
4. Faites en sorte que vos producteurs et consommateurs utilisent le registre des schémas pour écrire et lire des enregistrements sur to/from le sujet Amazon MSK ou Apache Kafka. Des exemples de code producteur et consommateur se trouvent dans le ReadMe fichier des bibliothèques Serde. La bibliothèque du registre de schémas du producteur sérialisera automatiquement l'enregistrement et décorera l'enregistrement avec un ID de version de schéma.
5. Si le schéma de cet enregistrement a été saisi, ou si l'enregistrement automatique est activé, le schéma aura été enregistré dans le registre de schémas.
6. La lecture de l'application consommateur à partir de la rubrique Amazon MSK ou Apache Kafka, à l'aide de la bibliothèque du registre de schémas AWS Glue, recherche automatiquement le schéma à partir du registre de schémas.

Cas d'utilisation : intégration d'Amazon Kinesis Data Streams au registre de schémas AWS Glue

Cette intégration nécessite que vous ayez un flux de données Amazon Kinesis existant. Pour plus d'informations, veuillez consulter Présentation des Amazon Kinesis Data Streams dans le Guide du développeur Amazon Kinesis Data Streams.

Il existe deux façons d'interagir avec les données d'un flux de données Kinesis.

- Via les bibliothèques Kinesis Producer Library (KPL) et Kinesis Client Library (KCL) en Java. La prise en charge multilingue n'est pas fournie.

- Grâce au `PutRecordsPutRecord`, et aux `GetRecords` Kinesis Data APIs Streams disponibles dans le AWS SDK pour Java.

Si vous utilisez actuellement les KPL/KCL bibliothèques, nous vous recommandons de continuer à utiliser cette méthode. Il existe des versions KCL et KPL mises à jour avec le registre de schémas intégré, comme illustré dans les exemples. Sinon, vous pouvez utiliser l'exemple de code pour tirer parti du registre des AWS Glue schémas si vous utilisez APIs directement le KDS.

L'intégration du registre de schémas n'est disponible qu'avec KPL version 0.14.2 ou ultérieure et avec KCL version 2.3 ou ultérieure. L'intégration du registre de schémas avec le format de données JSON est disponible avec KPL version 0.14.8 ou ultérieure et avec KCL version 2.3.6 ou ultérieure.

Interagir avec les données à l'aide du kit SDK Kinesis V2

Cette section décrit l'interaction avec Kinesis à l'aide du kit SDK Kinesis V2

```
// Example JSON Record, you can construct a AVRO record also
private static final JsonDataWithSchema record =
 JsonDataWithSchema.builder(schemaString, payloadString);
private static final DataFormat dataFormat = DataFormat.JSON;

//Configurations for Schema Registry
GlueSchemaRegistryConfiguration gsrConfig = new GlueSchemaRegistryConfiguration("us-
east-1");

GlueSchemaRegistrySerializer glueSchemaRegistrySerializer =
        new GlueSchemaRegistrySerializerImpl(awsCredentialsProvider, gsrConfig);
GlueSchemaRegistryDataFormatSerializer dataFormatSerializer =
        new GlueSchemaRegistrySerializerFactory().getInstance(dataFormat, gsrConfig);

Schema gsrSchema =
        new Schema(dataFormatSerializer.getSchemaDefinition(record), dataFormat.name(),
 "MySchema");

byte[] serializedBytes = dataFormatSerializer.serialize(record);

byte[] gsrEncodedBytes = glueSchemaRegistrySerializer.encode(streamName, gsrSchema,
 serializedBytes);

PutRecordRequest putRecordRequest = PutRecordRequest.builder()
        .streamName(streamName)
        .partitionKey("partitionKey")
```

```
        .data(SdkBytes.fromByteArray(gsrEncodedBytes))
        .build();
shardId = kinesisClient.putRecord(putRecordRequest)
        .get()
        .shardId();

GlueSchemaRegistryDeserializer glueSchemaRegistryDeserializer = new
 GlueSchemaRegistryDeserializerImpl(awsCredentialsProvider, gsrConfig);

GlueSchemaRegistryDataFormatDeserializer gsrDataFormatDeserializer =
        glueSchemaRegistryDeserializerFactory.getInstance(dataFormat, gsrConfig);

GetShardIteratorRequest getShardIteratorRequest = GetShardIteratorRequest.builder()
        .streamName(streamName)
        .shardId(shardId)
        .shardIteratorType(ShardIteratorType.TRIM_HORIZON)
        .build();

String shardIterator = kinesisClient.getShardIterator(getShardIteratorRequest)
        .get()
        .shardIterator();

GetRecordsRequest getRecordRequest = GetRecordsRequest.builder()
        .shardIterator(shardIterator)
        .build();
GetRecordsResponse recordsResponse = kinesisClient.getRecords(getRecordRequest)
        .get();

List<Object> consumerRecords = new ArrayList<>();
List<Record> recordsFromKinesis = recordsResponse.records();

for (int i = 0; i < recordsFromKinesis.size(); i++) {
    byte[] consumedBytes = recordsFromKinesis.get(i)
            .data()
            .asByteArray();

    Schema gsrSchema = glueSchemaRegistryDeserializer.getSchema(consumedBytes);
    Object decodedRecord =
 gsrDataFormatDeserializer.deserialize(ByteBuffer.wrap(consumedBytes),

 gsrSchema.getSchemaDefinition());
    consumerRecords.add(decodedRecord);
}
```

Interaction avec les données à l'aide KPL/KCL des bibliothèques

Cette section décrit l'intégration de Kinesis Data Streams à Schema Registry à KPL/KCL l'aide des bibliothèques. Pour en savoir plus sur l'utilisation de KPL/KPC, veuillez consulter Développement d'applications producteur à l'aide de la bibliothèque producteur Amazon Kinesis dans le Guide du développeur Amazon Kinesis Data Streams..

Configuration du registre de schémas dans KPL

1. Définissez la définition de schéma pour les données, le format de données et le nom de schéma créés dans le registre de schémas AWS Glue.
2. Le cas échéant, configurez l'objet `GlueSchemaRegistryConfiguration`.
3. Transmettez l'objet du schéma à `addUserRecord API`.

```
private static final String SCHEMA_DEFINITION = "{"namespace": "example.avro",\n"
+ " "type": "record",\n"
+ " "name": "User",\n"
+ " "fields": [\n"
+ " {"name": "name", "type": "string"},\n"
+ " {"name": "favorite_number", "type": ["int", "null"]},\n"
+ " {"name": "favorite_color", "type": ["string", "null"]}\n"
+ " ]\n"
+ "}";

KinesisProducerConfiguration config = new KinesisProducerConfiguration();
config.setRegion("us-west-1")

//[Optional] configuration for Schema Registry.

GlueSchemaRegistryConfiguration schemaRegistryConfig =
new GlueSchemaRegistryConfiguration("us-west-1");

schemaRegistryConfig.setCompression(true);

config.setGlueSchemaRegistryConfiguration(schemaRegistryConfig);

///Optional configuration ends.

final KinesisProducer producer =
      new KinesisProducer(config);

final ByteBuffer data = getDataToSend();
```

```
com.amazonaws.services.schemaregistry.common.Schema gsrSchema =
    new Schema(SCHEMA_DEFINITION, DataFormat.AVRO.toString(), "demoSchema");

ListenableFuture<UserRecordResult> f = producer.addUserRecord(
config.getStreamName(), TIMESTAMP, Utils.randomExplicitHashKey(), data, gsrSchema);

private static ByteBuffer getDataToSend() {
      org.apache.avro.Schema avroSchema =
        new org.apache.avro.Schema.Parser().parse(SCHEMA_DEFINITION);

      GenericRecord user = new GenericData.Record(avroSchema);
      user.put("name", "Emily");
      user.put("favorite_number", 32);
      user.put("favorite_color", "green");

      ByteArrayOutputStream outBytes = new ByteArrayOutputStream();
      Encoder encoder = EncoderFactory.get().directBinaryEncoder(outBytes, null);
      new GenericDatumWriter<>(avroSchema).write(user, encoder);
      encoder.flush();
      return ByteBuffer.wrap(outBytes.toByteArray());
 }
```

Configuration de la bibliothèque client Kinesis

Vous allez développer votre application consommateur de la bibliothèque client Kinesis en Java. Pour de plus amples informations, veuillez consulter Développement d'une application consommateur de la bibliothèque client Kinesis en Java dans le Guide du développeur Amazon Kinesis Data Streams.

1. Créez une instance de `GlueSchemaRegistryDeserializer` en transmettant un objet `GlueSchemaRegistryConfiguration`.
2. Transmettez le `GlueSchemaRegistryDeserializer` à `retrievalConfig.glueSchemaRegistryDeserializer`.
3. Accédez au schéma des messages entrants en appelant `kinesisClientRecord.getSchema()`.

```
GlueSchemaRegistryConfiguration schemaRegistryConfig =
    new GlueSchemaRegistryConfiguration(this.region.toString());

 GlueSchemaRegistryDeserializer glueSchemaRegistryDeserializer =
```

```
   new
GlueSchemaRegistryDeserializerImpl(DefaultCredentialsProvider.builder().build(),
schemaRegistryConfig);

RetrievalConfig retrievalConfig =
configsBuilder.retrievalConfig().retrievalSpecificConfig(new
PollingConfig(streamName, kinesisClient));
retrievalConfig.glueSchemaRegistryDeserializer(glueSchemaRegistryDeserializer);

 Scheduler scheduler = new Scheduler(
           configsBuilder.checkpointConfig(),
           configsBuilder.coordinatorConfig(),
           configsBuilder.leaseManagementConfig(),
           configsBuilder.lifecycleConfig(),
           configsBuilder.metricsConfig(),
           configsBuilder.processorConfig(),
           retrievalConfig
       );

public void processRecords(ProcessRecordsInput processRecordsInput) {
           MDC.put(SHARD_ID_MDC_KEY, shardId);
           try {
               log.info("Processing {} record(s)",
               processRecordsInput.records().size());
               processRecordsInput.records()
               .forEach(
                   r ->
                       log.info("Processed record pk: {} -- Seq: {} : data {} with
schema: {}",
                       r.partitionKey(),
r.sequenceNumber(), recordToAvroObj(r).toString(), r.getSchema()));
           } catch (Throwable t) {
               log.error("Caught throwable while processing records. Aborting.");
               Runtime.getRuntime().halt(1);
           } finally {
               MDC.remove(SHARD_ID_MDC_KEY);
           }
}

private GenericRecord recordToAvroObj(KinesisClientRecord r) {
   byte[] data = new byte[r.data().remaining()];
   r.data().get(data, 0, data.length);
   org.apache.avro.Schema schema = new
org.apache.avro.Schema.Parser().parse(r.schema().getSchemaDefinition());
```

```
    DatumReader datumReader = new GenericDatumReader<>(schema);

    BinaryDecoder binaryDecoder = DecoderFactory.get().binaryDecoder(data, 0,
 data.length, null);
    return (GenericRecord) datumReader.read(null, binaryDecoder);
 }
```

Interaction avec les données à l'aide des Kinesis Data Streams APIs

Cette section décrit l'intégration de Kinesis Data Streams à Schema Registry à l'aide des Kinesis Data Streams. APIs

1. Mettez à jour ces dépendances Maven :

```
<dependencyManagement>
        <dependencies>
            <dependency>
                <groupId>com.amazonaws</groupId>
                <artifactId>aws-java-sdk-bom</artifactId>
                <version>1.11.884</version>
                <type>pom</type>
                <scope>import</scope>
            </dependency>
        </dependencies>
    </dependencyManagement>

    <dependencies>
        <dependency>
            <groupId>com.amazonaws</groupId>
            <artifactId>aws-java-sdk-kinesis</artifactId>
        </dependency>

        <dependency>
            <groupId>software.amazon.glue</groupId>
            <artifactId>schema-registry-serde</artifactId>
            <version>1.1.5</version>
        </dependency>

        <dependency>
            <groupId>com.fasterxml.jackson.dataformat</groupId>
            <artifactId>jackson-dataformat-cbor</artifactId>
            <version>2.11.3</version>
```

```
        </dependency>
    </dependencies>
```

2. Dans l'application producteur, ajoutez des informations d'en-tête de schéma à l'aide de l'API `PutRecords` ou `PutRecord`dans Kinesis Data Streams.

```
//The following lines add a Schema Header to the record
        com.amazonaws.services.schemaregistry.common.Schema awsSchema =
            new com.amazonaws.services.schemaregistry.common.Schema(schemaDefinition,
 DataFormat.AVRO.name(),
                schemaName);
        GlueSchemaRegistrySerializerImpl glueSchemaRegistrySerializer =
            new
 GlueSchemaRegistrySerializerImpl(DefaultCredentialsProvider.builder().build(), new
 GlueSchemaRegistryConfiguration(getConfigs()));
        byte[] recordWithSchemaHeader =
            glueSchemaRegistrySerializer.encode(streamName, awsSchema,
 recordAsBytes);
```

3. Dans l'application producteur, utilisez l'API `PutRecords` ou `PutRecord` pour placer l'enregistrement dans le flux de données.
4. Dans l'application consommateur, supprimez l'enregistrement de schéma de l'en-tête et sérialisez un enregistrement de schéma Avro.

```
//The following lines remove Schema Header from record
        GlueSchemaRegistryDeserializerImpl glueSchemaRegistryDeserializer =
            new
 GlueSchemaRegistryDeserializerImpl(DefaultCredentialsProvider.builder().build(),
 getConfigs());
        byte[] recordWithSchemaHeaderBytes = new
 byte[recordWithSchemaHeader.remaining()];
        recordWithSchemaHeader.get(recordWithSchemaHeaderBytes, 0,
 recordWithSchemaHeaderBytes.length);
        com.amazonaws.services.schemaregistry.common.Schema awsSchema =
            glueSchemaRegistryDeserializer.getSchema(recordWithSchemaHeaderBytes);
        byte[] record =
 glueSchemaRegistryDeserializer.getData(recordWithSchemaHeaderBytes);

        //The following lines serialize an AVRO schema record
        if (DataFormat.AVRO.name().equals(awsSchema.getDataFormat())) {
            Schema avroSchema = new
 org.apache.avro.Schema.Parser().parse(awsSchema.getSchemaDefinition());
```

```
            Object genericRecord = convertBytesToRecord(avroSchema, record);
            System.out.println(genericRecord);
        }
```

Interaction avec les données à l'aide des Kinesis Data Streams APIs

Voici un exemple de code pour utiliser le `PutRecords` et `GetRecords` APIs.

```
//Full sample code
import
 com.amazonaws.services.schemaregistry.deserializers.GlueSchemaRegistryDeserializerImpl;
import
 com.amazonaws.services.schemaregistry.serializers.GlueSchemaRegistrySerializerImpl;
import com.amazonaws.services.schemaregistry.utils.AVROUtils;
import com.amazonaws.services.schemaregistry.utils.AWSSchemaRegistryConstants;
import org.apache.avro.Schema;
import org.apache.avro.generic.GenericData;
import org.apache.avro.generic.GenericDatumReader;
import org.apache.avro.generic.GenericDatumWriter;
import org.apache.avro.generic.GenericRecord;
import org.apache.avro.io.Decoder;
import org.apache.avro.io.DecoderFactory;
import org.apache.avro.io.Encoder;
import org.apache.avro.io.EncoderFactory;
import software.amazon.awssdk.auth.credentials.DefaultCredentialsProvider;
import software.amazon.awssdk.services.glue.model.DataFormat;

import java.io.ByteArrayOutputStream;
import java.io.File;
import java.io.IOException;
import java.nio.ByteBuffer;
import java.util.Collections;
import java.util.HashMap;
import java.util.Map;


public class PutAndGetExampleWithEncodedData {
    static final String regionName = "us-east-2";
    static final String streamName = "testStream1";
    static final String schemaName = "User-Topic";
    static final String AVRO_USER_SCHEMA_FILE = "src/main/resources/user.avsc";
    KinesisApi kinesisApi = new KinesisApi();
```

```
    void runSampleForPutRecord() throws IOException {
        Object testRecord = getTestRecord();
        byte[] recordAsBytes = convertRecordToBytes(testRecord);
        String schemaDefinition =
AVROUtils.getInstance().getSchemaDefinition(testRecord);

        //The following lines add a Schema Header to a record
        com.amazonaws.services.schemaregistry.common.Schema awsSchema =
            new com.amazonaws.services.schemaregistry.common.Schema(schemaDefinition,
DataFormat.AVRO.name(),
                schemaName);
        GlueSchemaRegistrySerializerImpl glueSchemaRegistrySerializer =
            new
GlueSchemaRegistrySerializerImpl(DefaultCredentialsProvider.builder().build(), new
GlueSchemaRegistryConfiguration(regionName));
        byte[] recordWithSchemaHeader =
            glueSchemaRegistrySerializer.encode(streamName, awsSchema, recordAsBytes);

        //Use PutRecords api to pass a list of records
        kinesisApi.putRecords(Collections.singletonList(recordWithSchemaHeader),
streamName, regionName);

        //OR
        //Use PutRecord api to pass single record
        //kinesisApi.putRecord(recordWithSchemaHeader, streamName, regionName);
    }

    byte[] runSampleForGetRecord() throws IOException {
        ByteBuffer recordWithSchemaHeader = kinesisApi.getRecords(streamName,
regionName);

        //The following lines remove the schema registry header
        GlueSchemaRegistryDeserializerImpl glueSchemaRegistryDeserializer =
            new
GlueSchemaRegistryDeserializerImpl(DefaultCredentialsProvider.builder().build(), new
GlueSchemaRegistryConfiguration(regionName));
        byte[] recordWithSchemaHeaderBytes = new
byte[recordWithSchemaHeader.remaining()];
        recordWithSchemaHeader.get(recordWithSchemaHeaderBytes, 0,
recordWithSchemaHeaderBytes.length);

        com.amazonaws.services.schemaregistry.common.Schema awsSchema =
            glueSchemaRegistryDeserializer.getSchema(recordWithSchemaHeaderBytes);
```

```
        byte[] record =
glueSchemaRegistryDeserializer.getData(recordWithSchemaHeaderBytes);

        //The following lines serialize an AVRO schema record
        if (DataFormat.AVRO.name().equals(awsSchema.getDataFormat())) {
            Schema avroSchema = new
org.apache.avro.Schema.Parser().parse(awsSchema.getSchemaDefinition());
            Object genericRecord = convertBytesToRecord(avroSchema, record);
            System.out.println(genericRecord);
        }

        return record;
    }

    private byte[] convertRecordToBytes(final Object record) throws IOException {
        ByteArrayOutputStream recordAsBytes = new ByteArrayOutputStream();
        Encoder encoder = EncoderFactory.get().directBinaryEncoder(recordAsBytes,
null);
        GenericDatumWriter datumWriter = new
GenericDatumWriter<>(AVROUtils.getInstance().getSchema(record));
        datumWriter.write(record, encoder);
        encoder.flush();
        return recordAsBytes.toByteArray();
    }

    private GenericRecord convertBytesToRecord(Schema avroSchema, byte[] record) throws
IOException {
        final GenericDatumReader<GenericRecord> datumReader = new
GenericDatumReader<>(avroSchema);
        Decoder decoder = DecoderFactory.get().binaryDecoder(record, null);
        GenericRecord genericRecord = datumReader.read(null, decoder);
        return genericRecord;
    }

    private Map<String, String> getMetadata() {
        Map<String, String> metadata = new HashMap<>();
        metadata.put("event-source-1", "topic1");
        metadata.put("event-source-2", "topic2");
        metadata.put("event-source-3", "topic3");
        metadata.put("event-source-4", "topic4");
        metadata.put("event-source-5", "topic5");
        return metadata;
    }
```

```
    private GlueSchemaRegistryConfiguration getConfigs() {
        GlueSchemaRegistryConfiguration configs = new
 GlueSchemaRegistryConfiguration(regionName);
        configs.setSchemaName(schemaName);
        configs.setAutoRegistration(true);
        configs.setMetadata(getMetadata());
        return configs;
    }

    private Object getTestRecord() throws IOException {
        GenericRecord genericRecord;
        Schema.Parser parser = new Schema.Parser();
        Schema avroSchema = parser.parse(new File(AVRO_USER_SCHEMA_FILE));

        genericRecord = new GenericData.Record(avroSchema);
        genericRecord.put("name", "testName");
        genericRecord.put("favorite_number", 99);
        genericRecord.put("favorite_color", "red");

        return genericRecord;
    }
}
```

Cas d'utilisation : Service géré Amazon pour Apache Flink

Apache Flink est un cadre open source populaire et un moteur de traitement distribué pour les calculs avec état sur des flux de données sans limite et limités. Amazon Managed Service pour Apache Flink est un AWS service entièrement géré qui vous permet de créer et de gérer des applications Apache Flink pour traiter des données de streaming.

Apache Flink open source fournit un certain nombre de sources et de récepteurs. Par exemple, les sources de données prédéfinies incluent la lecture à partir de fichiers, de répertoires et de sockets, ainsi que l'ingestion de données à partir de collections et d'itérateurs. DataStream Les connecteurs Apache Flink fournissent du code permettant à Apache Flink de s'interfacer avec divers systèmes tiers, tels qu'Apache Kafka ou Kinesis, en tant que récepteurs de sources. and/or

Pour plus d'informations, veuillez consulter le Guide du développeur Amazon Kinesis Data Analytics.

Connecteur Kafka Apache Flink

Apache Flink fournit un connecteur de flux de données Apache Kafka pour lire et écrire des données sur des sujets Kafka avec des garanties en une seule fois. L'application consommateur Kafka de

Flink, `FlinkKafkaConsumer`, permet d'accéder à la lecture d'une ou de plusieurs rubriques Kafka. L'application producteur kafka d'Apache Flink, `FlinkKafkaProducer`, permet d'écrire un flux d'enregistrements à une ou plusieurs rubriques Kafka. Pour de plus amples informations, veuillez consulter Apache Kafka Connector.

Connecteur de flux Kinesis Apache Flink

Le connecteur de flux de données Kinesis vous permet d'accéder à Amazon Kinesis Data Streams. `FlinkKinesisConsumer`Il s'agit d'une source de données de streaming parallèle qui s'abonne à plusieurs flux Kinesis au sein de la même région de AWS service et peut gérer de manière transparente le repartitionnement des flux pendant l'exécution de la tâche. Chaque sous-tâche de l'application consommateur est responsable de la récupération des enregistrements de données à partir de plusieurs partitions Kinesis. Le nombre de partitions récupérées par chaque sous-tâche change au fur et à mesure que les partitions sont fermées et créées par Kinesis. Le `FlinkKinesisProducer` utilise Kinesis Producer Library (KPL) pour placer les données d'un flux Apache Flink dans un flux Kinesis. Pour plus d'informations, veuillez consulter Amazon Kinesis Streams Connector.

Pour plus d'informations, veuillez consulter le Référentiel GitHub de schémas AWS Glue.

Intégration à Apache Flink

La SerDes bibliothèque fournie avec Schema Registry s'intègre à Apache Flink. Pour travailler avec Apache Flink, vous devez implémenter des interfaces `SerializationSchema` et `DeserializationSchema` appelées `GlueSchemaRegistryAvroSerializationSchema` et `GlueSchemaRegistryAvroDeserializationSchema`, que vous pouvez brancher sur des connecteurs Apache Flink.

Ajout d'une dépendance du registre de schémas AWS Glue dans l'application Apache Flink

Pour configurer les dépendances d'intégration sur le registre de schémas AWS Glue dans l'application Apache Flink :

1. Ajoutez la dépendance au fichier `pom.xml`.

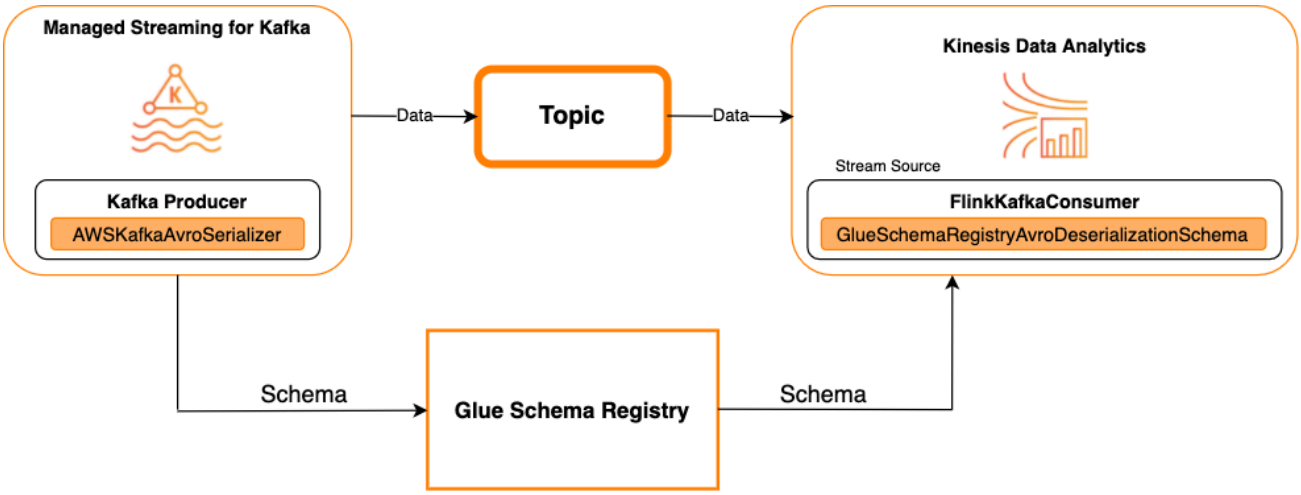
```
<dependency>
    <groupId>software.amazon.glue</groupId>
    <artifactId>schema-registry-flink-serde</artifactId>
    <version>1.0.0</version>
</dependency>
```

Intégration de Kafka ou Amazon MSK à Apache Flink

Vous pouvez utiliser Service géré pour Apache Flink pour Apache Flink, avec Kafka comme source ou Kafka comme collecteur.

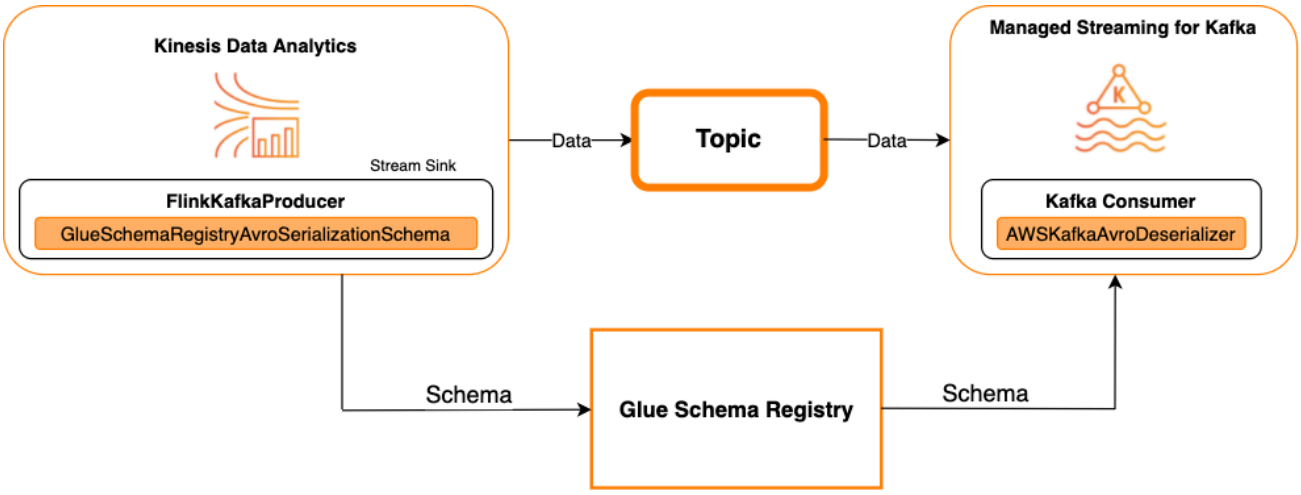
Kafka en tant que source

Le diagramme suivant illustre l'intégration de Kinesis Data Streams Kinesis à Service géré pour Apache Flink pour Apache Flink, avec Kafka en tant que source.



Kafka en tant que récepteur

Le diagramme suivant illustre l'intégration de Kinesis Data Streams Kinesis à Service géré pour Apache Flink pour Apache Flink, avec Kafka en tant que collecteur.



Pour intégrer Kafka (ou Amazon MSK) à Service géré pour Apache Flink pour Apache Flink, avec Kafka comme source ou Kafka comme collecteur, apportez les modifications de code ci-dessous. Ajoutez les blocs de code en gras à votre code respectif dans les sections analogues.

Si Kafka est la source, utilisez le code de désérialiseur (bloc 2). Si Kafka est le récepteur, utilisez le code de sérialiseur (bloc 3).

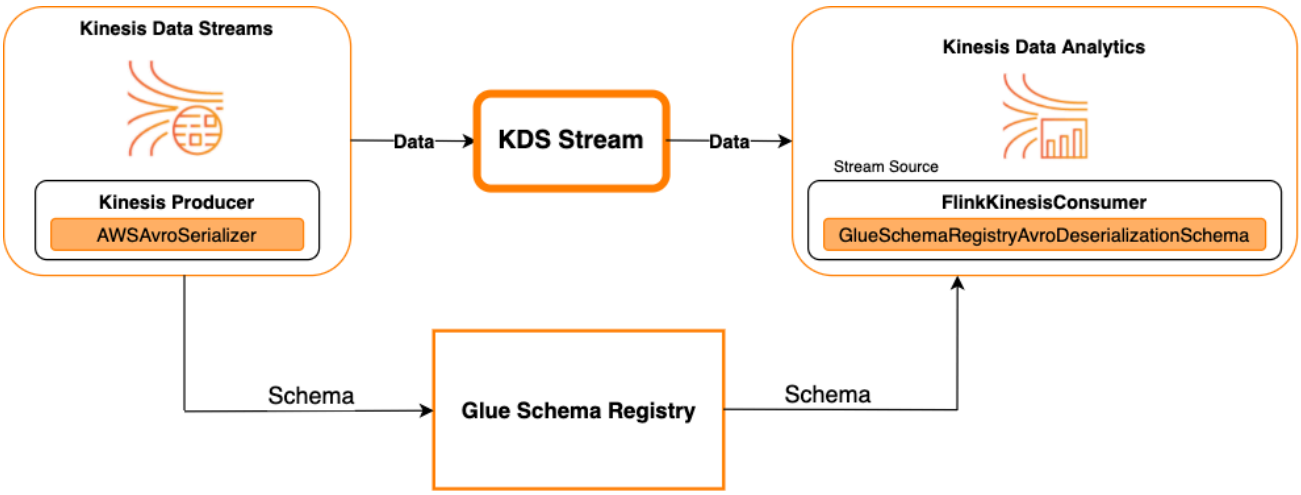
```
StreamExecutionEnvironment env = StreamExecutionEnvironment.getExecutionEnvironment();

String topic = "topic";
Properties properties = new Properties();
properties.setProperty("bootstrap.servers", "localhost:9092");
properties.setProperty("group.id", "test");

// block 1
Map<String, Object> configs = new HashMap<>();
configs.put(AWSSchemaRegistryConstants.AWS_REGION, "aws-region");
configs.put(AWSSchemaRegistryConstants.SCHEMA_AUTO_REGISTRATION_SETTING, true);
configs.put(AWSSchemaRegistryConstants.AVRO_RECORD_TYPE,
 AvroRecordType.GENERIC_RECORD.getName());

FlinkKafkaConsumer<GenericRecord> consumer = new FlinkKafkaConsumer<>(
    topic,
    // block 2
    GlueSchemaRegistryAvroDeserializationSchema.forGeneric(schema, configs),
    properties);

FlinkKafkaProducer<GenericRecord> producer = new FlinkKafkaProducer<>(
    topic,
    // block 3
    GlueSchemaRegistryAvroSerializationSchema.forGeneric(schema, topic, configs),
    properties);

DataStream<GenericRecord> stream = env.addSource(consumer);
stream.addSink(producer);
env.execute();
```

Intégration de Kinesis Data Streams à Apache Flink

Vous pouvez utiliser Service géré pour Apache Flink pour Apache Flink, avec Data Streams comme source ou comme collecteur.

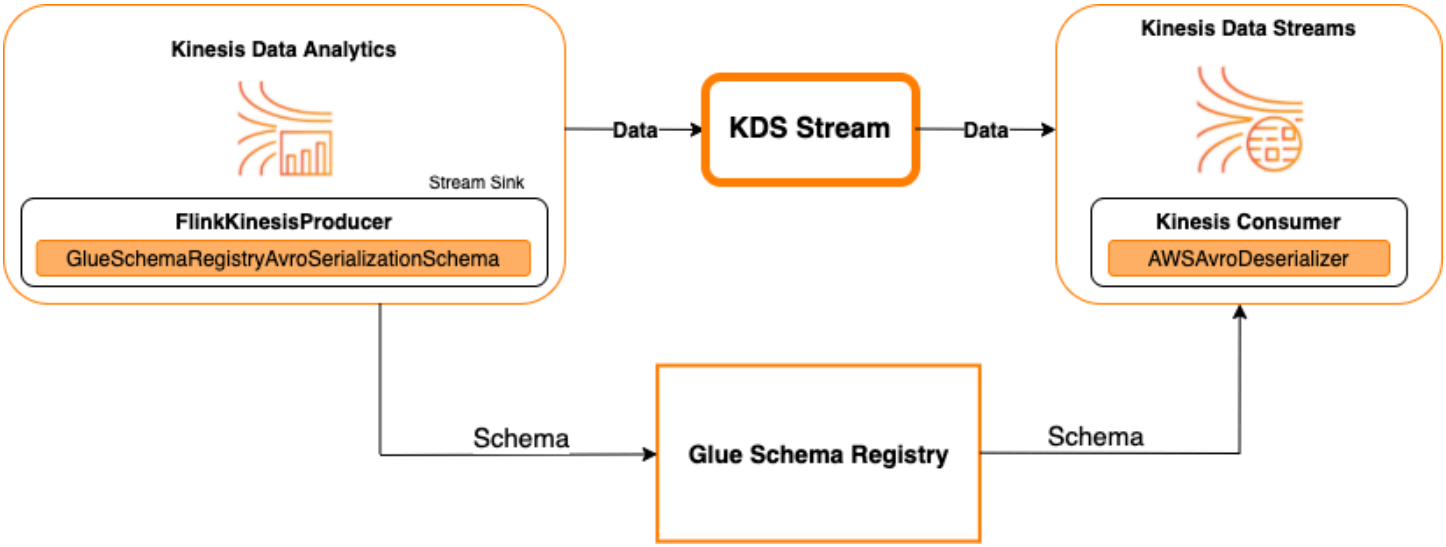
Kinesis Data Streams en tant que source

Le diagramme suivant illustre l'intégration de Kinesis Data Streams Kinesis à Service géré pour Apache Flink pour Apache Flink, avec Kinesis Data Streams en tant que source.



Kinesis Data Streams en tant que récepteur

Le diagramme suivant illustre l'intégration de Kinesis Data Streams Kinesis à Service géré pour Apache Flink pour Apache Flink, avec Kinesis Data Streams en tant que collecteur.



Pour intégrer Kinesis Data Streams à Service géré pour Apache Flink pour Apache Flink, avec Kinesis Data Streams en tant que source ou Kinesis Data Streams en tant que collecteur, apportez les modifications de code ci-dessous. Ajoutez les blocs de code en gras à votre code respectif dans les sections analogues.

Si Kinesis Data Streams est la source, utilisez le code de désérialiseur (bloc 2). Si Kinesis Data Streams est le récepteur, utilisez le code de sérialiseur (bloc 3).

```
StreamExecutionEnvironment env = StreamExecutionEnvironment.getExecutionEnvironment();
```

```
String streamName = "stream";
Properties consumerConfig = new Properties();
consumerConfig.put(AWSConfigConstants.AWS_REGION, "aws-region");
consumerConfig.put(AWSConfigConstants.AWS_ACCESS_KEY_ID, "aws_access_key_id");
consumerConfig.put(AWSConfigConstants.AWS_SECRET_ACCESS_KEY, "aws_secret_access_key");
consumerConfig.put(ConsumerConfigConstants.STREAM_INITIAL_POSITION, "LATEST");

// block 1
Map<String, Object> configs = new HashMap<>();
configs.put(AWSSchemaRegistryConstants.AWS_REGION, "aws-region");
configs.put(AWSSchemaRegistryConstants.SCHEMA_AUTO_REGISTRATION_SETTING, true);
configs.put(AWSSchemaRegistryConstants.AVRO_RECORD_TYPE,
 AvroRecordType.GENERIC_RECORD.getName());

FlinkKinesisConsumer<GenericRecord> consumer = new FlinkKinesisConsumer<>(
    streamName,
    // block 2
    GlueSchemaRegistryAvroDeserializationSchema.forGeneric(schema, configs),
    properties);

FlinkKinesisProducer<GenericRecord> producer = new FlinkKinesisProducer<>(
    // block 3
    GlueSchemaRegistryAvroSerializationSchema.forGeneric(schema, topic, configs),
    properties);
producer.setDefaultStream(streamName);
producer.setDefaultPartition("0");

DataStream<GenericRecord> stream = env.addSource(consumer);
stream.addSink(producer);
env.execute();
```

Cas d'utilisation : intégration avec AWS Lambda

Pour utiliser une AWS Lambda fonction en tant que consommateur Apache Kafka/Amazon MSK et désérialiser les messages codés en AVRO à l'aide de AWS Glue Schema Registry, visitez la page MSK Labs.

Cas d'utilisation : AWS Glue Data Catalog

Les tables AWS Glue prennent en charge des schémas que vous pouvez spécifier manuellement ou par référence au registre de schémas AWS Glue. Le registre de schéma s'intègre au catalogue de données pour vous permettre d'utiliser en option des schémas stockés dans le registre de schémas lors de la création ou de la mise à jour de tables AWS Glue ou partitions dans le catalogue de

données. Pour identifier une définition de schéma dans le registre de schémas, vous devez au minimum connaître l'ARN du schéma dont il fait partie. Une version de schéma d'un schéma, qui contient une définition de schéma, peut être référencée par son UUID ou son numéro de version. Il y a toujours une version de schéma, la « dernière » version, qui peut être recherchée sans connaître son numéro de version ou son UUID.

Lors de l'appel des opérations `CreateTable` ou `UpdateTable`, vous transmettrez une structure `TableInput` qui contient un `StorageDescriptor`, qui peut avoir une `SchemaReference` à un schéma existant dans le registre de schémas. De même, lorsque vous appelez le `GetTable` ou `GetPartition` APIs, la réponse peut contenir le schéma et le`SchemaReference`. Lorsqu'une table ou une partition a été créée à l'aide de références de schéma, le catalogue de données tente d'extraire le schéma pour cette référence de schéma. Dans le cas où il ne parvient pas à trouver le schéma dans le registre de schémas, il renvoie un schéma vide dans la réponse `GetTable` ; sinon la réponse aura à la fois le schéma et la référence du schéma.

Vous pouvez aussi effectuer les actions depuis la console AWS Glue.

Pour effectuer ces opérations et créer, mettre à jour ou afficher les informations de schéma, vous devez accorder un rôle IAM à l'appelant qui fournit les autorisations pour l'API `GetSchemaVersion`.

Ajout d'une table ou mise à jour du schéma pour une table

L'ajout d'une nouvelle table à partir d'un schéma existant lie la table à une version de schéma spécifique. Une fois que les nouvelles versions de schéma sont enregistrées, vous pouvez mettre à jour cette définition de table à partir de la page Affichage de la table dans la console AWS Glue ou à l'aide de l'API UpdateTable action (Python : update_table).

Ajout d'une table à partir d'un schéma existant

Vous pouvez créer une table AWS Glue à partir d'une version de schéma dans le registre à l'aide de la console AWS Glue ou de l'API `CreateTable`.

API AWS Glue

Lors de l'appel de l'API `CreateTable`, vous transmettrez une `TableInput` qui contient un `StorageDescriptor`, qui a une `SchemaReference` à un schéma existant dans le registre de schémas.

Console AWS Glue

Pour créer une table à partir de la console AWS Glue :

1. Connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/.
2. Dans le panneau de navigation, sous Data catalog (Catalogue de données), choisissez Tables.
3. Dans le menu Add tables (Ajouter des tables), choisissez Add table from existing schema (Ajouter une table à partir d'un schéma existant).
4. Configurez les propriétés de la table et le magasin de données selon le Guide du développeur AWS Glue.
5. Sur la page Choisir un schéma de Glue, sélectionnez le registre où réside le schéma.
6. Choisissez le nom du schéma et sélectionnez la version du schéma à appliquer.
7. Passez en revue la prévisualisation du schéma, puis choisissez Next (Suivant).
8. Vérifiez et créez la table.

Le schéma et la version appliqués à la table s'affichent dans la colonne Glue schema (Schéma Glue) dans la liste des tables. Vous pouvez afficher le tableau pour voir plus de détails.

Mise à jour du schéma pour une table

Lorsqu'une nouvelle version de schéma devient disponible, vous pouvez mettre à jour le schéma d'une table à l'aide de l'API UpdateTable action (Python : update_table) ou de la console AWS Glue.

> **Important**
>
> Lors de la mise à jour du schéma d'une table existante dotée d'un schéma AWS Glue spécifié manuellement, le nouveau schéma référencé dans le registre de schémas peut être incompatible. Cela peut entraîner l'échec de vos tâches.

API AWS Glue

Lors de l'appel de l'API `UpdateTable`, vous transmettrez une `TableInput` qui contient un `StorageDescriptor`, qui a une `SchemaReference` à un schéma existant dans le registre de schémas.

Console AWS Glue

Pour mettre à jour le schéma pour une table à partir de la console AWS Glue :

1. Connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/.
2. Dans le panneau de navigation, sous Data catalog (Catalogue de données), choisissez Tables.
3. Affichez la table à partir de la liste des tables.
4. Cliquez sur Update schema (Mettre à jour le schéma) dans la zone qui vous informe d'une nouvelle version.
5. Examinez les différences entre le schéma actuel et le nouveau.
6. Choisissez Show all schema differences (Afficher toutes les différences du schéma) pour plus de détails.
7. Choisissez Save table (Enregistrer la table) pour accepter la nouvelle version.

Cas d'utilisation : streaming AWS Glue

Le streaming AWS Glue consomme des données provenant de sources de streaming et effectue des opérations d'ETL avant d'écrire dans un récepteur de sortie. La source de streaming d'entrée peut être spécifiée à l'aide d'une table de données ou directement en spécifiant la configuration de la source.

Le streaming AWS Glue prend en charge une table de catalogue de données pour la source de streaming créée avec le schéma présent dans le registre de schémas AWS Glue. Vous pouvez créer un schéma dans le registre de schémas AWS Glue et créer une table AWS Glue avec une source de streaming utilisant ce schéma. Cette table AWS Glue peut être utilisée comme entrée dans une tâche de streaming AWS Glue pour désérialiser les données dans le flux d'entrée.

Remarque : lorsque le schéma du registre de schémas AWS Glue change, vous devez redémarrer la tâche de streaming AWS Glue afin de refléter les modifications dans le schéma.

Cas d'utilisation : Apache Kafka Streams

L'API Apache Kafka Streams est une bibliothèque client pour le traitement et l'analyse des données stockées dans Apache Kafka. Cette section décrit l'intégration d'Apache Kafka Streams au registre de schémas AWS Glue, qui vous permet de gérer et d'appliquer des schémas sur vos applications de streaming de données. Pour de plus amples informations sur Apache Kafka Streams, veuillez consulter Apache Kafka Streams.

Intégration aux SerDes bibliothèques

Il existe une classe `GlueSchemaRegistryKafkaStreamsSerde` avec laquelle vous pouvez configurer une application Streams.

Exemple de code d'application Kafka Streams

Pour utiliser le registre de schémas AWS Glue dans une application Apache Kafka Streams :

1. Configurez l'application Kafka Streams.

```
final Properties props = new Properties();
    props.put(StreamsConfig.APPLICATION_ID_CONFIG, "avro-streams");
    props.put(StreamsConfig.BOOTSTRAP_SERVERS_CONFIG, "localhost:9092");
    props.put(StreamsConfig.CACHE_MAX_BYTES_BUFFERING_CONFIG, 0);
    props.put(StreamsConfig.DEFAULT_KEY_SERDE_CLASS_CONFIG,
 Serdes.String().getClass().getName());
    props.put(StreamsConfig.DEFAULT_VALUE_SERDE_CLASS_CONFIG,
 AWSKafkaAvroSerDe.class.getName());
    props.put(ConsumerConfig.AUTO_OFFSET_RESET_CONFIG, "earliest");

    props.put(AWSSchemaRegistryConstants.AWS_REGION, "aws-region");
    props.put(AWSSchemaRegistryConstants.SCHEMA_AUTO_REGISTRATION_SETTING, true);
    props.put(AWSSchemaRegistryConstants.AVRO_RECORD_TYPE,
 AvroRecordType.GENERIC_RECORD.getName());
 props.put(AWSSchemaRegistryConstants.DATA_FORMAT, DataFormat.AVRO.name());
```

2. Créez un flux à partir de la rubrique avro-input.

```
StreamsBuilder builder = new StreamsBuilder();
final KStream<String, GenericRecord> source = builder.stream("avro-input");
```

3. Traitez les enregistrements de données (l'exemple filtre les enregistrements dont la valeur de favorite_color (couleur favorite) est pink (rose) ou dont la valeur de amount (montant) est 15).

```
final KStream<String, GenericRecord> result = source
    .filter((key, value) -
> !"pink".equals(String.valueOf(value.get("favorite_color"))));
    .filter((key, value) -> !"15.0".equals(String.valueOf(value.get("amount"))));
```

4. Réécrivez les résultats dans la rubrique avro-output.

```
result.to("avro-output");
```

5. Démarrez l'application Apache Kafka Streams.

```
KafkaStreams streams = new KafkaStreams(builder.build(), props);
streams.start();
```

Résultats de l'implémentation

Ces résultats montrent le processus de filtrage des enregistrements qui ont été filtrés à l'étape 3 sous la forme d'une valeur favorite_color (couleur favorite) « pink » (rose) ou d'une valeur de « 15.0 ».

Enregistrements avant le filtrage :

```
{"name": "Sansa", "favorite_number": 99, "favorite_color": "white"}
{"name": "Harry", "favorite_number": 10, "favorite_color": "black"}
{"name": "Hermione", "favorite_number": 1, "favorite_color": "red"}
{"name": "Ron", "favorite_number": 0, "favorite_color": "pink"}
{"name": "Jay", "favorite_number": 0, "favorite_color": "pink"}

{"id": "commute_1","amount": 3.5}
{"id": "grocery_1","amount": 25.5}
{"id": "entertainment_1","amount": 19.2}
{"id": "entertainment_2","amount": 105}
 {"id": "commute_1","amount": 15}
```

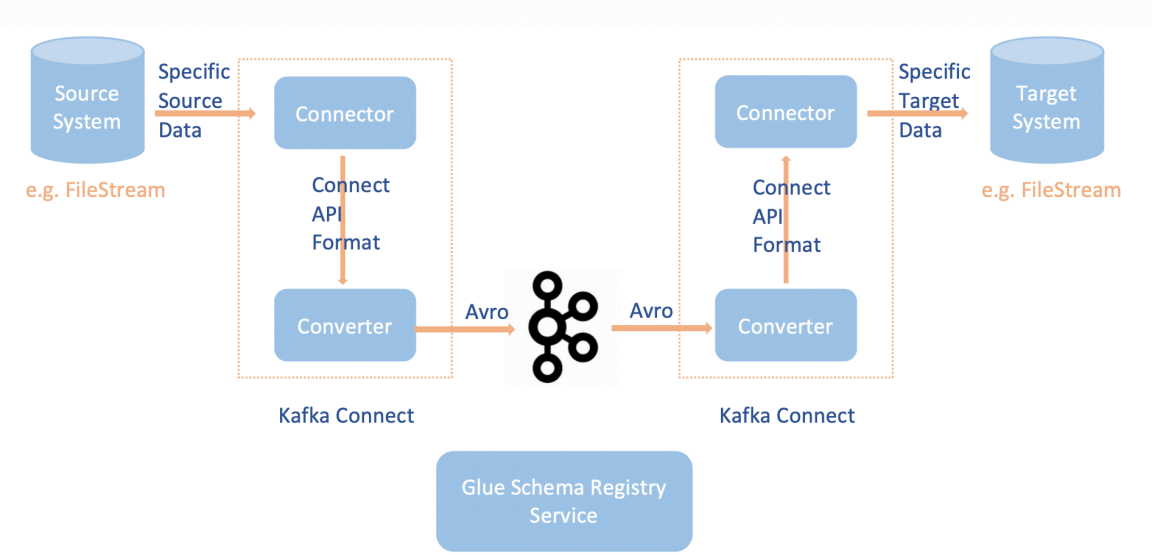
Enregistrements après filtrage :

```
{"name": "Sansa", "favorite_number": 99, "favorite_color": "white"}
{"name": "Harry", "favorite_number": 10, "favorite_color": "black"}
{"name": "Hermione", "favorite_number": 1, "favorite_color": "red"}
{"name": "Ron", "favorite_number": 0, "favorite_color": "pink"}

{"id": "commute_1","amount": 3.5}
{"id": "grocery_1","amount": 25.5}
{"id": "entertainment_1","amount": 19.2}
```

```
{"id": "entertainment_2","amount": 105}
```

Cas d'utilisation : Apache Kafka Connect

L'intégration d'Apache Kafka Connect au registre de schémas AWS Glue vous permet d'obtenir des informations de schéma à partir des connecteurs. Les convertisseurs Apache Kafka spécifient le format des données dans Apache Kafka et comment les traduire en données Apache Kafka Connect. Chaque utilisateur d'Apache Kafka Connect devra configurer ces convertisseurs en fonction du format souhaité pour leurs données lorsqu'elles sont chargées depuis ou stockés dans Apache Kafka. De cette façon, vous pouvez définir vos propres convertisseurs pour traduire les données Apache Kafka Connect dans le type utilisé dans le registre de schémas AWS Glue (par exemple : Avro) et utiliser notre sérialiseur pour enregistrer son schéma et effectuer la sérialisation. Ensuite, les convertisseurs peuvent également utiliser notre désérialiseur pour désérialiser les données reçues d'Apache Kafka et les convertir en données Apache Kafka Connect. Un exemple de diagramme de flux de travail est présenté ci-dessous.



1. Installez le projet `aws-glue-schema-registry` en clonant le référentiel Github pour le registre de schémas AWS Glue.

```
git clone git@github.com:awslabs/aws-glue-schema-registry.git
cd aws-glue-schema-registry
mvn clean install
mvn dependency:copy-dependencies
```

2. Si vous prévoyez d'utiliser Apache Kafka Connect en mode autonome, mettez à jour connect-standalone.properties à l'aide des instructions suivantes pour cette étape. Si vous prévoyez d'utiliser Apache Kafka Connect en mode distribué, mettez à jour le connect-avro-distributedfichier .properties en suivant les mêmes instructions.

   a. Ajoutez également ces propriétés au fichier de propriétés de connexion Apache Kafka :

   ```
   key.converter.region=aws-region
   value.converter.region=aws-region
   key.converter.schemaAutoRegistrationEnabled=true
   value.converter.schemaAutoRegistrationEnabled=true
   key.converter.avroRecordType=GENERIC_RECORD
   value.converter.avroRecordType=GENERIC_RECORD
   ```

   b. Ajoutez la commande ci-dessous à la section Mode de lancement sous kafka-run-class.sh :

   ```
   -cp $CLASSPATH:"<your AWS GlueSchema Registry base directory>/target/dependency/*"
   ```

3. Ajoutez la commande ci-dessous à la section Mode de lancement sous kafka-run-class.sh

   ```
   -cp $CLASSPATH:"<your AWS GlueSchema Registry base directory>/target/dependency/*"
   ```

   Elle doit ressembler à ce qui suit :

   ```
   # Launch mode
   if [ "x$DAEMON_MODE" = "xtrue" ]; then
     nohup "$JAVA" $KAFKA_HEAP_OPTS $KAFKA_JVM_PERFORMANCE_OPTS $KAFKA_GC_LOG_OPTS
    $KAFKA_JMX_OPTS $KAFKA_LOG4J_OPTS -cp $CLASSPATH:"/Users/johndoe/aws-glue-schema-
   registry/target/dependency/*" $KAFKA_OPTS "$@" > "$CONSOLE_OUTPUT_FILE" 2>&1 < /dev/
   null &
   else
     exec "$JAVA" $KAFKA_HEAP_OPTS $KAFKA_JVM_PERFORMANCE_OPTS $KAFKA_GC_LOG_OPTS
    $KAFKA_JMX_OPTS $KAFKA_LOG4J_OPTS -cp $CLASSPATH:"/Users/johndoe/aws-glue-schema-
   registry/target/dependency/*" $KAFKA_OPTS "$@"
   fi
   ```

4. Si vous utilisez bash, exécutez les commandes ci-dessous pour configurer votre CLASSPATH dans votre bash_profile. Pour tout autre shell, mettez à jour l'environnement en conséquence.

   ```
   echo 'export GSR_LIB_BASE_DIR=<>' >>~/.bash_profile
   echo 'export GSR_LIB_VERSION=1.0.0' >>~/.bash_profile
   ```

```
echo 'export KAFKA_HOME=<your Apache Kafka installation directory>' >>~/.bash_profile
echo 'export CLASSPATH=$CLASSPATH:$GSR_LIB_BASE_DIR/avro-kafkaconnect-converter/
target/schema-registry-kafkaconnect-converter-$GSR_LIB_VERSION.jar:$GSR_LIB_BASE_DIR/
common/target/schema-registry-common-$GSR_LIB_VERSION.jar:$GSR_LIB_BASE_DIR/
avro-serializer-deserializer/target/schema-registry-serde-$GSR_LIB_VERSION.jar'
 >>~/.bash_profile
source ~/.bash_profile
```

5. (Facultatif) Si vous souhaitez tester avec une source de fichier simple, clonez le connecteur de source de fichier.

```
git clone https://github.com/mmolimar/kafka-connect-fs.git
cd kafka-connect-fs/
```

   a. Sous la configuration du connecteur source, modifiez le format de données sur Avro, le lecteur de fichiers sur `AvroFileReader` et mettez à jour un exemple d'objet Avro à partir du chemin d'accès du fichier à partir duquel vous lisez. Par exemple :

```
vim config/kafka-connect-fs.properties
```

```
fs.uris=<path to a sample avro object>
policy.regexp=^.*\.avro$
file_reader.class=com.github.mmolimar.kafka.connect.fs.file.reader.AvroFileReader
```

   b. Installez le connecteur source.

```
mvn clean package
echo "export CLASSPATH=\$CLASSPATH:\"\$(find target/ -type f -name '*.jar'| grep
 '\-package' | tr '\n' ':')\"" >>~/.bash_profile
source ~/.bash_profile
```

   c. Mettez à jour les propriétés du récepteur sous *<your Apache Kafka installation directory>*`/config/connect-file-sink.properties`, mettez à jour le nom de rubrique et le nom de fichier de sortie.

```
file=<output file full path>
topics=<my topic>
```

6. Démarrez le connecteur source (dans cet exemple, il s'agit d'un connecteur source de fichier).

```
$KAFKA_HOME/bin/connect-standalone.sh $KAFKA_HOME/config/connect-
standalone.properties config/kafka-connect-fs.properties
```

7. Exécutez le connecteur du récepteur (dans cet exemple, il s'agit d'un connecteur de récepteur de fichiers).

   ```
   $KAFKA_HOME/bin/connect-standalone.sh $KAFKA_HOME/config/connect-
   standalone.properties $KAFKA_HOME/config/connect-file-sink.properties
   ```

   Pour un exemple d'utilisation de Kafka Connect, regardez le script run-local-tests .sh situé dans le dossier integration-tests du référentiel Github pour le registre des schémas. AWS Glue

Migration d'un registre de schémas tiers vers le registre de schémas AWS Glue

La migration d'un registre de schémas tiers vers le registre de schémas AWS Glue a une dépendance sur le registre de schéma tiers existant et actuel. S'il existe des enregistrements dans une rubrique Apache Kafka qui ont été envoyés à l'aide d'un registre de schémas tiers, les applications consommateur ont besoin du registre de schémas tiers pour désérialiser ces enregistrements. Le `AWSKafkaAvroDeserializer` permet de spécifier une classe de désérialiseur secondaire qui pointe vers le désérialiseur tiers et qui est utilisée pour désérialiser ces enregistrements.

Il existe deux critères pour le retrait d'un schéma tiers. Tout d'abord, le retrait ne peut avoir lieu qu'après que les enregistrements dans les rubriques Apache Kafka utilisant le registre de schéma tiers ne soient plus requis par et pour les applications consommateur. Deuxièmement, le retrait peut se produire suite au vieillissement des rubriques Apache Kafka, en fonction de la période de rétention spécifiée pour ces rubriques. Notez que si vous avez des rubriques qui ont une rétention infinie, vous pouvez toujours migrer vers le registre de schémas AWS Glue, mais vous ne serez pas en mesure de retirer le registre de schémas tiers. Comme solution de contournement, vous pouvez utiliser une application ou Mirror Maker 2 pour lire à partir de la rubrique actuelle et produire vers une nouvelle rubrique à l'aide du registre de schémas AWS Glue.

Pour migrer d'un registre de schémas tiers vers le registre de schémas AWS Glue :

1. Créez un registre dans le registre de schémas AWS Glue ou utilisez le registre par défaut.
2. Arrêtez l'application consommateur. Modifiez-le pour inclure le registre de schémas AWS Glue en tant que désérialiseur principal et le registre de schéma tiers en tant que secondaire.

- Définissez les propriétés de l'application consommateur. Dans cet exemple, la valeur secondary_deserializer est définie sur un autre désérialiseur. Le comportement est le suivant : l'application consommateur récupère les enregistrements auprès d'Amazon MSK et essaie d'abord d'utiliser le `AWSKafkaAvroDeserializer`. S'il n'est pas en mesure de lire l'octet magique qui contient l'ID de schéma Avro pour le registre de schémas AWS Glue, le `AWSKafkaAvroDeserializer` essaie ensuite d'utiliser la classe du désérialiseur fournie dans la valeure secondary_deserializer. Les propriétés spécifiques au désérialiseur secondaire doivent également être fournies dans les propriétés de l'application consommateur, telles que schema_registry_url_config et specific_avro_reader_config, comme indiqué ci-dessous.

```
consumerProps.setProperty(ConsumerConfig.KEY_DESERIALIZER_CLASS_CONFIG,
 StringDeserializer.class.getName());
consumerProps.setProperty(ConsumerConfig.VALUE_DESERIALIZER_CLASS_CONFIG,
 AWSKafkaAvroDeserializer.class.getName());
consumerProps.setProperty(AWSSchemaRegistryConstants.AWS_REGION,
 KafkaClickstreamConsumer.gsrRegion);
consumerProps.setProperty(AWSSchemaRegistryConstants.SECONDARY_DESERIALIZER,
 KafkaAvroDeserializer.class.getName());
consumerProps.setProperty(KafkaAvroDeserializerConfig.SCHEMA_REGISTRY_URL_CONFIG,
 "URL for third-party schema registry");
consumerProps.setProperty(KafkaAvroDeserializerConfig.SPECIFIC_AVRO_READER_CONFIG,
 "true");
```

3. Redémarrez l'application consommateur.
4. Arrêtez l'application producteur et pointez-la vers le registre de schémas AWS Glue.
   a. Définissez les propriétés du producteur. Dans cet exemple, le producteur utilisera le registre par défaut et enregistrera automatiquement les versions de schéma.

```
producerProps.setProperty(ProducerConfig.KEY_SERIALIZER_CLASS_CONFIG,
 StringSerializer.class.getName());
producerProps.setProperty(ProducerConfig.VALUE_SERIALIZER_CLASS_CONFIG,
 AWSKafkaAvroSerializer.class.getName());
producerProps.setProperty(AWSSchemaRegistryConstants.AWS_REGION, "us-east-2");
producerProps.setProperty(AWSSchemaRegistryConstants.AVRO_RECORD_TYPE,
 AvroRecordType.SPECIFIC_RECORD.getName());
producerProps.setProperty(AWSSchemaRegistryConstants.SCHEMA_AUTO_REGISTRATION_SETTING,
 "true");
```

5. (Facultatif) Déplacez manuellement les schémas et les versions de schéma existants du registre de schémas tiers actuel vers le registre de schémas AWS Glue, soit vers le registre par défaut dans le registre de schémas AWS Glue ou vers un registre spécifique autre que celui par défaut dans le registre de schémas AWS Glue. Cela peut être fait en exportant des schémas à partir de registres de schémas tiers au format JSON et en créant de nouveaux schémas dans le registre de AWS Glue schémas à l'aide du ou du AWS Management Console . AWS CLI

   Cette étape peut être importante si vous devez activer les contrôles de compatibilité avec les versions de schéma précédentes pour les versions de schéma nouvellement créées à l'aide du AWS CLI et AWS Management Console, ou lorsque les producteurs envoient des messages avec un nouveau schéma avec l'enregistrement automatique des versions de schéma activé.
6. Démarrez l'application producteur.

Connexion aux données

Une AWS Glue connexion est un objet de catalogue de données qui stocke les informations d'identification de connexion, les chaînes d'URI, les informations du cloud privé virtuel (VPC), etc. pour un magasin de données spécifique. AWS Glue les robots d'exploration, les jobs et les terminaux de développement utilisent des connexions pour accéder à certains types de magasins de données. Vous pouvez utiliser des connexions pour les sources et les cibles, mais aussi réutiliser la même connexion pour plusieurs tâches de crawler ou extraction, transformation et chargement (ETL).

La dernière version du schéma de AWS Glue connexions fournit un moyen unifié de gérer les connexions de données entre les AWS services et les applications, tels que AWS Glue Amazon Athena, et Amazon SageMaker AI Unified Studio.

Présentation de l'utilisation des connecteurs et des connexions

Une connexion contient les propriétés requises pour se connecter à un magasin de données particulier. Lorsque vous créez une connexion, elle est stockée dans AWS Glue Data Catalog. Vous sélectionnez un connecteur, puis créez une connexion basée sur celui-ci.

Vous pouvez vous abonner à des connecteurs pour les magasins de données non pris en charge de manière native AWS Marketplace, puis utiliser ces connecteurs lorsque vous créez des connexions. Les développeurs peuvent également créer leurs propres connecteurs et vous pouvez les utiliser lors de la création de connexions.

> **Note**
>
> Les connexions créées à l'aide de AWS Marketplace connecteurs personnalisés ou de connecteurs AWS Glue Studio apparaissent dans la AWS Glue console avec le type défini sur`UNKNOWN`.

Les étapes suivantes décrivent le processus global d'utilisation des connecteurs dans AWS Glue Studio :

1. Abonnez-vous à un connecteur dans AWS Marketplace, ou développez votre propre connecteur et téléchargez-le sur AWS Glue Studio. Pour de plus amples informations, veuillez consulter Ajouter des connecteurs àAWS Glue Studio.

2. Vérifiez les informations d'utilisation du connecteur. Vous pouvez trouver ces informations sous l'onglet Usage (Utilisation) de la page produit du connecteur. Par exemple, si vous cliquez sur l'onglet Utilisation de cette page produit, AWS Glue Connector for Google BigQuery, vous pouvez voir dans la section Ressources supplémentaires un lien vers un blog sur l'utilisation de ce connecteur.
3. Créez une connexion. Vous sélectionnez le connecteur à utiliser et fournissez des informations supplémentaires pour la connexion, telles que les informations d'identification de connexion, les chaînes d'URI et les informations de Virtual Private Cloud (VPC). Pour de plus amples informations, veuillez consulter Création de connexions pour les connecteurs.
4. Créez un rôle IAM pour votre tâche. La tâche dispose des autorisations du rôle IAM que vous spécifiez quand vous la créez. Ce rôle IAM doit disposer des autorisations nécessaires pour s'authentifier, extraire des données et écrire des données dans vos magasins de données.
5. Créez une tâche ETL et configurez les propriétés de la source de données pour cette tâche. Fournissez les options de connexion et les informations d'authentification comme indiqué par le fournisseur de connecteur personnalisé. Pour de plus amples informations, veuillez consulter Création de tâches avec des connecteurs personnalisés.
6. Personnalisez votre tâche ETL en ajoutant des transformations ou des magasins de données supplémentaires, comme décrit dans Démarrage de tâches ETL visuelles dans AWS Glue Studio.
7. Si vous utilisez un connecteur pour la cible de données, configurez les propriétés de la cible de données pour votre tâche ETL. Fournissez les options de connexion et les informations d'authentification comme indiqué par le fournisseur de connecteur personnalisé. Pour de plus amples informations, veuillez consulter the section called “Création de tâches avec des connecteurs personnalisés”.
8. Personnalisez l'environnement d'exécution de la tâche en configurant ses propriétés, comme décrit dans Modifier les propriétés de tâche.
9. Exécutez la tâche.

Connexions unifiées

AWS a récemment introduit une nouvelle fonctionnalité appelée « SageMaker LakeHouse Connexions » ou « Connexions AWS Glue unifiées ». Cette fonctionnalité vous permet de créer des connexions qui peuvent être utilisées par plusieurs AWS services, tels que AWS Glue et Amazon Athena. Lorsque vous créez une source de données dans Amazon Athena, vous remarquerez une section qui fait référence aux entrées de AWS Glue connexion. Dans ce cas, il Amazon Athena

créera une AWS Glue connexion pour vous, y compris toutes les propriétés Amazon Athena spécifiques dans la section `AthenaProperties` de la connexion.

En revanche, si vous créez une connexion directement dans Apache Spark AWS Glue, vous serez uniquement invité à saisir les propriétés spécifiques à AWS Glue Apache Spark, qui seront stockées dans les sections ConnectionProperties `` et SparkProperties `` de la connexion.

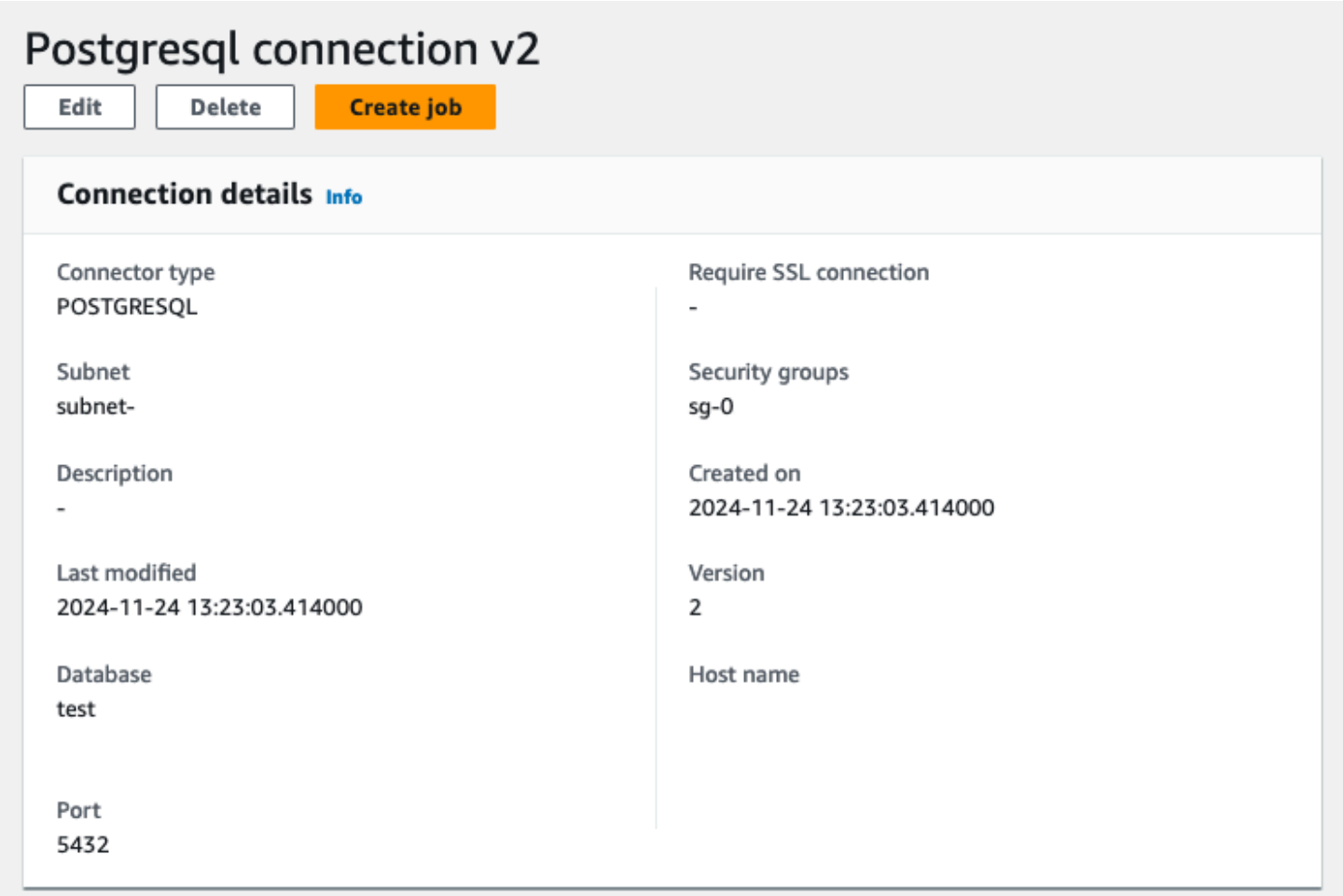
Ces deux scénarios aboutissent à la création d'une « connexion unifiée », mais les connexions créées dans ne Amazon Athena sont configurées que pour une utilisation interne Amazon Athena, tandis que les connexions créées dans ne AWS Glue sont configurées que pour une utilisation interne AWS Glue. Cependant, il est possible de mettre à jour ces connexions avec les propriétés manquantes (propriétés Spark Amazon Athena ou propriétés Spark) afin qu'elles puissent être utilisées par les deux services. Amazon SageMaker AI Unified Studio s'en charge automatiquement en renseignant toutes les propriétés nécessaires (`ConnectionProperties`, AthenaProperties `et` SparkProperties `) sur la AWS Glue connexion, en veillant à ce que la connexion puisse être utilisée par AWS Glue les deux Amazon Athena.

Il est important de noter que même si nous les appelons « connexions unifiées », les connexions créées dans AWS Glue ou Amazon Athena individuellement ne sont pas réellement unifiées à moins qu'elles ne soient correctement configurées pour être utilisées par les deux services. Seules les connexions créées via SageMaker Unified Studio sont réellement unifiées et utilisables par plusieurs services prêts à l'emploi.

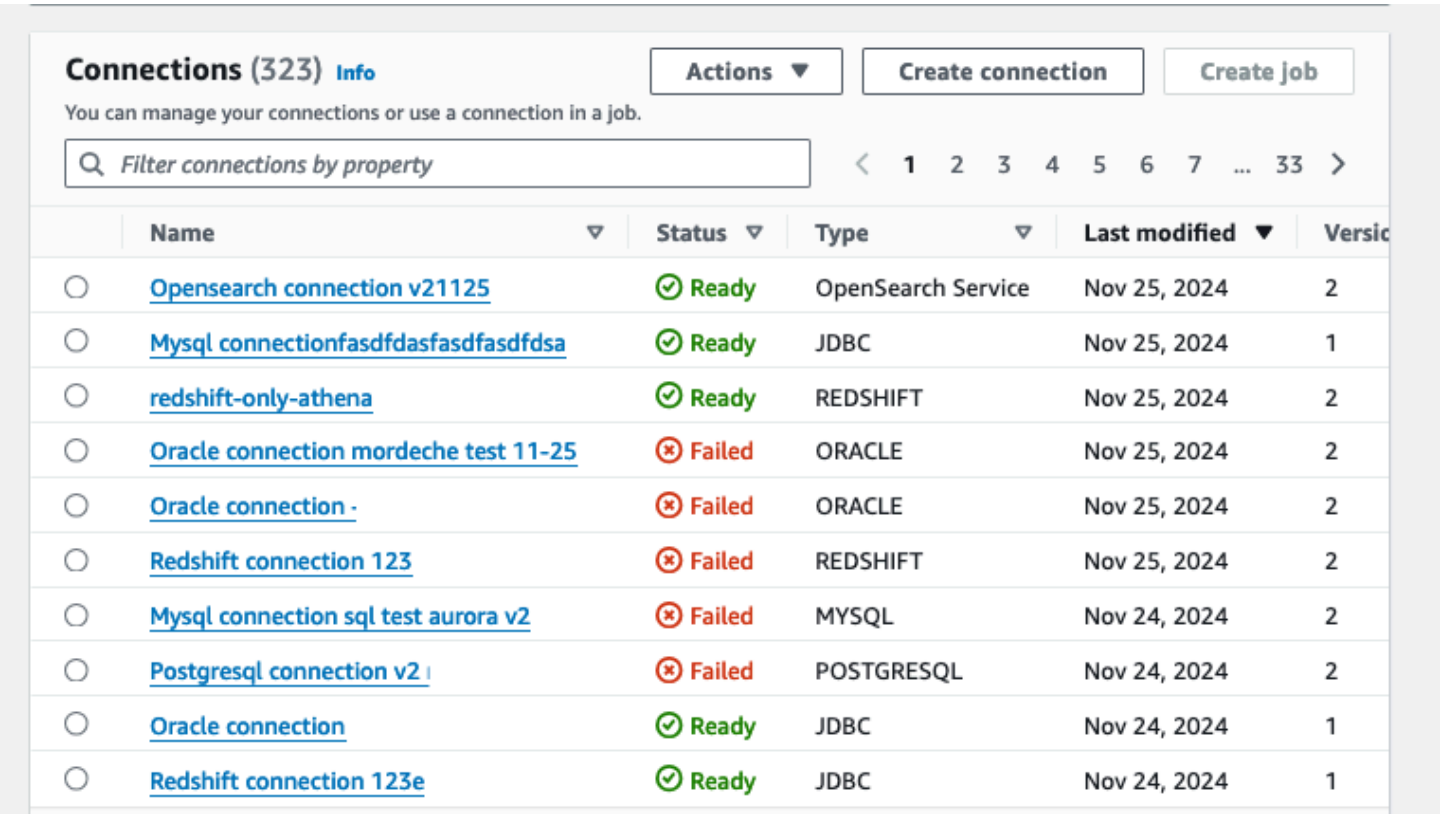
En outre, les connexions créées dans ne AWS Glue sont pas visibles dans Amazon Athena car Amazon Athena affiche les sources de données, qui incluent une référence à une AWS Glue connexion mais ne sont pas la AWS Glue connexion elle-même. De même, les connexions créées dans ne Amazon Athena sont pas visibles dans AWS Glue Studio car AWS Glue Studio les connexions qui n'ont pas été configurées avec les paramètres nécessaires pour sont filtrées AWS Glue.

AWS Glue Studio crée des connexions unifiées par défaut. Dans la AWS Glue console, vous pouvez voir la version de la connexion dans le tableau des connexions sur la page des connexions, sur la page de détail des connexions, et le tableau des connexions sur la page des détails de la tâche.

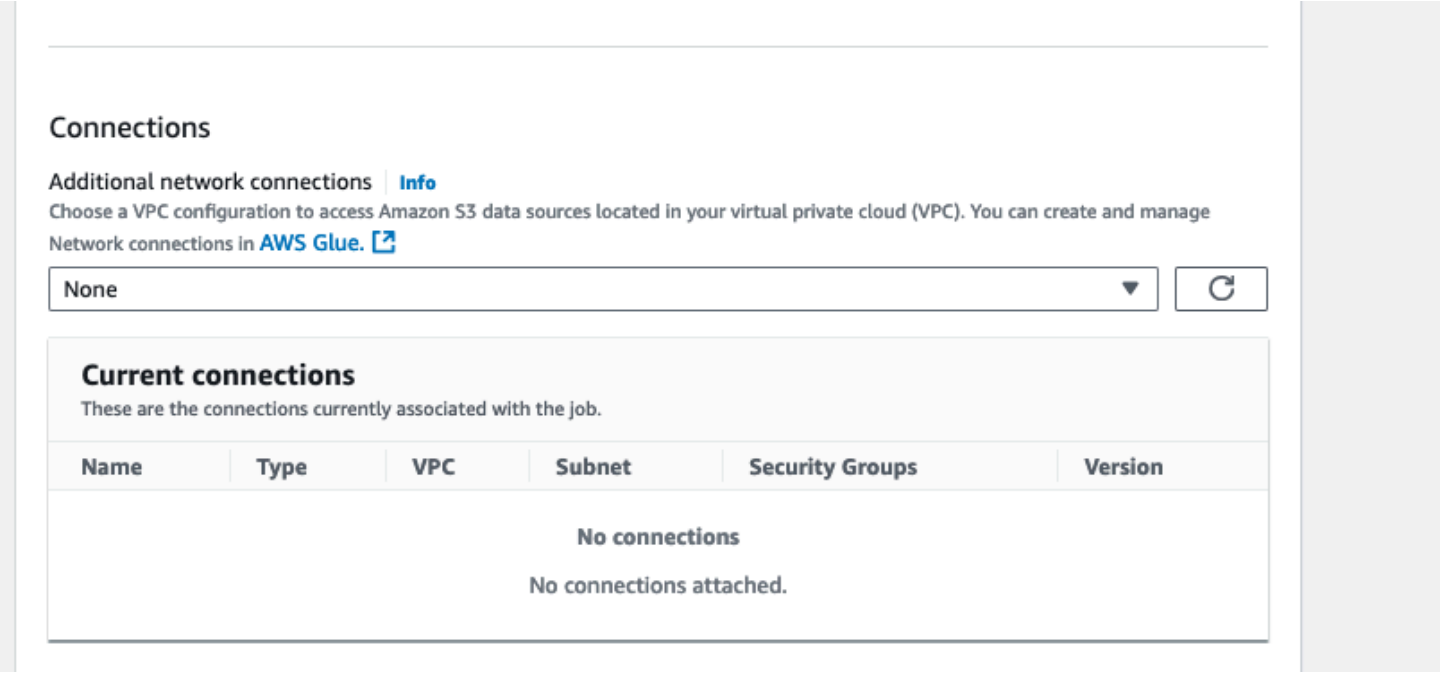
La version de connexion est visible dans Détails de la connexion :



La version de connexion est également visible lorsque vous consultez toutes vos connexions.



Enfin, la version de connexion est visible dans l'onglet Informations de la tâche pour une tâche.



Avec les connexions version 2, vous disposez des fonctionnalités de connectivité de données étendues suivantes :

- Découverte du type de connexion : Support pour la création de connexions à l'aide de modèles standardisés. AWS Glue découvre automatiquement les types de connexion auxquels vous avez accès ainsi que les entrées requises et facultatives pour un type de connexion donné.
- Réutilisabilité : définitions de connexion réutilisables sur tous les moteurs et outils de traitement des AWS données tels que AWS Glue Amazon Athena, et Amazon SageMaker AI. Les connexions contiennent désormais AthenaProperties SparkProperties, PythonProperties qui permettent de spécifier des propriétés de connexion environment/service spécifiques au calcul en plus des propriétés communes stockées dans ConnectionProperties. Athéna crée désormais des connexions en AWS Glue spécifiant des propriétés spécifiques à Athéna dans le plan des propriétés. AthenaProperties
- Aperçu des données : possibilité de parcourir les métadonnées et les données d’aperçu à partir de sources connectées.
- Métadonnées du connecteur : des connexions réutilisables peuvent être utilisées pour découvrir les métadonnées des tables.
- Secrets liés au service : les utilisateurs peuvent fournir les informations d'authentification nécessaires OAuth, de base ou personnalisées dans la `CreateConnection` demande. L' CreateConnection API crée un secret lié au service dans votre compte et stocke les informations d'identification en votre nom.

Types d’authentification pris en charge

Les connexions unifiées prennent en charge les types d’authentification suivants :

- BASIC — La plupart des types de connexion à la base de données et les types de AWS Glue connexion existants prennent en charge l'authentification de base, qui consiste en un nom d'utilisateur et un mot de passe. Auparavant, le nom des clés SecretsManager était spécifique au connecteur et, par exemple, pouvait être user, username, UserName, opensearch.net.http.auth.user, etc. C’est là que les connexions unifiées ont normalisé les types de connexion d’authentification de base sur les clés USERNAME et PASSWORD.
- OAUTH2— La majorité des nouveaux types de connexion SaaS prennent en charge OAuth2 le protocole.
- PERSONNALISÉ — Certains types de connexion disposent d'un autre mécanisme d'authentification, tel que Google, BigQuery où les utilisateurs sont tenus de fournir le JSON qu'ils obtiennent de Google BigQuery.

Considérations

Lorsque vous créez une connexion unifiée pour les sources de données, tenez compte des différences suivantes :

- Lors de la création d'une connexion unifiée via AWS Glue Studio, les informations d'identification de l'utilisateur sont stockées au AWS Secrets Manager lieu de la connexion elle-même. Cela signifie que les tâches doivent désormais accéder à Secrets Manager.
- Si les tâches sont exécutées dans un VPC, elles nécessitent soit un point de terminaison d’un VPC, soit une passerelle NAT pour accéder à AWS Secrets Manager , ainsi qu’un Secure Token Service (STS), ce qui entraîne des coûts supplémentaires.
- Pour certaines sources de données (Redshift, SQL Server, MySQL, Oracle, PostgreSQL), la création d'une connexion unifiée via nécessite l'accès à et. AWS Glue Studio AWS STS AWS Secrets Manager Cela est nécessaire pour établir une connexion sécurisée et récupérer les informations d’identification requises pour accéder à ces sources de données au sein de votre cloud privé virtuel (VPC).
- La création d'une connexion unifiée via AWS Glue Studio nécessite un rôle IAM autorisé à accéder aux ressources VPC AWS Secrets Manager et à les gérer (si vous utilisez un VPC) :
  - responsable des secrets : GetSecretValue
  - responsable des secrets : PutSecretValue
  - responsable des secrets : DescribeSecret
  - EC2 : CreateNetworkInterface
  - EC2 : DeleteNetworkInterface
  - EC2 : DescribeNetworkInterfaces
  - EC2 : DescribeSubnets

Connexions disponibles

AWS Glue prend en charge les types de connexion suivants :

- Adobe Analytics
- Adobe Marketo Engage
- Amazon Aurora (pris en charge si le pilote JDBC natif est utilisé ; l’exploitation de toutes les fonctionnalités du pilote n’est pas possible)

- Amazon DocumentDB
- Amazon DynamoDB
- Amazon OpenSearch Service, à utiliser avec AWS Glue for Spark.
- Amazon Redshift
- Asana
- Azure Cosmos, pour l'utilisation d'Azure Cosmos DB pour NoSQL avec AWS Glue des tâches ETL
- Azure SQL, à utiliser avec AWS Glue for Spark.
- Blackbaud
- CircleCI
- Datadog
- Docusign Monitor
- Domo
- Dynatrace
- Facebook Ads
- Facebook Page Insights
- Freshdesk
- Freshsales
- Google Ads
- Google Analytics 4
- Google BigQuery, à utiliser avec AWS Glue for Spark.
- Google Search Console
- Google Sheets
- HubSpot
- Instagram Ads
- Intercom
- JDBC
- Jira Cloud
- Kafka
- Kustomer

- LinkedIn
- Mailchimp
- Microsoft Dynamics 365 CRM
- Microsoft Teams
- Mixpanel
- Lundi
- MongoDB
- Atlas MongoDB
- Okta
- Oracle NetSuite
- PayPal
- Pendo
- Pipedrive
- Productboard
- QuickBooks
- API REST (nécessite la configuration du type de connexion, voir Connexions à l'API REST)
- Salesforce
- Salesforce Commerce Cloud
- Salesforce Marketing Cloud
- Salesforce Marketing Cloud Account Engagement (anciennement Salesforce Pardot)
- SAP HANA, à utiliser avec AWS Glue for Spark.
- SAP OData
- SendGrid
- ServiceNow
- Slack
- Smartsheet
- Snapchat Ads
- Stripe
- Snowflake, à utiliser avec AWS Glue for Spark.
- Teradata Vantage, lors de l'utilisation AWS Glue pour Spark.

- Twilio
- Vertica, à utiliser avec AWS Glue for Spark.
- WooCommerce
- Zendesk
- Zoho CRM
- Zoom Meetings
- Diverses offres Amazon Relational Database Service (Amazon RDS).
- Réseau (désigne une connexion à une source de données qui se trouve dans un Amazon Virtual Private Cloud (Amazon VPC))

Avec AWS Glue Studio, vous pouvez également créer une connexion pour un connecteur. Un connecteur est un package de code facultatif qui facilite l'accès aux magasins de données dans AWS Glue Studio. Pour de plus amples informations, veuillez consulter Utilisation des connecteurs et des connexions avec AWS Glue Studio

Pour plus d'informations sur la connexion aux bases de données locales, consultez Comment accéder aux banques de données locales et les analyser AWS Glueà l'aide du site Web AWS Big Data Blog.

Pour créer une connexion avec une configuration VPC tout en utilisant un rôle IAM personnalisé, celui-ci doit disposer des actions d'accès VPC suivantes :

- responsable des secrets : GetSecretValue
- responsable des secrets : PutSecretValue
- responsable des secrets : DescribeSecret
- EC2 : CreateNetworkInterface
- EC2 : DeleteNetworkInterface
- EC2 : DescribeNetworkInterfaces
- EC2 : DescribeSubnets

Limitations

- Vous ne pouvez pas modifier les connexions via la AWS Glue console si vous avez créé une connexion v2 en utilisant AWS Glue APIs :

- Amazon DocumentDB
- Amazon Aurora
- MariaDB
- Atlas MongoDB
- MongoDB

Connexions à l'API REST

AWS Glue les connecteurs couvrent un large éventail de sources de données, qu' AWS elles soient externes ou non. Cependant, il se peut qu'il existe des systèmes propriétaires ou des plateformes émergentes que les AWS Glue connecteurs natifs ne prennent pas en charge. Le connecteur AWS Glue REST fournit un mécanisme permettant de configurer un connecteur personnalisé pour toute source de données dotée d'une API basée sur REST qui fonctionne ensuite de la même manière que les connecteurs pris en charge AWS Glue de manière native. Cette fonctionnalité réduit le besoin de créer des bibliothèques personnalisées ou des scripts ETL pour accéder à ces sources de données.

Pour activer le connecteur d'API REST, AWS Glue a introduit une nouvelle AWS ressource : le type de connexion. Les opérations suivantes peuvent être utilisées pour gérer la connectivité aux sources de données basées sur l'API REST :

- RegisterConnectionType- Enregistre un type de connexion en AWS Glue fonction de la configuration fournie. Pour la structure des demandes, voir RegisterConnectionType.
- DeleteConnectionType- Supprime une configuration de type de connexion dans. AWS Glue Voir DeleteConnectionType.
- DescribeConnectionType- Renvoie tous les détails des options prises en charge pour un type de connexion donné dans AWS Glue. Voir DescribeConnectionType.
- ListConnectionTypes- Renvoie la liste des types de connexion avec des informations détaillées sur ce qui est pris en charge pour chaque type de AWS Glue connexion. Consultez ListConnectionTypes.

Création d'un type de connexion REST dans AWS Glue

Pour créer un type de connexion REST, utilisez le AWS CLI AWS SDK ou AWS l'API pour appeler l' RegisterConnectionType API. Les détails de configuration sont fournis en fonction de l'API REST à laquelle vous êtes connecté.

> **Note**
>
> Les connexions REST ne peuvent pas être créées à l'aide de AWS Glue la console

Prérequis

Avant de tenter de configurer votre connecteur d' AWS Glue API REST, vous devez vous familiariser avec le fonctionnement interne de l'API REST. Il est utile de disposer de la documentation de l'API REST pour la source de données et d'un outil pour inspecter la structure de demande/réponse.

Les propriétés suivantes de l'API REST doivent être identifiées :

- Mécanisme d'authentification utilisé pour les appels effectués vers le point de terminaison REST. AWS Glue Les connecteurs d'API REST prennent en charge : l'authentification de base, OAuth2 - ClientCredentials, OAuth2 - le code d'autorisation, OAuth2 - l'authentification JWT et l'authentification personnalisée.
- Entités (objets de données) que vous pouvez transférer via le point de terminaison REST et les métadonnées associées (telles que les noms de champs et les types de données).
- Structure de requête et de réponse HTTP. Cette structure inclut la méthode HTTP, l'URL du point de terminaison, les en-têtes, les paramètres de requête, ainsi que le format du corps de la demande et de la charge utile de la réponse.
- Le schéma de pagination. AWS Glue Les connecteurs d'API REST prennent en charge deux schémas de pagination : `cursor-based` et la `offset-based` pagination.

RegisterConnectionType

Enregistre un ConnectionType identifiant en AWS Glue fonction de la configuration fournie. La ConnectionType ressource est une nouvelle AWS Glue ressource qui stocke des informations sur la façon dont les demandes et les réponses sont interprétées par la source de données. Aujourd'hui, seul le protocole REST est supporté. Il encapsule des détails sur la source de l'API REST, tels que :

- Propriétés de connexion - Définit BaseUrl la connexion à l'API REST et définit tous les paramètres de demande supplémentaires nécessaires pour prendre des entrées pendant AWS Glue CreateConnection.

- Configuration de l'authentification : définit la manière dont les demandes adressées à l'API REST sont authentifiées. Cette configuration est utilisée lors de la création d'une AWS Glue connexion pour le type de connexion.
- Configuration REST : définit la configuration des requêtes et réponses HTTP pour lire les données depuis la source de l'API REST. Les propriétés suivantes doivent être configurées au niveau global ou au niveau de l'entité :
  - Méthode de demande : définit la méthode HTTP REST.
  - Configuration de la réponse : définit la manière dont la réponse de l'API doit être interprétée (y compris les emplacements des enregistrements ou des erreurs dans le corps).
  - Configuration de la pagination : définit la stratégie utilisée par l'API REST pour paginer les enregistrements résultants et la manière dont les propriétés de pagination doivent être envoyées/récupérées.
  - Configuration du point de terminaison de validation : définit le point de terminaison à appeler pour valider une AWS Glue connexion.
  - Configurations d'entités : définit les entités REST prises en charge par le connecteur ainsi que leur schéma.

RegisterConnectionType Syntaxe de la demande

Voir RegisterConnectionType

Une fois qu'un type de connexion à l'API REST a été enregistré, il est prêt à être utilisé pour créer des connexions à la source de données configurée. Reportez-vous à la section Connexion à REST APIs pour obtenir des instructions détaillées.

AWS Glue propriétés de connexion

Cette rubrique contient des informations sur les propriétés des AWS Glue connexions.

Rubriques

- Propriétés de connexion requises
- AWS Glue Propriétés de connexion JDBC
- AWS Glue Propriétés de connexion MongoDB et MongoDB Atlas
- Propriétés de connexion Salesforce
- Connexion Snowflake

- Connexion Vertica
- Connexion SAP HANA
- Connexion Azure SQL
- Connexion Teradata Vantage
- OpenSearch Connexion au service
- Connexion Azure Cosmos
- AWS Glue Propriétés de connexion SSL
- Propriétés de connexion Apache Kafka pour l'authentification du client
- BigQuery Connexion Google
- Connexion Vertica

Propriétés de connexion requises

Lorsque vous définissez une connexion sur la AWS Glue console, vous devez fournir des valeurs pour les propriétés suivantes :

Connection name (Nom de la connexion)

Saisissez un nom unique pour votre connexion.

Type de connexion

Choisissez JDBC ou l'un des types de connexion spécifiques.

Pour plus d'informations sur le type de connexion JDBC, consultez the section called "Propriétés de connexion JDBC"

Choisissez Network (Réseau) pour vous connecter à une source de données dans un environnement Amazon Virtual Private Cloud (Amazon VPC).

Selon le type que vous choisissez, la console AWS Glue affiche d'autres champs obligatoires. Par exemple, si vous choisissez Amazon RDS, vous devez ensuite choisir le moteur de base de données.

Require SSL connection (Connexion SSL obligatoire)

Lorsque vous sélectionnez cette option, vous AWS Glue devez vérifier que la connexion au magasin de données est connectée via un protocole SSL (Secure Sockets Layer) approuvé.

Pour plus d'informations, y compris les options supplémentaires disponibles lorsque vous sélectionnez cette option, consultez the section called "Propriétés de connexion SSL".

Sélectionner un cluster MSK (Amazon Managed Streaming for Apache Kafka (MSK) uniquement)

Spécifie un cluster MSK provenant d'un autre AWS compte.

Serveur bootstrap Kafka URLs (Kafka uniquement)

Spécifie une liste de serveurs bootstrap séparés par des virgules. URLs Incluez le numéro de port. Par exemple : b-1.vpc-test-2.o4q88o.c6.kafka.us-east-1.amazonaws.com:9094, b-2.vpc-test-2.o4q88o.c6.kafka.us-east-1.amazonaws.com:9094, b-3.vpc-test-2.o4q88o.c6.kafka.us-east-1.amazonaws.com:9094

AWS Glue Propriétés de connexion JDBC

AWS Glue Studio crée désormais des connexions unifiées pour les sources de données MySQL, Oracle, PostgressQL, Redshift et SQL Server, ce qui nécessite des étapes supplémentaires pour accéder aux ressources de Secrets Manager et VPC, ce qui peut entraîner des coûts supplémentaires. Vous pouvez accéder à ces connexions en AWS Glue Studio choisissant le nom de la connexion correspondante.

Pour de plus amples informations, veuillez consulter Considérations.

AWS Glue peut se connecter aux magasins de données suivants via une connexion JDBC :

- Amazon Redshift
- Amazon Aurora
- Microsoft SQL Server
- MySQL
- Oracle
- PostgreSQL
- Flocon de neige, lorsque vous utilisez AWS Glue des chenilles.
- Aurora (pris en charge si le pilote JDBC natif est utilisé ; l'exploitation de toutes les fonctionnalités du pilote n'est pas possible)
- Amazon RDS for MariaDB

> **Important**
>
> Actuellement, une tâche ETL peut utiliser des connexions JDBC dans un seul sous-réseau. Si vous avez plusieurs magasins de données dans une tâche, ils doivent être sur le même sous-réseau ou accessibles depuis le sous-réseau.
> Si vous choisissez d'utiliser vos propres versions de pilotes JDBC pour les robots d' AWS Glue exploration, ceux-ci consommeront des ressources dans les tâches et AWS Glue Amazon S3 pour s'assurer que les pilotes que vous avez fournis sont exécutés dans votre environnement. L'utilisation supplémentaire des ressources sera reflétée sur votre compte. De plus, le fait de fournir votre propre pilote JDBC ne signifie pas que le Crawler est capable de tirer parti de toutes les fonctionnalités du pilote. Les pilotes sont limités aux propriétés décrites dans Defining connections in the Data Catalog.

Voici des propriétés supplémentaires pour le type de connexion JDBC.

URL JDBC

Saisissez l'URL de votre magasin de données JDBC. Pour la plupart des moteurs de base de données, ce champ est au format suivant. Dans ce format, remplacez *protocolhost*,*port*, et *db_name* par vos propres informations.

`jdbc:`*protocol*`://`*host*`:`*port*`/`*db_name*

En fonction du moteur de base de données, un autre format d'URL JDBC peut être requis. Ce format peut avoir une utilisation légèrement différente des deux-points (:) et de la barre oblique (/), ou des mots-clés différents pour spécifier les bases de données.

Pour que JDBC se connecte au magasin de données, un `db_name` dans le magasin de données est obligatoire. Le `db_name` permet d'établir une connexion réseau avec les `username` et `password` fournis. Une fois connecté, AWS Glue vous pouvez accéder à d'autres bases de données du magasin de données pour exécuter un robot d'exploration ou une tâche ETL.

Les exemples d'URL JDBC suivants montrent la syntaxe pour plusieurs moteurs de base de données.

- Pour se connecter au magasin de données d'un cluster Amazon Redshift avec une base de données `dev` :

```
jdbc:redshift://xxx.us-east-1.redshift.amazonaws.com:8192/dev
```

- Pour se connecter à un magasin de données Amazon RDS for MySQL avec une base de données `employee` :

  ```
  jdbc:mysql://xxx-cluster.cluster-xxx.us-east-1.rds.amazonaws.com:3306/employee
  ```

- Pour se connecter à un magasin de données Amazon RDS for PostgreSQL avec une base de données `employee` :

  ```
  jdbc:postgresql://xxx-cluster.cluster-xxx.us-east-1.rds.amazonaws.com:5432/employee
  ```

- Pour se connecter à un magasin de données Amazon RDS for Oracle avec un nom de service `employee` :

  ```
  jdbc:oracle:thin://@xxx-cluster.cluster-xxx.us-east-1.rds.amazonaws.com:1521/employee
  ```

  La syntaxe d'Amazon RDS for Oracle peut suivre les modèles suivants. Dans ces modèles, remplacez *`hostport`*,*`service_name`*, et *`SID`* par vos propres informations.

  - `jdbc:oracle:thin://@`*`host`*`:`*`port`*`/`*`service_name`*
  - `jdbc:oracle:thin://@`*`host`*`:`*`port`*`:`*`SID`*

- Pour se connecter à un magasin de données Amazon RDS for Microsoft SQL Server avec une base de données `employee` :

  ```
  jdbc:sqlserver://xxx-cluster.cluster-xxx.us-east-1.rds.amazonaws.com:1433;databaseName=employee
  ```

  La syntaxe d'Amazon RDS for SQL Server peut suivre les modèles suivants. Dans ces modèles, remplacez *`server_nameport`*, et *`db_name`* par vos propres informations.

  - `jdbc:sqlserver://`*`server_name`*`:`*`port`*`;database=`*`db_name`*
  - `jdbc:sqlserver://`*`server_name`*`:`*`port`*`;databaseName=`*`db_name`*

- Pour vous connecter à une Amazon Aurora PostgreSQL instance de la `employee` base de données, spécifiez le point de terminaison de l'instance de base de données, le port et le nom de la base de données :

  ```
  jdbc:postgresql://employee_instance_1.xxxxxxxxxxxx.us-east-2.rds.amazonaws.com:5432/employee
  ```

- Pour vous connecter à un magasin de Amazon RDS for MariaDB données avec une `employee` base de données, spécifiez le point de terminaison de l'instance de base de données, le port et le nom de la base de données :

  ```
  jdbc:mysql://xxx-cluster.cluster-xxx.aws-
  region.rds.amazonaws.com:3306/employee
  ```

- > **Warning**
  >
  > Les connexions JDBC Snowflake ne sont prises en charge que par les robots d'exploration. AWS Glue Lorsque vous utilisez le connecteur Snowflake dans des AWS Glue tâches, utilisez le type de connexion Snowflake.

  Pour vous connecter à une l'instance Snowflake de la base de données `sample`, spécifiez le point de terminaison de l'instance Snowflake, l'utilisateur, le nom de base de données et le nom de rôle. Vous pouvez également ajouter le paramètre `warehouse`.

  ```
  jdbc:snowflake://account_name.snowflakecomputing.com/?
  user=user_name&db=sample&role=role_name&warehouse=warehouse_name
  ```

  > **Important**
  >
  > Pour les connexions Snowflake via JDBC, l'ordre des paramètres dans l'URL est imposé et doit être le suivant : `user`, db, `role_name` et `warehouse`.

- Pour vous connecter à une instance Snowflake de la `sample` base de données via un lien AWS privé, spécifiez l'URL JDBC Snowflake comme suit :

  ```
  jdbc:snowflake://account_name.region.privatelink.snowflakecomputing.com/?
  user=user_name&db=sample&role=role_name&warehouse=warehouse_name
  ```

Nom d'utilisateur

> **Note**
>
> Nous vous recommandons d'utiliser un AWS secret pour stocker les informations de connexion au lieu de fournir directement votre nom d'utilisateur et votre mot de passe.

Pour de plus amples informations, veuillez consulter Stockage des informations de connexion dans AWS Secrets Manager.

Fournissez un nom d'utilisateur qui est autorisé à accéder au magasin de données JDBC.

Mot de passe

Saisissez le mot de passe pour le nom d'utilisateur qui a l'autorisation d'accès au magasin de données JDBC.

Port

Saisissez le port utilisé dans l'URL JDBC pour vous connecter à une instance Amazon RDS Oracle. Ce champ est uniquement affiché lorsque Require SSL connection (Connexion SSL obligatoire) est sélectionné pour une instance Amazon RDS Oracle.

VPC

Choisissez le nom du cloud privé virtuel (VPC) qui contient votre magasin de données. La AWS Glue console répertorie tout VPCs pour la région actuelle.

> **Important**
>
> Lorsque vous travaillez sur une connexion JDBC hébergée hors de AWS, par exemple avec des données provenant de Snowflake, votre VPC doit disposer d'une passerelle NAT qui divise le trafic en sous-réseaux publics et privés. Le sous-réseau public est utilisé pour la connexion à la source externe, et le sous-réseau interne est utilisé pour le traitement par. AWS Glue Pour plus d'informations sur la configuration de votre Amazon VPC pour les connexions externes, consultez Connectez-vous à Internet ou à d'autres réseaux à l'aide de périphériques NAT et Configuration d'Amazon VPC pour les connexions JDBC aux entrepôts de données Amazon RDS depuis AWS Glue.

Sous-réseau

Choisissez le sous-réseau du VPC qui contient votre magasin de données. La console AWS Glue répertorie tous les sous-réseaux pour le magasin de données de votre VPC.

Groupes de sécurité

Choisissez les groupes de sécurité associés à votre banque de données. AWS Glue nécessite un ou plusieurs groupes de sécurité dotés d'une règle de source entrante qui permet de se AWS

Glue connecter. La AWS Glue console répertorie tous les groupes de sécurité auxquels un accès entrant est accordé à votre VPC. AWS Glue associe ces groupes de sécurité à l'interface elastic network attachée à votre sous-réseau VPC.

Nom de la classe de pilote JDBC (facultatif)

Indiquez le nom de classe de pilote JDBC personnalisé :

- Postgres : org.postgresql.Driver
- MySQL : com.mysql.jdbc.Driver, com.mysql.cj.jdbc.Driver
- Redshift : com.amazon.redshift.jdbc.Driver, com.amazon.redshift.jdbc42.Driver
- Oracle : oracle.jdbc.driver. OracleDriver
- SQL Server : com.microsoft.sqlserver.jdbc. SQLServerChauffeur

Chemin S3 du pilote JDBC (facultatif)

Indiquez l'emplacement Amazon S3 au pilote JDBC personnalisé. Il s'agit d'un chemin absolu vers un fichier .jar. Si vous souhaitez fournir vos propres pilotes JDBC afin de vous connecter à vos sources de données pour vos bases de données prises en charge par un Crawler, vous pouvez spécifier des valeurs pour les paramètres `customJdbcDriverS3Path` et `customJdbcDriverClassName`.
L'utilisation d'un pilote JDBC fourni par un client est limitée au Propriétés de connexion requises nécessaire.

AWS Glue Propriétés de connexion MongoDB et MongoDB Atlas

Voici des propriétés supplémentaires pour le type de connexion MongoDB ou MongoDB Atlas.

URL MongoDB

Saisissez l'URL de votre magasin de données MongoDB ou MongoDB Atlas :

- Pour MongoDB : mongodb://host:port/database. L'hôte peut être un nom d'hôte, une adresse IP ou un socket de domaine UNIX. Si la chaîne de connexion ne spécifie aucun port, elle utilise le port MongoDB par défaut, 27017.

- Pour MongoDB Atlas : mongodb+srv://server.example.com/database. L'hôte peut être un nom d'hôte qui suit et correspond à un enregistrement DNS SRV. Le format SRV ne nécessite pas de port et utilisera le port MongoDB par défaut, 27017.

Nom d'utilisateur

> **Note**
>
> Nous vous recommandons d'utiliser un AWS secret pour stocker les informations de connexion au lieu de fournir directement votre nom d'utilisateur et votre mot de passe. Pour de plus amples informations, veuillez consulter Stockage des informations de connexion dans AWS Secrets Manager.

Fournissez un nom d'utilisateur qui est autorisé à accéder au magasin de données JDBC.

Mot de passe

Saisissez le mot de passe pour le nom d'utilisateur qui a l'autorisation d'accès au magasin de données MongoDB ou MongoDB Atlas.

Propriétés de connexion Salesforce

Voici des propriétés supplémentaires pour le type de connexion Salesforce.

- `ENTITY_NAME`(Chaîne) - (Obligatoire) Utilisé pour la lecture/l'écriture. Le nom de votre objet dans Salesforce.
- `API_VERSION`(Chaîne) - (Obligatoire) Utilisé pour la lecture/l'écriture. Version d'API Rest de Salesforce que vous souhaitez utiliser.
- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.
- `PARTITION_FIELD`(Chaîne) - Utilisé pour la lecture. Champ à utiliser pour partitionner la requête.
- `LOWER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite inférieure inclusive du champ de partition choisi.

- `UPPER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS`(Entier) - Par défaut : 1. Utilisé pour la lecture. Nombre de partitions pour la lecture.
- `IMPORT_DELETED_RECORDS`(Chaîne) - Par défaut : FALSE. Utilisé pour la lecture. Pour obtenir les enregistrements supprimés lors de la requête.
- `WRITE_OPERATION`(Chaîne) - Par défaut : INSERT. Utilisé pour l'écriture La valeur doit être INSERT, UPDATE, UPSERT, DELETE.
- `ID_FIELD_NAMES`(Chaîne) - Valeur par défaut : null. Utilisé uniquement pour UPSERT.

Connexion Snowflake

Les propriétés suivantes sont utilisées pour configurer une connexion Snowflake utilisée dans les tâches AWS Glue ETL. Lors de l'indexation de Snowflake, utilisez une connexion JDBC.

URL de Snowflake

L'URL de votre point de terminaison Snowflake. Pour plus d'informations sur le point de terminaison Snowflake URLs, consultez la section Connexion à vos comptes dans la documentation Snowflake.

AWS Secret

Le nom secret d'un secret dans AWS Secrets Manager. AWS Glue se connectera à Snowflake en utilisant les `sfPassword` clés `sfUser` et de votre secret.

Rôle Snowflake (facultatif)

Un rôle de sécurité Snowflake AWS Glue sera utilisé lors de la connexion.

Utilisez les propriétés suivantes lors de la configuration d'une connexion à un point de terminaison Snowflake hébergé dans Amazon VPC en utilisant AWS PrivateLink.

VPC

Choisissez le nom du cloud privé virtuel (VPC) qui contient votre magasin de données. La AWS Glue console répertorie tout VPCs pour la région actuelle.

Sous-réseau

Choisissez le sous-réseau du VPC qui contient votre magasin de données. La console AWS Glue répertorie tous les sous-réseaux pour le magasin de données de votre VPC.

Groupes de sécurité

Choisissez les groupes de sécurité associés à votre banque de données. AWS Glue nécessite un ou plusieurs groupes de sécurité dotés d'une règle de source entrante qui permet de se AWS Glue connecter. La AWS Glue console répertorie tous les groupes de sécurité auxquels un accès entrant est accordé à votre VPC. AWS Glue associe ces groupes de sécurité à l'interface elastic network attachée à votre sous-réseau VPC.

Connexion Vertica

Utilisez les propriétés suivantes pour configurer une connexion Vertica pour les tâches AWS Glue ETL.

Hôte Vertica

Le nom d’hôte de votre installation Vertica.

Port Vertica

Le port via lequel est disponible votre installation Vertica.

AWS Secret

Le nom secret d'un secret dans AWS Secrets Manager. AWS Glue se connectera à Vertica en utilisant les clés de votre secret.

Utilisez les propriétés suivantes lors de la configuration d’une connexion à un point de terminaison Vertica hébergé dans Amazon VPC.

VPC

Choisissez le nom du cloud privé virtuel (VPC) qui contient votre magasin de données. La AWS Glue console répertorie tout VPCs pour la région actuelle.

Sous-réseau

Choisissez le sous-réseau du VPC qui contient votre magasin de données. La console AWS Glue répertorie tous les sous-réseaux pour le magasin de données de votre VPC.

Groupes de sécurité

Choisissez les groupes de sécurité associés à votre banque de données. AWS Glue nécessite un ou plusieurs groupes de sécurité dotés d'une règle de source entrante qui permet de se AWS Glue connecter. La AWS Glue console répertorie tous les groupes de sécurité auxquels un accès entrant est accordé à votre VPC. AWS Glue associe ces groupes de sécurité à l'interface elastic network attachée à votre sous-réseau VPC.

Connexion SAP HANA

Utilisez les propriétés suivantes pour configurer une connexion SAP HANA pour les tâches AWS Glue ETL.

URL SAP HANA

UNE URL SAP JDBC.

SAP HANA JDBC URLs sont sous la forme `jdbc:sap://saphanaHostname:saphanaPort/?databaseName=saphanaDBname,ParameterNa`

AWS Glue nécessite les paramètres d'URL JDBC suivants :

- `databaseName` – une base de données par défaut dans SAP HANA à laquelle se connecter.

AWS Secret

Le nom secret d'un secret dans AWS Secrets Manager. AWS Glue se connectera à SAP HANA à l'aide des clés de votre secret.

Utilisez les propriétés suivantes lors de la configuration d'une connexion à un point de terminaison SAP HANA hébergé dans Amazon VPC :

VPC

Choisissez le nom du cloud privé virtuel (VPC) qui contient votre magasin de données. La AWS Glue console répertorie tout VPCs pour la région actuelle.

Sous-réseau

Choisissez le sous-réseau du VPC qui contient votre magasin de données. La console AWS Glue répertorie tous les sous-réseaux pour le magasin de données de votre VPC.

Groupes de sécurité

Choisissez les groupes de sécurité associés à votre banque de données. AWS Glue nécessite un ou plusieurs groupes de sécurité dotés d'une règle de source entrante qui permet de se AWS Glue connecter. La AWS Glue console répertorie tous les groupes de sécurité auxquels un accès entrant est accordé à votre VPC. AWS Glue associe ces groupes de sécurité à l'interface elastic network attachée à votre sous-réseau VPC.

Connexion Azure SQL

Utilisez les propriétés suivantes pour configurer une connexion Azure SQL pour les tâches AWS Glue ETL.

URL Azure SQL

URL JDBC d'un endpoint Azure SQL.

La URL doit avoir le format suivant : `jdbc:sqlserver://databaseServerName:databasePort;databaseName=azuresqlDBname;`.

AWS Glue nécessite les propriétés d'URL suivantes :

- `databaseName` – une base de données par défaut dans Azure SQL à laquelle se connecter.

Pour plus d'informations sur JDBC URLs pour Azure SQL Managed Instances, consultez la documentation Microsoft.

AWS Secret

Le nom secret d'un secret dans AWS Secrets Manager. AWS Glue se connectera à Azure SQL à l'aide des clés de votre secret.

Connexion Teradata Vantage

Utilisez les propriétés suivantes pour configurer une connexion Teradata Vantage pour les tâches AWS Glue ETL.

URL de Teradata

Pour vous connecter à une instance Teradata, spécifiez le nom d'hôte de l'instance de base de données et les paramètres Teradata pertinents :

`jdbc:teradata://teradataHostname/ParameterName=ParameterValue,ParameterName=Pa`

AWS Glue prend en charge les paramètres d'URL JDBC suivants :

- `DATABASE_NAME` – une base de données par défaut dans Teradata à laquelle se connecter.
- `DBS_PORT` – spécifie le port Teradata, s'il n'est pas standard.

AWS Secret

Le nom secret d'un secret dans AWS Secrets Manager. AWS Glue se connectera à Teradata Vantage à l'aide des clés de votre secret.

Utilisez les propriétés suivantes lors de la configuration d'une connexion à un point de terminaison Teradata Vantage hébergé dans Amazon VPC :

VPC

Choisissez le nom du cloud privé virtuel (VPC) qui contient votre magasin de données. La AWS Glue console répertorie tout VPCs pour la région actuelle.

Sous-réseau

Choisissez le sous-réseau du VPC qui contient votre magasin de données. La console AWS Glue répertorie tous les sous-réseaux pour le magasin de données de votre VPC.

Groupes de sécurité

Choisissez les groupes de sécurité associés à votre banque de données. AWS Glue nécessite un ou plusieurs groupes de sécurité dotés d'une règle de source entrante qui permet de se AWS Glue connecter. La AWS Glue console répertorie tous les groupes de sécurité auxquels un accès entrant est accordé à votre VPC. AWS Glue associe ces groupes de sécurité à l'interface elastic network attachée à votre sous-réseau VPC.

OpenSearch Connexion au service

Utilisez les propriétés suivantes pour configurer une connexion de OpenSearch service pour les tâches AWS Glue ETL.

Point de terminaison du domaine

Un point de terminaison de domaine Amazon OpenSearch Service aura le formulaire par défaut suivant, https://search - *domainName* -*unstructuredIdContent*. *region*.es.amazonaws.com.

Pour plus d'informations sur l'identification du point de terminaison de votre domaine, consultez la section Création et gestion des domaines Amazon OpenSearch Service dans la documentation Amazon OpenSearch Service.

Port

Le port ouvert sur le point de terminaison.

AWS Secret

Le nom secret d'un secret dans AWS Secrets Manager. AWS Glue se connectera au OpenSearch Service en utilisant les clés de votre secret.

Utilisez les propriétés suivantes lors de la configuration d'une connexion à un point de terminaison de OpenSearch service hébergé dans Amazon VPC :

VPC

Choisissez le nom du cloud privé virtuel (VPC) qui contient votre magasin de données. La AWS Glue console répertorie tout VPCs pour la région actuelle.

Sous-réseau

Choisissez le sous-réseau du VPC qui contient votre magasin de données. La console AWS Glue répertorie tous les sous-réseaux pour le magasin de données de votre VPC.

Groupes de sécurité

Choisissez les groupes de sécurité associés à votre banque de données. AWS Glue nécessite un ou plusieurs groupes de sécurité dotés d'une règle de source entrante qui permet de se AWS Glue connecter. La AWS Glue console répertorie tous les groupes de sécurité auxquels un accès entrant est accordé à votre VPC. AWS Glue associe ces groupes de sécurité à l'interface elastic network attachée à votre sous-réseau VPC.

Connexion Azure Cosmos

Utilisez les propriétés suivantes pour configurer une connexion Azure Cosmos pour les tâches AWS Glue ETL.

URI du point de terminaison du compte Azure Cosmos DB

Point de terminaison utilisé pour vous connecter à Azure Cosmos. Pour en savoir plus, consultez la documentation Azure.

AWS Secret

Le nom secret d'un secret dans AWS Secrets Manager. AWS Glue se connectera à Azure Cosmos à l'aide des clés de votre secret.

AWS Glue Propriétés de connexion SSL

Vous trouverez ci-dessous des détails sur la propriété Require SSL connection (Connexion SSL obligatoire).

Si vous n'avez pas besoin d'une connexion SSL, AWS Glue ignore les échecs lorsqu'il utilise le protocole SSL pour chiffrer une connexion au magasin de données. Consultez la documentation de votre magasin de données pour obtenir les instructions de configuration. Lorsque vous sélectionnez cette option, l'exécution de la tâche, le crawler ou les déclarations ETL dans un point de terminaison de développement échouent lorsque AWS Glue ne parvient pas à se connecter.

> **Note**
>
> Snowflake prend en charge une connexion SSL par défaut. Cette propriété n'est donc pas applicable à Snowflake.

Cette option est validée côté AWS Glue client. Pour les connexions JDBC, se connecte AWS Glue uniquement via SSL avec validation du certificat et du nom d'hôte. La prise en charge de la connexion SSL est disponible pour :

- Oracle Database
- Microsoft SQL Server
- PostgreSQL
- Amazon Redshift
- MySQL (instances Amazon RDS uniquement)
- Amazon Aurora (instances Amazon RDS uniquement)
- Amazon Aurora PostgreSQL (Instances Amazon RDS uniquement)
- Kafka, qui inclut Amazon Managed Streaming pour Apache Kafka
- MongoDB

> **Note**
>
> Pour permettre à un magasin de données Amazon RDS Oracle d'utiliser Require SSL connection (Connexion SSL obligatoire), vous devez créer et attacher un groupe d'options à l'instance Oracle.
>
> 1. Connectez-vous à la console Amazon RDS AWS Management Console et ouvrez-la à https://console.aws.amazon.com/rds/l'adresse.
> 2. Ajoutez un groupe d'options à l'instance Amazon RDS Oracle. Pour plus d'informations sur la façon d'ajouter un groupe d'options à la console Amazon RDS, veuillez consulter Création d'un groupe d'options
> 3. Ajoutez une Option au groupe d'options pour SSL. Le port que vous spécifiez pour SSL est ensuite utilisé lorsque vous créez une URL de connexion AWS Glue JDBC pour l'instance Oracle Amazon RDS. Pour plus d'informations sur la façon d'ajouter une option à la console Amazon RDS, veuillez consulter Ajout d'une option à un groupe d'options dans le Guide de l'utilisateur Amazon RDS. Pour de plus amples informations sur l'option Oracle SSL, veuillez consulter Oracle SSL dans le Guide de l'utilisateur Amazon RDS.
> 4. Sur la AWS Glue console, créez une connexion à l'instance Oracle Amazon RDS. Dans la définition de connexion, sélectionnez Require SSL connection (Connexion SSL obligatoire). Lorsque vous y êtes invité, saisissez le port que vous avez utilisé dans l'option Oracle SSL d'Amazon RDS.

Les propriétés facultatives supplémentaires suivantes sont disponibles lorsque l'option Require SSL connection (Connexion SSL obligatoire) est sélectionnée pour une connexion :

Certificat JDBC personnalisé dans S3

Si vous utilisez actuellement un certificat pour les communications SSL avec vos bases de données sur site ou dans le cloud, vous pouvez utiliser ce certificat pour les connexions SSL aux sources de AWS Glue données ou aux cibles. Entrez un emplacement Amazon Simple Storage Service (Amazon S3) contenant un certificat racine personnalisé. AWS Glue utilise ce certificat pour établir une connexion SSL avec la base de données. AWS Glue gère uniquement les certificats X.509. Le certificat doit être codé DER et fourni au format PEM d'encodage Base64.

Si ce champ est laissé vide, le certificat par défaut est utilisé.

Chaîne de certificat JDBC personnalisé

Saisissez les informations de certificat spécifiques à votre base de données JDBC. Il s'agit d'une chaîne utilisée pour la mise en correspondance des domaines ou des noms uniques (DN). Pour Oracle Database, cette chaîne est mappée sur le paramètre `SSL_SERVER_CERT_DN` dans la section de sécurité du fichier `tnsnames.ora`. Pour Microsoft SQL Server, elle est utilisée comme `hostNameInCertificate`.

Voici un exemple de paramètre `SSL_SERVER_CERT_DN` pour Oracle Database.

```
cn=sales,cn=OracleContext,dc=us,dc=example,dc=com
```

Emplacement du certificat d'autorité de certification privé Kafka

Si vous avez un certificat que vous utilisez actuellement pour les communications SSL avec votre banque de données Kafka, vous pouvez utiliser ce certificat avec votre AWS Glue connexion. Cette option est obligatoire pour les magasins de données Kafka et facultative pour les magasins de Amazon Managed Streaming pour Apache Kafka données. Entrez un emplacement Amazon Simple Storage Service (Amazon S3) contenant un certificat racine personnalisé. AWS Glue utilise ce certificat pour établir une connexion SSL avec le magasin de données Kafka. AWS Glue gère uniquement les certificats X.509. Le certificat doit être codé DER et fourni au format PEM d'encodage Base64.

Ignorer la validation des certificats

Cochez la case Ignorer la validation du certificat pour ignorer la validation du certificat personnalisé par AWS Glue. Si vous choisissez de valider, AWS Glue valide l'algorithme de signature et l'algorithme de clé publique du sujet du certificat. Si la validation du certificat échoue, toute tâche ETL ou crawler qui utilise la connexion échoue.

Les seuls algorithmes de signature autorisés sont SHA256with RSA, SHA384with RSA ou SHA512with RSA. Pour l'algorithme de clé publique d'objet, la longueur de clé doit être d'au moins 2048.

Emplacement du magasin de clés client Kafka

Emplacement Amazon S3 du fichier de magasin de clés client pour l'authentification côté client Kafka. Le chemin doit être au format s3 ://bucket/prefix/filename.jks. Il doit se terminer par le nom du fichier et l'extension .jks.

Mot de passe du magasin de clés client Kafka (facultatif)

Le mot de passe pour accéder au magasin de clés fourni.

Mot de passe de la clé client Kafka (facultatif)

Un magasin de clés peut être composé de plusieurs clés, il s'agit donc du mot de passe pour accéder à la clé client à utiliser avec la clé côté serveur Kafka.

Propriétés de connexion Apache Kafka pour l'authentification du client

AWS Glue prend en charge le framework SASL (Simple Authentication and Security Layer) pour l'authentification lorsque vous créez une connexion Apache Kafka. Le cadre SASL prend en charge divers mécanismes d'authentification, et AWS Glue propose à la fois les protocoles SCRAM (nom d'utilisateur et mot de passe), GSSAPI (protocole Kerberos) et PLAIN.

AWS Glue Studio À utiliser pour configurer l'une des méthodes d'authentification client suivantes. Pour plus d'informations, consultez la section Création de connexions pour les connecteurs dans le guide de AWS Glue Studio l'utilisateur.

- Aucune - Aucune authentification. Cette option est utile si vous créez une connexion pour des raisons de tests.
- SASL/SCRAM-SHA-512 - Le choix de cette méthode d'authentification vous permettra de spécifier les informations d'identification d'authentification. Deux options s'offrent à vous :
  - Utiliser AWS Secrets Manager (recommandé) : si vous sélectionnez cette option, vous pouvez enregistrer votre nom d'utilisateur et votre mot de passe dans AWS Secrets Manager et y AWS Glue accéder en cas de besoin. Spécifiez le secret qui stocke les informations d'identification d'authentification SSL ou SASL. Pour plus d'informations, consultez Stockage des informations de connexion dans AWS Secrets Manager.
  - Fournissez directement un nom d'utilisateur et un mot de passe.
- SASL/GSSAPI (Kerberos) - if you select this option, you can select the location of the keytab file, krb5.conf file and enter the Kerberos principal name and Kerberos service name. The locations for the keytab file and krb5.conf file must be in an Amazon S3 location. Since MSK does not yet support SASL/GSSAPI, cette option n'est disponible que pour les clusters Apache Kafka gérés par le client. Pour en savoir plus, consultez MIT Kerberos Documentation: Keytab (Documentation du MIT Kerberos : Keytab).
- SASL/PLAIN : choisissez cette méthode d'authentification pour spécifier les informations d'identification d'authentification. Deux options s'offrent à vous :

  - Utiliser AWS Secrets Manager (recommandé) : si vous sélectionnez cette option, vous pouvez enregistrer vos informations d'identification dans AWS Secrets Manager et autoriser l' AWS Glue accès aux informations en cas de besoin. Spécifiez le secret qui stocke les informations d'identification d'authentification SSL ou SASL.
  - Fournissez directement un nom d'utilisateur et un mot de passe.
- Authentification client SSL : si vous sélectionnez cette option, vous pouvez sélectionner l'emplacement du centre de stockage des clés client Kafka en naviguant sur Amazon S3. Vous pouvez également entrer le mot de passe du centre de stockage des clés client Kafka et le mot de passe de la clé client Kafka.

BigQuery Connexion Google

Les propriétés suivantes sont utilisées pour configurer une BigQuery connexion Google utilisée dans les tâches AWS Glue ETL. Pour de plus amples informations, veuillez consulter the section called "BigQuery connexions".

AWS Secret

Le nom secret d'un secret dans AWS Secrets Manager. AWS Glue Les tâches ETL se connecteront à Google à l' BigQuery aide de la `credentials` clé de votre secret.

Connexion Vertica

Les propriétés suivantes sont utilisées pour configurer une connexion Vertica utilisée dans les tâches AWS Glue ETL. Pour de plus amples informations, veuillez consulter the section called "Connexions Vertica".

Stockage des informations de connexion dans AWS Secrets Manager

Nous vous recommandons de l'utiliser AWS Secrets Manager pour fournir des informations d'identification de connexion pour votre banque de données. L'utilisation de Secrets Manager permet d' AWS Glue accéder à votre secret au moment de l'exécution pour les tâches ETL et les exécutions de robots d'exploration, et contribue à la sécurité de vos informations d'identification.

Conditions préalables

Pour utiliser Secrets Manager avec AWS Glue, vous devez accorder à votre rôle IAM AWS Glue l'autorisation de récupérer des valeurs secrètes. La politique AWS gérée `AWSGlueServiceRole` n'inclut pas AWS Secrets Manager les autorisations. Pour obtenir des exemples de politiques IAM, consultez la rubrique Exemple : Autorisation pour récupérer des valeurs de secrets dans le Guide de l'utilisateur AWS Secrets Manager.

En fonction de votre configuration réseau, vous devrez peut-être également créer un point de terminaison d'un VPC pour établir une connexion privée entre votre VPC et gestionaire de secrets Pour plus d'informations, consultez la rubrique Utilisation d'un point de terminaison d'un VPC AWS Secrets Manager.

Pour créer un secret pour AWS Glue

1. Suivez les instructions de la rubrique Création et gestion des secrets dans le AWS Secrets Manager Guide de l'utilisateur . L'exemple JSON suivant montre comment spécifier vos informations d'identification dans l'onglet Texte brut lorsque vous créez un secret pour AWS Glue.

   ```
   {
     "username": "EXAMPLE-USERNAME",
     "password": "EXAMPLE-PASSWORD"
   }
   ```

2. Associez votre secret à une connexion via l' AWS Glue Studio interface. Pour plus d'informations, consultez Creating connections for connectors dans le Guide de l'utilisateur AWS Glue Studio .

Ajouter une AWS Glue connexion

Vous pouvez vous connecter aux sources de données AWS Glue de Spark par programmation. Pour de plus amples informations, consultez Types et options de connexion pour ETL dans AWS Glue pour Spark.

Vous pouvez également utiliser la AWS Glue console pour ajouter, modifier, supprimer et tester des connexions. Pour plus d'informations sur AWS Glue les connexions, consultezConnexion aux données.

Rubriques

- Connexion à Adobe Analytics

- Connexion à Adobe Marketo Engage
- Connexion à Amazon Redshift dans AWS Glue Studio
- Connexion à Asana
- Connexion à Azure Cosmos DB dans AWS Glue Studio
- Connexion à Azure SQL dans AWS Glue Studio
- Connexion à Blackbaud Raiser's Edge NXT
- Connexion à CircleCI
- Connexion à Datadog
- Connexion à Docusign Monitor
- Connexion à Domo
- Connexion à Dynatrace
- Connexion à Facebook Ads
- Connexion à Facebook Page Insights
- Connexion à Freshdesk
- Connexion à Freshsales
- Connexion à Google Ads
- Connexion à Google Analytics 4
- Connexion à Google BigQuery en AWS Glue Studio
- Connexion à Google Search Console
- Connexion à Google Sheets
- Connexion à HubSpot
- Connexion à Instagram Ads
- Connexion à Intercom dans AWS Glue Studio
- Connexion à Jira Cloud
- Connexion à Kustomer
- Connexion à LinkedIn
- Connexion à Mailchimp.
- Connexion à Microsoft Dynamics 365 CRM

- Connexion à Microsoft Teams
- Connexion à Mixpanel
- Connexion à Monday
- Connexion à MongoDB dans AWS Glue Studio
- Connexion à Oracle NetSuite
- Connexion au OpenSearch service dans AWS Glue Studio
- Connexion à Okta
- Connexion à PayPal
- Connexion à Pendo
- Connexion à Pipedrive
- Connexion à Productboard
- Connexion à QuickBooks
- Connexion à une API REST
- Connexion à Salesforce
- Connexion à Salesforce Marketing Cloud
- Connexion à Salesforce Commerce Cloud
- Connexion à Salesforce Marketing Cloud Account Engagement
- Connexion à SAP HANA dans AWS Glue Studio
- Connexion à SAP OData
- Connexion à SendGrid
- Connexion à ServiceNow
- Connexion à Slack dans AWS Glue Studio
- Connexion à Smartsheet
- Connexion à Snapchat Ads dans AWS Glue Studio
- Connexion à Snowflake dans AWS Glue Studio
- Connexion à Stripe dans AWS Glue Studio
- Connexion à Teradata Vantage dans AWS Glue Studio
- Connexion à Twilio

- Connexion à Vertica dans AWS Glue Studio
- Connexion à WooCommerce
- Connexion à Zendesk
- Connexion à Zoho CRM
- Connexion à Zoom Meetings
- Ajout d'une connexion JDBC à l'aide de vos propres pilotes JDBC

Connexion à Adobe Analytics

Adobe Analytics est une plateforme d'analyse de données robuste qui collecte des données à partir d'expériences numériques multicanales prenant en charge le parcours client et qui fournit des outils pour analyser les données. Il s'agit d'une plateforme couramment utilisée par les spécialistes du marketing et les analystes commerciaux à des fins d'analytique métier. Si vous êtes un utilisateur d'Adobe Analytics, vous pouvez vous connecter AWS Glue à votre compte Adobe Analytics. Vous pouvez ensuite utiliser Adobe Analytics comme source de données dans vos tâches ETL. Exécutez ces tâches pour transférer des données entre Adobe Analytics et AWS des services ou d'autres applications prises en charge.

Rubriques

- AWS Glue support pour Adobe Analytics
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration d'Adobe Analytics
- Configuration des connexions Adobe Analytics
- Lecture à partir d'entités Adobe Analytics
- Options de connexion Adobe Analytics
- Création d'un compte Adobe Analytics
- Limitations

AWS Glue support pour Adobe Analytics

AWS Glue prend en charge Adobe Analytics comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser les tâches AWS Glue ETL pour interroger les données d'Adobe Analytics.

Prise en charge en tant que cible ?

Non.

Versions d'API Adobe Analytics prises en charge

v2.0

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les AWS autorisations requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Si vous ne voulez pas utiliser la méthode ci-dessus, vous pouvez également utiliser les politiques IAM gérées suivantes :

- AWSGlueServiceRole — Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources

spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.

- AWSGlueConsoleFullAccess — Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration d'Adobe Analytics

Avant de pouvoir AWS Glue effectuer un transfert depuis Adobe Analytics, vous devez satisfaire aux exigences suivantes :

Configuration requise

- Vous disposez d'un compte Adobe Analytics avec adresse e-mail et mot de passe. Pour plus d'informations sur la création d'un compte, consultez Creating a Adobe Analytics account.
- Votre compte Adobe Analytics est activé pour accéder à l'API. L'accès à l'API est activé par défaut pour les éditions Select, Prime et Ultimate.

Si vous répondez à ces exigences, vous êtes prêt à vous connecter AWS Glue à votre compte Adobe Analytics. Pour les connexions classiques, vous n'avez rien d'autre à faire dans Adobe Analytics.

Configuration des connexions Adobe Analytics

Adobe Analytics prend en charge le type d'autorisation `AUTHORIZATION_CODE` pour `OAuth2`.

Ce type d'autorisation est considéré comme `OAuth` tridimensionnel, car il repose sur la redirection des utilisateurs vers le serveur d'autorisation tiers pour authentifier l'utilisateur. Les utilisateurs peuvent choisir de créer leur propre application connectée dans Adobe Analytics et de fournir leur propre identifiant client et leur propre secret client lors de la création de connexions via la AWS Glue console. Dans ce scénario, ils seront toujours redirigés vers Adobe Analytics pour se connecter et autoriser AWS Glue à accéder à leurs ressources.

Ce type d'autorisation donne lieu à un jeton d'actualisation et à un jeton d'accès. Le jeton d'accès est de courte durée et peut être actualisé automatiquement sans interaction de l'utilisateur à l'aide du jeton d'actualisation.

Pour consulter la documentation publique d'Adobe Analytics sur la création d'une application connectée pour le OAuth flux AUTHORIZATION_CODE, consultez Adobe Analytics. APIs

Pour configurer une connexion Adobe Analytics :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes :

   Pour l'application connectée gérée par le client : le secret doit contenir le secret du consommateur de l'application connectée avec `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` comme clé.

   > **Note**
   >
   > Il est indispensable de créer un secret par connexion dans AWS Glue.

2. Dans AWS Glue Studio, créez une connexion sous Connexions de données en suivant les étapes ci-dessous :
   a. Lorsque vous sélectionnez un type de connexion, sélectionnez Adobe Analytics.
   b. Indiquez l'`x_api_key, instanceUrl` d'Adobe Analytics à laquelle vous souhaitez vous connecter.
   c. Sélectionnez le rôle IAM pour lequel AWS Glue vous pouvez assumer et qui dispose des autorisations nécessaires pour effectuer les actions suivantes :

      JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
```

```
}
```

d. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.

e. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.

3. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.

Lecture à partir d'entités Adobe Analytics

Conditions préalables

Un objet Adobe Analytics à partir duquel vous souhaitez lire. Reportez-vous au tableau des entités prises en charge ci-dessous pour vérifier les entités disponibles.

Entités prises en charge

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Annotation	Oui	Oui	Oui	Oui	Non
Calculated Metrics	Oui	Oui	Oui	Oui	Non
Calculate d Metrics Function	Oui	Non	Non	Oui	Non
Component Metadata Shares	Oui	Oui	Non	Oui	Non
Date Ranges	Oui	Oui	Non	Oui	Non
Dimensions	Oui	Non	Non	Oui	Non
Métriques	Oui	Non	Non	Oui	Non

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Projets	Oui	Non	Non	Oui	Non
Reports Top Item	Oui	Oui	Non	Oui	Non
Segments	Oui	Oui	Oui	Oui	Non
Usage Logs	Oui	Oui	Non	Oui	Non

Exemple

```
adobeAnalytics_read = glueContext.create_dynamic_frame.from_options(
     connection_type="adobeanalytics",
     connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "annotation/ex*****",
        "API_VERSION": "v2.0"
 })
```

Détails des entités et des champs Adobe Analytics

- Annotations
- Calculated Metrics
- Component Meta Data
- Date Ranges
- Dimensions
- Métriques
- Projects
- Rapports
- Segments
- Utilisateurs
- Usage Logs

Options de connexion Adobe Analytics

Les options de connexion pour Adobe Analytics sont les suivantes :

- `ENTITY_NAME`(Chaîne) - (Obligatoire) Utilisé pour la lecture/l'écriture. Le nom de votre objet dans Adobe Analytics.
- `API_VERSION`(Chaîne) - (Obligatoire) Utilisé pour la lecture/l'écriture. Version d'API Rest Adobe Analytics que vous souhaitez utiliser. Par exemple : v2.0.
- `X_API_KEY`(Chaîne) - (Obligatoire) Utilisé pour la lecture/l'écriture. Il est nécessaire d'authentifier le développeur ou l'application qui envoie des demandes à l'API.
- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.

Création d'un compte Adobe Analytics

1. Inscrivez-vous au programme de partenaire d'échange en accédant au programme Adobe Partner.
2. Choisissez Rejoindre le programme d'échange.
3. Inscrivez-vous ou créez un compte en utilisant votre adresse e-mail professionnelle.
4. Dans la boîte à suggestions, sélectionnez l'entreprise appropriée qui possède un abonnement à un produit Adobe Analytics.
5. Assurez-vous que le compte est enregistré auprès d'une organisation valide (dans la liste disponible) disposant d'un abonnement Adobe Analytics actif.
6. Après l'approbation de l'administration de l'entreprise, activez votre compte en cliquant sur le lien contenu dans l'e-mail d'approbation.

Vérifiez si le compte que vous avez créé a accès au service Adobe Analytics.

1. Connectez-vous à Adobe Admin Console.
2. Vérifiez le nom de l'organisation dans le coin supérieur droit de la page pour vous assurer que vous vous êtes connecté à la bonne entreprise.

3. Sélectionnez Produits et vérifiez si Adobe Analytics est disponible.

> **Note**
>
> Si aucune organisation n'est disponible, ou si le produit Adobe Analytics est grisé ou n'est pas disponible, il est probable que votre compte n'est pas associé à une organisation n' and/or ayant aucun abonnement Adobe Analytics actif. Contactez votre administrateur système pour demander l'accès à ce service pour votre compte.

Création d'un projet et d'informations d'identification **`OAuth2.0`**

1. Connectez-vous au compte Adobe Analytics sur lequel vous souhaitez créer l'application OAuth 2.0.
2. Sélectionnez Projet, puis Créer un projet.
3. Pour ajouter un projet, sélectionnez Ajouter au projet, puis API.
4. Sélectionnez l'API Adobe Analytics.
5. Sélectionnez OAUTH comme authentification utilisateur.
6. Sélectionnez Web en tant que `OAUTH` et indiquez l'URI de redirection.

   Pour l'URI de redirection et son modèle, consultez ce qui suit :

   - URI de redirection par défaut `OAuth 2.0` : l'URI de redirection par défaut est l'URL de la page à laquelle Adobe accèdera pendant le processus d'authentification. Par exemple, `https://ap-southeast-2.console.aws.amazon.com/appflow/oauth`
   - OAuth Modèle d'URI de redirection 2.0 — Un modèle d'URI de redirection est un chemin d'URI (ou une liste de chemins séparés par des virgules) vers lequel Adobe peut être redirigé (si demandé) une fois le flux de connexion terminé. Par exemple, `https://ap-southeast-2\\.console\\.aws\\.amazon\\.com`
7. Ajoutez les portées suivantes :

   - `openid`
   - `read_organizations`
   - `additional_info.projectedProductContext`
   - `additional_info.job_function`
8. Choisissez Enregistrer les informations d'identification.

9. Une fois l'application créée, copiez les valeurs `Client ID` et `Client Secret` dans un fichier texte.

Limitations

Les restrictions du connecteur Adobe Analytics sont les suivantes :

- Adobe Analytics ne prend pas en charge le partitionnement basé sur les champs et basé sur les enregistrements. Le partitionnement basé sur les champs n'est pas pris en charge, car vous ne pouvez pas interroger les champs que vous partitionnez. Le partitionnement basé sur les enregistrements ne peut pas être pris en charge, car il n'existe aucune disposition permettant d'obtenir un « décalage » pour la pagination.
- Dans l'entité `Report Top Item`, les paramètres de requête `startDate` et `endDate` ne fonctionnent pas comme prévu. La réponse n'est pas filtrée en fonction de ces paramètres, ce qui entraîne des problèmes avec le filtre et le flux incrémentiel pour cette entité.
- Pour les entités `Annotation`, `Calculated Metrics`, `Calculated Metrics Function`, `Date Ranges`, `Dimension`, `Metric`, `Project`, `Report Top Items` et `Segment`, le paramètre de requête `locale` indique la langue à utiliser pour les sections localisées des réponses et ne filtre pas les enregistrements. Par exemple, `locale="ja_JP"` affichera les données en japonais.
- Entité `Report Top Item` : le filtre est activé et les champs `dateRange` et `lookupNoneValues` ne fonctionnent pas actuellement.
- Entité `Segment` : avec une valeur de filtre `includeType="templates"`, les filtres sur les autres champs ne fonctionnent pas.
- Entité `Date Range` : le filtre sur le champ `curatedRsid` ne fonctionne pas.
- Entité `Metric entity` : le filtre sur un champ segmentable avec une valeur « false » donne un résultat à la fois pour une valeur true et une valeur false.

Connexion à Adobe Marketo Engage

Adobe Marketo Engage est une plateforme d'automatisation du marketing qui permet aux spécialistes du marketing de gérer des programmes et des campagnes multicanaux personnalisés destinés aux prospects et aux clients.

Rubriques

- AWS Glue support pour Adobe Marketo Engage

- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration d'Adobe Marketo Engage
- Configuration des connexions Adobe Marketo Engage
- Lecture à partir d'entités Adobe Marketo Engage
- Écriture aux entités Adobe Marketo Engage
- Options de connexion Adobe Marketo Engage
- Restrictions et remarques relatives au connecteur Adobe Marketo Engage

AWS Glue support pour Adobe Marketo Engage

AWS Glue prend en charge Adobe Marketo Engage comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser les tâches AWS Glue ETL pour interroger les données d'Adobe Marketo Engage.

Prise en charge en tant que cible ?

Oui. Vous pouvez utiliser des tâches AWS Glue ETL pour écrire des données dans Adobe Marketo.

Versions de l'API Adobe Marketo Engage prises en charge

Les versions de l'API Adobe Marketo Engage suivantes sont prises en charge :

- v1

Pour connaître la prise en charge des entités par version, consultez la section Entités prises en charge pour Source.

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les autorisations AWS IAM requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
```

```
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Si vous ne voulez pas utiliser la méthode ci-dessus, vous pouvez également utiliser les politiques IAM gérées suivantes :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration d'Adobe Marketo Engage

Avant de pouvoir AWS Glue transférer des données d'Adobe Marketo Engage, vous devez satisfaire aux exigences suivantes :

Configuration requise

Les exigences minimales sont les suivantes :

- Vous possédez un compte Adobe Marketo Engage avec des informations d'identification client.
- Votre compte Adobe Marketo Engage dispose d'un accès à l'API avec une licence valide.

Si vous répondez à ces exigences, vous êtes prêt à vous connecter AWS Glue à votre compte Adobe Marketo Engage. Pour les connexions classiques, vous n'avez rien d'autre à faire dans Adobe Marketo Engage.

Obtenir des informations d'identification OAuth 2.0

Pour obtenir des informations d'identification d'API afin de passer des appels authentifiés vers votre instance, consultez Adobe Marketo Rest API.

Configuration des connexions Adobe Marketo Engage

Adobe Marketo Engage prend en charge le type de subvention CLIENT CREDENTIALS pour OAuth2.

- Ce type de subvention est considéré comme bidimensionnel OAuth 2.0 car il est utilisé par les clients pour obtenir un jeton d'accès en dehors du contexte d'un utilisateur. AWS Glue est capable d'utiliser l'ID client et le secret du client pour authentifier l'Adobe Marketo Engage APIs fournis par les services personnalisés que vous définissez.
- Chaque service personnalisé appartient à un utilisateur régi uniquement par l'API, qui dispose d'un ensemble de rôles et d'autorisations autorisant le service à effectuer des actions spécifiques. Un jeton d'accès est associé à un seul service personnalisé.
- Ce type d'autorisation donne lieu à un jeton d'accès de courte durée qui peut être renouvelé en appelant un point de terminaison d'identité.
- Pour consulter la documentation publique d'Adobe Marketo Engage OAuth 2.0 avec les informations d'identification du client, voir Authentification dans le guide du développeur d'Adobe Marketo Engage.

Pour configurer une connexion Adobe Marketo Engage :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes :
   a. Pour l'application connectée gérée par le client, le secret doit contenir le secret du consommateur de l'application connectée avec `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` comme clé.
   b. Remarque : vous devez créer un secret par connexion dans AWS Glue.

2. Dans AWS Glue Glue Studio, créez une connexion sous Data Connections en suivant les étapes ci-dessous :
   a. Lorsque vous sélectionnez un Type de connexion, sélectionnez Adobe Marketo Engage.
   b. Indiquez l'`INSTANCE_URL` de l'instance Adobe Marketo Engage à laquelle vous souhaitez vous connecter.
   c. Sélectionnez le rôle AWS IAM qui AWS Glue peut assumer et dispose des autorisations nécessaires pour les actions suivantes :

      JSON

      ```
      {
        "Version":"2012-10-17",
        "Statement": [
          {
            "Effect": "Allow",
            "Action": [
              "secretsmanager:DescribeSecret",
              "secretsmanager:GetSecretValue",
              "secretsmanager:PutSecretValue",
              "ec2:CreateNetworkInterface",
              "ec2:DescribeNetworkInterfaces",
              "ec2:DeleteNetworkInterface"
            ],
            "Resource": "*"
          }
        ]
      }
      ```

   d. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.
   e. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.
3. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.

Lecture à partir d'entités Adobe Marketo Engage

Prérequis

Un objet Adobe Marketo Engage à partir duquel vous souhaitez lire. Vous aurez besoin du nom de l'objet, comme leads, activities ou customobjects. Les tableaux suivants présentent les entités prises en charge.

Entités prises en charge pour source (synchrone) :

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
leads	Oui	Oui	Non	Oui	Non
activités	Oui	Oui	Non	Oui	Non
customobj ects	Oui	Oui	Non	Oui	Non

Entités prises en charge pour source (asynchrone) :

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
leads	Oui	Non	Non	Oui	Oui
activités	Oui	Non	Non	Oui	Non
customobj ects	Oui	Non	Non	Oui	Oui

Exemple :

```
adobe-marketo-engage_read = glueContext.create_dynamic_frame.from_options(
    connection_type="adobe-marketo-engage",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "leads",
        "API_VERSION": "v2",
        "INSTANCE_URL": "https://539-t**-6**.mktorest.com"
    }
```

Détails des entités et des champs Adobe Marketo Engage :

Entités avec métadonnées statiques :

Entité	Champ	Type de données	Opérateurs pris en charge
activités	SinceDateTime (uniquement pris en charge en mode synchrone)	DateTime	>= (uniquement pour le mode synchrone)
	createdAt (uniquement pris en charge en mode asynchrone)	DateTime	between (uniquement pour le mode asynchrone)
	activitiesTypeId	Entier	=
	adobe-marketo-engageGUID	Long	= (uniquement pour le mode synchrone)
	leadId	Long	N/A
	activityDate	DateTime	N/A
	campaignId	Long	N/A
	primaryAttributeValueId	Entier	N/A
	primaryAttributeValue	String	N/A
	attributs	String	N/A

Entités avec métadonnées dynamiques :

Pour les entités suivantes, Adobe Marketo Engage fournit des points de terminaison permettant de récupérer les métadonnées de manière dynamique, afin que la prise en charge des opérateurs soit capturée au niveau du type de données pour chaque entité.

Entité	Type de données	Opérateurs pris en charge
leads	Entier	= (uniquement pour le mode synchrone)
	DateTime	between (uniquement pour le mode asynchrone)
	String	= (uniquement pour le mode synchrone)
	Long	N/A
	Booléen	N/A
	Date	N/A
	Float	N/A
customobjects	Entier	N/A
	DateTime	between (uniquement pour le mode asynchrone)
	String	= (uniquement pour le mode synchrone)
	Date	N/A
	Long	N/A
	Booléen	N/A
	Float	N/A

Requêtes de partitionnement

Vous pouvez indiquer les options Spark supplémentaires `PARTITION_FIELD`, `LOWER_BOUND`, `UPPER_BOUND` et `NUM_PARTITIONS` si vous souhaitez utiliser la simultanéité dans Spark. Avec ces

paramètres, la requête d'origine serait divisée en NUM_PARTITIONS nombres de sous-requêtes pouvant être exécutées simultanément par les tâches Spark.

- PARTITION_FIELD : le nom du champ à utiliser pour partitionner la requête.
- LOWER_BOUND : une valeur limite inférieure inclusive du champ de partition choisi.

  Pour le DateTime champ, nous acceptons la valeur au format ISO.

  Exemple de valeur valide :

  ```
  "2024-07-01T00:00:00.000Z"
  ```

- UPPER_BOUND : une valeur limite supérieure exclusive du champ de partition choisi.
- NUM_PARTITIONS : le nombre de partitions.

Le tableau suivant décrit les détails de prise en charge dans les champs de partitionnement d'entités :

Nom de l'entité	Champs de partitionnement	Type de données
leads	createdAt	DateTime
	updateAt	DateTime
customobjects	updatedAt	DateTime

Exemple :

```
adobe-marketo-engage_read = glueContext.create_dynamic_frame.from_options(
    connection_type="adobe-marketo-engage",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "leads",
        "API_VERSION": "v1",
        "PARTITION_FIELD": "createdAt"
        "LOWER_BOUND": "2024-07-01T00:00:00.000Z"
        "UPPER_BOUND": "2024-07-02T00:00:00.000Z"
        "NUM_PARTITIONS": "10"
    }
```

Écriture aux entités Adobe Marketo Engage

Conditions préalables

- Un objet Adobe Marketo auquel vous souhaitez écrire. Vous aurez besoin du nom de l'objet, comme `leads` ou `customobjects`.
- Le connecteur Adobe Marketo prend en charge trois opérations d'écriture :
  - INSERT
  - UPSERT
  - UPDATE
- Pour les opérations d'écriture `UPSERT` et `UPDATE`, vous devez proposer l'option `ID_FIELD_NAMES` afin de spécifier le champ d'ID pour les enregistrements. Lorsque vous travaillez avec l'entité `leads`, utilisez `email` en tant que `ID_FIELD_NAMES` pour les opérations `UPSERT` et `id` pour les opérations `UPDATE`. Pour l'entité `customobjects`, utilisez `marketoGUID` en tant que `ID_FIELD_NAMES` pour les opérations `UPDATE` et `UPSERT`.

Entités prises en charge pour destination (synchrone)

Nom de l'entité	Sera pris en charge en tant que connecteur de destination	Peut être inséré	Peut être mis à jour	Peut faire l'objet d'un upsert
leads	Oui	Oui (groupé)	Oui (groupé)	Oui (groupé)
customobjects	Oui	Oui (groupé)	Oui (groupé)	Oui (groupé)

Exemple :

Opération INSERT :

```
marketo_write = glueContext.write_dynamic_frame.from_options(
    frame=frameToWrite,
    connection_type="marketo",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "leads",
```

```
        "API_VERSION": "v1",
        "WRITE_OPERATION": "INSERT"
    }
```

Opération UPDATE :

```
marketo_write = glueContext.write_dynamic_frame.from_options(
    frame=frameToWrite,
    connection_type="marketo",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "leads",
        "API_VERSION": "v1",
        "WRITE_OPERATION": "UPDATE",
        "ID_FIELD_NAMES": "id"
    }
```

> **Note**
>
> Pour les entités `leads` et `customobjects`, Adobe Marketo fournit des points de terminaison permettant de récupérer les métadonnées de manière dynamique afin que les champs inscriptibles soient identifiés à partir de la réponse de l'API Marketo.

Options de connexion Adobe Marketo Engage

Les options de connexion pour Adobe Marketo Engage sont les suivantes :

- `ENTITY_NAME`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Le nom de votre objet dans Adobe Marketo Engage.
- `API_VERSION`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Version d'API Rest d'Adobe Marketo Engage que vous souhaitez utiliser. Par exemple : v1.
- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.
- `PARTITION_FIELD`(Chaîne) - Utilisé pour la lecture. Champ à utiliser pour partitionner la requête.

- `LOWER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite inférieure inclusive du champ de partition choisi.
- `UPPER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS`(Entier) - Par défaut : 1. Utilisé pour la lecture. Nombre de partitions pour la lecture.
- `TRANSFER_MODE`(Chaîne) - Par défaut : SYNC. Utilisé pour la lecture asynchrone.
- `WRITE_OPERATION`(Chaîne) - Par défaut : INSERT. Utilisé pour l'écriture La valeur doit être INSERT, UPDATE, UPSERT.
- `ID_FIELD_NAMES`(Chaîne) - Valeur par défaut : null. Nécessaire pour UPDATE et UPSERT.

Restrictions et remarques relatives au connecteur Adobe Marketo Engage

Voici les restrictions ou les remarques relatives au connecteur Adobe Marketo Engage :

- « SinceDateTime » et « activityTypeId » sont des paramètres de filtre obligatoires pour l'entité Sync Activities.
- Les abonnements se voient allouer 50 000 appels d'API par jour (réinitialisés tous les jours à minuit CST). Une capacité quotidienne supplémentaire peut être achetée dans le cadre d'un abonnement Adobe Marketo Engage. Pour référence, consultez Adobe Marketo Rest API.
- La durée maximale pour le filtre de plage de dates (`createdAt` ou `updatedAt`) est de 31 jours. Pour référence, consultez Bulk Extract - Marketo Developers.
- Les abonnements se voient attribuer un maximum de 10 tâches d'extraction en bloc dans la file d'attente à un moment précis. Pour référence, consultez Bulk Extract - Marketo Developers.
- Par défaut, les tâches d'extraction sont limitées à 500 Mo par jour (réinitialisation quotidienne à minuit CST). Une capacité quotidienne supplémentaire peut être achetée dans le cadre d'un abonnement Adobe Marketo Engage. Pour référence, consultez Bulk Extract - Marketo Developers.
- Le nombre maximum de tâches d'exportation simultanées est de deux. Pour référence, consultez Bulk Extract - Marketo Developers.
- Le nombre maximum de tâches d'exportation en attente (y compris les tâches actuellement en cours d'exportation) est de dix. Pour référence, consultez Bulk Extract - Marketo Developers.
- La taille de fichier maximale autorisée est de 1 Go pour l'extraction à partir d'une tâche groupée.
- Une fois qu'une tâche asynchrone est créée, la période de conservation des fichiers sera de sept jours avant l'expiration. Pour référence, consultez Bulk Extract - Marketo Developers.

- `createdAt` ou `updatedAt` sont des paramètres de filtre obligatoires pour l'entité Async Leads.
- Pour l'entité Async Activities, `createdAt` est un paramètre de filtre obligatoire.
- Pour l'entité Async Custom Object, `updatedAt` est un paramètre de filtre obligatoire.
- Lorsqu'ils utilisent des connecteurs AWS Glue SaaS, les utilisateurs ne peuvent pas identifier les enregistrements spécifiques qui ont échoué lors d'une opération d'écriture sur les plateformes SaaS de destination en cas de défaillance partielle.
- Lorsque vous utilisez les opérations d'écriture Sync, tous les champs contenant des valeurs nulles dans le fichier d'entrée seront automatiquement supprimés et ne seront pas envoyés à la plateforme SaaS.
- Vous pouvez créer ou mettre à jour jusqu'à 300 enregistrements par lot pour l'écriture Sync.

Pour plus d'informations, consultez Adobe Marketo Engage Integration Best Practices et Bulk Extract.

Connexion à Amazon Redshift dans AWS Glue Studio

> **Note**
>
> Vous pouvez utiliser Spark AWS Glue pour lire et écrire dans des tables de Amazon Redshift bases de données extérieures àAWS Glue Studio. Pour configurer Amazon Redshift des AWS Glue tâches par programmation, voirConnexions Redshift.

AWS Gluefournit un support intégré pour Amazon Redshift. AWS Glue Studiofournit une interface visuelle permettant de se connecter Amazon Redshift, de créer des tâches d'intégration de données et de les exécuter sur un environnement d'exécution Spark AWS Glue Studio sans serveur.

Rubriques

- Création d'une Amazon Redshift connexion
- Création d'un nœud Amazon Redshift source
- Création d'un nœud Amazon Redshift cible
- Options avancées

Création d'une Amazon Redshift connexion

Autorisations nécessaires

Des autorisations supplémentaires sont nécessaires pour utiliser les Amazon Redshift clusters et les environnements Amazon Redshift sans serveur. Pour plus d'informations sur la façon d'ajouter des autorisations aux tâches ETL, consultez Review IAM permissions needed for ETL jobs.

- décalage vers le rouge : DescribeClusters
- redshift-serverless : ListWorkgroups
- redshift-serverless : ListNamespaces

Présentation de

Lorsque vous ajoutez une Amazon Redshift connexion, vous pouvez choisir une Amazon Redshift connexion existante ou en créer une nouvelle lorsque vous ajoutez une source de données - nœud Redshift. AWS Glue Studio

AWS Glueprend en charge à la fois les Amazon Redshift clusters et les environnements Amazon Redshift sans serveur. Lorsque vous créez une connexion, les environnements Amazon Redshift sans serveur affichent l'étiquette sans serveur à côté de l'option de connexion.

Pour plus d'informations sur la création d'une Amazon Redshift connexion, consultez la section Déplacement de données vers et depuis Amazon Redshift.

Création d'un nœud Amazon Redshift source

Autorisations nécessaires

AWS Glue Studio les tâches utilisant Amazon Redshift des sources de données nécessitent des autorisations supplémentaires. Pour plus d'informations sur la façon d'ajouter des autorisations aux tâches ETL, consultez Review IAM permissions needed for ETL jobs.

Les autorisations suivantes sont nécessaires pour utiliser une Amazon Redshift connexion.

- données redshift : ListSchemas
- données redshift : ListTables
- données redshift : DescribeTable

- données redshift : ExecuteStatement
- données redshift : DescribeStatement
- données redshift : GetStatementResult

Ajouter une source Amazon Redshift de données

Pour ajouter un nœud Source de données – Amazon Redshift :

1. Choisissez le type Amazon Redshift d'accès :

   - Connexion directe aux données (recommandée) : choisissez cette option si vous souhaitez accéder directement à vos données Amazon Redshift . Il s'agit de l'option recommandée, mais également de l'option par défaut.
   - Data Catalog tables : choisissez cette option si vous souhaitez utiliser des tables du catalogue de données.
2. Si vous choisissez Connexion directe aux données, choisissez la connexion pour votre source de Amazon Redshift données. Cela suppose que la connexion existe déjà et que vous pouvez la sélectionner parmi les connexions existantes. Si vous devez créer une connexion, choisissez Créer une connexion Redshift. Pour en savoir plus, consultez Overview of using connectors and connections .

   Une fois que vous avez choisi une connexion, vous pouvez afficher ses propriétés en cliquant sur Afficher les propriétés. Les informations relatives à la connexion sont visibles, notamment l'URL, les groupes de sécurité, le sous-réseau, la zone de disponibilité, la description et les horodatages de création (UTC) et de dernière mise à jour (UTC).
3. Choisissez une option Amazon Redshift de source :

   - Choisissez une seule table : il s'agit de la table qui contient les données auxquelles vous souhaitez accéder à partir d'une seule Amazon Redshift table.
   - Saisir une requête personnalisée : vous permet d'accéder à un jeu de données à partir de plusieurs tables Amazon Redshift en fonction de votre requête personnalisée.
4. Si vous avez choisi une seule table, choisissez le Amazon Redshift schéma. La liste des schémas disponibles parmi lesquels choisir est déterminée par la table sélectionnée.

   Vous pouvez également choisir Saisir une requête personnalisée. Choisissez cette option pour accéder à un jeu de données personnalisé à partir de plusieurs tables Amazon Redshift . Lorsque vous choisissez cette option, entrez la Amazon Redshift requête.

Lorsque vous vous connectez à un environnement Amazon Redshift sans serveur, ajoutez l'autorisation suivante à la requête personnalisée :

```
GRANT SELECT ON ALL TABLES IN <schema> TO PUBLIC
```

Vous pouvez choisir Déduire un schéma pour lire le schéma en fonction de la requête que vous avez saisie. Vous pouvez également choisir l'éditeur de requêtes Open Redshift pour saisir une Amazon Redshift requête. Pour plus d'informations, consultez Interrogation d'une base de données à l'aide de l'éditeur de requête.

5. Dans Performances et sécurité, choisissez le répertoire intermédiaire Amazon S3 et le rôle IAM.
   - Répertoire intermédiaire Amazon S3 : choisissez l'emplacement Amazon S3 pour les données intermédiaires temporaires.
   - Rôle IAM : choisissez le rôle IAM qui peut écrire sur l'emplacement Amazon S3 que vous avez sélectionné.
6. Dans Paramètres Redshift personnalisés – facultatif, saisissez le paramètre et la valeur.

Création d'un nœud Amazon Redshift cible

Autorisations nécessaires

AWS Glue Studio les tâches utilisant une cible de Amazon Redshift données nécessitent des autorisations supplémentaires. Pour plus d'informations sur la façon d'ajouter des autorisations aux tâches ETL, consultez Review IAM permissions needed for ETL jobs.

Les autorisations suivantes sont nécessaires pour utiliser une Amazon Redshift connexion.

- données redshift : ListSchemas
- données redshift : ListTables

Ajouter un nœud Amazon Redshift cible

Pour créer un nœud Amazon Redshift cible :

1. Choisissez une Amazon Redshift table existante comme cible ou entrez un nouveau nom de table.
2. Lorsque vous utilisez le nœud cible Cible de données – Redshift, vous pouvez choisir l'une des options suivantes :

   - APPREND : si une table existe déjà, insérez toutes les nouvelles données dans la table sous forme d'insertion. Si la table n'existe pas, créez-la, puis insérez toutes les nouvelles données.

     Cochez également la case si vous souhaitez mettre à jour (UPSERT) les enregistrements existants dans la table cible. La table doit d'abord exister, sinon l'opération échouera.
   - MERGE : AWS Glue met à jour ou ajoute des données à votre table cible en fonction des conditions que vous spécifiez.

     > **Note**
     >
     > Pour utiliser l'action de fusion dansAWS Glue, vous devez activer la fonctionnalité de Amazon Redshift fusion. Pour savoir comment activer la fusion pour votre Amazon Redshift instance, consultez MERGE (aperçu).

     Choisissez parmi les options :

     - Choisir des clés et des actions simples : choisissez les colonnes à utiliser comme clés de correspondance entre les données source et votre jeu de données cible.

       Spécifiez les options suivantes lorsqu'elles correspondent :

       - Mettez à jour l'enregistrement dans votre jeu de données cible avec les données de la source.
       - Supprimez l'enregistrement dans votre jeu de données cible.

       Spécifiez les options suivantes lorsqu'elles ne correspondent pas :

       - Insérez les données source en tant que nouvelle ligne dans votre jeu de données cible.
       - Ne rien faire.

    - Saisir une instruction MERGE personnalisée : vous pouvez ensuite choisir Valider l'instruction de fusion pour vérifier si l'instruction est valide ou non.
- TRUNCATE : si une table existe déjà, tronquez les données de la table en effaçant d'abord le contenu de la table cible. Si la troncature est réussie, insérez toutes les données. Si la table n'existe pas, créez-la, puis insérez toutes les nouvelles données. Si la troncature échoue, l'opération échouera.
- DROP : si une table existe déjà, supprimez les métadonnées et les données de la table. Si la suppression est réussie, insérez toutes les données. Si la table n'existe pas, créez-la, puis insérez toutes les nouvelles données. Si la suppression échoue, l'opération échouera.
- CREATE : créez une table avec le nom par défaut. Si le nom de la table existe déjà, créez-en une avec le suffixe `job_datetime` au nom pour garantir son unicité. Toutes les données seront alors insérées dans la nouvelle table. Si la table existe, le suffixe sera ajouté au nom final de la table. Si elle n'existe pas, une table sera créée. Dans les deux cas, une nouvelle table sera créée.

Options avancées

Consultez la section Utilisation du connecteur Amazon Redshift Spark sur AWS Glue.

Connexion à Asana

Asana est une solution de collaboration d'équipe cloud qui aide les équipes à organiser, à planifier et à exécuter des tâches et des projets. Si vous utilisez Asana, votre compte contient des données sur vos espaces de travail, vos projets, vos tâches, vos équipes, etc. Vous pouvez transférer des données d'Asana vers certains services AWS ou d'autres applications compatibles.

Rubriques

- AWS Glue support pour Asana
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration d'Asana
- Configuration de connexions Asana
- Lecture à partir d'entités Asana
- Options de connexion Asana
- Création d'un compte Asana
- Limitations

AWS Glue support pour Asana

AWS Glue prend en charge Asana comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser les tâches AWS Glue ETL pour interroger les données d'Asana.

Prise en charge en tant que cible ?

Non.

Versions d’API Asana prises en charge

1.0

Politiques contenant les opérations API pour la création et l’utilisation des connexions

L'exemple de politique suivant décrit les AWS autorisations requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Si vous ne voulez pas utiliser la méthode ci-dessus, vous pouvez également utiliser les politiques IAM gérées suivantes :

- AWSGlueServiceRole — Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess — Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration d'Asana

Avant de pouvoir AWS Glue effectuer un transfert depuis Asana, vous devez remplir les conditions suivantes :

Configuration requise

- Vous disposez d'un compte Asana avec adresse e-mail et mot de passe. Pour plus d'informations sur la création d'un compte, consultez Creating a Asana account.
- Vous devez avoir créé AWS un compte avec l'accès au service à AWS Glue.
- Assurez-vous d'avoir créé l'une des ressources suivantes dans votre compte Asana :
  - Une application pour développeurs qui prend en charge l'authentification `OAuth 2.0`. Pour plus d'instructions, consultez la documentation destinée OAuthaux développeurs d'Asana. Sinon, consultez the section called "Création d'un compte Asana".
  - Un jeton d'accès personnel. Pour plus d'informations, consultez le jeton d'accès personnel https://developers.asana.com/docs/personal-access-token dans la documentation destinée aux développeurs Asana.

Si vous répondez à ces exigences, vous êtes prêt à vous connecter AWS Glue à votre compte Adobe Analytics. Pour les connexions classiques, vous n'avez rien d'autre à faire dans Adobe Analytics.

Configuration de connexions Asana

Asana prend en charge le type d'autorisation `AUTHORIZATION_CODE` pour `OAuth2`.

Ce type d'autorisation est considéré comme `0Auth` tridimensionnel, car il repose sur la redirection des utilisateurs vers le serveur d'autorisation tiers pour authentifier l'utilisateur. Les utilisateurs peuvent choisir de créer leur propre application connectée sur Asana et de fournir leur propre identifiant client et leur secret client lors de la création de connexions via la AWS Glue console. Dans ce scénario, ils seront toujours redirigés vers Asana pour se connecter et autoriser l'accès AWS Glue à leurs ressources.

Ce type d'autorisation donne lieu à un jeton d'actualisation et à un jeton d'accès. Le jeton d'accès est de courte durée et peut être actualisé automatiquement sans interaction de l'utilisateur à l'aide du jeton d'actualisation.

Pour consulter la documentation publique d'Asana sur la création d'une application connectée pour `AUTHORIZATION_CODE 0Auth` Flow, consultez Asana APIs.

Pour configurer une connexion Asana :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes :
   - Pour l'application connectée gérée par le client : le secret doit contenir le secret du consommateur de l'application connectée avec `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` comme clé.
   -
     > **Note**
     >
     > Vous devez créer un secret pour la connexion dans AWS Glue.
2. Dans AWS Glue Studio, créez une connexion sous Connexions de données en suivant les étapes ci-dessous :
   a. Lorsque vous sélectionnez un type de connexion, sélectionnez Asana.
   b. Indiquez l'environnement Asana.
   c. Sélectionnez le rôle IAM pour lequel AWS Glue vous pouvez assumer et qui dispose des autorisations nécessaires pour effectuer les actions suivantes :

      JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
```

```
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

d. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.

e. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.

3. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.

Lecture à partir d'entités Asana

Conditions préalables

Un objet Asana à partir duquel vous souhaitez lire. Reportez-vous au tableau des entités prises en charge ci-dessous pour vérifier les entités disponibles.

Entités prises en charge pour la source

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Espace de travail	Non	Oui	Non	Oui	Non
Balise	Non	Oui	Non	Oui	Non
Utilisateur	Non	Oui	Non	Oui	Non
Portfolio	Non	Oui	Non	Oui	Non
Team	Non	Oui	Non	Oui	Non

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Project	Oui	Oui	Non	Oui	Non
Section	Non	Oui	Non	Oui	Non
Sous-tâche	Oui	Non	Non	Oui	Oui
Objectif	Oui	Oui	Non	Oui	Non
AuditLogE vent	Oui	Oui	Non	Oui	Non
Status Update	Oui	Oui	Non	Oui	Non
Custom Field	Non	Oui	Non	Oui	Non
Project Brief	Oui	Non	Non	Oui	Oui

Exemple

```
read_read = glueContext.create_dynamic_frame.from_options(
    connection_type="Asana",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "task/workspace:xxxx",
        "API_VERSION": "1.0",
        "PARTITION_FIELD": "created_at",
        "LOWER_BOUND": "2024-02-05T14:09:30.115Z",
        "UPPER_BOUND": "2024-06-07T13:30:00.134Z",
        "NUM_PARTITIONS": "3"
    }
```

Détails des entités et des champs Asana

- Espace de travail
- Balise

- Utilisateur
- Portfolio
- Team
- Project
- Rubrique
- Tâche
- Objectif
- AuditLogEvent
- Status Update
- Custom Field
- Project Brief

Requêtes de partitionnement

Les options Spark supplémentaires `PARTITION_FIELD`, `LOWER_BOUND`, `UPPER_BOUND` et `NUM_PARTITIONS` peuvent être indiquées si vous souhaitez utiliser la simultanéité dans Spark. Avec ces paramètres, la requête d'origine serait divisée en `NUM_PARTITIONS` nombres de sous-requêtes pouvant être exécutées simultanément par les tâches Spark.

- `PARTITION_FIELD` : le nom du champ à utiliser pour partitionner la requête.
- `LOWER_BOUND` : une valeur limite inférieure inclusive du champ de partition choisi.

  Pour le champ Date, nous acceptons le format de date Spark utilisé dans les requêtes SQL Spark. Exemples de valeurs valides : `2024-06-07T13:30:00.134Z`.
- `UPPER_BOUND` : une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS` : nombre de partitions.

Les détails relatifs à la prise en charge des champs de partitionnement par entité sont présentés dans le tableau suivant.

Nom de l'entité	Champ de partitionnement	Type de données
Sous-tâche	created_at	DateTime

Nom de l'entité	Champ de partitionnement	Type de données
Sous-tâche	modified_at	DateTime

Exemple

```
read_read = glueContext.create_dynamic_frame.from_options(
    connection_type="Asana",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "task/workspace:xxxx",
        "API_VERSION": "1.0",
        "PARTITION_FIELD": "created_at",
        "LOWER_BOUND": "2024-02-05T14:09:30.115Z",
        "UPPER_BOUND": "2024-06-07T13:30:00.134Z",
        "NUM_PARTITIONS": "3"
    }
```

Options de connexion Asana

Les options de connexion pour Asana sont les suivantes :

- `ENTITY_NAME`(Chaîne) - (Obligatoire) Utilisé pour la lecture/l'écriture. Le nom de votre objet dans Asana.
- `API_VERSION`(Chaîne) - (Obligatoire) Utilisé pour la lecture/l'écriture. Version d'API Rest Asana que vous souhaitez utiliser. Par exemple : 1.0.
- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.
- `PARTITION_FIELD`(Chaîne) - Utilisé pour la lecture. Champ à utiliser pour partitionner la requête.
- `LOWER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite inférieure inclusive du champ de partition choisi.
- `UPPER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite supérieure exclusive du champ de partition choisi.

- `NUM_PARTITIONS`(Entier) - Par défaut : 1. Utilisé pour la lecture. Nombre de partitions pour la lecture.

Création d'un compte Asana

1. Créez un compte Asana et choisissez S'inscrire.
2. Une fois connecté, vous serez redirigé vers la page Configuration du compte. Procédez comme suit :
   - Consultez le formulaire de configuration du compte.
   - Renseignez toutes les informations nécessaires pour créer votre compte Asana.
   - Vérifiez l'exactitude des informations.
3. Choisissez Créer un compte ou Soumettre (le texte exact du bouton peut varier) pour finaliser la configuration de votre compte.

Création de l'application dans Asana pour **`OAuth2.0`**

1. Connectez-vous à votre compte Asana à l'aide de vos informations d'identification client Asana.
2. Choisissez l'icône de votre profil utilisateur dans le coin supérieur droit et sélectionnez Mes paramètres dans le menu déroulant.
3. Sélectionnez l'onglet Applications, puis Gérer les applications de développeurs.
4. Sélectionnez Créer une application et entrez les informations pertinentes.
5. Sélectionnez Créer des applications.
6. Sur la page Mes applications :
   a. Sélectionnez OAuthet dans la section Informations d'identification de l'application, notez votre identifiant client et votre secret client.
   b. Dans la URLs section Redirection, ajoutez les URL de redirection nécessaires.

> **Note**
>
> Entrez l'URI de redirection en utilisant ce format : `https://{aws-region-code}.console.aws.amazon.com/gluestudio/oauth`. Par exemple, pour USA Est (Virginie du Nord), utilisez : `https://us-east-1.console.aws.amazon.com/gluestudio/oauth`

Création de l'application dans Asana pour le jeton **PAT**

1. Connectez-vous à votre compte Asana à l'aide de vos informations d'identification client Asana.
2. Choisissez l'icône de votre profil utilisateur dans le coin supérieur droit et sélectionnez Mes paramètres de profil dans le menu déroulant.
3. Sélectionnez l'onglet Applications, puis Comptes de service.
4. Sélectionnez Créer une application et entrez les informations pertinentes.
5. Choisissez Ajouter un compte de service.
6. La page suivante affiche votre jeton, copiez-le et stockez-le en toute sécurité.

> **Important**
>
> Ce jeton ne sera affiché qu'une seule fois. Assurez-vous de le copier et de le stocker en toute sécurité.

Limitations

Les restrictions du connecteur Asana sont les suivantes :

- Les comptes de service dans Enterprise Domains ne peuvent accéder qu'aux points de terminaison de l'API du journal d'audit. L'authentification à l'aide du jeton d'accès personnel d'un compte de service est requise pour accéder à ces points de terminaison.
- L'entité Goal n'est accessible que pour les comptes utilisateur dotés d'un forfait Premium ou supérieur.
- `Audit Log Event Entity` : dans le connecteur, les champs `start_at` et `end_at` sont combinés en un seul champ « start_end_at » pour prendre en charge le filtrage et le transfert incrémentiel.
- Le partitionnement ne peut pas être pris en charge pour le `Date` champ, même s'il prend en charge les opérateurs greater-than-or-equal -to et less-than-or-equal -to. Scénario : création d'une tâche avec `partitionField` en tant que due_on (datatype : date), `lowerBound` en tant que `2019-09-14`, upperBound en tant que `2019-09-16` et `numPartition` en tant que 2. La partie filtre de l'URL du point de terminaison est créée comme suit :
  - partition1 : due_on.before=2019-09-14&due_on.after=2019-09-14
  - partition2 : due_on.before=2019-09-15&due_on.after=2019-09-15 Sortie :

  - Dans partition1, nous obtenons des données dont la date d'échéance (due_date) est le 14/09/2019 et le 15/09/2019.
  - Dans partition2, nous obtenons les mêmes données avec une date d'échéance le 15/09/2019 (qui se trouvaient dans partition1), ainsi que d'autres données, ce qui entraîne une duplication des données.
- Le filtrage et le partitionnement ne peuvent pas être pris en charge sur le même champ, car une erreur de demande erronée est renvoyée du côté de SaaS.
- L'entité Task nécessite un minimum d'un champ dans les critères de filtre. Asana présente une restriction : la pagination n'est pas identifiée si les enregistrements ne sont pas triés en fonction d'un champ temporel. Par conséquent, le champ Created_at est utilisé avec la pagination pour distinguer le prochain ensemble d'enregistrements. Le champ Created_at est marqué comme obligatoire dans le filtre, avec une valeur par défaut de 2000-01-01T00:00:00Z s'il n'est pas fourni. Pour plus d'informations sur la pagination, consultez Tasks in a workspace.

Connexion à Azure Cosmos DB dans AWS Glue Studio

AWS Glue fournit une prise en charge intégrée pour Azure Cosmos DB. AWS Glue Studio fournit une interface visuelle pour se connecter à Azure Cosmos DB pour NoSQL, créer des tâches d'intégration de données et les exécuter sur l'exécution Spark AWS Glue Studio sans serveur.

Rubriques

- Création d'une connexion Azure Cosmos DB
- Création d'un nœud source Azure Cosmos DB
- Création d'un nœud cible Azure Cosmos DB
- Options avancées

Création d'une connexion Azure Cosmos DB

Prérequis :

- Dans Azure, vous devez identifier ou générer une clé de base de données Azure Cosmos à utiliser par AWS Glue,`cosmosKey`. Pour plus d'informations, consultez la section Accès sécurisé aux données dans Azure Cosmos DB dans la documentation Azure.

Pour configurer une connexion à Azure Cosmos DB :

1. Dans AWS Secrets Manager, créez un secret à l'aide de votre clé de base de données Azure Cosmos. Pour créer un secret dans Secrets Manager, suivez le didacticiel disponible dans la section Créer un AWS Secrets Manager secret dans la AWS Secrets Manager documentation. Après avoir créé le secret, conservez le nom du secret *secretName* pour l'étape suivante.

   - Lorsque vous sélectionnez des paires clé/valeur, créez une paire pour la clé `spark.cosmos.accountKey` contenant la valeur. *cosmosKey*

2. Dans la AWS Glue console, créez une connexion en suivant les étapes décrites dansthe section called "Ajouter une AWS Glue connexion". Après avoir créé la connexion, conservez le nom de la connexion pour une utilisation future dans AWS Glue. *connectionName*

   - Lorsque vous sélectionnez un type de connexion, sélectionnez Azure Cosmos DB.
   - Lorsque vous sélectionnez un AWS secret, fournissez*secretName*.

Création d'un nœud source Azure Cosmos DB

Prérequis

- Une connexion AWS Glue Azure Cosmos DB, configurée avec un AWS Secrets Manager secret, comme décrit dans la section précédente,the section called "Création d'une connexion Azure Cosmos DB".
- Les autorisations appropriées sur votre tâche pour lire le secret utilisé par la connexion.
- Un conteneur Azure Cosmos DB pour NoSQL à partir duquel vous souhaitez lire. Vous aurez besoin des informations d'identification du conteneur.

  Un conteneur Azure Cosmos pour NoSQL est identifié par sa base de données et son conteneur. Vous devez fournir les noms de base de données et de conteneur lorsque vous vous connectez à l'API Azure Cosmos pour NoSQL. *cosmosDBName cosmosContainerName*

Ajout d'une source de données Azure Cosmos DB

Pour ajouter un nœud Source de données – Azure Cosmos DB :

1. Choisissez la connexion pour votre source de données Azure Cosmos DB. Puisque vous l'avez créé, il devrait être disponible dans le menu déroulant. Si vous devez créer une connexion,

choisissez Créer une connexion Azure Cosmos DB. Pour de plus amples informations, veuillez consulter la section précédente the section called "Création d'une connexion Azure Cosmos DB".

Une fois que vous avez choisi une connexion, vous pouvez afficher ses propriétés en cliquant sur Afficher les propriétés.

2. Choisissez le nom de la base de données Cosmos DB : indiquez le nom de la base de données à partir de laquelle vous souhaitez lire*cosmosDBName*.
3. Choisissez Azure Cosmos DB Container : indiquez le nom du conteneur à partir duquel vous souhaitez lire*cosmosContainerName*.
4. Choisissez éventuellement la Requête personnalisé Azure Cosmos DB : fournissez une requête SQL SELECT pour récupérer des informations spécifiques à partir d'Azure Cosmos DB.
5. Dans les Propriétés Azure Cosmos personnalisées, saisissez les paramètres et les valeurs nécessaires.

Création d'un nœud cible Azure Cosmos DB

Prérequis

- Une connexion AWS Glue Azure Cosmos DB, configurée avec un AWS Secrets Manager secret, comme décrit dans la section précédente,the section called "Création d'une connexion Azure Cosmos DB".
- Les autorisations appropriées sur votre tâche pour lire le secret utilisé par la connexion.
- Une table Azure Cosmos DB dans laquelle vous souhaitez écrire. Vous aurez besoin des informations d'identification du conteneur. Vous devez créer le conteneur avant d'appeler la méthode de connexion.

  Un conteneur Azure Cosmos pour NoSQL est identifié par sa base de données et son conteneur. Vous devez fournir les noms de base de données et de conteneur lorsque vous vous connectez à l'API Azure Cosmos pour NoSQL. *cosmosDBName cosmosContainerName*

Ajout d'une cible de données Azure Cosmos DB

Pour ajouter un nœud Cible de données – Azure Cosmos DB :

1. Choisissez la connexion pour votre source de données Azure Cosmos DB. Puisque vous l'avez créé, il devrait être disponible dans le menu déroulant. Si vous devez créer une connexion,

choisissez Créer une connexion Azure Cosmos DB. Pour de plus amples informations, veuillez consulter la section précédente the section called "Création d'une connexion Azure Cosmos DB".

Une fois que vous avez choisi une connexion, vous pouvez afficher ses propriétés en cliquant sur Afficher les propriétés.

2. Choisissez le nom de la base de données Cosmos DB : indiquez le nom de la base de données à partir de laquelle vous souhaitez lire*cosmosDBName*.
3. Choisissez Azure Cosmos DB Container : indiquez le nom du conteneur à partir duquel vous souhaitez lire*cosmosContainerName*.
4. Dans les Propriétés Azure Cosmos personnalisées, saisissez les paramètres et les valeurs nécessaires.

Options avancées

Vous pouvez fournir des options avancées lors de la création d'un nœud Azure Cosmos DB. Ces options sont les mêmes que celles disponibles lors de la programmation AWS Glue de scripts Spark.

Consultez the section called "Connexions Azure Cosmos DB".

Connexion à Azure SQL dans AWS Glue Studio

AWS Glue fournit une prise en charge intégrée pour Azure SQL. AWS Glue Studio fournit une interface visuelle pour se connecter à Azure SQL, créer des tâches d'intégration de données et les exécuter sur l'exécution Spark AWS Glue Studio sans serveur.

Rubriques

- Création d'une connexion Azure SQL
- Création d'un nœud source Azure SQL
- Création d'un nœud cible Azure SQL
- Options avancées

Création d'une connexion Azure SQL

Pour vous connecter à Azure SQL depuis AWS Glue, vous devez créer et stocker vos informations d'identification Azure SQL dans un AWS Secrets Manager secret, puis associer ce secret à une AWS Glue connexion Azure SQL.

Pour configurer une connexion à Azure SQL :

1. Dans AWS Secrets Manager, créez un secret à l'aide de vos informations d'identification Azure SQL. Pour créer un secret dans Secrets Manager, suivez le didacticiel disponible dans la section Créer un AWS Secrets Manager secret dans la AWS Secrets Manager documentation. Après avoir créé le secret, conservez le nom du secret *secretName* pour l'étape suivante.

   - Lorsque vous sélectionnez des paires clé/valeur, créez une paire pour la clé `user` contenant la valeur. *azuresqlUsername*
   - Lorsque vous sélectionnez des paires clé/valeur, créez une paire pour la clé `password` contenant la valeur. *azuresqlPassword*

2. Dans la AWS Glue console, créez une connexion en suivant les étapes décrites dansthe section called "Ajouter une AWS Glue connexion". Après avoir créé la connexion, conservez le nom de la connexion pour une utilisation future dans AWS Glue. *connectionName*

   - Lorsque vous sélectionnez un type de connexion, sélectionnez Azure SQL.
   - Lorsque vous fournissez une URL Azure SQL, fournissez une URL de point de terminaison JDBC.

     La URL doit avoir le format suivant :
     `jdbc:sqlserver://`*databaseServerName*`:`*databasePort*`;databaseName=`*azuresqlDBnam*

     AWS Glue nécessite les propriétés d'URL suivantes :

     - `databaseName` – une base de données par défaut dans Azure SQL à laquelle se connecter.

     Pour plus d'informations sur JDBC URLs pour Azure SQL Managed Instances, consultez la documentation Microsoft.

   - Lorsque vous sélectionnez un AWS secret, fournissez*secretName*.

Création d'un nœud source Azure SQL

Prérequis

- Une connexion AWS Glue Azure SQL, configurée avec un AWS Secrets Manager secret, comme décrit dans la section précédente,the section called "Création d'une connexion Azure SQL".
- Les autorisations appropriées sur votre tâche pour lire le secret utilisé par la connexion.

- Une table SQL Azure que vous souhaitez lire,*tableName*.

  Une table Azure SQL est identifiée par sa base de données, son schéma et son nom de table. Vous devez fournir le nom de la base de données et le nom de la table lorsque vous vous connectez à Azure SQL. Vous devez également fournir le schéma s'il n'est pas « public » par défaut. La base de données est fournie via une propriété URL dans*connectionName*, un schéma et un nom de table via le`dbtable`.

Ajout d'une source de données Azure SQL

Pour ajouter un nœud Source de données – Azure SQL :

1. Choisissez la connexion pour votre source de données Azure SQL. Puisque vous l'avez créé, il devrait être disponible dans le menu déroulant. Si vous devez créer une connexion, choisissez Créer une connexion Azure SQL. Pour de plus amples informations, veuillez consulter la section précédente the section called "Création d'une connexion Azure SQL".

   Une fois que vous avez choisi une connexion, vous pouvez afficher ses propriétés en cliquant sur Afficher les propriétés.
2. Choisissez une option Source Azure SQL :

   - Choisissez une seule table : accéder à toutes les données à partir d'une seule table.
   - Saisir une requête personnalisée : accéder à un jeu de données à partir de plusieurs tables en fonction de votre requête personnalisée.
3. Si vous avez choisi une seule table, entrez*tableName*.

   Si vous avez choisi Saisir une requête personnalisée, saisissez une requête TransactSQL SELECT.
4. Dans les Propriétés Azure SQL personnalisées, saisissez les paramètres et les valeurs nécessaires.

Création d'un nœud cible Azure SQL

Prérequis

- Une connexion AWS Glue Azure SQL, configurée avec un AWS Secrets Manager secret, comme décrit dans la section précédente,the section called "Création d'une connexion Azure SQL".

- Les autorisations appropriées sur votre tâche pour lire le secret utilisé par la connexion.
- Une table SQL Azure dans laquelle vous souhaitez écrire,*tableName*.

  Une table Azure SQL est identifiée par sa base de données, son schéma et son nom de table. Vous devez fournir le nom de la base de données et le nom de la table lorsque vous vous connectez à Azure SQL. Vous devez également fournir le schéma s'il n'est pas « public » par défaut. La base de données est fournie via une propriété d'URL dans*connectionName*, un schéma et un nom de table via le`dbtable`.

Ajout d'une cible de données Azure SQL

Pour ajouter un nœud Cible de données – Azure SQL :

1. Choisissez la connexion pour votre source de données Azure SQL. Puisque vous l'avez créé, il devrait être disponible dans le menu déroulant. Si vous devez créer une connexion, choisissez Créer une connexion Azure SQL. Pour de plus amples informations, veuillez consulter la section précédente the section called “Création d'une connexion Azure SQL”.

   Une fois que vous avez choisi une connexion, vous pouvez afficher ses propriétés en cliquant sur Afficher les propriétés.
2. Configurez le nom de la table en fournissant*tableName*.
3. Dans les Propriétés Azure SQL personnalisées, saisissez les paramètres et les valeurs nécessaires.

Options avancées

Vous pouvez fournir des options avancées lors de la création d'un nœud Azure SQL. Ces options sont les mêmes que celles disponibles lors de la programmation AWS Glue de scripts Spark.

Consultez the section called “Connexions Azure SQL”.

Connexion à Blackbaud Raiser's Edge NXT

Blackbaud Raiser's Edge NXT est une solution logicielle complète de collecte de fonds et de gestion des donateurs basée sur le cloud, spécialement conçue pour les organisations à but non lucratif et l'ensemble de la communauté sociale. Ce connecteur repose sur l'API SKY de Blackbaud Raiser's Edge NXT et fournit des opérations pour permettre de gérer les entités présentes dans le Raisers Edge NXT.

Rubriques

- AWS Glue support pour Blackbaud Raiser's Edge NXT
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration de Blackbaud Raiser's Edge NXT
- Configuration des connexions Blackbaud Raiser's Edge NXT
- Lecture à partir des entités Blackbaud Raiser's Edge NXT
- Options de connexion Blackbaud Raiser's Edge NXT
- Restrictions de Blackbaud Raiser's Edge NXT

AWS Glue support pour Blackbaud Raiser's Edge NXT

AWS Glue prend en charge Blackbaud Raiser's Edge NXT comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser des tâches AWS Glue ETL pour interroger des données depuis Blackbaud Raiser's Edge NXT.

Prise en charge en tant que cible ?

Non.

Versions de l'API Blackbaud Raiser's Edge NXT prises en charge

Les versions suivantes de l'API Blackbaud Raiser's Edge NXT sont prises en charge :

- v1

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les autorisations AWS IAM requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
```

```
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Si vous ne voulez pas utiliser la méthode ci-dessus, vous pouvez également utiliser les politiques IAM gérées suivantes :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration de Blackbaud Raiser's Edge NXT

Avant de pouvoir AWS Glue transférer des données depuis Blackbaud Raiser's Edge NXT, vous devez répondre aux exigences suivantes :

Configuration requise

Les exigences minimales sont les suivantes :

- Vous disposez d'un compte Blackbaud Raiser's Edge NXT.
- Vous avez généré un jeton d'accès sur votre compte Blackbaud Raiser's Edge NXT avec l' read/write étendue appropriée attribuée pour accéder au. APIs Pour plus d'informations, consultez Authorization.

Si vous répondez à ces exigences, vous êtes prêt à vous connecter AWS Glue à votre compte Blackbaud Raiser's Edge NXT.

Configuration des connexions Blackbaud Raiser's Edge NXT

Blackbaud Raiser's Edge NXT prend en charge le type de subvention AUTHORIZATION_CODE pour. OAuth2

- Ce type de subvention est considéré comme « à trois étapes » OAuth car il repose sur la redirection des utilisateurs vers un serveur d'autorisation tiers pour authentifier l'utilisateur. Il est utilisé lors de la création de connexions via la AWS Glue console. La AWS Glue console redirige l'utilisateur vers Blackbaud Raiser's Edge NXT où il doit se connecter et autoriser AWS Glue les autorisations demandées pour accéder à son instance Blackbaud Raiser's Edge NXT.
- Les utilisateurs peuvent choisir de créer leur propre application connectée dans Blackbaud Raiser's Edge NXT et de fournir leur propre identifiant client, clé d'abonnement et URL d'instance lors de la création de connexions via la console. AWS Glue Dans ce scénario, ils seront toujours redirigés vers Blackbaud Raiser's Edge NXT pour se connecter et autoriser AWS Glue l'accès à leurs ressources.
- Ce type d'autorisation donne lieu à un jeton d'actualisation et à un jeton d'accès. Le jeton d'accès est de courte durée et peut être actualisé automatiquement sans interaction de l'utilisateur à l'aide du jeton d'actualisation.
- Pour consulter la documentation publique de Blackbaud Raiser's Edge NXT sur la création d'une application connectée pour le OAuth flux de code d'autorisation, voir Autorisation.

Pour configurer une connexion Blackbaud Raiser's Edge NXT :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes :
   a. Pour l'application connectée gérée par le client, le secret doit contenir la clé d'API de l'application connectée avec *USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET* comme clé.
   b. Remarque : vous devez créer un secret pour vos connexions dans AWS Glue.

1. Dans AWS Glue Glue Studio, créez une connexion sous Data Connections en suivant les étapes ci-dessous :
   a. Lorsque vous sélectionnez une source de données, sélectionnez Blackbaud Raiser's Edge NXT.
   b. Indiquez l'`INSTANCE_URL` du compte Blackbaud Raiser's Edge NXT auquel vous souhaitez vous connecter.
   c. Indiquez le `clientId` de l'application du client gérée par l'utilisateur.
   d. Indiquez la clé d'abonnement associée à votre compte.
   e. Sélectionnez le rôle AWS IAM qui AWS Glue peut assumer les actions suivantes et qui dispose des autorisations nécessaires pour les actions suivantes :

      JSON

      ```
      {
        "Version":"2012-10-17",
        "Statement": [
          {
            "Effect": "Allow",
            "Action": [
              "secretsmanager:DescribeSecret",
              "secretsmanager:GetSecretValue",
              "secretsmanager:PutSecretValue",
              "ec2:CreateNetworkInterface",
              "ec2:DescribeNetworkInterfaces",
              "ec2:DeleteNetworkInterface"
            ],
            "Resource": "*"
          }
        ]
      }
      ```

   f. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.
   g. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.
2. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.

Lecture à partir des entités Blackbaud Raiser's Edge NXT

Prérequis

Un objet Blackbaud Raiser's Edge NXT à partir duquel vous souhaitez lire. Vous aurez besoin du nom de l'objet.

Entités prises en charge pour la source :

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Constituent Address	Oui	Oui	Non	Oui	Oui
Constituent Education	Oui	Oui	Non	Oui	Oui
Constitue nt Email address	Oui	Oui	Non	Oui	Oui
Constituent Phone	Oui	Oui	Non	Oui	Oui
Constituent Note	Oui	Oui	Non	Oui	Oui
Constituent Relationship	Oui	Oui	Non	Oui	Oui
Constitue nt Online presence	Oui	Oui	Non	Oui	Oui
Opportunity	Oui	Oui	Non	Oui	Oui
Appeal	Oui	Oui	Non	Oui	Oui
Campaign	Oui	Oui	Non	Oui	Oui
Fund	Oui	Oui	Non	Oui	Oui

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Package	Oui	Oui	Non	Oui	Oui
Gift Batch	Oui	Oui	Non	Oui	Non
Event Participant	Oui	Oui	Oui	Oui	Oui
Constituent Fundraiser Assignment	Non	Non	Non	Oui	Non
Gift	Oui	Oui	Oui	Oui	Oui
Membres	Oui	Oui	Non	Oui	Oui
Action	Oui	Oui	Non	Oui	Non
Constituent	Oui	Oui	Oui	Oui	Oui
Constituent Goods	Oui	Oui	Non	Oui	Oui
Événement	Oui	Oui	Oui	Oui	Oui
Gift custom field	Oui	Oui	Non	Oui	Oui

Exemple :

```
blackbaud_read = glueContext.create_dynamic_frame.from_options(
    connection_type="BLACKBAUD",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "API_VERSION": "v1",
        "SUBSCRIPTION_KEY": <Subscription key associated with one's developer account>
```

```
    }
```

Détails des entités et des champs Blackbaud Raiser's Edge NXT

Pour plus d'informations sur les entités et de détails sur les champs, consultez :

- Action
- Constituent
- Constituent Address
- Constituent Membership
- Constituent Fundraiser Assignment
- Constituent Education
- Constituent Email address
- Constituent Phone
- Constituent Note
- Constituent Online presence
- Constituent Relationship
- Event
- Event Participant
- Appeal
- Campaign
- Fund
- Package
- Gift
- Gift Custom Field
- Gift Batch
- Opportunity
- Constituent Codes

> **Note**
>
> Les types de données Struct et List sont convertis en type de données String, et le type de DateTime données est converti en Timestamp dans la réponse des connecteurs.

Requêtes de partitionnement

Partitionnement basé sur les champs :

Blackbaud Raiser's Edge NXT ne prend pas en charge le partitionnement basé sur les champs ou les enregistrements.

Partitionnement basé sur des enregistrements :

Vous pouvez indiquer l'option Spark supplémentaire `NUM_PARTITIONS` si vous souhaitez utiliser la simultanéité dans Spark. Avec ce paramètre, la requête d'origine serait divisée en `NUM_PARTITIONS` nombres de sous-requêtes pouvant être exécutées simultanément par les tâches Spark.

Dans le partitionnement basé sur les enregistrements, le nombre total d'enregistrements présents est demandé par l'API Blackbaud Raiser's Edge NXT, puis divisé par le nombre `NUM_PARTITIONS` fourni. Le nombre d'enregistrements qui en résulte est ensuite extrait simultanément par chaque sous-requête.

- `NUM_PARTITIONS` : le nombre de partitions.

Exemple :

```
blackbaud_read = glueContext.create_dynamic_frame.from_options(
    connection_type="BLACKBAUD",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "API_VERSION": "v1",
        "NUM_PARTITIONS": "2",
        "SUBSCRIPTION_KEY": <Subscription key associated with one's developer account>
    }
```

Options de connexion Blackbaud Raiser's Edge NXT

Les options de connexion pour Blackbaud Raiser's Edge NXT sont les suivantes :

- `ENTITY_NAME`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Le nom de votre objet dans Blackbaud Raiser's Edge NXT.
- `API_VERSION`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Version que vous souhaitez utiliser de l'API Rest Blackbaud Raiser's Edge NXT.
- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.
- `NUM_PARTITIONS`(Entier) - Par défaut : 1. Utilisé pour la lecture. Nombre de partitions pour la lecture. Exemple de valeur : 10.
- `SUBSCRIPTION_KEY`(Chaîne) - (Obligatoire) Par défaut : vide. Utilisé pour la lecture. Clé d'abonnement associée à son compte de développeur.

Restrictions de Blackbaud Raiser's Edge NXT

Voici les restrictions ou remarques relatives à Blackbaud Raiser's Edge NXT :

- Le SaaS ne prend en charge que l'opérateur `EQUAL_TO`, qui renvoie les résultats créés ou modifiés à la date spécifiée ou après cette date. De plus, le champ « id » est un type de données String. Il n'y a pas non plus d'identification des champs n'autorisant pas les valeurs nulles. Par conséquent, le partitionnement basé sur les champs n'est pas pris en charge.
- L'attraction incrémentielle n'est prise en charge par l'entité `Event` qu'avec des fréquences quotidiennes, mensuelles et hebdomadaires.
- L'entité Constituent Fundraiser Assignment renvoie un maximum de 20 enregistrements.
- Partitionnement basé sur les enregistrements :
  - Non pris en charge par les entités `Action`, `Constituent Fundraiser Assignment` ou `Gift Batch`.
  - Le partitionnement basé sur les enregistrements avec le prédicat de filtre n'est pris en charge que par les entités `Event` et `Event Participant`. Si un prédicat de filtre est utilisé avec d'autres entités prises en charge basées sur des enregistrements, une exception sera émise.

- Dans l'entité `Gift Custom Field`, le champ « value » doit être utilisé conjointement avec le champ « category », faute de quoi une réponse non filtrée sera envoyée. Ainsi, pour obliger l'utilisateur à saisir le champ « category » lors du filtrage avec le champ « value », une exception sera émise si l'exigence susmentionnée n'a pas été respectée.
- Les champs `date_added` et `last_modified` de toutes les entités applicables ne prennent en charge aucun opérateur comparatif. Ils ne prennent en charge que l'opérateur égal à. De plus, aucun champ ne peut être associé aux champs susmentionnés pour fournir une gamme d'enregistrements. Par conséquent, ces champs sont uniquement interrogeables et ne peuvent pas prendre en charge le transfert incrémentiel.
- Le champ `added_by` de l'entité `Gift Batch` ne sera pas considéré comme filtrable, car il risque de ne pas émettre les bons résultats.
- Il existe une latence d'environ 30 minutes pour que les enregistrements soient récupérés via le point de terminaison `/GET Gift List` lors de l'insertion des données dans l'entité `Gift`.
- La prise en charge du transfert incrémentiel a été abandonnée pour l'entité Gift en raison de restrictions liées à la source de données.
- Il existe une latence de 10 minutes pour le champ de statut dans l'entité Opportunity.
- L'entité `Fundraiser Assignment` a `Constituent` comme entité dépendante. Le connecteur charge au maximum 5 000 personnes IDs parmi lesquelles choisir, afin d'éviter que la taille de réponse ne dépasse la taille de charge utile maximale autorisée.

Connexion à CircleCI

CircleCI est une plateforme d'intégration et de livraison continues. Votre compte CircleCI contient des données sur vos projets, vos pipelines, vos flux de travail, etc. Si vous êtes un utilisateur de CircleCI, vous pouvez vous connecter AWS Glue à votre compte CircleCI. Vous pouvez ensuite utiliser CircleCI comme source de données dans vos tâches ETL. Exécutez ces tâches pour transférer des données entre CircleCI AWS et des services ou d'autres applications prises en charge.

Rubriques

- AWS Glue support pour CircleCI
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration de CircleCI
- Configuration de connexions CircleCI
- Lecture à partir d'entités CircleCI

- Options de connexion CircleCI
- Restrictions de CircleCI

AWS Glue support pour CircleCI

AWS Glue prend en charge CircleCI comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser des tâches AWS Glue ETL pour interroger les données de CircleCI.

Prise en charge en tant que cible ?

Non.

Versions de l'API CircleCI prises en charge

Les versions d'API CircleCI suivantes sont prises en charge :

- v2

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les autorisations AWS IAM requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
```

```
      ],
      "Resource": "*"
    }
  ]
}
```

Si vous ne voulez pas utiliser la méthode ci-dessus, vous pouvez également utiliser les politiques IAM gérées suivantes :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration de CircleCI

Avant de pouvoir AWS Glue transférer des données depuis CircleCI, vous devez satisfaire aux exigences suivantes :

Configuration requise

Les exigences minimales sont les suivantes :

- Vous disposez d'un compte CircleCI qui contient les données que vous souhaitez transférer.
- Dans les paramètres utilisateur de votre compte, vous avez créé un jeton d'API personnel. Pour plus d'informations, consultez Creating a personal API token.
- Vous fournissez le jeton d'API personnel à AWS Glue lors de la création de la connexion.

Si vous répondez à ces exigences, vous êtes prêt à vous connecter AWS Glue à votre compte CircleCI.

Configuration de connexions CircleCI

CircleCI prend en charge l'authentification personnalisée.

Pour configurer une connexion CircleCI :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes :
   a. Pour l'application connectée gérée par le client, le secret doit contenir la clé d'API de l'application connectée avec `Circle-Token` comme clé.
   b. Remarque : vous devez créer un secret pour vos connexions dans AWS Glue.

1. Dans AWS Glue Glue Studio, créez une connexion sous Data Connections en suivant les étapes ci-dessous :
   a. Lorsque vous sélectionnez une source de données, sélectionnez CircleCI.
   b. Sélectionnez le rôle AWS IAM qui AWS Glue peut assumer et dispose des autorisations nécessaires pour les actions suivantes :

      JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

   c. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.
   d. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.

2. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.

Lecture à partir d'entités CircleCI

Prérequis

Un objet CircleCI à partir duquel vous souhaitez lire. Vous aurez besoin du nom de l'objet.

Entités prises en charge pour la source :

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Contexte	Oui	Non	Non	Oui	Non
Organizat ion Summary Metric	Oui	Non	Non	Oui	Non
Pipeline	Non	Non	Non	Oui	Non
Flux de travail d'un pipeline	Oui	Non	Non	Oui	Non
Project Branch	Oui	Non	Non	Oui	Non
Project Flaky Test	Non	Non	Non	Oui	Non
Project Summary Metric	Oui	Non	Non	Oui	Non
Planning	Non	Non	Non	Oui	Non
Workflow Job Timeseries	Oui	Non	Non	Oui	Non

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Workflow Metric And Trend	Oui	Non	Non	Oui	Non
Workflow Recent Run	Oui	Non	Non	Oui	Non
Workflow Summary Metric	Oui	Non	Non	Oui	Non
Workflow Test Metric	Oui	Non	Non	Oui	Non

Exemple :

```
circleci_read = glueContext.create_dynamic_frame.from_options(
    connection_type="circleci",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "context/e7ea2945-dccb-4205-b673-8391fe1b3a4c",
        "API_VERSION": "v2"
    }
```

Détails des entités et des champs CircleCI

Pour plus d'informations sur les entités et de détails sur les champs, consultez :

- Contexts
- Project Summary Metrics
- Workflow Job Timeseries
- Organization Summary Metrics
- Project Branches
- Project Flaky Tests

- Workflow Recent Runs
- Workflow Summary Metrics
- Workflow Metrics and Trends
- Workflow Test Metrics
- Pipelines
- Pipeline Workflows
- Horaires

Entités avec métadonnées statiques :

Entité	Champ	Type de données	Opérateurs pris en charge
Contexte	Created At	String	
	ID	String	
	Name	String	
	Owner Type	String	EQUAL_TO
Organization Summary Metric	All Projects	List	
	Org Data	Struct	
	Org Project Data	List	
	Project Names	String	EQUAL_TO
	Reporting Window	String	EQUAL_TO
Pipeline	Branche	String	EQUAL_TO
	Created At	String	
	Erreurs	List	
	ID	String	

Entité	Champ	Type de données	Opérateurs pris en charge
	Number	Entier	
	Project Slug	String	
	State	String	
	Déclencheur	Struct	
	Trigger Parameters	Struct	
	Updated At.	String	
	VCS	Struct	
Flux de travail d'un pipeline	Canceled By	String	
	Created At	String	
	Errorer By	String	
	ID	String	
	Name	String	
	ID du pipeline	String	
	Pipeline Number	Entier	
	Project Slug	String	
	Started By	String	
	Statut	String	
	Stopped At	String	
	Balise	String	
Project Branch	Branches	List	

Entité	Champ	Type de données	Opérateurs pris en charge
	Org ID	String	
	Project ID	String	
	Nom du flux de travail	String	EQUAL_TO
Project Flaky Test	Classname	String	
	Fichier	String	
	Nom de la tâche	String	
	Job Number	Entier	
	Pipeline Number	Entier	
	Source	String	
	Test Name	String	
	Time Wasted	Entier	
	Times Flaked	Entier	
	Workflow Created At	String	
	ID de flux de travail	String	
	Nom du flux de travail	String	
Project Summary Metric	All Branches	List	
	All Workflows	List	
	Branches	String	EQUAL_TO
	Organization ID	String	
	Project Data	Struct	

Entité	Champ	Type de données	Opérateurs pris en charge
	Project ID	String	
	Project Workflow Branch Data	List	
	Project Workflow Data	List	
	Reporting Window	String	EQUAL_TO
	Workflow Names	String	EQUAL_TO
Planning	Acteur	Struct	
	Created At	String	
	Description	String	
	ID	String	
	Name	String	
	Parameters	Struct	
	Project Slug	String	
	Timetable	Struct	
	Updated At.	String	
Workflow Job Timeseries	Branche	String	EQUAL_TO
	Granularité	String	EQUAL_TO
	Max Ended At	String	
	Métriques	Struct	
	Min Started At	String	

Entité	Champ	Type de données	Opérateurs pris en charge
	Name	String	
	Start End Date	DateTime	EQUAL_TO, BETWEEN
	Horodatage	String	
Workflow Metric and Trend	All Branches	Booléen	EQUAL_TO
	Branches	String	EQUAL_TO
	Métriques	Struct	
	Tendances	Struct	
	Workflow Names	List	
Workflow Recent Run	All Brances	Booléen	EQUAL_TO
	Branche	String	EQUAL_TO
	Created At	String	
	Credits Used	Entier	
	Duration	Entier	
	ID	String	
	Is Approval	Booléen	
	Start End Date	DateTime	EQUAL_TO, BETWEEN
	Statut	String	
	Stopped At	String	

Entité	Champ	Type de données	Opérateurs pris en charge
Workflow Summary Metric	All Branches	Booléen	EQUAL_TO
	Branche	String	EQUAL_TO
	Métriques	Struct	
	Name	String	
	Project ID	String	
	Reporting Window	String	EQUAL_TO
	Window End	String	
	Window Start	String	
Workflow Test Metric	Average Test Count	Entier	
	Branche	String	EQUAL_TO
	Most Failed Tests	List	
	Most Failed Tests Extra	Entier	
	Slowest Tests	List	
	Slowest Tests Extra	Entier	
	Test Runs	List	
	Total Test Runs	Entier	

> **Note**
>
> Les types de données Struct et List sont convertis en type de données String dans la réponse du connecteur.

Requêtes de partitionnement

CircleCI ne prend pas en charge le partitionnement basé sur les champs ou les enregistrements.

Options de connexion CircleCI

Les options de connexion pour CircleCI sont les suivantes :

- `ENTITY_NAME`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Le nom de votre objet dans CircleCI.
- `API_VERSION`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Version d'API Rest CircleCI que vous souhaitez utiliser.
- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.

Restrictions de CircleCI

Les restrictions ou remarques suivantes concernent CircleCI :

- CircleCI ne prend en charge ni le partitionnement basé sur les champs, ni le partitionnement basé sur les enregistrements.
- Les champs de filtre contenant un « - » (trait d'union) ne fonctionneront que s'ils sont encadrés par des accents graves. Par exemple : `workflow-name` = "abc"
- Le type GitLab VCS ne peut pas être pris en charge car il n'existe aucun moyen programmatique de récupérer l' « ID de projet » requis pour le chemin de l' GitLab entité VCS.

Connexion à Datadog

Datadog est une plateforme de surveillance et d'analytique pour les applications à l'échelle du cloud, notamment les infrastructures, les applications, les services et les outils.

Rubriques

- AWS Glue support pour Datadog
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration de Datadog

- Configuration de connexions Datadog
- Lecture à partir d'entités Datadog
- Options de connexion Datadog
- Création d'un compte Datadog
- Limitations

AWS Glue support pour Datadog

AWS Glue supporte Datadog comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser les jobs AWS Glue ETL pour interroger les données de Datadog.

Prise en charge en tant que cible ?

Non.

Versions d'API Datadog prises en charge

- v1
- v2

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les AWS autorisations requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
```

```
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Si vous ne voulez pas utiliser la méthode ci-dessus, vous pouvez également utiliser les politiques IAM gérées suivantes :

- AWSGlueServiceRole — Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess — Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration de Datadog

Avant de pouvoir AWS Glue effectuer un transfert depuis Datadog, vous devez satisfaire aux exigences suivantes :

Configuration requise

- Vous possédez un compte Datadog avec DD-API-KEY et. DD-APPLICATION-KEY Pour plus d'informations sur la création d'un compte, consultez Creating a Datadog account.
- Votre compte Datadog dispose d'un accès à l'API avec une licence valide.

Datadog prend en charge les six suivants. URLs Tous les clients d'API Datadog sont configurés par défaut pour utiliser le site US1 Datadog. APIs Si vous êtes sur le site européen de Datadog,

vous devez sélectionner l'URL https://api.datadoghq.eu avec le `DD-API-KEY` et du site européen `DD-APPLICATION-KEY` de Datadog pour accéder au. APIs De même, pour les autres sites, vous devez sélectionner le `DD-API-KEY and DD-APPLICATION-KEY` correspondant URLs avec le site concerné.

- US1 URL de l'API — https://api.datadoghq.com https://api.datadoghq.com
- URL de l'API EU – https://api.datadoghq.eu
- US3 URL de l'API — https://api.us3.datadoghq.com
- US5 URL de l'API — https://api.us5.datadoghq.com
- URL DE L'API S1-FED — https://api.ddog-gov.com
- URL de l'API pour le Japon — https://api.ap1.datadoghq.com

Si vous répondez à ces exigences, vous êtes prêt à vous connecter AWS Glue à votre compte Datadog.

Configuration de connexions Datadog

Datadog prend en charge l'authentification personnalisée. Voici les étapes à suivre pour configurer la connexion Datadog :

Pour configurer une connexion Datadog :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes :

   Pour l'application connectée gérée par le client : le secret doit contenir le secret du consommateur de l'application connectée avec **`API_KEY`** et **`APPLICATION_KEY`** comme clés.

   > **Note**
   >
   > Il est indispensable de créer un secret par connexion dans AWS Glue.

2. Dans AWS Glue Studio, créez une connexion sous Connexions de données en suivant les étapes ci-dessous :
   a. Lorsque vous sélectionnez un type de connexion, sélectionnez Datadog.
   b. Indiquez l'`Instance_Url` de Datadog à laquelle vous souhaitez vous connecter.
   c. Sélectionnez le rôle IAM pour lequel AWS Glue vous pouvez assumer et qui dispose des autorisations nécessaires pour effectuer les actions suivantes :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

d. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.

e. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.

3. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.

Lecture à partir d'entités Datadog

Conditions préalables

Un objet Datadog à partir duquel vous souhaitez lire. Reportez-vous au tableau des entités prises en charge ci-dessous pour vérifier les entités disponibles.

Entités prises en charge

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Metrics Timeseries	Oui	Non	Non	Oui	Non

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Log Queries	Oui	Oui	Oui	Oui	Non

Exemple

```
Datadog_read = glueContext.create_dynamic_frame.from_options(
    connection_type="datadog",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "log-queries",
        "API_VERSION": "v2",
        "INSTANCE_URL": "https://api.datadoghq.com",
        "FILTER_PREDICATE": "from = `2023-10-03T09:00:26Z`"
    }
```

Détails des entités et des champs Datadog

Entité	Champ	Type de données	Opérateurs pris en charge
Metrics Timeseries	error	String	NA
	aggr	String	NA
	attributs	Struct	NA
	display_name	String	NA
	end	DateTime	NA
	expression	String	NA
	interval	Entier	NA
	longueur	Entier	NA
	métrique	String	NA

Entité	Champ	Type de données	Opérateurs pris en charge
	pointlist	List	NA
	query_index	Entier	NA
	scope	String	NA
	démarrer	DateTime	NA
	tag_set	List	NA
	unité	Struct	NA
	from_to_date	DateTime	BETWEEN
	query	String	EQUAL_TO
	status	String	NA
	type	String	NA
	hôte	String	NA
Log Queries	id	String	NA
	attributs	Struct	NA
	timestamp	DateTime	NA
	type	String	NA
	from	DateTime	BETWEEN,EQUAL_TO
	index	List	EQUAL_TO
	storage_tier	String	EQUAL_TO
	query	String	EQUAL_TO

Options de connexion Datadog

Les options de connexion pour Datadog sont les suivantes :

- `ENTITY_NAME`(Chaîne) - (Obligatoire) Utilisé pour la lecture/l'écriture. Le nom de votre objet dans Datadog.
- `API_VERSION`(Chaîne) - (Obligatoire) Utilisé pour la lecture/l'écriture. Version d'API Rest Datadog que vous souhaitez utiliser. La version `v1` prend en charge l'entité `metrics-timeseries` tandis que la version `v2` prend en charge l'entité `log-queries`.
- `INSTANCE_URL`(Chaîne) – (Obligatoire) Utilisé pour la lecture. URL d'instance Datadog. L'URL de l'instance Datadog varie selon les régions.
- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.

Création d'un compte Datadog

1. Accédez à https://www.datadoghq.com/.
2. Choisissez COMMENCER GRATUITEMENT.
3. Saisissez les informations requises et inscrivez-vous.
4. Installez le programme d'installation de l'agent Datadog comme suggéré.
5. Assurez-vous que le compte est enregistré auprès d'une organisation valide (dans la liste disponible) disposant d'un abonnement Datadog actif.
6. Une fois connecté à votre compte Datadog, survolez votre nom d'utilisateur dans le coin supérieur droit pour afficher les détails des clés :
   a. Pour obtenir votre clé d'API, choisissez Clés d'API.
   b. Pour obtenir votre clé d'application, choisissez Clés d'application.

Limitations

Les restrictions du connecteur Datadog sont les suivantes :

- Datadog ne prend en charge ni le partitionnement basé sur les champs ni le partitionnement basé sur les enregistrements.
- `from` est un paramètre de filtre obligatoire pour l'entité `Log Queries`.
- `from_to_date` et `query` sont des paramètres de filtre obligatoires pour l'entité `Metrics Timeseries`.

Connexion à Docusign Monitor

Docusign Monitor aide les entreprises à protéger leurs accords grâce au suivi des round-the-clock activités. L'API Monitor fournit ces informations de suivi des activités directement aux piles de sécurité ou aux outils de visualisation des données existants, ce qui permet aux équipes de détecter les activités non autorisées, d'enquêter sur les incidents et de réagir rapidement aux menaces vérifiées. Il offre également la flexibilité dont les équipes de sécurité ont besoin pour personnaliser les tableaux de bord et les alertes afin de répondre aux besoins métier spécifiques.

Rubriques

- AWS Glue support pour Docusign Monitor
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration de Docusign Monitor
- Configuration des connexions Docusign Monitor
- Lecture depuis les entités Docusign Monitor
- Options de connexion Docusign Monitor
- Restrictions de Docusign Monitor

AWS Glue support pour Docusign Monitor

AWS Glue prend en charge Docusign Monitor comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser des tâches AWS Glue ETL pour interroger des données à partir de Docusign Monitor.

Prise en charge en tant que cible ?

Non.

Versions d'API Docusign Monitor prises en charge

Les versions d'API Docusign Monitor suivantes sont prises en charge :

- v2.0

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les autorisations AWS IAM requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Si vous ne voulez pas utiliser la méthode ci-dessus, vous pouvez également utiliser les politiques IAM gérées suivantes :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.

- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration de Docusign Monitor

Avant de pouvoir AWS Glue transférer des données depuis Docusign Monitor vers des destinations prises en charge, vous devez satisfaire aux exigences suivantes :

Configuration requise

Les exigences minimales sont les suivantes :

- Vous disposez d'un compte Docusign sur lequel vous pouvez utiliser le produit Docusign Software dans Docusign Monitor.
- Dans la console de développement de votre compte Docusign, vous avez créé une application d'intégration OAuth 2.0 pour. AWS Glue

  Cette application fournit les informations d'identification du client qui AWS Glue permettent d'accéder à vos données en toute sécurité lorsqu'elle passe des appels authentifiés vers votre compte. Pour plus d'informations, consultez la OAuth version 2.0 dans la documentation de Docusign Monitor.

Si vous répondez à ces exigences, vous êtes prêt à vous connecter AWS Glue à votre compte Docusign Monitor.

Configuration des connexions Docusign Monitor

Docusign Monitor prend en charge le type d'autorisation AUTHORIZATION_CODE.

- Ce type de subvention est considéré comme tridimensionnel OAuth car il repose sur la redirection des utilisateurs vers le serveur d'autorisation tiers pour authentifier l'utilisateur. Il est utilisé lors de la création de connexions via la AWS Glue console.
- Les utilisateurs peuvent choisir de créer leur propre application connectée dans Docusign Monitor et de fournir leur propre identifiant client et leur propre secret client lors de la création de

connexions via la AWS Glue console. Dans ce scénario, ils seront toujours redirigés vers Docusign Monitor pour se connecter et autoriser l'accès AWS Glue à leurs ressources.

- Ce type d'autorisation donne lieu à un jeton d'actualisation et à un jeton d'accès. Le jeton d'accès est de courte durée et peut être actualisé automatiquement sans interaction de l'utilisateur à l'aide du jeton d'actualisation.
- Pour consulter la documentation publique de Docusign Monitor sur la création d'une application connectée pour le OAuth flux de code d'autorisation, consultez Docusign OAuth Connect.

Pour configurer une connexion Docusign Monitor :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes :
   a. Pour l'application connectée gérée par le client, le secret doit contenir la clé d'API de l'application connectée avec `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` comme clé.
   b. Remarque : vous devez créer un secret pour vos connexions dans AWS Glue.

1. Dans AWS Glue Glue Studio, créez une connexion sous Data Connections en suivant les étapes ci-dessous :
   a. Sous Connexions, choisissez Créer une connexion.
   b. Lorsque vous sélectionnez une source de données, sélectionnez Docusign Monitor.
   c. Sélectionnez le rôle AWS IAM qui AWS Glue peut assumer les actions suivantes et qui dispose des autorisations nécessaires pour les actions suivantes :

      JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
```

```
      ],
      "Resource": "*"
    }
  ]
}
```

d. Fournissez l'application client gérée par l'utilisateur ClientId de l'application Docusign Monitor.

e. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.

f. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.

2. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.

Lecture depuis les entités Docusign Monitor

Prérequis

Un objet Docusign Monitor à partir duquel vous souhaitez lire.

Entités prises en charge pour la source :

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Surveillance des données	Oui	Oui	Non	Oui	Non

Exemple :

```
docusignmonitor_read = glueContext.create_dynamic_frame.from_options(
    connection_type="docusign_monitor",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "monitoring-data",
        "API_VERSION": "v2.0"
    }
```

Détails des entités et des champs Docusign Monitor

Entités avec métadonnées statiques :

Entité	Champ	Type de données	Opérateurs pris en charge
Surveillance des données	timestamp	DateTime	N/A
	eventId	String	N/A
	vigie	String	N/A
	environment	String	N/A
	site	String	N/A
	traceToken	String	N/A
	organizationId	String	N/A
	accountId	String	N/A
	userId	String	N/A
	objet	String	N/A
	action	String	N/A
	property	String	N/A
	field	String	N/A
	result	String	N/A
	IntegratorKey	String	N/A
	customerVisible	String	N/A
	version	String	N/A
	userAgent	String	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	userAgentClientInfos	Struct	N/A
	ipAddress	String	N/A
	ipAddressLocation	Struct	N/A
	data	String	N/A
	source	String	N/A
	latitude	Double	N/A
	longitude	Double	N/A
	city	String	N/A
	state	String	N/A
	country	String	N/A
	usUserMemberOfDomain	Booléen	N/A
	affectedUserIsMemberOfDomain	Booléen	N/A
	proxyStatus	String	N/A
	proxyType	String	N/A
	proxyLevel	String	N/A
	referencedUserId	String	N/A
	device	String	N/A
	browser	String	N/A
	curseur	DateTime	EQUAL_TO

Requêtes de partitionnement

Docusign Monitor ne prend en charge ni le partitionnement basé sur les champs ni le partitionnement basé sur les enregistrements.

Options de connexion Docusign Monitor

Les options de connexion pour Docusign Monitor sont les suivantes :

- `ENTITY_NAME`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Le nom de votre objet dans Docusign Monitor.
- `API_VERSION`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Version de l'API Rest de Docusign Monitor que vous souhaitez utiliser.
- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.
- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.

Restrictions de Docusign Monitor

Les restrictions ou remarques suivantes concernent Docusign Monitor :

- Lorsqu'un filtre est appliqué à l'aide du champ `cursor`, l'API récupère les enregistrements des sept prochains jours à compter de la date spécifiée.
- Si aucun filtre n'est fourni, l'API récupère les enregistrements des sept jours précédents à compter de la date actuelle de la demande d'API.
- Docusign Monitor ne prend en charge ni le partitionnement basé sur les champs, ni le partitionnement basé sur les enregistrements.
- Docusign Monitor ne prend pas en charge la fonctionnalité Order By.

Connexion à Domo

Domo est un outil de tableau de bord cloud. Avec la plateforme d'applications d'entreprise de Domo, les bases dont vous avez besoin pour étendre Domo sont en place, afin que vous puissiez créer vos solutions personnalisées plus rapidement.

Rubriques

- AWS Glue support pour Domo
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration de Domo
- Configuration de connexions Domo
- Lecture à partir d'entités Domo
- Options de connexion Domo
- Restrictions de Domo

AWS Glue support pour Domo

AWS Glue supporte Domo comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser les tâches AWS Glue ETL pour interroger les données de Domo.

Prise en charge en tant que cible ?

Non.

Versions de l'API Domo prises en charge

Les versions d'API Domo suivantes sont prises en charge :

- v1

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les autorisations AWS IAM requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
```

```
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Si vous ne voulez pas utiliser la méthode ci-dessus, vous pouvez également utiliser les politiques IAM gérées suivantes :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration de Domo

Avant de pouvoir transférer des AWS Glue données depuis Domo vers des destinations prises en charge, vous devez satisfaire aux exigences suivantes :

Configuration requise

Les exigences minimales sont les suivantes :

- Vous disposez d'un compte Domo activé pour accéder à l'API.
- Vous avez une application dans votre compte de développeur Domo qui fournit les informations d'identification du client qui permettent d' AWS Glue accéder à vos données en toute sécurité

lorsqu'il passe des appels authentifiés vers votre compte. Pour de plus amples informations, veuillez consulter Création d'une application pour développeurs Domo.

Si vous remplissez ces conditions, vous êtes prêt à vous connecter AWS Glue à votre compte Domo.

Création d'une application pour développeurs Domo

Pour obtenir l'ID client et le secret client, vous devez créer un compte développeur.

1. Accédez à la page de connexion du développeur Domo.
2. Choisissez Login (Connexion).
3. Indiquez le nom de domaine et cliquez sur Continuer.
4. Passez la souris sur Mon compte et choisissez Nouveau client.
5. Indiquez le nom et la description, sélectionnez la portée (« données ») et choisissez Créer.
6. Récupérez l'ID client et le secret client générés auprès du nouveau client créé.

Configuration de connexions Domo

Domo prend en charge le type de subvention CLIENT_CREDENTIALS pour. OAuth2

- Ce type de subvention est considéré comme bidimensionnel OAuth car seule l'application cliente s'authentifie auprès du serveur, sans aucune implication de l'utilisateur.
- Les utilisateurs peuvent choisir de créer leur propre application connectée dans Domo et de fournir leur propre identifiant client et leur propre secret client lors de la création de connexions via la AWS Glue console.
- Pour consulter la documentation publique de Domo sur la création d'une application connectée pour le OAuth flux de code d'autorisation, consultez OAuth Authentification.

Pour configurer une connexion Domo :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes :
   a. Pour l'application connectée gérée par le client, le secret doit contenir le jeton d'accès à l'application connectée, `client_id` et `client_secret`.
   b. Remarque : vous devez créer un secret pour vos connexions dans AWS Glue.

1. Dans AWS Glue Glue Studio, créez une connexion sous Data Connections en suivant les étapes ci-dessous :
   a. Sous Connexions, choisissez Créer une connexion.
   b. Lorsque vous sélectionnez une source de données, sélectionnez Domo.
   c. Sélectionnez le rôle AWS IAM qui AWS Glue peut assumer et dispose des autorisations nécessaires pour les actions suivantes :

      JSON

      ```
      {
        "Version":"2012-10-17",
        "Statement": [
          {
            "Effect": "Allow",
            "Action": [
              "secretsmanager:DescribeSecret",
              "secretsmanager:GetSecretValue",
              "secretsmanager:PutSecretValue",
              "ec2:CreateNetworkInterface",
              "ec2:DescribeNetworkInterfaces",
              "ec2:DeleteNetworkInterface"
            ],
            "Resource": "*"
          }
        ]
      }
      ```

   d. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.
   e. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.
2. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.

Lecture à partir d'entités Domo

Prérequis

Un objet Domo à partir duquel vous souhaitez lire. Vous aurez besoin du nom de l'objet, tel que le jeu de données ou les politiques d'autorisation des données. Le tableau suivant présente les entités prises en charge.

Entités prises en charge pour la source :

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Ensemble de données	Oui	Oui	Oui	Oui	Oui
Politiques d'autorisation des données	Non	Non	Non	Oui	Non

Exemple :

```
Domo_read = glueContext.create_dynamic_frame.from_options(
    connection_type="domo",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "dataset",
        "API_VERSION": "v1"
    }
```

Détails des entités et des champs Domo

Entités avec métadonnées statiques :

Entité	Champ	Type de données	Opérateurs pris en charge
Politiques d'autorisation des données	id	Long	N/A
	type	String	N/A
	name	String	N/A
	filtres	List	N/A
	users	List	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	virtualUsers	List	N/A
	groups	List	N/A

Pour l'entité suivante, Domo fournit des points de terminaison permettant de récupérer les métadonnées de manière dynamique, afin que la prise en charge des opérateurs soit capturée au niveau du type de données pour l'entité.

Entité	Type de données	Opérateurs pris en charge
Ensemble de données	Entier	=, !=, <, >, >=, <=
	Long	=, !=, <, >, >=, <=
	String	=, !=, CONTAINS
	Date	=, >, >=, <, <=, BETWEEN
	DateTime	=, <, <=, >, >=, BETWEEN
	Booléen	=, !=
	Double	=, !=, <, >, >=, <=
	List	N/A
	Struct	N/A

Requêtes de partitionnement

Partitionnement basé sur les champs

Vous pouvez indiquer les options Spark supplémentaires `PARTITION_FIELD`, `LOWER_BOUND`, `UPPER_BOUND` et `NUM_PARTITIONS` si vous souhaitez utiliser la simultanéité dans Spark. Avec ces paramètres, la requête d'origine serait divisée en `NUM_PARTITIONS` nombres de sous-requêtes pouvant être exécutées simultanément par les tâches Spark.

- PARTITION_FIELD : le nom du champ à utiliser pour partitionner la requête.
- LOWER_BOUND : une valeur limite inférieure inclusive du champ de partition choisi.

  Pour le DateTime champ, nous acceptons la valeur au format ISO.

  Exemple de valeur valide :

  ```
  "2023-01-15T11:18:39.205Z"
  ```

  Pour le champ Date, nous acceptons la valeur au format ISO.

  Exemple de valeur valide :

  ```
  "2023-01-15"
  ```

- UPPER_BOUND : une valeur limite supérieure exclusive du champ de partition choisi.

  Exemple de valeur valide :

  ```
  "2023-02-15T11:18:39.205Z"
  ```

- NUM_PARTITIONS : le nombre de partitions.

Les détails relatifs à la prise en charge des champs de partitionnement par entité sont présentés dans le tableau suivant :

Nom de l'entité	Champs de partitionnement	Type de données
Jeu de données	N'importe quel champ Date/Time basé [métadonnées dynamiques]	DateTime
	N'importe quel champ basé sur la date [métadonnées dynamiques]	Date

Exemple :

```
Domo_read = glueContext.create_dynamic_frame.from_options(
    connection_type="domo",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "dataset",
        "API_VERSION": "v1",
        "PARTITION_FIELD": "permissionTime"
        "LOWER_BOUND": "2023-01-15T11:18:39.205Z"
        "UPPER_BOUND": "2023-02-15T11:18:39.205Z"
        "NUM_PARTITIONS": "2"
    }
```

Partitionnement basé sur les enregistrements

Vous pouvez indiquer l'option Spark supplémentaire `NUM_PARTITIONS` si vous souhaitez utiliser la simultanéité dans Spark. Avec ce paramètre, la requête d'origine serait divisée en `NUM_PARTITIONS` nombres de sous-requêtes pouvant être exécutées simultanément par les tâches Spark.

Dans le partitionnement basé sur les enregistrements, le nombre total d'enregistrements présents est demandé par Domo, puis divisé par le nombre `NUM_PARTITIONS` fourni. Le nombre d'enregistrements qui en résulte est ensuite extrait simultanément par chaque sous-requête.

Exemple :

```
Domo_read = glueContext.create_dynamic_frame.from_options(
    connection_type="domo",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "dataset",
        "API_VERSION": "v1",
        "NUM_PARTITIONS": "2"
    }
```

Options de connexion Domo

Les options de connexion pour Domo sont les suivantes :

- `ENTITY_NAME`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Le nom de votre objet dans Domo.
- `API_VERSION`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Version d'API Rest Domo que vous souhaitez utiliser.

- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.
- `PARTITION_FIELD`(Chaîne) - Utilisé pour la lecture. Champ à utiliser pour partitionner la requête.
- `LOWER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite inférieure inclusive du champ de partition choisi.
- `UPPER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS`(Entier) - Par défaut : 1. Utilisé pour la lecture. Nombre de partitions pour la lecture.

Restrictions de Domo

Les restrictions ou remarques suivantes concernent Domo :

- En raison d'une limitation du kit SDK, le filtrage ne fonctionne pas comme prévu pour les champs interrogeables commençant par « _ » (par exemple : _BATCH_ID).
- En raison d'une restriction de l'API, le filtrage fonctionne à la date antérieure à la date que vous avez indiquée. Cela affecte également l'extraction incrémentielle. Pour contourner cette limitation, sélectionnez une date en fonction de votre fuseau horaire par rapport à l'UTC, afin d'obtenir les données correspondant à la date requise.

Connexion à Dynatrace

Dynatrace est une plateforme qui propose des outils d'analytique et d'automatisation pour une observabilité et une sécurité complètes. Elle est spécialisée dans la surveillance et l'optimisation des performances des applications, de l'infrastructure et de l'expérience utilisateur.

Rubriques

- AWS Glue support pour Dynatrace
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration de Dynatrace

- Configuration de connexions Dynatrace
- Lecture à partir d'entités Dynatrace
- Options de connexion Dynatrace
- Restrictions de Dynatrace

AWS Glue support pour Dynatrace

AWS Glue prend en charge Dynatrace comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser les tâches AWS Glue ETL pour interroger les données de Dynatrace.

Prise en charge en tant que cible ?

Non.

Versions de l'API Dynatrace prises en charge

Les versions d'API Dynatrace suivantes sont prises en charge :

- v2

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les autorisations AWS IAM requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
```

```
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Si vous ne voulez pas utiliser la méthode ci-dessus, vous pouvez également utiliser les politiques IAM gérées suivantes :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration de Dynatrace

Avant de pouvoir AWS Glue transférer des données depuis Dynatrace, vous devez satisfaire aux exigences suivantes :

Configuration requise

Les exigences minimales sont les suivantes :

- Vous disposez d'un compte Dynatrace.
- Vous avez généré un jeton d'accès dans votre compte Dynatrace avec l' read/write étendue appropriée attribuée pour accéder au. APIs Pour plus d'informations, consultez Generate a token.

Si vous remplissez ces conditions, vous êtes prêt à vous connecter AWS Glue à votre compte Dynatrace.

Configuration de connexions Dynatrace

Dynatrace prend en charge l'authentification personnalisée.

Pour configurer une connexion Dynatrace :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes :
   a. Pour l'application connectée gérée par le client, le secret doit contenir la clé d'API de l'application connectée avec *apiToken* comme clé.
   b. Remarque : vous devez créer un secret pour vos connexions dans AWS Glue.

1. Dans AWS Glue Glue Studio, créez une connexion sous Data Connections en suivant les étapes ci-dessous :
   a. Lorsque vous sélectionnez une source de données, sélectionnez Dynatrace.
   b. Indiquez l'`INSTANCE_URL` du compte Dynatrace auquel vous souhaitez vous connecter.
   c. Sélectionnez le rôle AWS IAM qui AWS Glue peut assumer et dispose des autorisations nécessaires pour les actions suivantes :

      JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

   d. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.

e. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.

2. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.

Lecture à partir d'entités Dynatrace

Prérequis

Un objet Dynatrace à partir duquel vous souhaitez lire. Vous aurez besoin du nom de l'objet, comme « problem ».

Entités prises en charge pour la source :

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Problème	Oui	Oui	Oui	Oui	Non

Exemple :

```
Dynatrace_read = glueContext.create_dynamic_frame.from_options(
    connection_type="Dynatrace",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "problem",
        "API_VERSION": "v2",
        "INSTANCE_URL": "https://[instanceName].live.dynatrace.com"
    }
```

Détails des entités et des champs Dynatrace :

Dynatrace fournit des points de terminaison permettant de récupérer les métadonnées de manière dynamique pour les entités prises en charge. Par conséquent, la prise en charge des opérateurs est capturée au niveau du type de données.

Entité	Champ	Type de données	Opérateurs pris en charge
Problème	affectedEntities	List	EQUAL_TO
	displayId	String	EQUAL_TO
	endTime	DateTime	
	entityTags	List	
	evidenceDetails	Struct	
	impactAnalysis	Struct	
	impactLevel	String	EQUAL_TO
	impactedEntities	List	EQUAL_TO
	linkedProblemInfo	Struct	
	managementZones	List	EQUAL_TO
	problemFilters	List	
	recentComments	Struct	
	rootCauseEntity	Struct	EQUAL_TO
	problemId	String	EQUAL_TO
	severityLevel	String	EQUAL_TO
	startTime	DateTime	BETWEEN
	status	String	EQUAL_TO
	title	String	
	from	DateTime	EQUAL_TO, BETWEEN

Entité	Champ	Type de données	Opérateurs pris en charge
	problemFilterIds	String	EQUAL_TO
	problemFilterNames	String	EQUAL_TO
	managementZoneIds	String	EQUAL_TO
	text	String	EQUAL_TO
	underMaintenance	Booléen	EQUAL_TO
	message	String	

Requêtes de partitionnement

Dynatrace ne prend pas en charge le partitionnement basé sur les champs ou le partitionnement basé sur les enregistrements.

Options de connexion Dynatrace

Les options de connexion pour Dynatrace sont les suivantes :

- `ENTITY_NAME`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Le nom de votre objet dans Dynatrace.
- `API_VERSION`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Version d'API Rest Dynatrace que vous souhaitez utiliser.
- `INSTANCE_URL`(Chaîne) - Utilisé pour la lecture. URL d'instance Dynatrace valide.
- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.

Restrictions de Dynatrace

Les restrictions ou remarques suivantes concernent Dynatrace :

- Dynatrace ne prend en charge ni le partitionnement basé sur les champs, ni le partitionnement basé sur les enregistrements.
- Pour la fonctionnalité Tout sélectionner, si vous fournissez le « champ » dans le filtre, le nombre d'enregistrements ne pourra pas dépasser 10 par page.
- La taille maximale de la page prise en charge est de 500. Si vous sélectionnez l'un des champs [`evidenceDetails, impactAnalysis, recentComments`] lors de la création du flux, le nombre d'enregistrements par page sera défini par défaut sur 10.

Connexion à Facebook Ads

Facebook Ads est une plateforme de publicité numérique performante utilisée par les entreprises de toutes tailles pour toucher leur public cible et atteindre divers objectifs marketing. La plateforme permet aux annonceurs de créer des publicités personnalisées qui peuvent être affichées sur l'ensemble de la gamme d'applications et de services de Facebook, notamment Facebook et Messenger. Grâce à ses fonctionnalités de ciblage avancées, Facebook Ads permet aux entreprises d'atteindre des groupes démographiques, des centres d'intérêt, des comportements et des emplacements spécifiques.

Rubriques

- AWS Glue support pour Facebook Ads
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration de Facebook Ads
- Configuration des connexions Facebook Ads
- Lecture depuis les entités Facebook Ads
- Options de connexion Facebook Ads
- Restrictions et remarques relatives au connecteur Facebook Ads

AWS Glue support pour Facebook Ads

AWS Glue prend en charge les publicités Facebook comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser des tâches AWS Glue ETL pour interroger des données provenant de Facebook Ads.

Prise en charge en tant que cible ?

Non.

Versions de l'API Facebook Ads prises en charge

Les versions suivantes de l'API Facebook Ads sont prises en charge :

- v17.0
- v18.0
- v19.0
- v20.0

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les autorisations AWS IAM requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Si vous ne voulez pas utiliser la méthode ci-dessus, vous pouvez également utiliser les politiques IAM gérées suivantes :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration de Facebook Ads

Avant de pouvoir AWS Glue transférer des données depuis Facebook Ads, vous devez satisfaire aux exigences suivantes :

Configuration requise

Les exigences minimales sont les suivantes :

- Les comptes Facebook Standard sont accessibles directement via Facebook.
- L'authentification de l'utilisateur est nécessaire pour générer le jeton d'accès.
- Le connecteur du SDK Facebook Ads implémentera le OAuth flux de jetons d'accès utilisateur.
- Nous utilisons OAuth2 .0 pour authentifier nos demandes d'API auprès de Facebook Ads. Cette authentification Web s'inscrit dans le cadre de l'architecture d'authentification multifactorielle (MFA), qui est un sur-ensemble de l'authentification à deux facteurs.
- L'utilisateur doit accorder des autorisations pour accéder aux points de terminaison. Pour accéder aux données de l'utilisateur, l'autorisation du point de terminaison est gérée par le biais d'autorisations et de fonctionnalités.

Obtenir des informations d'identification OAuth 2.0

Pour obtenir des informations d'identification d'API afin de pouvoir passer des appels authentifiés vers votre instance, consultez REST API dans le Guide du développeur Facebook Ads.

Configuration des connexions Facebook Ads

Facebook Ads prend en charge le type de subvention AUTHORIZATION_CODE pour. OAuth2

- Ce type de subvention est considéré comme tridimensionnel OAuth car il repose sur la redirection des utilisateurs vers le serveur d'autorisation tiers pour authentifier l'utilisateur. Il est utilisé lors de la création de connexions via la AWS Glue console.
- Les utilisateurs peuvent toujours choisir de créer leur propre application connectée dans Facebook Ads et de fournir leur propre identifiant client et leur secret client lors de la création de connexions via la AWS Glue console. Dans ce scénario, ils seront toujours redirigés vers Facebook Ads pour se connecter et autoriser AWS Glue l'accès à leurs ressources.
- Ce type d'autorisation donne lieu à un jeton d'accès. Un jeton d'utilisateur système qui arrive à expiration est valide pendant 60 jours à compter de la date de génération ou d'actualisation. Pour assurer la continuité, le développeur doit actualiser le jeton d'accès dans les 60 jours. À défaut, cela entraîne la perte du jeton d'accès et oblige le développeur à en obtenir un nouveau pour retrouver l'accès à l'API. Consultez Actualiser un token d'accès.
- Pour consulter la documentation relative aux publicités Facebook publiques sur la création d'une application connectée pour le OAuth flux de code d'autorisation, consultez la section Utiliser la OAuth version 2.0 pour accéder à Google APIs dans le guide Google pour les développeurs.

Pour configurer une connexion Facebook Ads :

1. Dans AWS Glue Glue Studio, créez une connexion sous Data Connections en suivant les étapes ci-dessous :
   a. Lorsque vous sélectionnez un type de connexion, sélectionnez Facebook Ads.
   b. Indiquez l'`INSTANCE_URL` de l'instance Facebook Ads à laquelle vous souhaitez vous connecter.
   c. Sélectionnez le rôle AWS IAM qui AWS Glue peut assumer et dispose des autorisations nécessaires pour les actions suivantes :

      JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
```

```
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

d. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.

e. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.

2. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.

Lecture depuis les entités Facebook Ads

Prérequis

Un objet Facebook Ads à partir duquel vous souhaitez lire. Vous aurez besoin du nom de l'objet. Les tableaux suivants présentent les entités prises en charge.

Entités prises en charge pour la source :

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Campaign	Oui	Oui	Non	Oui	Oui
Ad Set	Oui	Oui	Non	Oui	Oui
Ads	Oui	Oui	Non	Oui	Oui
Ad Creative	Non	Oui	Non	Oui	Non

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Insights - Account	Non	Oui	Non	Oui	Non
Adaccounts	Oui	Oui	Non	Oui	Non
Insights - Ad	Oui	Oui	Non	Oui	Oui
Perspectives - AdSet	Oui	Oui	Non	Oui	Oui
Insights - Campaign	Oui	Oui	Non	Oui	Oui

Exemple :

```
FacebookAds_read = glueContext.create_dynamic_frame.from_options(
    connection_type="FacebookAds",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "API_VERSION": "v20.0"
    }
```

Détails des entités et des champs Facebook Ads

Pour plus d'informations sur les entités et de détails sur les champs, consultez :

- Ad Account
- Campaign
- Ad Set
- Ad
- Ad Creative
- Insight Ad Account
- Insights Ads

- Perspectives AdSets
- Insights Campaigns

Pour plus d'informations, consultez API Marketing.

> **Note**
>
> Les types de données Struct et List sont convertis en type de données String dans la réponse des connecteurs.

Requêtes de partitionnement

Vous pouvez indiquer les options Spark supplémentaires `PARTITION_FIELD`, `LOWER_BOUND`, `UPPER_BOUND` et `NUM_PARTITIONS` si vous souhaitez utiliser la simultanéité dans Spark. Avec ces paramètres, la requête d'origine serait divisée en `NUM_PARTITIONS` nombres de sous-requêtes pouvant être exécutées simultanément par les tâches Spark.

- `PARTITION_FIELD` : le nom du champ à utiliser pour partitionner la requête.
- `LOWER_BOUND` : une valeur limite inférieure inclusive du champ de partition choisi.

  Pour le DateTime champ, nous acceptons le format d'horodatage Spark utilisé dans les requêtes SQL Spark.

  Exemple de valeur valide :

  ```
  "2022-01-01"
  ```
- `UPPER_BOUND` : une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS` : le nombre de partitions.

Exemple :

```
FacebookADs_read = glueContext.create_dynamic_frame.from_options(
    connection_type="FacebookAds",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "API_VERSION": "v20.0",
```

```
        "PARTITION_FIELD": "created_time"
        "LOWER_BOUND": "2022-01-01"
        "UPPER_BOUND": "2024-01-02"
        "NUM_PARTITIONS": "10"
    }
```

Options de connexion Facebook Ads

Les options de connexion pour Facebook Ads sont les suivantes :

- `ENTITY_NAME`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Le nom de votre objet dans Facebook Ads.
- `API_VERSION`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Version d'API Facebook Ads que vous souhaitez utiliser. Par exemple : v1.
- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.
- `PARTITION_FIELD`(Chaîne) - Utilisé pour la lecture. Champ à utiliser pour partitionner la requête.
- `LOWER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite inférieure inclusive du champ de partition choisi.
- `UPPER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS`(Entier) - Par défaut : 1. Utilisé pour la lecture. Nombre de partitions pour la lecture.
- `TRANSFER_MODE`(Chaîne) - Par défaut : SYNC. Utilisé pour la lecture asynchrone.

Restrictions et remarques relatives au connecteur Facebook Ads

Voici les restrictions ou les remarques relatives au connecteur Facebook Ads :

- Comme Facebook Ads prend en charge les métadonnées dynamiques, tous les champs peuvent être interrogés. Tous les champs prennent en charge le filtrage et les enregistrements sont récupérés si les données sont disponibles, sinon Facebook renvoie une réponse Bad request (400) avec un message d'erreur approprié.

- Le nombre d'appels d'une application est le nombre d'appels qu'un utilisateur peut passer pendant une période d'une heure continue, 200 multiplié par le nombre d'utilisateurs. Pour plus de détails sur les plafonds, consultez Plafonds et Plafonds Business Use Case.

Connexion à Facebook Page Insights

Facebook Page permet aux entreprises et aux autres groupes d'intérêt de créer des pages pour le réseau social Facebook.com. Les entreprises utilisent ces pages pour partager leurs heures d'ouverture, faire des annonces et interagir avec les clients en ligne. Si vous êtes un utilisateur de Facebook Page Insights, vous pouvez vous connecter AWS Glue à votre compte Facebook Page Insights. Vous pouvez utiliser Facebook Page Insights comme source de données dans vos tâches ETL. Exécutez ces tâches pour transférer les données de Facebook Page Insights vers AWS des services ou d'autres applications compatibles.

Rubriques

- AWS Glue support pour Facebook Page Insights
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration de Facebook Page Insights
- Configuration des connexions Facebook Page Insights
- Lecture à partir d'entités Facebook Page Insights
- Options de connexion Facebook Page Insights
- Restrictions et remarques relatives au connecteur Facebook Page Insights

AWS Glue support pour Facebook Page Insights

AWS Glue prend en charge Facebook Page Insights comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser les tâches AWS Glue ETL pour interroger les données de Facebook Page Insights.

Prise en charge en tant que cible ?

Non.

Versions de l'API Facebook Page Insights prises en charge

Les versions suivantes de l'API Facebook Page Insights sont prises en charge :

- v17
- v18
- v19
- v20
- v21

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les autorisations AWS IAM requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Si vous ne voulez pas utiliser la méthode ci-dessus, vous pouvez également utiliser les politiques IAM gérées suivantes :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources

spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.

- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration de Facebook Page Insights

Avant de pouvoir AWS Glue transférer des données depuis Facebook Page Insights, vous devez satisfaire aux exigences suivantes :

Configuration requise

Les exigences minimales sont les suivantes :

- Les comptes Facebook Standard sont accessibles directement via Facebook.
- L'authentification de l'utilisateur est nécessaire pour générer le jeton d'accès.
- Le connecteur Facebook Page Insights implémente le OAuth flux de jetons d'accès utilisateur.
- Le connecteur utilise la valeur OAuth2 .0 pour authentifier nos demandes d'API auprès de Facebook Page Insights. Cela s'inscrit dans le cadre de l'architecture d'authentification multifactorielle (MFA), qui est un sur-ensemble de l'authentification à deux facteurs. Il s'agit d'une authentification Web.
- L'utilisateur doit accorder des autorisations pour accéder aux points de terminaison. Pour accéder aux données de l'utilisateur, l'autorisation du point de terminaison est gérée par le biais d'autorisations et de fonctionnalités.

Configuration des connexions Facebook Page Insights

Pour configurer une connexion Facebook Page Insights :

1. Dans AWS Glue Glue Studio, créez une connexion sous Data Connections en suivant les étapes ci-dessous :
   a. Lorsque vous sélectionnez un type de connexion, sélectionnez Facebook Page Insights.

b. Sélectionnez le rôle AWS IAM qui AWS Glue peut assumer les actions suivantes et qui dispose des autorisations nécessaires pour les actions suivantes :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

c. Sélectionnez l’URL du code d’autorisation.

d. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.

e. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.

2. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.

Lecture à partir d’entités Facebook Page Insights

Prérequis

Un objet Facebook Page Insights à partir duquel vous souhaitez lire. Vous aurez besoin du nom de l’objet.

Entités prises en charge pour la source :

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Page Content	Oui	Non	Oui	Oui	Oui
Page CTA Clicks	Oui	Non	Non	Oui	Oui
Page Engagement	Oui	Non	Non	Oui	Oui
Page Impressions	Oui	Non	Non	Oui	Oui
Page Posts	Oui	Non	Non	Oui	Oui
Page Post Engagement	Non	Non	Non	Oui	Non
Page Post Reactions	Non	Non	Non	Oui	Non
Page Reactions	Oui	Non	Non	Oui	Oui
Narrations	Oui	Non	Non	Oui	Oui
Page User Demograph ics	Oui	Non	Non	Oui	Oui
Page Video Views	Oui	Non	Non	Oui	Oui
Page Views	Oui	Non	Non	Oui	Oui
Page Video Posts	Oui	Non	Non	Oui	Oui

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Pages	Non	Oui	Non	Oui	Non
Feeds	Oui	Oui	Non	Oui	Oui

Exemple :

```
facebookPageInsights_read = glueContext.create_dynamic_frame. from options(
    connection_type="facebookpageinsights",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "API_VERSION": "v21"
   }
```

Détails de champs Facebook Page Insights :

Entité	Champ	Type de données	Opérateurs pris en charge
Page Content	Name	String	N/A
	Period	Period	EQUAL_TO
	Since	DateTime	EQUAL_TO
	Valeurs	List	N/A
	Titre	String	N/A
	Description	String	N/A
	description_from_api_doc	String	N/A
	Id	String	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
Page CTA Clicks	Name	String	N/A
	Period	Period	EQUAL_TO
	Since	DateTime	EQUAL_TO
	Valeurs	List	N/A
	Titre	String	N/A
	Description	String	N/A
	description_from_api_doc	String	N/A
	Id	String	N/A
Page Engagement	Name	String	N/A
	Period	Period	EQUAL_TO
	Since	DateTime	EQUAL_TO
	Valeurs	List	N/A
	Titre	String	N/A
	Description	String	N/A
	description_from_api_doc	String	N/A
	Id	String	N/A
Page Impressions	Name	String	N/A
	Period	Period	EQUAL_TO
	Since	DateTime	EQUAL_TO

Entité	Champ	Type de données	Opérateurs pris en charge
	Valeurs	List	N/A
	Titre	String	N/A
	Description	String	N/A
	description_from_api_doc	String	N/A
	Id	String	N/A
Page Posts	Name	String	N/A
	Period	Period	EQUAL_TO
	Since	DateTime	EQUAL_TO
	Valeurs	List	N/A
	Titre	String	N/A
	Description	String	N/A
	description_from_api_doc	String	N/A
	Id	String	N/A
Page Post Engagement	Name	String	N/A
	Period	Period	EQUAL_TO
	Valeurs	List	N/A
	Titre	String	N/A
	Description	String	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	description_from_api_doc	String	N/A
	Id	String	N/A
Page Post Reactions	Name	String	N/A
	Period	Period	EQUAL_TO
	Valeurs	List	N/A
	Titre	String	N/A
	Description	String	N/A
	description_from_api_doc	String	N/A
	Id	String	N/A
Page User Demographics	Name	String	N/A
	Period	Period	EQUAL_TO
	Since	DateTime	EQUAL_TO
	Valeurs	List	N/A
	Titre	String	N/A
	Description	String	N/A
	description_from_api_doc	String	N/A
	Id	String	N/A
Page Video Views	Name	String	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	Period	Period	EQUAL_TO
	Since	DateTime	EQUAL_TO
	Valeurs	List	N/A
	Titre	String	N/A
	Description	String	N/A
	description_from_api_doc	String	N/A
	Id	String	N/A
Page Views	Name	String	N/A
	Period	Period	EQUAL_TO
	Since	DateTime	EQUAL_TO
	Valeurs	List	N/A
	Titre	String	N/A
	Description	String	N/A
	description_from_api_doc	String	N/A
	Id	String	N/A
Page Video Posts	Name	String	N/A
	Period	Period	EQUAL_TO
	Since	DateTime	EQUAL_TO
	Valeurs	List	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	Titre	String	N/A
	Description	String	N/A
	description_from_api_doc	String	N/A
	Id	String	N/A
Pages	Name	String	N/A
	À propos	String	N/A
	access_token	String	N/A
	ad_campaign	String	N/A
	Affiliation	String	N/A
	app_id	String	N/A
	artists_we_like	String	N/A
	Attire	String	N/A
	Awards	String	N/A
	band_interests	String	N/A
	band_members	String	N/A
	best_page	String	N/A
	Bio	String	N/A
	Anniversaire	String	N/A
	booking_agent	String	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	Built	String	N/A
	can_checkin	String	N/A
	can_post	String	N/A
	Catégorie	String	N/A
	category_list	List	N/A
	Checkins	Entier	N/A
	company_overview	String	N/A
	connected_instagram_account	String	N/A
	contact_address	String	N/A
	country_page_likes	Entier	N/A
	Couverture	Struct	N/A
	culinary_team	String	N/A
	current_location	String	N/A
	delivery_and_pickup_option_info	List	N/A
	Description	String	N/A
	description_html	String	N/A
	differently_open_offerings	List	N/A
	directed_by	String	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	display_subtext	String	N/A
	displayed_message_response_time	String	N/A
	Emails	String	N/A
	Engagement	String	N/A
	fan_count	Entier	N/A
	featured_video	String	N/A
	Caractéristiques	String	N/A
	followers_count	Entier	N/A
	food_styles	List	N/A
	Founded	String	N/A
	general_info	String	N/A
	general_manager	String	N/A
	Genre	String	N/A
	global_brand_page_name	String	N/A
	global_brand_root_id	String	N/A
	has_added_app	Booléen	N/A
	has_transitioned_to_new_page_experience	Booléen	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	has_whatsapp_business_number	Booléen	N/A
	has_whatsapp_number	Booléen	N/A
	Hometown	String	N/A
	Heures	Struct	N/A
	Impressum	String	N/A
	Influences	String	N/A
	instagram_business_account	String	N/A
	is_always_open	Booléen	N/A
	is_chain	Booléen	N/A
	is_community_page	Booléen	N/A
	is_eligible_for_branded_content	Booléen	N/A
	is_messenger_bot_get_started_enabled	Booléen	N/A
	is_messenger_platform_bot	Booléen	N/A
	is_owned	Booléen	N/A
	is_permanently_closed	Booléen	N/A
	is_published	Booléen	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	Name	String	N/A
	Tâches	List	N/A
	is_unclaimed	Booléen	N/A
	is_webhooks_subscribed	Booléen	N/A
	leadgen_tos_acceptance_time	DateTime	N/A
	leadgen_tos_accepted	Booléen	N/A
	leadgen_tos_accepting_user	String	N/A
	leadgen_tos_accepting_user	Struct	N/A
	Lien	Lien	N/A
	Location	Struct	N/A
	Members	String	N/A
	merchant_review_status	String	N/A
	messenger_ads_default_icebreakers	List	N/A
	messenger_ads_default_page_welcome_message	Struct	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	messenger_ads_default_quick_replies	List	N/A
	messenger_ads_quick_replies_type	String	N/A
	Mission	String	N/A
	Mpg	String	N/A
	name_with_location_descriptor	String	N/A
	Réseau	String	N/A
	new_like_count	Entier	N/A
	offer_eligible	Booléen	N/A
	overall_star_rating	Float	N/A
	page_token	String	N/A
	parent_page	String	N/A
	Parking	String	N/A
	payment_options	Struct	N/A
	personal_info	String	N/A
	personal_interests	String	N/A
	pharma_safety_info	String	N/A
	Téléphone	String	N/A
	pickup_options	List	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	place_type	String	N/A
	plot_outline	String	N/A
	press_contact	String	N/A
	price_range	String	N/A
	privacy_info_url	String	N/A
	produced_by	String	N/A
	Produits	String	N/A
	promotion_eligible	Booléen	N/A
	promotion_ineligible_reason	String	N/A
	public_transit	String	N/A
	rating_count	Entier	N/A
	record_label	String	N/A
	release_date	String	N/A
	restaurant_services	Struct	N/A
	restaurant_specialties	Struct	N/A
	Planning	String	N/A
	screenplay_by	String	N/A
	Saison	String	N/A
	single_line_address	String	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	Starring	String	N/A
	start_info	Struct	N/A
	store_code	String	N/A
	store_location_descriptor	String	N/A
	store_number	Entier	N/A
	Studio	String	N/A
	supports_donate_button_in_live_video	Booléen	N/A
	talking_about_count	Entier	N/A
	temporary_status	String	N/A
	unread_message_count	Entier	N/A
	unread_notif_count	Entier	N/A
	unseen_message_count	Entier	N/A
	Nom d'utilisateur	String	N/A
	verification_status	String	N/A
	voip_info	Struct	N/A
	Site Web	String	N/A
	were_here_count	Entier	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	whatsapp_number	String	N/A
written_by	String	N/A	
Feeds	Id	String	N/A
	Actions	List	N/A
	admin_creator	Objet	N/A
	Application	Objet	N/A
	Pièces jointes	Objets	N/A
	backdated_time	DateTime	N/A
	call_to_action	Objet	N/A
	can_reply_privately	Booléen	N/A
	child_attachments	List	N/A
	Coordinates	Struct	N/A
	created_time	DateTime	N/A
	Événement	Struct	N/A
	expanded_height	Entier	N/A
	expanded_width	Entier	N/A
	feed_targeting	Objet	N/A
	De	Objet	N/A
	full_picture	String	N/A
	Height	Entier	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	Icône	String	N/A
	instagram_eligibility	String	N/A
	is_eligible_for_promotion	Booléen	N/A
	is_expired	Booléen	N/A
	is_hidden	Booléen	N/A
	is_inline_created	Booléen	N/A
	is_instagram_eligible	Booléen	N/A
	is_popular	Booléen	N/A
	is_published	Booléen	N/A
	is_spherical	Booléen	N/A
	Message	String	N/A
	message_tags	List	N/A
	multi_share_end_card	Booléen	N/A
	multi_share_optimized	Booléen	N/A
	parent_id	String	N/A
	permalink_url	String	N/A
	Lieu	String	N/A
	Confidentialité	Objet	N/A
	promotable_id	String	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	promotion_status	String	N/A
	Propriétés	List	N/A
	scheduled_publish_time	Float	N/A
	Shares	Objet	N/A
	status_type	String	N/A
	Story	String	N/A
	story_tags	List	N/A
	Subscribed	Booléen	N/A
	Cible	Struct	N/A
	Ciblage	Objet	N/A
	À	Objet	N/A
	timeline_visibility	String	N/A
	updated_time	DateTime	N/A
	Via	Struct	N/A
	video_buying_eligibility	List	N/A
	Width	Entier	N/A
	Since	DateTime	EQUAL_TO

Requêtes de partitionnement

Partitionnement basé sur des filtres :

Vous pouvez indiquer les options Spark supplémentaires `PARTITION_FIELD`, `LOWER_BOUND`, `UPPER_BOUND` et `NUM_PARTITIONS` si vous souhaitez utiliser la simultanéité dans Spark. Avec ces paramètres, la requête d'origine serait divisée en `NUM_PARTITIONS` nombres de sous-requêtes pouvant être exécutées simultanément par les tâches Spark.

- `PARTITION_FIELD` : le nom du champ à utiliser pour partitionner la requête.
- `LOWER_BOUND` : une valeur limite inférieure inclusive du champ de partition choisi.

  Pour le champ Datetime, nous acceptons le format d'horodatage Spark utilisé dans les requêtes SQL Spark.

  Exemples de valeurs valides :

  ```
  "2024-09-30T01:01:01.000Z"
  ```

- `UPPER_BOUND` : une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS` : le nombre de partitions.

Exemple :

```
facebookPageInsights_read = glueContext.create_dynamic_frame.from_options(
     connection_type="facebookpageinsights",
     connection_options={
         "connectionName": "connectionName",
         "ENTITY_NAME": "entityName",
         "API_VERSION": "v21",
         "PARTITION_FIELD": "created_Time"
         "LOWER_BOUND": "2024-10-27T07:00:00+0000"
         "UPPER_BOUND": "2024-10-27T07:00:00+0000"
         "NUM_PARTITIONS": "10"
     }
```

Options de connexion Facebook Page Insights

Les options de connexion pour Facebook Page Insights sont les suivantes :

- `ENTITY_NAME`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Le nom de votre objet dans Facebook Page Insights.
- `API_VERSION`(Chaîne) - (Obligatoire) Utilisé pour la lecture. La version de l'API Rest Facebook Page Insights que vous souhaitez utiliser.
- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.
- `PARTITION_FIELD`(Chaîne) - Utilisé pour la lecture. Champ à utiliser pour partitionner la requête.
- `LOWER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite inférieure inclusive du champ de partition choisi.
- `UPPER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS`(Entier) - Par défaut : 1. Utilisé pour la lecture. Nombre de partitions pour la lecture.
- `INSTANCE_URL`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Une URL d'instance Facebook Page Insights valide.

Restrictions et remarques relatives au connecteur Facebook Page Insights

Voici les restrictions ou les remarques relatives au connecteur Facebook Page Insights :

- La plupart des métriques sont actualisées toutes les 24 heures.
- Seules les données d'information des deux dernières années sont disponibles.
- Seuls 90 jours d'informations peuvent être consultés à la fois lorsque vous utilisez les paramètres `since` et `until`.

Connexion à Freshdesk

Freshdesk est un logiciel de support client cloud qui est à la fois riche en fonctionnalités et facile à utiliser. Grâce aux multiples canaux d'assistance disponibles, notamment le chat en direct, le courrier électronique, le téléphone et les médias sociaux, vous pouvez aider les clients par le biais de leur

méthode de communication préférée. Si vous êtes un utilisateur de Freshdesk, vous pouvez vous connecter AWS Glue à votre compte Freshdesk. Vous pouvez utiliser Freshdesk comme source de données dans vos tâches ETL. Exécutez ces tâches pour transférer des données de Freshdesk vers AWS des services ou d'autres applications prises en charge.

Rubriques

- AWS Glue support pour Freshdesk
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration de Freshdesk
- Configuration de connexions Freshdesk
- Lecture à partir d'entités Freshdesk
- Options de connexion Freshdesk
- Restrictions et remarques relatives au connecteur Freshdesk

AWS Glue support pour Freshdesk

AWS Glue prend en charge Freshdesk comme suit :

Prise en charge en tant que source ?

Oui : synchrone et asynchrone. Vous pouvez utiliser les tâches AWS Glue ETL pour interroger les données de Freshdesk.

Prise en charge en tant que cible ?

Non.

Versions de l'API Freshdesk prises en charge

Les versions d'API Freshdesk suivantes sont prises en charge :

- v2

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les autorisations AWS IAM requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Si vous ne voulez pas utiliser la méthode ci-dessus, vous pouvez également utiliser les politiques IAM gérées suivantes :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration de Freshdesk

Avant de pouvoir AWS Glue transférer des données depuis Freshdesk, vous devez satisfaire aux exigences suivantes :

Configuration requise

Les exigences minimales sont les suivantes :

- Un compte Freshdesk. Vous pouvez choisir entre les éditions Free, Growth, Pro ou Enterprise.
- Clé d'API d'un utilisateur de Freshdesk.

Configuration de connexions Freshdesk

Freshdesk prend en charge l'authentification personnalisée.

Pour consulter la documentation publique de Freshdesk sur la génération des clés d'API requises pour une authentification personnalisée, consultez Freshdesk authentication.

Voici les étapes à suivre pour configurer la connexion Freshdesk :

- Dans AWS Secrets Manager, créez un secret avec les informations suivantes :
  - Pour l'application connectée gérée par le client : le secret doit contenir la clé d'API de l'application connectée avec `apiKey` comme clé. Notez que vous devez créer un secret par connexion dans AWS Glue.
- Dans le AWS Glue Studio, créez une connexion sous Connexions de données en suivant les étapes ci-dessous :
  - Lorsque vous sélectionnez une source de données, sélectionnez Freshdesk.
  - Indiquez l'`INSTANCE_URL` de l'instance Freshdesk à laquelle vous souhaitez vous connecter.
  - Sélectionnez le rôle AWS IAM qui AWS Glue peut assumer les actions suivantes et qui dispose des autorisations nécessaires pour les actions suivantes :

    JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
```

```
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

- Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.
- Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.

- Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.
- Dans la configuration de votre AWS Glue tâche, fournissez `connectionName` une connexion réseau supplémentaire.

Lecture à partir d'entités Freshdesk

Prérequis

Un objet Freshdesk à partir duquel vous souhaitez lire. Vous aurez besoin du nom de l'objet.

Entités prises en charge pour la source de synchronisation :

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Agents	Oui	Oui	Non	Oui	Oui
Business Hours	Non	Oui	Non	Oui	Oui
Société	Oui	Oui	Non	Oui	Oui
Contacts	Oui	Oui	Non	Oui	Oui
Conversat ions	Non	Oui	Non	Oui	Non
Email Configs	Non	Oui	Non	Oui	Non

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Email Inboxes	Oui	Oui	Oui	Oui	Non
Forum Categories	Non	Oui	Non	Oui	Non
Forums	Non	Oui	Non	Oui	Non
Groupes	Non	Oui	Non	Oui	Non
Produits	Non	Oui	Non	Oui	Non
Roles	Non	Oui	Non	Oui	Non
Satisfaction Ratings	Oui	Oui	Non	Oui	Non
Skills	Non	Oui	Non	Oui	Non
Solutions	Oui	Oui	Non	Oui	Non
Enquêtes	Non	Oui	Non	Oui	Non
Tickets	Oui	Oui	Oui	Oui	Oui
Time Entries	Oui	Oui	Non	Oui	Non
Rubriques	Non	Oui	Non	Oui	Non
Topic Comments	Non	Oui	Non	Oui	Non

Entités prises en charge pour la source asynchrone :

Entité	Version de l'API	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Companies	v2	Non	Non	Non	Non	Non
Contacts	v2	Non	Non	Non	Non	Non

Exemple :

```
freshdesk_read = glueContext.create_dynamic_frame.from_options(
    connection_type="freshdesk",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "API_VERSION": "v2"
    }
```

Détails des entités et des champs Freshdesk :

Entité	Champ
Agents	https://developers.freshdesk.com/api/#list_all_agents
Business-hours	https://developers.freshdesk.com/api/#list_all_business_hours
Commentaires	https://developers.freshdesk.com/api/#comment_attributess
Société	https://developers.freshdesk.com/api/#companies
Contacts	https://developers.freshdesk.com/api/#list_all_contacts

Entité	Champ
Conversations	https://developers.freshdesk.com/api/#list_all_ticket_notes
Email-configs	https://developers.freshdesk.com/api/#list_all_email_configs
Email-inboxes	https://developers.freshdesk.com/api/#list_all_email_mailboxes
Forum-categories	https://developers.freshdesk.com/api/#category_attributes
Forums	https://developers.freshdesk.com/api/#forum_attributes
Groupes	https://developers.freshdesk.com/api/#list_all_groups
Produits	https://developers.freshdesk.com/api/#list_all_products
Roles	https://developers.freshdesk.com/api/#list_all_roles
Satisfaction-rating	https://developers.freshdesk.com/api/#view_all_satisfaction_ratingss
Skills	https://developers.freshdesk.com/api/#list_all_skills
Solutions	https://developers.freshdesk.com/api/#solution_content
Enquêtes	https://developers.freshdesk.com/api/#list_all_survey
Tickets	https://developers.freshdesk.com/api/#list_all_tickets

Entité	Champ
Time-entries	https://developers.freshdesk.com/api/#list_all_time_entries
Rubriques	https://developers.freshdesk.com/api/#topic_attributes

Requêtes de partitionnement

Partitionnement basé sur des filtres :

Vous pouvez indiquer les options Spark supplémentaires `PARTITION_FIELD`, `LOWER_BOUND`, `UPPER_BOUND` et `NUM_PARTITIONS` si vous souhaitez utiliser la simultanéité dans Spark. Avec ces paramètres, la requête d'origine serait divisée en `NUM_PARTITIONS` nombres de sous-requêtes pouvant être exécutées simultanément par les tâches Spark.

- `PARTITION_FIELD` : le nom du champ à utiliser pour partitionner la requête.
- `LOWER_BOUND` : une valeur limite inférieure inclusive du champ de partition choisi.

  Pour le champ Datetime, nous acceptons le format d'horodatage Spark utilisé dans les requêtes SQL Spark.

  Exemples de valeurs valides :

  ```
  "2024-09-30T01:01:01.000Z"
  ```

- `UPPER_BOUND` : une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS` : le nombre de partitions.

Exemple :

```
freshDesk_read = glueContext.create_dynamic_frame.from_options(
     connection_type="freshdesk",
     connection_options={
         "connectionName": "connectionName",
         "ENTITY_NAME": "entityName",
         "API_VERSION": "v2",
         "PARTITION_FIELD": "Created_Time"
```

```
        "LOWER_BOUND": " 2024-10-27T23:16:08Z“
        "UPPER_BOUND": " 2024-10-27T23:16:08Z"
        "NUM_PARTITIONS": "10"
    }
```

Options de connexion Freshdesk

Les options de connexion pour Freshdesk sont les suivantes :

- `ENTITY_NAME`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Le nom de votre objet dans Freshdesk.
- `API_VERSION`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Version d'API Rest Freshdesk que vous souhaitez utiliser.
- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.
- `PARTITION_FIELD`(Chaîne) - Utilisé pour la lecture. Champ à utiliser pour partitionner la requête.
- `LOWER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite inférieure inclusive du champ de partition choisi.
- `UPPER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS`(Entier) - Par défaut : 1. Utilisé pour la lecture. Nombre de partitions pour la lecture.
- `INSTANCE_URL`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Une URL d'instance Freshdesk valide.
- `TRANSFER_MODE`(Chaîne) - Utilisé pour indiquer si le type de traitement tel que `SYNC` ou `ASYNC` est défini sur `SYNC` par défaut. (Facultatif)

Restrictions et remarques relatives au connecteur Freshdesk

Voici les restrictions ou les remarques relatives au connecteur Freshdesk :

- Les entités `Company`, `Contacts` et `Tickets` avec filtrage ont des limites de pagination. Elles ne renvoient que 30 enregistrements par page et la valeur de la page peut être définie jusqu'à 10 (pour un maximum de 300 enregistrements).
- L'entité `Tickets` ne récupère pas les enregistrements datant de plus de 30 jours.
- Les entités `Company`, `Contacts` et `Tickets` prennent en charge le type de données « Date » dans le filtrage. Vous devez sélectionner les fréquences de déclenchement « Quotidiennes » et suivantes pour ces trois entités. Si vous sélectionnez « Minutes » ou « Horaire », vous risquez de dupliquer les données. De plus, lors de la sélection de ces champs pour le filtrage, seule la valeur de date doit être sélectionnée, car elle ne prendra en compte que la partie Date de l'horodatage sélectionné.
- Le nombre d'appels d'API par minute dépend de votre forfait. Cette limite est appliquée à l'ensemble du compte, indépendamment de facteurs tels que le nombre d'agents ou d'adresses IP utilisés pour passer les appels. Pour tous les utilisateurs de la version d'essai, la limite d'API par défaut est de 50 appels/minute. Pour plus d'informations, consultez Freshdesk.
- Pour chaque entité, un seul Export/Async Job est traité à la fois. Une nouvelle tâche ne sera traitée que lorsque la tâche existante aura abouti ou échoué. Pour plus d'informations, consultez Freshdesk.
- Les champs suivants sont pris en charge pour les appels d'API de synchronisation, mais ne doivent pas supported/allowed être transmis dans le corps de la demande d'API asynchrone.
  - id
  - created_at
  - updated_at
  - updated_since
  - actif
  - company_id
  - other_companies
  - avatar
  - view_all_tickets
  - deleted
  - other_emails
  - state
  - balise
  - tags

Connexion à Freshsales

Freshsales est un CRM intuitif qui aide les commerciaux à éliminer les incertitudes liées aux ventes. Grâce au téléphone, au courrier électronique, aux tâches, aux rendez-vous et aux notes intégrés, les commerciaux n'ont pas à passer d'un onglet à l'autre pour suivre les prospects. Vous pouvez mieux gérer vos transactions grâce à la vue du pipeline et faire en sorte qu'un plus grand nombre de transactions soient conclues. Si vous êtes un utilisateur de Freshsales, vous pouvez vous connecter AWS Glue à votre compte Freshsales. Vous pouvez utiliser Freshsales comme source de données dans vos tâches ETL. Exécutez ces tâches pour transférer les données de Freshsales vers des AWS services ou d'autres applications prises en charge.

Rubriques

- AWS Glue support pour Freshsales
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration de Freshsales
- Configuration de connexions Freshsales
- Lecture à partir d'entités Freshsales
- Options de connexion Freshsales
- Restrictions de Freshsales

AWS Glue support pour Freshsales

AWS Glue soutient Freshsales comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser les tâches AWS Glue ETL pour interroger les données de Freshsales.

Prise en charge en tant que cible ?

Non.

Versions de l'API Freshsales prises en charge

Les versions d'API Freshsales suivantes sont prises en charge :

- v1.0

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les autorisations AWS IAM requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Si vous ne voulez pas utiliser la méthode ci-dessus, vous pouvez également utiliser les politiques IAM gérées suivantes :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration de Freshsales

Avant de pouvoir AWS Glue transférer des données depuis Freshsales, vous devez répondre aux exigences suivantes :

Configuration requise

Les exigences minimales sont les suivantes :

- Vous disposez d'un compte Freshsales.
- Vous disposez d'une clé d'API utilisateur.

Si vous remplissez ces conditions, vous êtes prêt à vous connecter AWS Glue à votre compte Freshsales. Pour les connexions classiques, vous n'avez rien d'autre à faire dans Freshsales.

Configuration de connexions Freshsales

Freshsales prend en charge l'authentification personnalisée.

Pour consulter la documentation publique de Freshsales sur la génération des clés d'API requises pour une authentification personnalisée, consultez Authentication.

Pour configurer une connexion Freshsales :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes :
   a. Pour l'application connectée gérée par le client, le secret doit contenir la clé d'API de l'application connectée avec *`apiSecretKey`* comme clé. Le secret doit également contenir une autre paire clé-valeur avec *`apiKey`* comme clé et *`token`* comme valeur.
   b. Remarque : vous devez créer un secret pour vos connexions dans AWS Glue.

1. Dans AWS Glue Glue Studio, créez une connexion sous Data Connections en suivant les étapes ci-dessous :
   a. Lorsque vous sélectionnez une source de données, sélectionnez Freshsales.
   b. Indiquez l'`INSTANCE_URL` du compte Freshsales auquel vous souhaitez vous connecter.
   c. Sélectionnez le rôle AWS IAM qui AWS Glue peut assumer et dispose des autorisations nécessaires pour les actions suivantes :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

d. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.

e. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.

2. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.

Lecture à partir d'entités Freshsales

Prérequis

Un objet Freshsales à partir duquel vous souhaitez lire. Vous aurez besoin du nom de l'objet.

Entités prises en charge pour la source :

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Comptes	Oui	Oui	Oui	Oui	Oui
Contacts	Oui	Oui	Oui	Oui	Oui

Exemple :

```
freshSales_read = glueContext.create_dynamic_frame.from_options(
     connection_type="freshsales",
     connection_options={
         "connectionName": "connectionName",
         "ENTITY_NAME": "entityName",
         "API_VERSION": "v1.0"
     }
```

Détails des entités et des champs Freshsales :

Freshsales fournit des points de terminaison permettant de récupérer les métadonnées de manière dynamique pour les entités prises en charge. Par conséquent, la prise en charge des opérateurs est capturée au niveau du type de données.

Entité	Type de données	Opérateurs pris en charge
Entités Freshsale (toutes)	Entier	!=,=,<,<=,>,>=,BETWEEN
	String	Like, =, !=
	BigInteger	!=,=,<,<=,>,>=,BETWEEN
	Booléen	=
	Double	!=,=,<,<=,>,>=,BETWEEN
	BigDecimal	!=,=,<,<=,>,>=,BETWEEN
	Date	!=,=,<,<=,>,>=,BETWEEN
	DateTime	!=,=,<,<=,>,>=,BETWEEN
	Struct	N/A
	List	N/A

Requêtes de partitionnement

Partitionnement basé sur des filtres :

Vous pouvez indiquer les options Spark supplémentaires PARTITION_FIELD, LOWER_BOUND, UPPER_BOUND et NUM_PARTITIONS si vous souhaitez utiliser la simultanéité dans Spark. Avec ces paramètres, la requête d'origine serait divisée en NUM_PARTITIONS nombres de sous-requêtes pouvant être exécutées simultanément par les tâches Spark.

- PARTITION_FIELD : le nom du champ à utiliser pour partitionner la requête.
- LOWER_BOUND : une valeur limite inférieure inclusive du champ de partition choisi.

  Pour le champ Datetime, nous acceptons la valeur au format ISO.

  Exemples de valeurs valides :

  ```
  "2024-09-30T01:01:01.000Z"
  ```

- UPPER_BOUND : une valeur limite supérieure exclusive du champ de partition choisi.
- NUM_PARTITIONS : le nombre de partitions.

Exemple :

```
freshSales_read = glueContext.create_dynamic_frame.from_options(
     connection_type="freshsales",
     connection_options={
         "connectionName": "connectionName",
         "ENTITY_NAME": "entityName",
         "API_VERSION": "v1",
         "PARTITION_FIELD": "Created_Time"
         "LOWER_BOUND": " 2024-10-15T21:16:25Z"
         "UPPER_BOUND": " 2024-10-20T21:25:50Z"
         "NUM_PARTITIONS": "10"
     }
```

Options de connexion Freshsales

Les options de connexion pour Freshsales sont les suivantes :

- ENTITY_NAME(Chaîne) - (Obligatoire) Utilisé pour la lecture. Le nom de votre objet dans Freshsales.
- API_VERSION(Chaîne) - (Obligatoire) Utilisé pour la lecture. Version d'API Rest Freshsales que vous souhaitez utiliser.

- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.
- `PARTITION_FIELD`(Chaîne) - Utilisé pour la lecture. Champ à utiliser pour partitionner la requête.
- `LOWER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite inférieure inclusive du champ de partition choisi.
- `UPPER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS`(Entier) - Par défaut : 1. Utilisé pour la lecture. Nombre de partitions pour la lecture.
- `INSTANCE_URL`(Chaîne) - Utilisé pour la lecture. Une URL d'instance Freshsales valide.

Restrictions de Freshsales

Les restrictions ou remarques suivantes concernent Freshsales :

- Dans Freshsales, la limite de débit d'API est de 1 000 demandes d'API par heure et par compte (consultez Errors). Cependant, cette limite est extensible avec l'abonnement Enterprise (consultez le comparatif des forfaits).

Connexion à Google Ads

L'API Google Ads est l'interface programmatique de Google Ads, utilisée pour gérer des comptes et des campagnes Google Ads volumineux ou complexes. Si vous utilisez Google Ads, vous pouvez vous connecter AWS Glue à votre compte Google Ads. Vous pouvez ensuite utiliser Google Ads comme source de données dans vos tâches ETL. Exécutez ces tâches pour transférer des données entre Google Ads et AWS des services ou d'autres applications compatibles.

Rubriques

- AWS Glue support pour Google Ads
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration de Google Ads

- Configuration des connexions Google Ads
- Lecture à partir d'entités Google Ads
- Options de connexion Google Ads
- Création d'un compte Google Ads
- Limitations

AWS Glue support pour Google Ads

AWS Glue prend en charge Google Ads comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser les tâches AWS Glue ETL pour interroger les données de Google Ads.

Prise en charge en tant que cible ?

Non.

Versions de l'API Google Ads prises en charge

v18

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les AWS autorisations requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
```

```
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Vous pouvez également utiliser les politiques IAM gérées suivantes pour autoriser l'accès :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration de Google Ads

Avant de pouvoir AWS Glue effectuer un transfert depuis Google Ads, vous devez remplir les conditions suivantes :

Configuration requise

- Vous disposez d'un compte Google Ads avec e-mail et mot de passe. Pour plus d'informations sur la création d'un compte, consultez Creating a Google Ads account.
- Votre compte Google Ads est activé pour accéder à l'API. Toute utilisation de l'API Google Ads est disponible sans frais supplémentaires.
- Votre compte Google Ads vous permet d'installer des applications connectées. Si vous n'avez pas accès à cette fonctionnalité, contactez votre administrateur Google Ads.

Si vous remplissez ces conditions, vous êtes prêt à vous connecter AWS Glue à votre compte Google Ads.

Configuration des connexions Google Ads

Google Ads prend en charge les types de `AUTHORIZATION_CODE` subventions pour OAuth2.

Ce type de subvention est considéré comme « à trois étapes » OAuth car il repose sur la redirection des utilisateurs vers le serveur d'autorisation tiers pour authentifier l'utilisateur. Il est utilisé lors de la création de connexions via la AWS Glue console. La AWS Glue console redirige l'utilisateur vers Google Ads où il doit se connecter et autoriser AWS Glue les autorisations demandées pour accéder à son instance Google Ads.

Les utilisateurs peuvent choisir de créer leur propre application connectée dans Google Ads et de fournir leur propre identifiant client et leur secret client lors de la création de connexions via la AWS Glue console. Dans ce scénario, ils seront toujours redirigés vers Google Ads pour se connecter et autoriser AWS Glue l'accès à leurs ressources.

Ce type d'autorisation donne lieu à un jeton d'actualisation et à un jeton d'accès. Le jeton d'accès est de courte durée et peut être actualisé automatiquement sans interaction de l'utilisateur à l'aide du jeton d'actualisation.

Pour plus d'informations, consultez la documentation publique de Google Ads sur la création d'une application connectée pour le OAuth flux de code d'autorisation.

Pour configurer une connexion Google Ads :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes. Il est nécessaire de créer un secret pour chaque connexion dans AWS Glue.
   a. Pour le type de AuthorizationCode subvention :
      - Pour l'application connectée gérée par le client : le secret doit contenir le secret du consommateur de l'application connectée avec `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` comme clé.
2. Dans AWS Glue Glue Studio, créez une connexion sous Data Connections en suivant les étapes ci-dessous :
   a. Lorsque vous sélectionnez un Type de connexion, sélectionnez Google Ads.
   b. Indiquez l'`developer token` de Google Ads à laquelle vous souhaitez vous connecter.
   c. Indiquez le `MANAGER ID` de Google Ads si vous souhaitez vous connecter en tant que responsable.
   d. Sélectionnez le rôle IAM qui AWS Glue peut assumer les actions suivantes et qui dispose des autorisations nécessaires pour les actions suivantes :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

e. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.

f. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.

3. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.

Lecture à partir d'entités Google Ads

Conditions préalables

- Un objet Google Ads à partir duquel vous souhaitez lire. Reportez-vous au tableau des entités prises en charge ci-dessous pour vérifier les entités disponibles.

Entités prises en charge

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Ad Group Ad	Oui	Oui	Oui	Non	Oui

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Ad Group	Oui	Oui	Oui	Non	Oui
Campaign Budget	Oui	Oui	Oui	Oui	Oui
Account Budget	Oui	Non	Oui	Oui	Non
Campaign	Oui	Oui	Oui	Oui	Oui
Compte	Oui	Non	Oui	Non	Non

Exemple

```
googleAds_read = glueContext.create_dynamic_frame.from_options(
    connection_type="googleads",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "campaign-3467***",
        "API_VERSION": "v16"
    }
```

Détails des entités et des champs Google Ads

Entité	Champ	Type de données	Opérateurs pris en charge
Compte	resourceName	String	!=, =
Compte	callReportingEnabled	Booléen	!=, =
Compte	callConversionReporting Activé	Booléen	!=, =
Compte	callConversionAction	String	!=, =

Entité	Champ	Type de données	Opérateurs pris en charge
Compte	conversionTrackingId	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Compte	crossAccountConversionTrackingId	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Compte	payPerConversionEligibilityFailureReasons	List	
Compte	id	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Compte	currencyCode	String	!=, =, LIKE
Compte	timeZone	String	!=, =, LIKE
Compte	autoTaggingEnabled	Booléen	!=, =
Compte	hasPartnersBadge	Booléen	!=, =
Compte	manager	Booléen	!=, =
Compte	testAccount	Booléen	!=, =
Compte	date	Date	BETWEEN, =, <, >, <=, >=
Compte	costMicros	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Compte	acceptedCustomerDataTermes	Booléen	
Compte	conversionTrackingStatus	String	!=, =, LIKE

Entité	Champ	Type de données	Opérateurs pris en charge
Compte	enhancedConversionsForLeadsEnabled	Booléen	
Compte	googleAdsConversionClient	String	
Compte	status	String	!=, =
Compte	allConversionsByConversionDate	Double	!=, =, <, >
Compte	allConversionsValueByConversionDate	Double	!=, =, <, >
Compte	conversionsByConversionDate	Double	!=, =, <, >
Compte	conversionsValueByConversionDate	Double	!=, =, <, >
Compte	valuePerAllConversionsByConversionDate	Double	!=, =, <, >
Compte	videoViews	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Compte	clicks	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Compte	invalidClicks	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Compte	costPerAllConversions	Double	!=, =, <, >
Compte	costPerConversion	Double	!=, =, <, >

Entité	Champ	Type de données	Opérateurs pris en charge
Compte	conversions	Double	!=, =, <, >
Compte	absoluteTopImpressionPourcentage	Double	!=, =, <, >
Compte	impressions	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Compte	topImpressionPercentage	Double	!=, =, <, >
Compte	averageCpc	Double	!=, =, <, >
Compte	activeViewMeasurableCostMicros	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Compte	averageCost	Double	!=, =, <, >
Compte	ctr	Double	!=, =, <, >
Compte	activeViewCtr	Double	!=, =, <, >
Compte	searchImpressionShare	Double	!=, =, <, >
Compte	conversionAction	String	!=, =
Compte	conversionActionCategory	String	!=, =
Compte	conversionActionName	String	!=, =, LIKE
Account Budget	resourceName	String	!=, =
Account Budget	status	String	!=, =

Entité	Champ	Type de données	Opérateurs pris en charge
Account Budget	proposedEndTimeType	String	!=, =
Account Budget	approvedEndTimeType	String	!=, =
Account Budget	id	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Account Budget	billingSetup	String	!=, =
Account Budget	name	String	!=, =, LIKE
Account Budget	approvedStartDateHeure	DateTime	BETWEEN, =, <, >, <=, >=
Account Budget	proposedSpendingLimitMicros	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Account Budget	approvedSpendingLimitMicros	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Account Budget	adjustedSpendingLimitMicros	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Account Budget	amountServedMicros	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Ad Group	resourceName	String	!=, =, LIKE
Ad Group	status	String	!=, =, LIKE
Ad Group	type	String	!=, =, LIKE
Ad Group	id	BigInteger	BETWEEN, =, !=, <, >, <=, >=

Entité	Champ	Type de données	Opérateurs pris en charge
Ad Group	name	String	!=, =, LIKE
Ad Group	campaign	String	!=, =
Ad Group	cpcBidMicros	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Ad Group	targetCpaMicros	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Ad Group	cpmBidMicros	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Ad Group	cpvBidMicros	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Ad Group	targetCpmMicros	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Ad Group	effectiveTargetCpaMicros	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Ad Group	date	Date	BETWEEN, =, <, >, <=, >=
Ad Group	costMicros	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Ad Group	useAudienceGrouped	Booléen	!=, =
Ad Group	effectiveCpcBidMicros	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Ad Group	allConversionsByConversionDate	Double	!=, =, <, >

Entité	Champ	Type de données	Opérateurs pris en charge
Ad Group	allConversionsValueByConversionDate	Double	!=, =, <, >
Ad Group	conversionsByConversionDate	Double	!=, =, <, >
Ad Group	conversionsValueByConversionDate	Double	!=, =, <, >
Ad Group	valuePerAllConversionsByConversionDate	Double	!=, =, <, >
Ad Group	valuePerConversionsByConversionDate	Double	!=, =, <, >
Ad Group	averageCost	Double	!=, =, <, >
Ad Group	costPerAllConversions	Double	!=, =, <, >
Ad Group	costPerConversion	Double	!=, =, <, >
Ad Group	averagePageViews	Double	!=, =, <, >
Ad Group	videoViews	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Ad Group	clicks	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Ad Group	allConversions	Double	!=, =, <, >
Ad Group	averageCpc	Double	!=, =, <, >
Ad Group	absoluteTopImpressionPourcentage	Double	!=, =, <, >

Entité	Champ	Type de données	Opérateurs pris en charge
Ad Group	impressions	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Ad Group	topImpressionPercentage	Double	!=, =, <, >
Ad Group	activeViewCtr	Double	!=, =, <, >
Ad Group	ctr	Double	!=, =, <, >
Ad Group	searchTopImpressionPartagez	Double	!=, =, <, >
Ad Group	searchImpressionShare	Double	!=, =, <, >
Ad Group	searchAbsoluteTopImpressionShare	Double	!=, =, <, >
Ad Group	relativeCtr	Double	!=, =, <, >
Ad Group	conversionAction	String	!=, =
Ad Group	conversionActionCategory	String	!=, =
Ad Group	conversionActionName	String	!=, =, LIKE
Ad Group	updateMask	String	
Ad Group	créer	Struct	
Ad Group	mise à jour	Struct	
Ad Group	primaryStatus	String	!=, =

Entité	Champ	Type de données	Opérateurs pris en charge
Ad Group	primaryStatusReasons	List	
Ad Group Ad	resourceName	String	!=, =
Ad Group Ad	id	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Ad Group Ad	status	String	!=, =
Ad Group Ad	labels	List	
Ad Group Ad	adGroup	String	!=, =
Ad Group Ad	costMicros	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Ad Group Ad	approvalStatus	String	!=, =
Ad Group Ad	reviewStatus	String	!=, =
Ad Group Ad	adStrength	String	!=, =
Ad Group Ad	type	String	!=, =
Ad Group Ad	businessName	String	!=, =, LIKE
Ad Group Ad	date	Date	BETWEEN, =, <, >, <=, >=
Ad Group Ad	allConversionsByConversionDate	Double	!=, =, <, >
Ad Group Ad	allConversionsValueByConversionDate	Double	!=, =, <, >
Ad Group Ad	conversionsByConversionDate	Double	!=, =, <, >

Entité	Champ	Type de données	Opérateurs pris en charge
Ad Group Ad	conversionsValueByConversionDate	Double	!=, =, <, >
Ad Group Ad	valuePerAllConversionsByConversionDate	Double	!=, =, <, >
Ad Group Ad	valuePerConversionsByConversionDate	Double	!=, =, <, >
Ad Group Ad	activeViewMeasurableCostMicros	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Ad Group Ad	averageCost	Double	!=, =, <, >
Ad Group Ad	costPerAllConversions	Double	!=, =, <, >
Ad Group Ad	costPerConversion	Double	!=, =, <, >
Ad Group Ad	clicks	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Ad Group Ad	averagePageViews	Double	!=, =, <, >
Ad Group Ad	videoViews	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Ad Group Ad	allConversions	Double	!=, =, <, >
Ad Group Ad	averageCpc	Double	!=, =, <, >
Ad Group Ad	topImpressionPercentage	Double	!=, =, <, >
Ad Group Ad	impressions	BigInteger	BETWEEN, =, !=, <, >, <=, >=

Entité	Champ	Type de données	Opérateurs pris en charge
Ad Group Ad	absoluteTopImpressionPourcentage	Double	!=, =, <, >
Ad Group Ad	activeViewCtr	Double	!=, =, <, >
Ad Group Ad	ctr	Double	!=, =, <, >
Ad Group Ad	conversionAction	String	!=, =
Ad Group Ad	conversionActionCategory	String	!=, =
Ad Group Ad	conversionActionName	String	!=, =, LIKE
Ad Group Ad	updateMask	String	
Ad Group Ad	créer	Struct	
Ad Group Ad	mise à jour	Struct	
Ad Group Ad	policyValidationParameter	Struct	
Ad Group Ad	primaryStatus	String	!=, =
Ad Group Ad	primaryStatusReasons	List	
Campaign	resourceName	String	!=, =
Campaign	status	String	!=, =
Campaign	baseCampaign	String	!=, =
Campaign	name	String	!=, =, LIKE

Entité	Champ	Type de données	Opérateurs pris en charge
Campaign	id	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Campaign	campaignBudget	String	!=, =, LIKE
Campaign	startDate	Date	BETWEEN, =, <, >, <=, >=
Campaign	endDate	Date	BETWEEN, =, <, >, <=, >=
Campaign	adServingOptimizationÉtat	String	!=, =
Campaign	advertisingChannelType	String	!=, =
Campaign	advertisingChannelSubType	String	!=, =
Campaign	experimentType	String	!=, =
Campaign	servingStatus	String	!=, =
Campaign	biddingStrategyType	String	!=, =
Campaign	domainName	String	!=, =, LIKE
Campaign	languageCode	String	!=, =, LIKE
Campaign	useSuppliedUrlsUniquement	Booléen	!=, =
Campaign	positiveGeoTargetType	String	!=, =

Entité	Champ	Type de données	Opérateurs pris en charge
Campaign	negativeGeoTargetType	String	!=, =
Campaign	paymentMode	String	!=, =
Campaign	optimizationGoalTypes	List	
Campaign	date	Date	BETWEEN, =, <, >, <=, >=
Campaign	averageCost	Double	
Campaign	clicks	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Campaign	costMicros	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Campaign	impressions	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Campaign	useAudienceGrouped	Booléen	!=, =
Campaign	activeViewMeasurableCostMicros	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Campaign	costPerAllConversions	Double	!=, =, <, >
Campaign	costPerConversion	Double	!=, =, <, >
Campaign	invalidClicks	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Campaign	publisherPurchasedClicks	BigInteger	BETWEEN, =, !=, <, >, <=, >=

Entité	Champ	Type de données	Opérateurs pris en charge
Campaign	averagePageViews	Double	!=, =, <, >
Campaign	videoViews	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Campaign	allConversionsByConversionDate	Double	!=, =, <, >
Campaign	allConversionsValueByConversionDate	Double	!=, =, <, >
Campaign	conversionsByConversionDate	Double	!=, =, <, >
Campaign	conversionsValueByConversionDate	Double	!=, =, <, >
Campaign	valuePerAllConversionsByConversionDate	Double	!=, =, <, >
Campaign	valuePerConversionsByConversionDate	Double	!=, =, <, >
Campaign	allConversions	Double	!=, =, <, >
Campaign	absoluteTopImpressionPourcentage	Double	!=, =, <, >
Campaign	searchAbsoluteTopImpressionShare	Double	!=, =, <, >
Campaign	averageCpc	Double	!=, =, <, >
Campaign	searchImpressionShare	Double	!=, =, <, >

Entité	Champ	Type de données	Opérateurs pris en charge
Campaign	searchTopImpressionPartagez	Double	!=, =, <, >
Campaign	activeViewCtr	Double	!=, =, <, >
Campaign	ctr	Double	!=, =, <, >
Campaign	relativeCtr	Double	!=, =, <, >
Campaign	updateMask	String	
Campaign	créer	Struct	
Campaign	mise à jour	Struct	
Campaign Budget	resourceName	String	!=, =
Campaign Budget	id	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Campaign Budget	status	String	!=, =
Campaign Budget	deliveryMethod	String	!=, =
Campaign Budget	point	String	!=, =
Campaign Budget	type	String	!=, =
Campaign Budget	name	String	!=, =, LIKE
Campaign Budget	amountMicros	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Campaign Budget	explicitlyShared	Booléen	!=, =
Campaign Budget	referenceCount	BigInteger	BETWEEN, =, !=, <, >, <=, >=

Entité	Champ	Type de données	Opérateurs pris en charge
Campaign Budget	hasRecommendedBudget	Booléen	!=, =
Campaign Budget	date	Date	BETWEEN, =, <, >, <=, >=
Campaign Budget	costMicros	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Campaign Budget	startDate	Date	BETWEEN, =, <, >, <=, >=
Campaign Budget	endDate	Date	BETWEEN, =, <, >, <=, >=
Campaign Budget	maximizeConversionValueTargetRoas	Double	!=, =, <, >
Campaign Budget	maximizeConversionsTargetCpaMicros	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Campaign Budget	selectiveOptimizationConversionActions	String	
Campaign Budget	averageCost	Double	!=, =, <, >
Campaign Budget	costPerAllConversions	Double	!=, =, <, >
Campaign Budget	costPerConversion	Double	!=, =, <, >
Campaign Budget	videoViews	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Campaign Budget	clicks	BigInteger	BETWEEN, =, !=, <, >, <=, >=

Entité	Champ	Type de données	Opérateurs pris en charge
Campaign Budget	allConversions	Double	!=, =, <, >
Campaign Budget	valuePerAllConversions	Double	!=, =, <, >
Campaign Budget	averageCpc	Double	!=, =, <, >
Campaign Budget	impressions	BigInteger	BETWEEN, =, !=, <, >, <=, >=
Campaign Budget	ctr	Double	!=, =, <, >
Campaign Budget	updateMask	String	
Campaign Budget	créer	Struct	
Campaign Budget	mise à jour	Struct	

Requêtes de partitionnement

Les options Spark supplémentaires `PARTITION_FIELD`, `LOWER_BOUND`, `UPPER_BOUND` et `NUM_PARTITIONS` peuvent être indiquées si vous souhaitez utiliser la simultanéité dans Spark. Avec ces paramètres, la requête d'origine serait divisée en `NUM_PARTITIONS` nombres de sous-requêtes pouvant être exécutées simultanément par les tâches Spark.

- `PARTITION_FIELD` : le nom du champ à utiliser pour partitionner la requête.
- `LOWER_BOUND` : une valeur limite inférieure inclusive du champ de partition choisi.

  Pour le champ Date, nous acceptons le format de date Spark utilisé dans les requêtes SQL Spark. Exemples de valeurs valides : `"2024-02-06"`.
- `UPPER_BOUND` : une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS` : nombre de partitions.

Les détails relatifs à la prise en charge des champs de partitionnement par entité sont présentés dans le tableau suivant.

Nom de l'entité	Champ de partitionnement	Type de données
Ad Group Ad	date	Date
Ad Group	date	Date
Campaign	date	Date
Campaign Budget	date	Date

Exemple

```
googleads_read = glueContext.create_dynamic_frame.from_options(
    connection_type="googleads",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "campaign-3467***",
        "API_VERSION": "v16",
        "PARTITION_FIELD": "date"
        "LOWER_BOUND": "2024-01-01"
        "UPPER_BOUND": "2024-06-05"
        "NUM_PARTITIONS": "10"
    }
)
```

Options de connexion Google Ads

Les options de connexion pour Google Ads sont les suivantes :

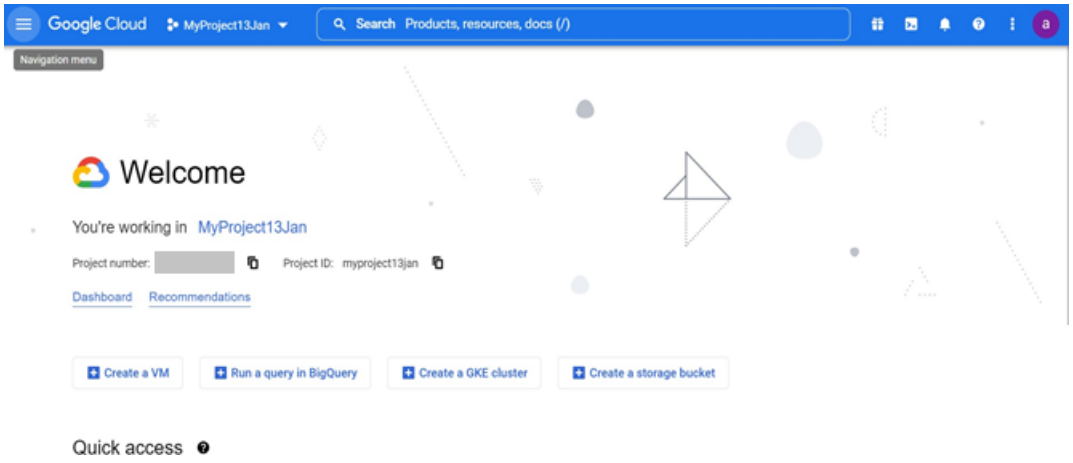
- `ENTITY_NAME`(Chaîne) - (Obligatoire) Utilisé pour la lecture/l'écriture. Le nom de votre objet dans Google Ads.
- `API_VERSION`(Chaîne) - (Obligatoire) Utilisé pour la lecture/l'écriture. Version d'API Rest Google Ads que vous souhaitez utiliser. Exemple : v16.
- `DEVELOPER_TOKEN`(Chaîne) - (Obligatoire) Utilisé pour la lecture/l'écriture. Nécessaire d'authentifier le développeur ou l'application qui envoie des demandes à l'API.
- `MANAGER_ID`(Chaîne) : utilisé pour la lecture/l'écriture. Identifiant unique qui vous permet de gérer plusieurs comptes Google Ads. Il s'agit de l'identifiant client du responsable autorisé. Si vous

accédez au compte client par le biais d'un compte de gestion, le `MANAGER_ID` est obligatoire. Pour de plus amples informations, veuillez consulter login-customer-id.

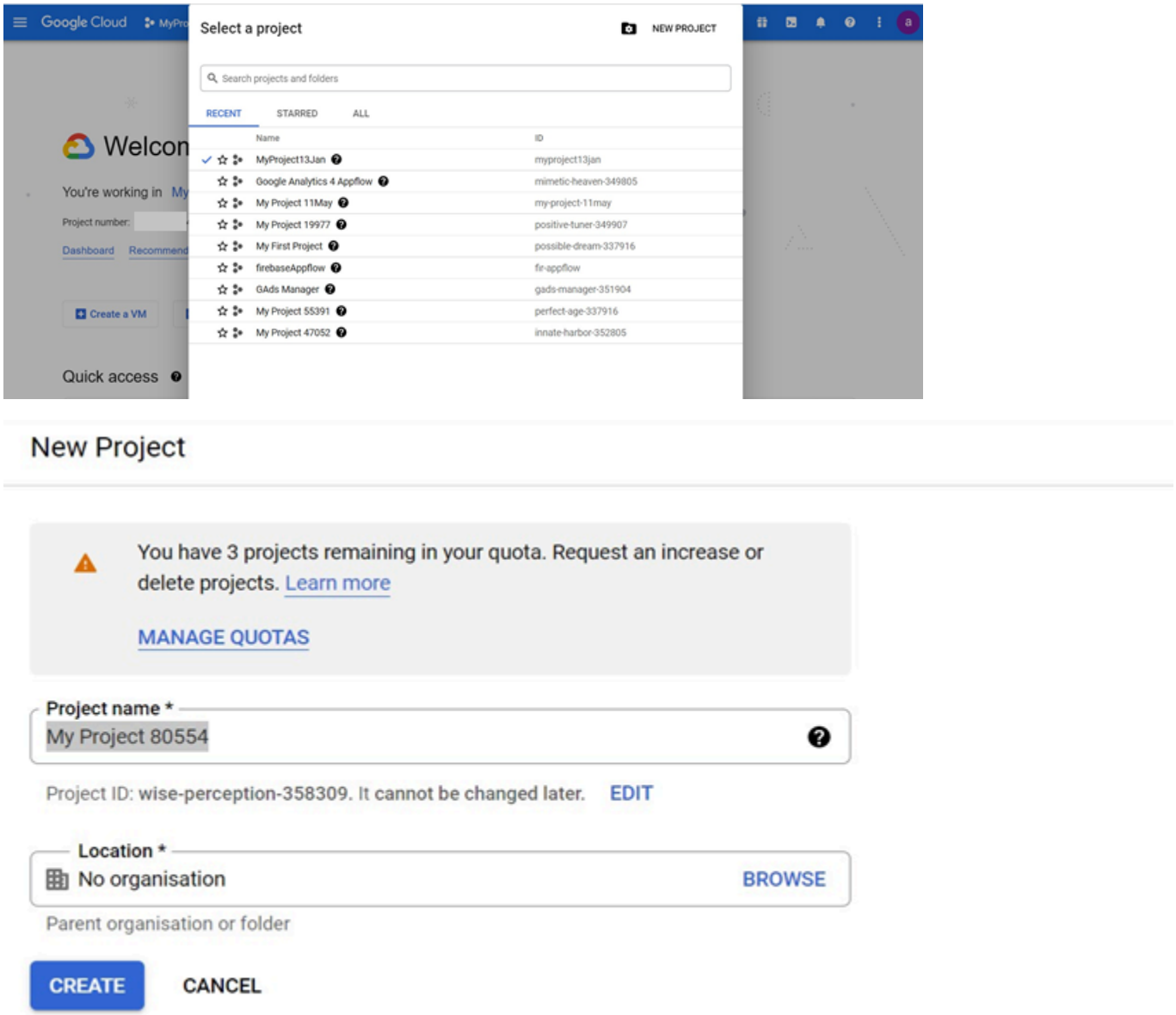
- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.
- `PARTITION_FIELD`(Chaîne) - Utilisé pour la lecture. Champ à utiliser pour partitionner la requête.
- `LOWER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite inférieure inclusive du champ de partition choisi.
- `UPPER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS`(Entier) - Par défaut : 1. Utilisé pour la lecture. Nombre de partitions pour la lecture.

Création d'un compte Google Ads

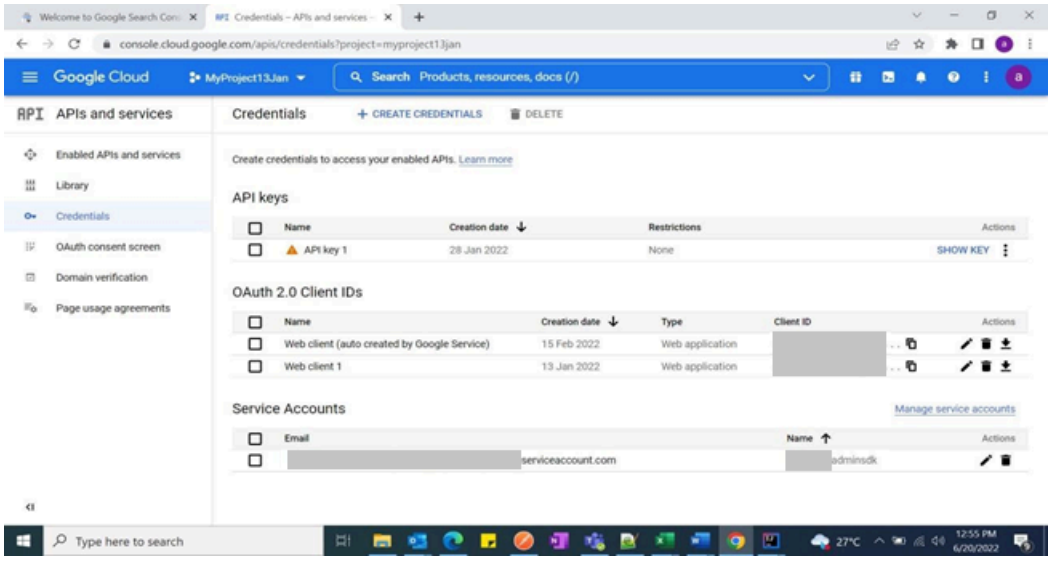
1. Connectez-vous au compte de développeur Google Ads à l'aide de vos informations d'identification, puis accédez à *MyProject.



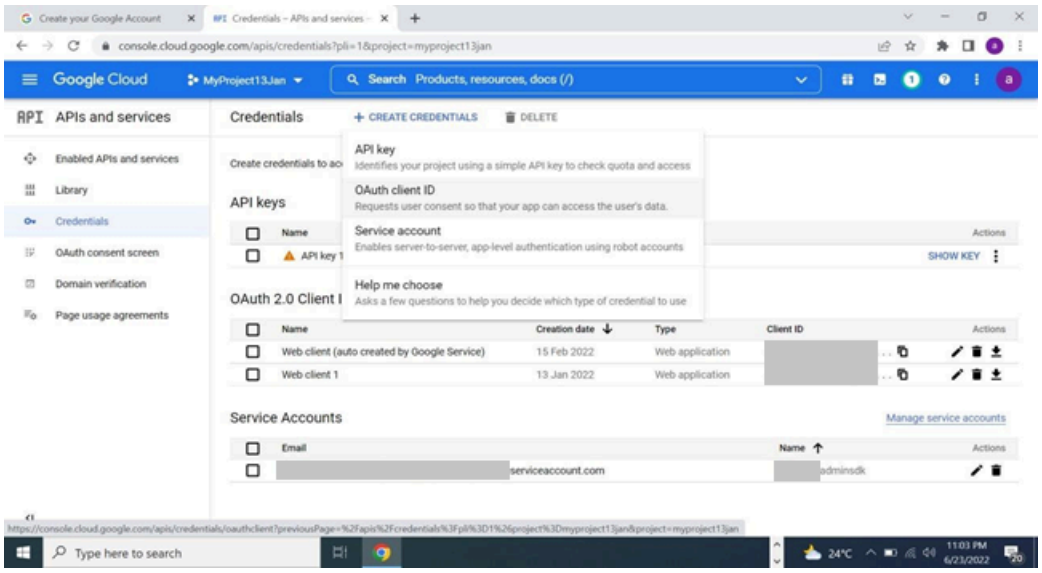
2. Choisissez Nouveau projet et fournissez les informations requises pour créer un projet Google si aucune application n'y est enregistrée.



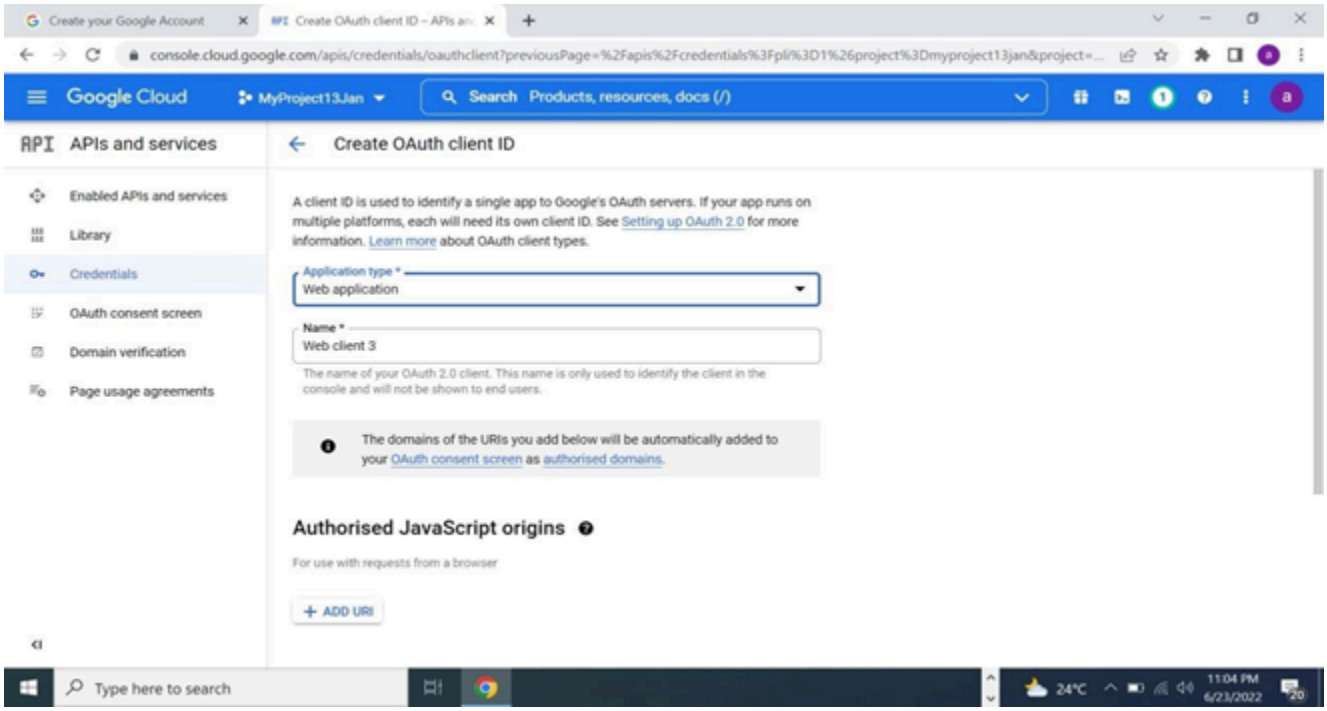
3. Choisissez l'onglet Navigation, puis API et paramètres, et Créer un identifiant client, ClientSecretqui nécessitera une configuration supplémentaire pour créer une connexion entre AWS Glue et GoogleAds. Pour en savoir plus, consultez Informations d'identification de l'API.



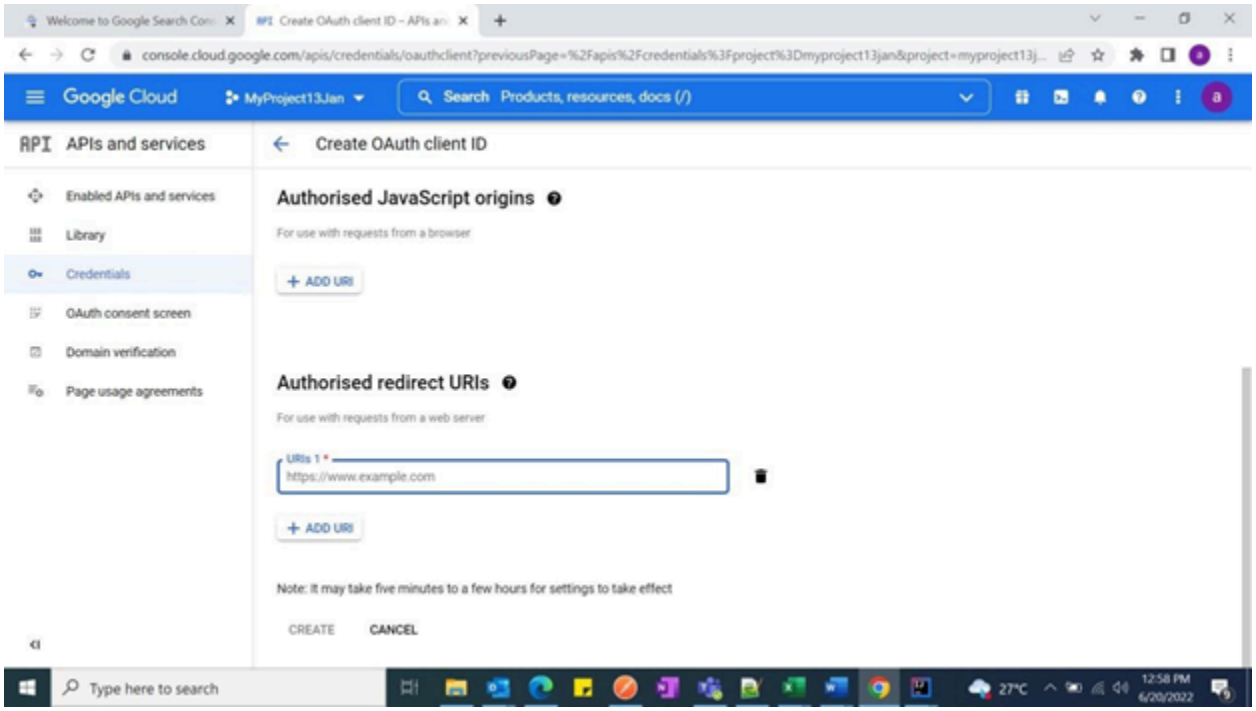
4. Choisissez CREATE CREDENTIALS et choisissez l'ID OAuth client.



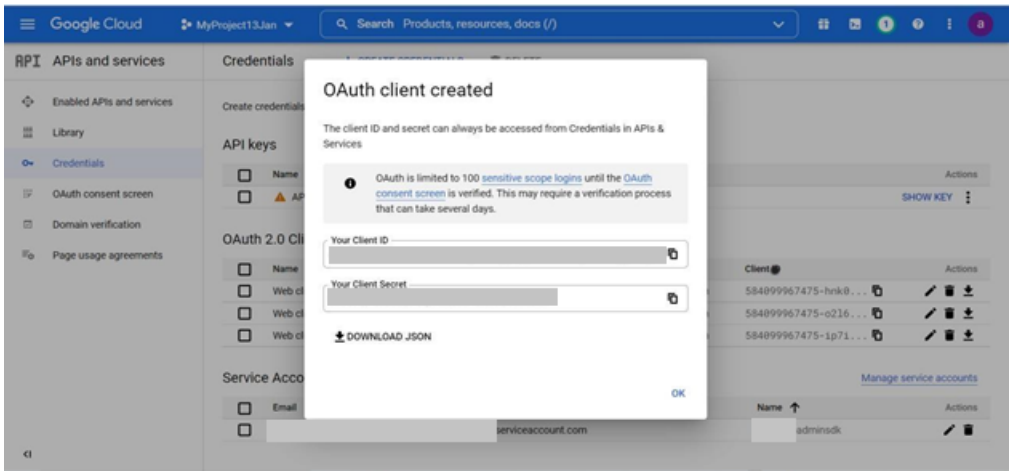
5. Sélectionnez Application Web comme Type d'application.



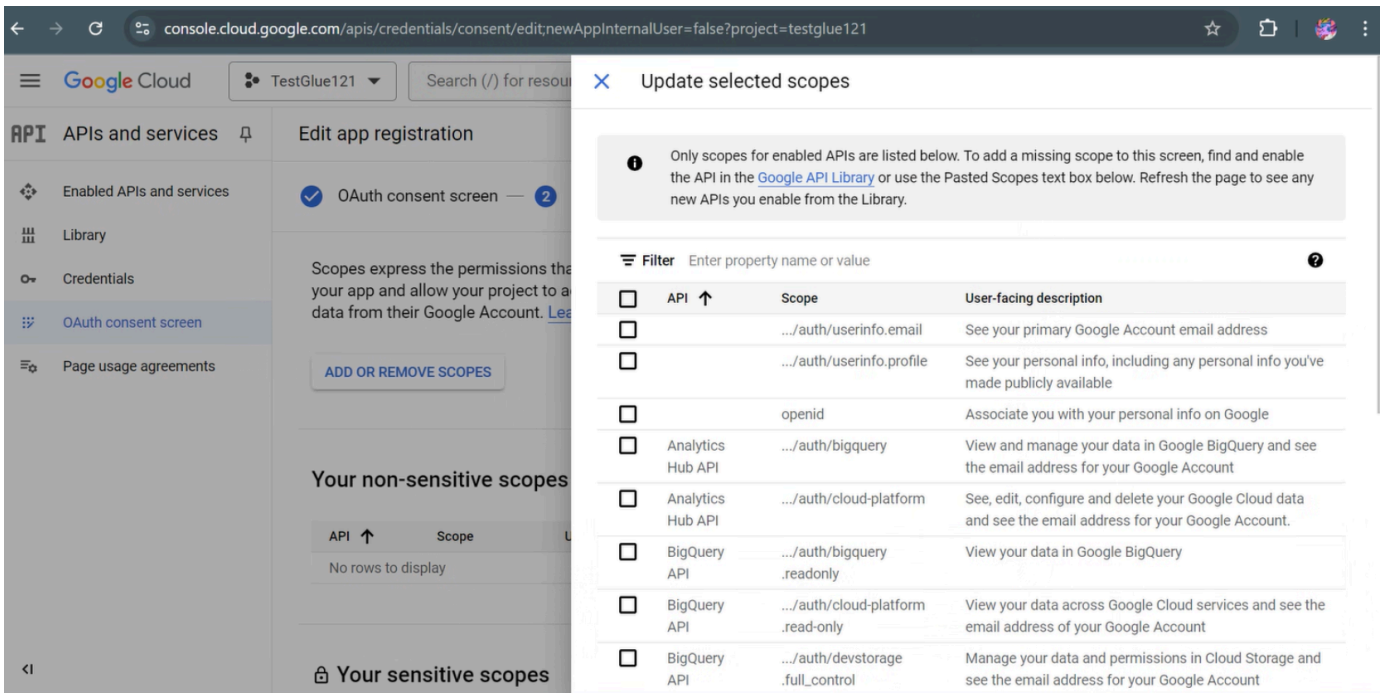
6. Sous Redirection autorisée URIs, ajoutez la OAuth redirection URIs et choisissez Créer. Vous pouvez ajouter plusieurs redirections URIs si nécessaire.



7. Votre identifiant client et votre secret client seront générés lors de la création d'une connexion entre AWS Glue et Google Ads.



8. Ajoutez les portées en fonction des besoins de votre application, choisissez l'écran de OAuth consentement, fournissez les informations requises et ajoutez les portées en fonction des exigences.



Limitations

Les restrictions du connecteur Google Ads sont les suivantes :

- `MANAGER_ID` est une entrée facultative lors de la création d'une connexion. Cependant, lorsque vous souhaitez accéder aux clients sous-jacents à un gestionnaire en particulier, `MANAGER_ID` est alors une entrée obligatoire. Le tableau ci-dessous explique les restrictions d'accès selon que `MANAGER_ID` est inclus ou non dans une connexion.

MANAGER_ID fourni lors de la création de la connexion ?	Clients accessibles
Oui	Les clients répertoriés sous le MANAGER_ID fourni avec le MANAGER_ID.
Non	Tous les clients seront répertoriés, mais les clients sous-jacents à un gestionnaire ne seront pas accessibles.

- Lorsqu'un compte de gestion est choisi comme objet, seul `Account` s'affichera comme sous-objet. Dans le connecteur Google Ads, les entités telles que les campagnes, les publicités, etc., sont extraites en fonction des comptes client individuels, et non du compte de gestion.
- Vous ne pouvez pas extraire les métriques pour le compte de gestion. Vous pouvez plutôt extraire des métriques pour des comptes client individuels.
- Chaque compte peut avoir jusqu'à 10 000 campagnes, y compris des campagnes actives et en suspens. Pour plus d'informations, consultez Campagne par compte.
- Lorsque vous créez un rapport, si vous choisissez d'afficher certaines métriques, les lignes dont les métriques sélectionnées sont toutes nulles ne seront pas renvoyées. Pour plus d'informations, consultez Zéro métrique.
- Avec les champs suivants, le flux de mappage complet ne fonctionnera pas pour les entités Account, Ad Group et Ad Group Ad Group Ad, en particulier pour ConversionAction,, conversionActionCategory. conversionActionName Pour plus d'informations, consultez Segment et métriques.
- Un filtre de plage de dates est obligatoire lorsque le champ `segments.date` est sélectionné.

Connexion à Google Analytics 4

Google Analytics 4 est un service d'analytique qui suit les métriques concernant les interactions des visiteurs avec vos applications et vos sites Web et en génère des rapports. Ces métriques incluent les pages vues, les utilisateurs actifs et les événements. Si vous êtes un utilisateur de Google Analytics 4, vous pouvez vous connecter AWS Glue à votre compte Google Analytics 4. Vous pouvez ensuite utiliser Google Analytics 4 comme source de données dans vos tâches ETL. Exécutez ces tâches pour transférer des données de Google Analytics 4 vers AWS des services ou d'autres applications compatibles.

Rubriques

- AWS Glue support pour Google Analytics 4
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration de Google Analytics 4
- Configuration des connexions Google Analytics 4
- Lecture à partir d'entités Google Analytics 4
- Options de connexion Google Analytics 4
- Création d'un compte Google Analytics 4
- Étapes pour créer une application cliente et des informations d'identification OAuth 2.0
- Limites et considérations

AWS Glue support pour Google Analytics 4

AWS Glue prend en charge Google Analytics 4 comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser les tâches AWS Glue ETL pour interroger les données de Google Analytics 4.

Prise en charge en tant que cible ?

Non.

Versions d'API Google Analytics 4 prises en charge

v1 Bêta.

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les AWS autorisations requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
```

```
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Vous pouvez également utiliser les politiques IAM gérées suivantes pour autoriser l'accès :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration de Google Analytics 4

Avant de pouvoir AWS Glue effectuer un transfert depuis Google Analytics 4, vous devez satisfaire aux exigences suivantes :

Configuration requise

- Vous disposez d'un compte Google Analytics avec un ou plusieurs flux de données qui collectent les données que vous souhaitez transférer.
- Vous disposez d'un compte Google Cloud Platform et d'un projet Google Cloud.
- Dans votre projet Google Cloud, vous avez activé les éléments suivants APIs :

  - API Google Analytics
  - API Google Analytics Admin
  - API Google Analytics Data
- Dans votre projet Google Cloud, vous avez configuré un écran de OAuth consentement pour les utilisateurs externes. Pour plus d'informations sur l'écran de OAuth consentement, consultez la section Configuration de votre écran de OAuth consentement dans l'aide de la console Google Cloud Platform.
- Dans votre projet Google Cloud, vous avez configuré un identifiant client OAuth 2.0. Pour plus d'informations, voir Configuration de la OAuth version 2.0.

Si vous répondez à ces exigences, vous êtes prêt à vous connecter AWS Glue à votre compte Google Analytics 4.

Configuration des connexions Google Analytics 4

Pour configurer une connexion à Google Sheet :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes. Il est nécessaire de créer un secret pour chaque connexion dans AWS Glue.
   a. Pour le type de AuthorizationCode subvention :
      - Pour l'application connectée gérée par le client : le secret doit contenir le secret du consommateur de l'application connectée avec `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` comme clé.
2. Dans AWS Glue Glue Studio, créez une connexion sous Data Connections en suivant les étapes ci-dessous :
   a. Lorsque vous sélectionnez un Type de connexion, sélectionnez Google Analytics 4.
   b. Indiquez l'`INSTANCE_URL` de Google Analytics 4 à laquelle vous souhaitez vous connecter.
   c. Sélectionnez le rôle IAM qui AWS Glue peut assumer et dispose des autorisations nécessaires pour les actions suivantes :

      JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
```

```
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

d. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.

e. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.

3. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.

Type d'autorisation `AUTHORIZATION_CODE`.

Ce type de subvention est considéré comme « à trois étapes » OAuth car il repose sur la redirection des utilisateurs vers le serveur d'autorisation tiers pour authentifier l'utilisateur. Il est utilisé lors de la création de connexions via la AWS Glue console. La AWS Glue console redirigera l'utilisateur vers Google Analytics 4 où il devra se connecter et autoriser AWS Glue les autorisations demandées pour accéder à son instance Google Analytics 4.

Les utilisateurs peuvent toujours choisir de créer leur propre application connectée dans Google Analytics 4 et de fournir leur propre identifiant client et leur propre secret client lors de la création de connexions via la AWS Glue console. Dans ce scénario, ils seront toujours redirigés vers Google Analytics 4 pour se connecter et autoriser AWS Glue l'accès à leurs ressources.

Ce type d'autorisation donne lieu à un jeton d'actualisation et à un jeton d'accès. Le jeton d'accès est de courte durée et peut être actualisé automatiquement sans interaction de l'utilisateur à l'aide du jeton d'actualisation.

Pour plus d'informations, consultez Utiliser Auth 2.0 pour accéder à Google APIs.

Lecture à partir d'entités Google Analytics 4

Conditions préalables

- Un objet Google Analytics 4 à partir duquel vous souhaitez lire. Reportez-vous au tableau des entités prises en charge ci-dessous pour vérifier les entités disponibles.

Entités prises en charge

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Real-Time Report	Oui	Oui	Oui	Oui	Non
Core Report	Oui	Oui	Oui	Oui	Oui

Exemple

```
googleAnalytics4_read = glueContext.create_dynamic_frame.from_options(
    connection_type="GoogleAnalytics4",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "API_VERSION": "v1beta"
    }
```

Détails des entités et des champs Google Analytics 4

Entité	Champ	Type de données	Opérateurs pris en charge
Core Report	Dynamic Fields		
Core Report	Champs de dimension s	String	LIKE, =
Core Report	Champs de dimension s	Date	LIKE, =

Entité	Champ	Type de données	Opérateurs pris en charge
Core Report	Champs de métriques	String	>, <, >=, <=, = BETWEEN
Core Report	Champs de dimensions et de métriques personnalisés	String	NA
Real-Time Report	appVersion	String	LIKE, =
Real-Time Report	audienceId	String	LIKE, =
Real-Time Report	audienceName	String	LIKE, =
Real-Time Report	city	String	LIKE, =
Real-Time Report	cityId	String	LIKE, =
Real-Time Report	country	String	LIKE, =
Real-Time Report	countryId	String	LIKE, =
Real-Time Report	deviceCategory	String	LIKE, =
Real-Time Report	eventName	String	LIKE, =
Real-Time Report	minutesAgo	String	LIKE, =
Real-Time Report	platform	String	LIKE, =
Real-Time Report	streamId	String	LIKE, =
Real-Time Report	streamName	String	LIKE, =
Real-Time Report	unifiedScreenName	String	LIKE, =
Real-Time Report	activeUsers	String	>, <, >=, <=, = BETWEEN

Entité	Champ	Type de données	Opérateurs pris en charge
Real-Time Report	conversions	String	>, <, >=, <=, = BETWEEN
Real-Time Report	eventCount	String	>, <, >=, <=, = BETWEEN
Real-Time Report	screenPageViews	String	>, <, >=, <=, = BETWEEN

Requêtes de partitionnement

1. Partition basée sur un filtre

   Les options Spark supplémentaires `PARTITION_FIELD`, `LOWER_BOUND`, `UPPER_BOUND` et `NUM_PARTITIONS` peuvent être indiquées si vous souhaitez utiliser la simultanéité dans Spark. Avec ces paramètres, la requête d'origine serait divisée en `NUM_PARTITIONS` nombres de sous-requêtes pouvant être exécutées simultanément par les tâches Spark.

   - `PARTITION_FIELD` : le nom du champ à utiliser pour partitionner la requête.
   - `LOWER_BOUND` : une valeur limite inférieure inclusive du champ de partition choisi.

     Pour le champ Date, nous acceptons le format de date Spark utilisé dans les requêtes SQL Spark. Exemples de valeurs valides : `"2024-02-06"`.
   - `UPPER_BOUND` : une valeur limite supérieure exclusive du champ de partition choisi.
   - `NUM_PARTITIONS` : nombre de partitions.

   Exemple

```
googleAnalytics4_read = glueContext.create_dynamic_frame.from_options(
    connection_type="GoogleAnalytics4",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "API_VERSION": "v1beta",
        "PARTITION_FIELD": "date"
        "LOWER_BOUND": "2022-01-01"
```

```
        "UPPER_BOUND": "2024-01-02"
        "NUM_PARTITIONS": "10"
    }
```

2. Partition basée sur des enregistrements

   Les options Spark supplémentaires NUM_PARTITIONS peuvent être indiquées si vous souhaitez utiliser la simultanéité dans Spark. Avec ces paramètres, la requête d'origine serait divisée en NUM_PARTITIONS nombres de sous-requêtes pouvant être exécutées simultanément par les tâches Spark.

   - NUM_PARTITIONS : nombre de partitions.

   Exemple

```
googleAnalytics4_read = glueContext.create_dynamic_frame.from_options(
    connection_type="GoogleAnalytics4",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "API_VERSION": "v1beta",
        "NUM_PARTITIONS": "10"
    }
```

Options de connexion Google Analytics 4

Les options de connexion pour Google Analytics 4 sont les suivantes :

- ENTITY_NAME(Chaîne) - (Obligatoire) Utilisé pour la lecture. Le nom de votre objet dans Google Analytics 4.
- API_VERSION(Chaîne) - (Obligatoire) Utilisé pour la lecture. Version d'API Rest Google Analytics 4 que vous souhaitez utiliser.
- SELECTED_FIELDS(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- FILTER_PREDICATE(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- QUERY(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.
- PARTITION_FIELD(Chaîne) - Utilisé pour la lecture. Champ à utiliser pour partitionner la requête.

- `LOWER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite inférieure inclusive du champ de partition choisi.
- `UPPER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS`(Entier) - Par défaut : 1. Utilisé pour la lecture. Nombre de partitions pour la lecture.
- `INSTANCE_URL`(Entier) - Utilisé pour la lecture. (Facultatif)

Création d'un compte Google Analytics 4

Suivez les étapes pour créer un compte Google Analytics 4 : https://support.google.com/analytics/answer/9304153 ? hl=fr

Étapes pour créer une application cliente et des informations d'identification OAuth 2.0

Pour plus d'informations, consultez la documentation de l'API Google Analytics4.

1. Créez et configurez votre compte en vous connectant à votre compte Google Analytics à l'aide de vos informations d'identification. Accédez ensuite à Admin > Créer un compte.
2. Créez une propriété pour le compte que vous avez créé en choisissant Créer une propriété. Configurez la propriété avec les informations requises. Une fois tous les détails fournis, l'identifiant de propriété correspondant sera généré.
3. Ajoutez un flux de données pour la propriété créée en choisissant Flux de données > Ajouter un flux > Web dans le menu déroulant. Indiquez les détails du site Web tels que l'URL et d'autres champs obligatoires. Après avoir fourni tous les détails, l'identifiant de flux et l'identifiant de mesure correspondants seront générés.
4. Configurez Google Analytics sur votre site Web en copiant l'identifiant de mesure et en l'ajoutant à la configuration de votre site Web.
5. Créez un rapport à partir de Google Analytics en accédant à Rapports et en générant le rapport requis.
6. Autorisez votre application en accédant à console.cloud.google.com et en recherchant l'API de données Google Analytics, puis en activant l'API.
    1. Accédez à la page API et services et choisissez Credentials > setup OAuth 2.0 Client IDs.
    2. Fournissez l'URL de redirection en ajoutant l'URL de AWS Glue redirection.

7. Copiez l'ID client et le secret client, qui nécessiteront des informations supplémentaires pour créer une connexion.

Limites et considérations

Les restrictions du connecteur Google Analytics 4 sont les suivantes :

- Pour l'entité Core Report, seuls neuf champs de dimension et dix champs de métriques sont autorisés à envoyer une demande. Si le nombre de champs autorisés est dépassé, la demande échouera et le connecteur enverra un message d'erreur.
- Pour l'entité Real-Time Report, seuls les champs de quatre dimensions sont autorisés à envoyer une demande. Si le nombre de champs autorisés est dépassé, la demande échouera et le connecteur enverra un message d'erreur.
- Google Analytics 4 est un outil gratuit en version bêta. Il y aura donc des mises à jour régulières sur les nouvelles fonctionnalités, l'amélioration des entités, l'ajout de nouveaux champs et le retrait des champs existants.
- Les champs Core Report étant remplis de manière dynamique, des champs seront ajoutés, rendus obsolètes et renommés, et l'imposition de nouvelles limites aux champs peut avoir lieu à tout moment.
- La date de début par défaut est de 30 jours et la date de fin est hier (un jour avant la date actuelle), et ces dates seront remplacées dans le code d'expression du filtre si l'utilisateur a défini la valeur OU si le flux est incrémentiel.
- Selon la documentation, l'entité Real-Time Report renvoie 10 000 enregistrements si la limite n'est pas transmise dans la demande, sinon l'API renvoie un maximum de 250 000 lignes par demande, quel que soit le nombre que vous demandez. Pour plus d'informations, voir Méthode : propriétés.runRealtimeReportdans la documentation de Google Analytics.
- L'entité Real-Time Report ne prend pas en charge la partition basée sur les enregistrements, car elle ne prend pas en charge la pagination. De plus, elle ne prend pas en charge la partition basée sur les champs, car aucun des champs ne répond aux critères définis.
- La raison est la restriction du nombre de champs qui peuvent être transmis dans une demande. Nous définissons les champs de dimension et de métrique par défaut dans les limites indiquées. Si « Tout sélectionner » est sélectionné, seules les données de ces champs prédéterminés seront extraites.
  - Core Report

    - Conformément aux restrictions de SAAS, les demandes sont autorisées pour un maximum de 9 dimensions et de 10 métriques uniquement (c'est-à-dire qu'une demande peut contenir un maximum de 19 champs [métriques + dimension]).
    - Conformément à l'implémentation : si l'utilisateur utilise SELECT_ALL ou sélectionne plus de 25 champs, les champs par défaut seront transmis dans la demande.
    - Les champs suivants sont considérés comme des champs par défaut pour Core Report : « country », « city », « EventName », « cityID », « browser », « date », « CurrencyCode », « DeviceCategory », « TransactionId », « active1 », « active28 », « active7 », DayUsers « ActiveUsers », « User », DayUsers « Engagement » Sessions », DayUsers « Nombre d'événements », averagePurchaseRevenue « Taux d'engagement »averageRevenuePer. averageSessionDuration
  - Real-Time Report
    - Conformément aux restrictions de SAAS, les demandes sont autorisées pour un maximum de quatre dimensions.
    - Si l'utilisateur transmet SELECT_ALL ou si les champs sélectionnés sont supérieurs à 15, les champs par défaut seront transmis dans la demande.
    - Les champs suivants sont considérés comme des champs par défaut pour le RealTime rapport : « country », « DeviceCategory », « city », « CityID », « ActiveUsers », « conversions », « EventCount », « ». screenPageViews
- Dans l'entité Core-Report, si la partition sur le champ de date et le filtre sur StartDate sont présents simultanément. Dans ce cas, la valeur dateRange est remplacée par la valeur du filtre startDate, mais comme la partition doit toujours être la priorité, le filtre startDate est supprimé si la partition sur le champ de date est déjà présente.
- Comme cohortSpecs fait désormais également partie du corps de la demande Core-Report, nous avons amélioré l'entité Core-Report actuelle pour inclure la prise en charge de l'attribut cohortSpec. Dans le corps de la demande cohortSpecs, presque tous les champs nécessitent une saisie par l'utilisateur. Pour résoudre ce problème, nous avons défini des valeurs par défaut pour celles-ci attributes/fields et avons prévu des dispositions permettant à l'utilisateur de remplacer ces valeurs si nécessaire.

FieldName	Valeurs par défaut	Exemple de requête pour transmettre les options filterPredicate afin de remplacer les valeurs par défaut
startDate	30 jours à partir de la date actuelle	"startDate between "2023-05-09" and "2023-05-10"
endDate	1 jour à partir de la date actuelle	"startDate between "2023-05-09" and "2023-05-10"
startOffset	0	startOffset=2
endOffset	1	endOffset=10
granularité	DAILY	granularity="WEEKLY"

- Vous pouvez également transmettre tous ces filtres ensemble en une seule fois ou avec d'autres filtres.
  - Exemple 1 : filterPredicate : startDate between "2023-05-09" and "2023-05-10" AND startOffset=1 AND endOffset=2 AND granularity="WEEKLY"
  - Example 2 : filterPredicate: city="xyz" AND startOffset=1 AND endOffset=2 AND granularity="WEEKLY"
- Dans le cadre d'une demande de cohorte :
  - Si « cohortNthMonth » est transmis dans la demande, la valeur de granularité interne sera définie comme « MENSUELLE »
  - De même, si « cohortNthWeek » est transmis, la valeur de granularité sera définie comme « HEBDOMADAIRE »
  - Et, pour « cohortNthDay », la valeur de granularité sera définie comme « TOUS LES JOURS ». Pour en savoir plus, consultez :
    - https://developers.google.com/analytics/devguides/reporting/data/v1/advanced
    - https://developers.google.com/analytics/devguides/reporting/data/v1/rest/v1beta/CohortSpec
  - Il est permis à l'utilisateur de remplacer la valeur par défaut dateRange et de granularité. Reportez-vous au tableau suivant.

Connexion à Google BigQuery en AWS Glue Studio

> **Note**
>
> Vous pouvez utiliser Spark AWS Glue pour lire et écrire dans des tables dans Google BigQuery dans les versions AWS Glue 4.0 et ultérieures. Pour configurer Google BigQuery avec des AWS Glue offres d'emploi par programmation, voir BigQuery connexions.

AWS Glue Studiofournit une interface visuelle pour se connecter BigQuery, créer des tâches d'intégration de données et les exécuter sur le runtime Spark AWS Glue Studio sans serveur.

Lors de la création d'une connexion à Google BigQuery In AWS Glue Studio, une connexion unifiée est créée. Pour de plus amples informations, veuillez consulter Considérations.

Au lieu de créer un secret avec les informations d'identification dans un format spécifique`{"credentials": "base64 encoded JSON"}`, désormais avec une connexion unifiée à Google BigQuery, vous pouvez créer un secret qui inclut directement le JSON de Google : BigQuery. `{"type": "service-account", ...}`

Rubriques

- Création d'une BigQuery connexion
- Création d'un nœud BigQuery source
- Création d'un nœud BigQuery cible
- Options avancées

Création d'une BigQuery connexion

Pour vous connecter à Google BigQuery depuis AWS Glue, vous devez créer et stocker vos informations d'identification Google Cloud Platform dans un AWS Secrets Manager secret, puis associer ce secret à une BigQuery AWS Glue connexion Google.

Pour configurer une connexion à BigQuery :

1. Dans Google Cloud Platform, créez et identifiez les ressources pertinentes :

- Créez ou identifiez un projet GCP contenant des BigQuery tables auxquelles vous souhaitez vous connecter.
- Activez l' BigQuery API. Pour plus d'informations, voir Utiliser l'API BigQuery Storage Read pour lire les données des tables.

2. Dans Google Cloud Platform, créez et exportez les informations d'identification du compte de service :

   Vous pouvez utiliser l'assistant BigQuery d'identification pour accélérer cette étape : créer des informations d'identification.

   Pour créer un compte de service dans GCP, suivez le didacticiel disponible dans la section Créer des comptes de service.

   - Lorsque vous sélectionnez un projet, sélectionnez le projet contenant votre BigQuery tableau.
   - Lorsque vous sélectionnez des rôles GCP IAM pour votre compte de service, ajoutez ou créez un rôle qui accordera les autorisations appropriées pour exécuter des BigQuery tâches de lecture, d'écriture ou de création BigQuery de tables.

   Pour créer des informations d'identification pour votre compte de service, suivez le didacticiel disponible dans la section Créer une clé de compte de service.

   - Lorsque vous sélectionnez le type de clé, sélectionnez JSON.

   Vous devriez maintenant avoir téléchargé un fichier JSON contenant les informations d'identification de votre compte de service. Il doit ressembler à l'exemple ci-dessous.

```
{
  "type": "service_account",
  "project_id": "*****",
  "private_key_id": "*****",
  "private_key": "*****",
  "client_email": "*****",
  "client_id": "*****",
  "auth_uri": "https://accounts.google.com/o/oauth2/auth",
  "token_uri": "https://oauth2.googleapis.com/token",
  "auth_provider_x509_cert_url": "https://www.googleapis.com/oauth2/v1/certs",
  "client_x509_cert_url": "*****",
  "universe_domain": "googleapis.com"
```

```
}
```

3. Dans AWS Secrets Manager, créez un secret à l'aide du fichier d'informations d'identification que vous avez téléchargé. Vous pouvez choisir l'onglet Texte brut et coller le contenu du fichier au format JSON. Pour créer un secret dans Secrets Manager, suivez le didacticiel disponible dans la section Créer un AWS Secrets Manager secret dans la AWS Secrets Manager documentation. Après avoir créé le secret, conservez le nom du secret *`secretName`* pour l'étape suivante.
4. Dans le catalogue de AWS Glue données, créez une connexion en suivant les étapes décrites danshttps://docs.aws.amazon.com/glue/latest/dg/console-connections.html. Après avoir créé la connexion, conservez le nom de la connexion pour l'étape suivante. *`connectionName`*
   - Lorsque vous sélectionnez un type de connexion, sélectionnez Google BigQuery.
   - Lorsque vous sélectionnez un AWS secret, fournissez*`secretName`*.
5. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire*`secretName`*.
6. Dans la configuration de votre AWS Glue tâche, fournissez *`connectionName`* une connexion réseau supplémentaire.

Création d'un nœud BigQuery source

Prérequis

- Une connexion BigQuery de type AWS Glue Data Catalog
- Un AWS Secrets Manager secret pour vos BigQuery informations d'identification Google, utilisées par la connexion.
- Les autorisations appropriées sur votre tâche pour lire le secret utilisé par la connexion.
- Le nom et le jeu de données de la table et du projet Google Cloud correspondant que vous souhaitez lire.

Ajouter une source BigQuery de données

Pour ajouter une source de données — BigQuery nœud :

1. Choisissez la connexion pour votre source BigQuery de données. Puisque vous l'avez créé, il devrait être disponible dans le menu déroulant. Si vous devez créer une connexion, choisissez Créer une BigQuery connexion. Pour en savoir plus, consultez Overview of using connectors and connections .

Une fois que vous avez choisi une connexion, vous pouvez afficher ses propriétés en cliquant sur Afficher les propriétés.

2. Identifiez les BigQuery données que vous souhaitez lire, puis choisissez une option BigQuery Source

   - Choisissez une seule table : cela vous permet d'extraire toutes les données d'une table.
   - Saisissez une requête personnalisée : cela vous permet de personnaliser les données extraites en fournissant une requête.

3. Décrivez les données que vous souhaitez lire.

   (Obligatoire) Définissez le Projet parent sur le projet contenant votre table, ou sur un projet parent de facturation, le cas échéant.

   Si vous avez choisi une seule table, attribuez à Table le nom d'une BigQuery table Google au format suivant : `[dataset].[table]`

   Si vous avez choisi une requête, indiquez-la à Requête. Dans votre requête, faites référence aux tables avec leur nom de table complet, au format suivant : `[project].[dataset].[tableName]`.

4. Fournir des BigQuery propriétés

   Si vous avez choisi une seule table, vous n'avez pas besoin de fournir de propriétés supplémentaires.

   Si vous avez choisi une requête, vous devez fournir les BigQuery propriétés Google personnalisées suivantes :

   - Définissez `viewsEnabled` sur true.
   - Définissez `materializationDataset` sur un jeu de données. Le principal GCP authentifié par les informations d'identification fournies via la AWS Glue connexion doit être en mesure de créer des tables dans cet ensemble de données.

Création d'un nœud BigQuery cible

Prérequis

- Une connexion BigQuery de type AWS Glue Data Catalog

- Un AWS Secrets Manager secret pour vos BigQuery informations d'identification Google, utilisées par la connexion.
- Les autorisations appropriées sur votre tâche pour lire le secret utilisé par la connexion.
- Le nom et le jeu de données de la table et du projet Google Cloud correspondant dans lequel vous souhaitez écrire.

Ajouter une cible BigQuery de données

Pour ajouter une cible de données — BigQuery nœud :

1. Choisissez la connexion pour votre cible BigQuery de données. Puisque vous l'avez créé, il devrait être disponible dans le menu déroulant. Si vous devez créer une connexion, choisissez Créer une BigQuery connexion. Pour en savoir plus, consultez Overview of using connectors and connections .

   Une fois que vous avez choisi une connexion, vous pouvez afficher ses propriétés en cliquant sur Afficher les propriétés.
2. Identifiez la BigQuery table dans laquelle vous souhaitez écrire, puis choisissez une méthode d'écriture.

   - Direct : écrit BigQuery directement sur le site à l'aide de l'API BigQuery Storage Write.
   - Indirect : écrit dans Google Cloud Storage, puis copie dans BigQuery.

   Si vous souhaitez écrire indirectement, indiquez un emplacement GCS de destination avec un Compartiment GCS temporaire. Vous devrez fournir une configuration supplémentaire dans votre AWS Glue connexion. Pour plus d'informations, consultez la section Utilisation de l'écriture indirecte avec Google BigQuery.
3. Décrivez les données que vous souhaitez lire.

   (Obligatoire) Définissez le Projet parent sur le projet contenant votre table, ou sur un projet parent de facturation, le cas échéant.

   Si vous avez choisi une seule table, attribuez à Table le nom d'une BigQuery table Google au format suivant : `[dataset].[table]`

Options avancées

Vous pouvez fournir des options avancées lors de la création d'un BigQuery nœud. Ces options sont les mêmes que celles disponibles lors de la programmation AWS Glue de scripts Spark.

Voir BigQuery la référence des options de connexion dans le guide du AWS Glue développeur.

Connexion à Google Search Console

Google Search Console est une plateforme gratuite mise à la disposition des propriétaires de sites Web pour surveiller la façon dont Google voit le site et pour optimiser sa présence organique. Cela inclut l'affichage des domaines de référence, des performances des sites mobiles, des résultats de recherche enrichis et des requêtes et pages les plus fréquentées. Si vous utilisez la console de recherche Google, vous pouvez vous connecter AWS Glue à votre compte Google Search Console. Vous pouvez utiliser Google Search Console comme source de données dans vos tâches ETL. Exécutez ces tâches pour transférer des données de la console de recherche Google vers AWS des services ou d'autres applications compatibles.

Rubriques

- AWS Glue support pour Google Search Console
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration de Google Search Console
- Configuration des connexions Google Search Console
- Lecture à partir des entités Google Search Console
- Options de connexion Google Search Console
- Restrictions de Google Search Console

AWS Glue support pour Google Search Console

AWS Glue prend en charge la console de recherche Google comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser des tâches AWS Glue ETL pour interroger des données à partir de la console de recherche Google.

Prise en charge en tant que cible ?

Non.

Versions de l'API Google Search Console prises en charge

Les versions de l'API Google Search Console prises en charge sont les suivantes :

- v3

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les autorisations AWS IAM requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Si vous ne voulez pas utiliser la méthode ci-dessus, vous pouvez également utiliser les politiques IAM gérées suivantes :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.

- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration de Google Search Console

Avant de pouvoir AWS Glue transférer des données depuis la console de recherche Google, vous devez satisfaire aux exigences suivantes :

Configuration requise

Les exigences minimales sont les suivantes :

- Vous disposez d'un compte Google Search Console.
- Vous disposez d'un compte Google Cloud Platform et d'un projet Google Cloud.
- Dans votre projet Google Cloud, vous avez activé l'API Google Search Console.
- Dans votre projet Google Cloud, vous avez configuré un écran de OAuth consentement pour les utilisateurs externes. Pour plus d'informations, consultez la section Configuration de votre écran de OAuth consentement dans l'aide de la console Google Cloud Platform.
- Dans votre projet Google Cloud, vous avez configuré un identifiant client OAuth 2.0. Consultez la section Configuration OAuth 2.0 pour connaître les informations d'identification du client AWS Glue qui permettent d'accéder à vos données en toute sécurité lorsqu'il effectue des appels authentifiés vers votre compte.

Si vous répondez à ces exigences, vous êtes prêt à vous connecter AWS Glue à votre compte Google Search Console. Pour les connexions classiques, vous n'avez rien d'autre à faire dans Google Search Console.

Configuration des connexions Google Search Console

La console de recherche Google prend en charge le type de subvention AUTHORIZATION_CODE pour. OAuth2 Le type de subvention détermine le mode AWS Glue de communication avec la console de recherche Google pour demander l'accès à vos données.

- Ce type de subvention est considéré comme « à trois étapes » OAuth car il repose sur la redirection des utilisateurs vers un serveur d'autorisation tiers pour authentifier l'utilisateur. Il est utilisé lors de la création de connexions via la AWS Glue console.
- Les utilisateurs peuvent toujours choisir de créer leur propre application connectée dans la console de recherche Google et de fournir leur propre identifiant client et leur secret client lors de la création de connexions via la AWS Glue console. Dans ce scénario, ils seront toujours redirigés vers la console de recherche Google pour se connecter et autoriser AWS Glue l'accès à leurs ressources.
- Ce type d'autorisation donne lieu à un jeton d'actualisation et à un jeton d'accès. Le jeton d'accès est de courte durée et peut être actualisé automatiquement sans interaction de l'utilisateur à l'aide du jeton d'actualisation.
- Pour consulter la documentation publique de la console de recherche Google sur la création d'une application connectée pour le OAuth flux de code d'autorisation, voir Utilisation de la OAuth version 2.0 pour accéder à Google APIs.

Pour configurer une connexion Google Search Console :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes :
   a. Pour l'application connectée gérée par le client, le secret doit contenir le secret du consommateur de l'application connectée avec `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` comme clé.
   b. Remarque : vous devez créer un secret pour vos connexions dans AWS Glue.

1. Dans AWS Glue Glue Studio, créez une connexion sous Data Connections en suivant les étapes ci-dessous :
   a. Lorsque vous sélectionnez un Type de connexion, sélectionnez Google Search Console.
   b. Sélectionnez le rôle AWS IAM qui AWS Glue peut assumer et dispose des autorisations nécessaires pour les actions suivantes :

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
```

```
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

c. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.

d. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.

2. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.

Lecture à partir des entités Google Search Console

Prérequis

Un objet Google Search Console à partir duquel vous souhaitez lire. Vous aurez besoin du nom de l'objet.

Entités prises en charge pour la source :

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Search Analytics	Oui	Oui	Non	Oui	Non
Sites	Non	Non	Non	Oui	Non
Sitemaps	Non	Non	Non	Oui	Non

Exemple :

```
googleSearchConsole_read = glueContext.create_dynamic_frame.from_options(
    connection_type="googlesearchconsole",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "API_VERSION": "v3"
    }
```

Détails des entités et des champs Google Search Console :

Google Search Console fournit des points de terminaison permettant de récupérer les métadonnées de manière dynamique pour les entités prises en charge. Par conséquent, la prise en charge des opérateurs est capturée au niveau du type de données.

Entité	Champ	Type de données	Opérateurs pris en charge	Remarque
Search Analytics	clés	List	N/A	
	clicks	Double	N/A	
	impressions	Double	N/A	
	ctr	BigDecimal	N/A	Pour le BigDecimal type de données, la valeur « 0 » est formatée comme « 0E-18 »
	position	Double	N/A	
	start_end_date	Date	BETWEEN	La valeur par défaut pour `start_end_date` est comprise entre <30 days ago from the current

Entité	Champ	Type de données	Opérateurs pris en charge	Remarque
				date> ET <yesterday: that is, 1 day ago from the current date>. Remarque : Il est attendu que vous transmettiez une valeur de date UTC. Exemple : start_end _date between '2022-01-01' AND '2024-09-09'
	country	String	EQUAL_TO, NOT_EQUAL _TO, CONTAINS	Les valeurs valides sont « IND », « CAN », etc.
	type	String	EQUAL_TO, NOT_EQUAL _TO	Les valeurs valides sont « discover », « googleNew s », « news », « image », « video », « web ».

Entité	Champ	Type de données	Opérateurs pris en charge	Remarque
	searchAppearance	String	EQUAL_TO, NOT_EQUAL_TO, CONTAINS	Consultez Search Appearance pour une liste de valeurs valides.
	device	String	EQUAL_TO, NOT_EQUAL_TO, CONTAINS	Les valeurs valides sont « DESKTOP », « MOBILE », « TABLET ».
	dimensions	String	EQUAL_TO	Les valeurs valides sont « country », « device ».
	page	String	EQUAL_TO, NOT_EQUAL_TO, CONTAINS	
	query	String	EQUAL_TO, NOT_EQUAL_TO, CONTAINS	
	dataState	String	EQUAL_TO	Les valeurs valides sont « all » et « final ».
Sites	siteUrl	String	N/A	
	permissionLevel	String	N/A	
Sitemaps	path	String	N/A	

Entité	Champ	Type de données	Opérateurs pris en charge	Remarque
	type	String	N/A	
	lastSubmitted	DateTime	N/A	
	isPending	Booléen	N/A	
	isSitemapsIndex	Booléen	N/A	
	lastDownloaded	DateTime	N/A	
	warnings	Long	N/A	
	errors	Long	N/A	
	contents	List	N/A	

> **Note**
>
> Pour obtenir une liste actualisée des valeurs valides pour les filtres, consultez la documentation relative à l'API Google Search Console.
> Le champ `start_end_date` est une combinaison de `start_date` et de `end_date`.

Requêtes de partitionnement

Le partitionnement basé sur les filtres et le partitionnement basé sur les enregistrements ne sont pas pris en charge.

Options de connexion Google Search Console

Les options de connexion pour Google Search Console sont les suivantes :

- `ENTITY_NAME`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Le nom de votre objet dans Google Search Console.
- `API_VERSION`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Version d'API Rest Google Search Console que vous souhaitez utiliser.

- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- `FILTER_PREDICATE`(Chaîne) - Par défaut : "start_end_date between <30 days ago from current date> AND <yesterday: that is, 1 day ago from the current date>". Utilisé pour la lecture. Doit être au format SQL Spark.
- `QUERY`(Chaîne) - Par défaut : "start_end_date between <30 days ago from current date> AND <yesterday: that is, 1 day ago from the current date>" Utilisé pour la lecture. Requête Spark SQL complète.
- `INSTANCE_URL`(Chaîne) - Utilisé pour la lecture. URL d'instance Google Search Console valide.

Restrictions de Google Search Console

Voici les restrictions ou les remarques relatives à Google Search Console :

- Google Search Console impose des limites d'utilisation à l'API. Pour plus d'informations, consultez Limites d'utilisation.
- Lorsqu'aucun filtre n'est appliqué à l'entité `Search Analytics`, l'API résume tous les clics, impressions, CTR et autres données pour l'ensemble de votre site dans la plage de dates par défaut spécifiée et les présente sous forme d'enregistrement unique.
- Pour décomposer les données en segments plus petits, vous devez ajouter des dimensions à votre requête. Les dimensions indiquent à l'API comment vous souhaitez segmenter vos données.
  - Par exemple, si vous ajoutez `filterPredicate: dimensions="country"`, vous obtiendrez un enregistrement pour chaque pays dans lequel votre site a reçu du trafic au cours de la période spécifiée.
  - Exemple pour transmettre plusieurs dimensions : `filterPredicate: dimensions="country" AND dimensions="device" AND dimensions="page"`. Dans ce cas, vous obtiendrez une ligne dans la réponse pour chaque combinaison unique de ces trois dimensions.
- Les valeurs par défaut sont définies pour les champs `start_end_date` et `dataState`.

Nom de champ	Expression de filtrage par défaut	Expression pour remplacer les valeurs par défaut
start_end_date	start_end_date between <30 days ago from current date> AND <yesterday: that is, 1 day ago from the current date>	start_end_date between "2024-01-01" AND "2024-05-05"
dateState	dataState="all"	dataState="final"

Connexion à Google Sheets

Google Sheets est un tableur en ligne qui vous permet d'organiser de grandes quantités de données, de créer des rapports personnalisés, d'automatiser les calculs et de collaborer avec d'autres personnes. Si vous utilisez Google Sheets, vous pouvez vous connecter AWS Glue à votre compte Google Sheets. Vous pouvez ensuite utiliser Google Sheets comme source de données dans vos tâches ETL. Exécutez ces tâches pour transférer des données entre Google Sheets et AWS des services ou d'autres applications compatibles.

Rubriques

- AWS Glue support pour Google Sheets
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration de Google Sheets
- Configuration des connexions Google Sheets
- Lecture à partir d'entités Google Sheets
- Options de connexion Google Sheets
- Configurer le OAuth flux de code d'autorisation pour Google Sheets
- Restrictions du connecteur Google Sheets

AWS Glue support pour Google Sheets

AWS Glue prend en charge Google Sheets comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser les tâches AWS Glue ETL pour interroger les données de Google Sheets.

Prise en charge en tant que cible ?

Non.

Versions de l'API Google Sheets prises en charge

API Google Sheets v4 et API Google Drive v3

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les AWS autorisations requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Vous pouvez également utiliser les politiques IAM gérées suivantes pour autoriser l'accès :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources

spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.

- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration de Google Sheets

Avant de pouvoir AWS Glue effectuer un transfert depuis Google Sheets, vous devez remplir les conditions suivantes :

Configuration requise

- Vous disposez d'un compte Google Sheets avec e-mail et mot de passe.
- Votre compte Google Sheets est activé pour accéder à l'API. Toute utilisation de l'API Google Sheets est disponible sans frais supplémentaires.
- Votre compte Google Sheets vous permet d'installer des applications connectées. Si vous n'avez pas accès à cette fonctionnalité, contactez votre administrateur Google Sheets.

Si vous répondez à ces exigences, vous êtes prêt à vous connecter AWS Glue à votre compte Google Sheets.

Configuration des connexions Google Sheets

Pour configurer une connexion à Google Sheet :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes :
   a. Pour le type de AuthorizationCode subvention :
      - Pour l'application connectée gérée par le client : le secret doit contenir le secret du consommateur de l'application connectée avec `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` comme clé.
2. Dans AWS Glue Glue Studio, créez une connexion sous Data Connections en suivant les étapes ci-dessous :

a. Lorsque vous sélectionnez une source de données, sélectionnez Google Sheets.

b. Indiquez l’environnement Google Sheets.

   i. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.

   ii. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.

3. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

Type d’autorisation AUTHORIZATION_CODE

Ce type de subvention est considéré comme « à trois étapes » OAuth car il repose sur la redirection des utilisateurs vers le serveur d'autorisation tiers pour authentifier l'utilisateur. Il est utilisé lors de la création de connexions via la AWS Glue console. La AWS Glue console redirige l'utilisateur vers Google Sheets où il doit se connecter et autoriser AWS Glue les autorisations demandées pour accéder à son instance Google Sheets.

Les utilisateurs peuvent choisir de créer leur propre application connectée dans Google Sheets et de fournir leur propre identifiant client et leur propre secret client lors de la création de connexions via la AWS Glue console. Dans ce scénario, ils seront toujours redirigés vers Google Sheets pour se connecter et autoriser AWS Glue l'accès à leurs ressources.

Ce type d'autorisation donne lieu à un jeton d'actualisation et à un jeton d'accès. Le jeton d'accès est de courte durée et peut être actualisé automatiquement sans interaction de l'utilisateur à l'aide du jeton d'actualisation.

Pour plus d'informations, consultez la documentation publique de Google Sheets sur la création d'une application connectée pour le OAuth flux de code d'autorisation.

Lecture à partir d'entités Google Sheets

Conditions préalables

- Un Google SpreadSheet que vous aimeriez lire. Vous aurez besoin de l' SpreadSheet ID et du TabName de la feuille de calcul.

Détails des entités et des champs Google Sheets :

Entité	Type de données	Opérateurs pris en charge
Feuille de calcul	String	N/A (le filtre n'est pas pris en charge)

Exemple

```
googleSheets_read = glueContext.create_dynamic_frame.from_options(
    connection_type="googlesheets",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "{SpreadSheetID}#{SheetTabName}",
        "API_VERSION": "v4"
    }
```

Requêtes de partitionnement

Pour Record Base Partitioning uniquement, vous pouvez fournir `NUM_PARTITIONS` en tant qu'options Spark supplémentaires si vous souhaitez utiliser la simultanéité dans Spark. Avec ce paramètre, la requête d'origine serait divisée en `NUM_PARTITIONS` nombres de sous-requêtes pouvant être exécutées simultanément par les tâches Spark.

Exemple avec **`NUM_PARTITIONS`**

```
googlesheets_read = glueContext.create_dynamic_frame.from_options(
    connection_type="googlesheets",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "{SpreadSheetID}#{SheetTabName}",
        "API_VERSION": "v4",
        "NUM_PARTITIONS": "10"
    }
```

Options de connexion Google Sheets

Les options de connexion pour Google Sheets sont les suivantes :

- `ENTITY_NAME`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Les éléments `SpreadSheet ID` et `sheetTabName` dans Google Sheets. Exemple: `{SpreadSheetID}#{SheetTabName}`.
- `API_VERSION`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Version d'API Rest Google Sheets que vous souhaitez utiliser.
- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.
- `NUM_PARTITIONS`(Entier) - Par défaut : 1. Utilisé pour la lecture. Nombre de partitions pour la lecture.

Configurer le OAuth flux de code d'autorisation pour Google Sheets

Conditions préalables

- Un compte Google auquel vous pouvez vous connecter pour utiliser l'application Google Sheets. Dans votre compte Google, Google Sheets contient les données que vous souhaitez transférer.
- Un compte Google Cloud Platform et un projet Google Cloud. Consultez Create Google Cloud Project pour plus de détails.

Pour configurer votre compte Google et obtenir des informations d'identification OAuth 2.0 :

1. Une fois le projet Google Cloud configuré, activez l'API Google Sheets et Google Drive APIs dans le projet. Pour connaître les étapes permettant de les activer, consultez la section Activer et désactiver APIs dans l'aide de la console API pour Google Cloud Platform.
2. Configurez ensuite un écran de OAuth consentement pour les utilisateurs externes. Pour plus d'informations sur l'écran de OAuth consentement, consultez la section Configuration de votre écran de OAuth consentement dans l'aide de la console Google Cloud Platform.
3. Dans l'écran de OAuth consentement, ajoutez les champs d'application suivants :

   - Le champ d'application en lecture seule de l'API Google Sheets
   - Le champ d'application en lecture seule de l'API Google Drive

   Pour plus d'informations sur ces étendues, consultez la section Scopes OAuth 2.0 pour Google APIs dans la documentation de Google Identity.
4. Générez un identifiant client OAuth 2.0 et un secret. Pour connaître les étapes de création de cet ID client, consultez la section Configuration de la OAuth version 2.0 dans l'aide de la console Google Cloud Platform.

   L'ID client OAuth 2.0 doit comporter une ou plusieurs redirections autorisées URLs.

   URLs Les redirections ont le format suivant :

   - https ://<aws-region>.console.aws.amazon. com/gluestudio/oauth
5. Notez l'ID client et le secret du client dans les paramètres de votre ID client OAuth 2.0.

Restrictions du connecteur Google Sheets

Les restrictions du connecteur Google Sheets sont les suivantes :

- Le connecteur Google Sheets ne prend pas en charge les filtres. Par conséquent, le partitionnement basé sur des filtres n'est pas pris en charge.
- Dans Record Base Partitioning, aucune disposition ne permet de renvoyer le nombre exact d'enregistrements par SAAS. Par conséquent, il peut exister des scénarios dans lesquels des fichiers contenant des enregistrements vides sont créés.

- Étant donné que le connecteur Google Sheets ne prend pas en charge le partitionnement basé sur des filtres, `partitionField`, `lowerbound` et `upperbound` ne sont pas des options de connexion valides. Si ces options sont fournies, la AWS Glue tâche est censée échouer.
- Il est essentiel de désigner la première ligne de la feuille comme ligne d'en-tête pour éviter les problèmes de traitement des données.
  - Si elle n'est pas fournie, la ligne d'en-tête sera remplacée par `Unnamed:1`, `Unnamed:2`, `Unnamed:3`, etc., si la feuille contient des données dont la première ligne est vide.
  - Si une ligne d'en-tête est fournie, les noms de colonnes vides seront remplacés par `Unnamed:<number of column>`. Par exemple, si la ligne d'en-tête est `['ColumnName1', 'ColumnName2', '', '', 'ColumnName5', 'ColumnName6']`, elle deviendra `['ColumnName1', 'ColumnName2', 'Unnamed:3', 'Unnamed:4', 'ColumnName5', 'ColumnName6'].`
- Le connecteur Google Sheets ne prend pas en charge le transfert incrémentiel.
- Le connecteur Google Sheets ne prend en charge que le type de données String.
- Les en-têtes dupliqués d'une feuille seront renommés de manière itérative avec un suffixe numérique. Les noms d'en-tête fournis par l'utilisateur auront priorité lors du changement de nom des en-têtes dupliqués. Par exemple, si la ligne d'en-tête est ["Name", "", "Name", null, "Unnamed:6", ""], elle deviendra : ["Name", "Unnamed:2", "Name1", "Unnamed:4", "Unnamed:6", "Unnamed:61"].
- Le connecteur Google Sheets ne prend pas en charge les espaces pour un élément tabName.
- Un nom de dossier ne peut pas contenir les caractères spéciaux suivants :
  - #
  - /

Connexion à HubSpot

HubSpotLa plateforme CRM dispose de tous les outils et intégrations dont vous avez besoin pour le marketing, les ventes, la gestion de contenu et le service client.

- Marketing Hub : un logiciel de marketing qui vous aide à augmenter le trafic, à convertir plus de visiteurs et à exécuter des campagnes de marketing entrant complètes à grande échelle.
- Sales Hub : un logiciel CRM pour les ventes qui vous aide à obtenir des informations détaillées sur les prospects, à automatiser vos tâches et à conclure plus de ventes plus rapidement.

- Service Hub : un logiciel de service client qui vous aide à entrer en contact avec les clients, à dépasser leurs attentes et à en faire des promoteurs capables de développer votre activité.
- Operations Hub : un logiciel d'exploitation qui synchronise vos applications, nettoie et crée les données client, et qui automatise les processus, afin que tous vos systèmes et équipes fonctionnent mieux ensemble.

Rubriques

- AWS Glue support pour HubSpot
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration HubSpot
- Configuration des HubSpot connexions
- Lecture à partir d' HubSpot entités
- Écrire à des HubSpot entités
- HubSpot options de connexion
- Limitations et remarques relatives au HubSpot connecteur

AWS Glue support pour HubSpot

AWS Glue prend en charge HubSpot les éléments suivants :

Prise en charge en tant que source ?

Oui : synchrone et asynchrone. Vous pouvez utiliser des tâches AWS Glue ETL pour interroger des données à partir de HubSpot.

Prise en charge en tant que cible ?

Oui. Vous pouvez utiliser des tâches AWS Glue ETL pour y écrire des données HubSpot.

Versions HubSpot d'API prises en charge

Les versions HubSpot d'API suivantes sont prises en charge :

- v1
- v2
- v3
- v4

Pour la prise en charge des entités par version spécifique, consultez Entités prises en charge pour la source de synchronisation et Entités prises en charge pour la source asynchrone.

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les autorisations AWS IAM requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Si vous ne voulez pas utiliser la méthode ci-dessus, vous pouvez également utiliser les politiques IAM gérées suivantes :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs

bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration HubSpot

Avant de pouvoir AWS Glue transférer des données depuis HubSpot, vous devez satisfaire aux exigences suivantes :

Configuration requise

Les exigences minimales sont les suivantes :

- Tu as un HubSpot compte. Pour de plus amples informations, veuillez consulter Création d'un HubSpot compte.
- Votre HubSpot compte est activé pour accéder à l'API.
- Vous devez disposer d'une application sous votre compte de HubSpot développeur qui fournit les informations d'identification du client qui AWS Glue permettent d'accéder à vos données en toute sécurité lorsqu'il passe des appels authentifiés vers votre compte. Pour de plus amples informations, veuillez consulter Création d'une application HubSpot pour développeurs.

Si vous remplissez ces conditions, vous êtes prêt à vous connecter AWS Glue à votre HubSpot compte. Pour les connexions classiques, vous n'avez rien d'autre à faire HubSpot.

Création d'un HubSpot compte

Pour créer un HubSpot compte, procédez comme suit :

1. Accédez à l' SignUp URL du HubSpot CRM.
2. Saisissez votre adresse e-mail et choisissez Vérifier l'adresse e-mail (vous pouvez également choisir de vous inscrire avec un compte Google, Microsoft ou Apple).
3. Vérifiez dans votre boîte de réception le code de vérification provenant de HubSpot.
4. Saisissez le code de vérification à six chiffres et cliquez sur Suivant.
5. Saisissez un mot de passe et cliquez sur Suivant.
6. Saisissez votre prénom et votre nom de famille et cliquez sur Suivant, ou inscrivez-vous en utilisant le lien S'inscrire avec Google.
7. Saisissez votre secteur d'activité et cliquez sur Suivant.

8. Saisissez votre fonction et cliquez sur Suivant.
9. Saisissez le nom de votre entreprise et cliquez sur Suivant.
10. Sélectionnez la taille de votre entreprise (nombre d’employés) et cliquez sur Suivant.
11. Saisissez le site Web de votre entreprise et cliquez sur Suivant.
12. Sélectionnez l’endroit où vos données doivent être hébergées (États-Unis ou Europe) et cliquez sur Créer un compte.
13. Sélectionnez le but de la création de votre compte et cliquez sur Suivant.
14. Choisissez Connect Google Account ou choisissez d'ajouter vous-même des contacts pour les associer à votre HubSpot compte.
15. Connectez-vous à votre compte Google si vous avez choisi l'option Connect Google Account pour associer vos contacts et commencer à utiliser votre HubSpot compte.

Création d'une application HubSpot pour développeurs

Les comptes de développeur d’applications servent à créer et à gérer des applications, des intégrations et des comptes de test pour développeurs. Vous pouvez également y créer et gérer des listes de la marketplace d’applications. Toutefois, les comptes de développeur d'applications et leurs comptes de test associés ne sont pas connectés à un HubSpot compte standard. Ils ne peuvent pas synchroniser les données ou les actifs depuis ou vers un autre HubSpot compte. Pour obtenir l’ID client et le secret client, vous devez créer un compte de développeur.

1. Accédez à https://developers.hubspot.com/
2. Choisissez Créer un compte de développeur et faites défiler l’écran vers le bas.
3. Il vous sera demandé si vous souhaitez créer un compte de développeurs d’applications, un compte d’application privé ou un compte d’environnement de test (sandbox) CMS Developer. Choisissez Créer un compte de développeurs d’applications.
4. Puisque vous avez déjà créé un compte auprès de cet utilisateur HubSpot, vous pouvez choisir Continuer avec cet utilisateur.
5. Cliquez sur Démarrer l’inscription.
6. Saisissez votre fonction et cliquez sur Suivant.
7. Donnez un nom à votre compte de développeur et cliquez sur Suivant, puis sur Ignorer.
8. Sélectionnez Create App (Créer une application).
9. Une fois votre application créée, choisissez Auth.
10. Sous Auth, notez l’ID client et le secret client.

11. Ajoutez l'URL de redirection spécifique à votre région sous la forme https : *//<aws-region>* .console.aws.amazon. com/gluestudio/oauth. For example, add https://us-east-1.console.aws.amazon.com/gluestudio/oauthpour la région us-east-1.
12. Faites défiler la page vers le bas et trouvez des portées. Il existe deux types de portées que vous devez sélectionner sous les rubriques « CRM » et « Standard ».
13. Ajoutez les portées suivantes :

```
content
automation
oauth
crm.objects.owners.read
forms
tickets
crm.objects.contacts.write
e-commerce
crm.schemas.custom.read
crm.objects.custom.read
sales-email-read
crm.objects.custom.write
crm.objects.companies.write
crm.lists.write
crm.objects.companies.read
crm.lists.read
crm.objects.deals.read
crm.objects.deals.write
crm.objects.contacts.read
```

14. Cliquez sur Enregistrer et votre compte de développeur est maintenant prêt à être utilisé.
15. Faites défiler la page ci-dessus pour trouver l'ID client.
16. Sur la même page, cliquez sur Afficher pour obtenir le secret client.

Création d'un compte test pour HubSpot développeurs

Dans les comptes de développement d'applications, vous pouvez créer des comptes de test pour développeurs afin de tester des applications et des intégrations sans affecter les HubSpot données réelles. Les comptes de test pour développeurs ne reflètent pas les comptes de production, mais ont plutôt accès à un essai de 90 jours des versions Enterprise de Marketing, Sales, Service, CMS et Operations Hub, ce qui permet de tester la plupart des HubSpot outils et APIs.

1. Cliquez sur Accueil.

2. Cliquez sur Créer un compte de test.
3. Cliquez sur Créer un compte de test d'application.
4. Une nouvelle fenêtre s'affiche. Saisissez le nom du compte de test d'application et cliquez sur Créer.

   Votre compte de test d'application est à présent créé.

> **Note**
>
> Le compte de développeur est lié aux activités de développement telles que l'intégration d'API, tandis que le compte de test d'application est utilisé pour voir les données créées ou extraites par le compte de développeur.

Configuration des HubSpot connexions

HubSpot prend en charge le type de subvention AUTHORIZATION_CODE pour. OAuth2

- Ce type de subvention est considéré comme « à trois étapes » OAuth car il repose sur la redirection des utilisateurs vers un serveur d'autorisation tiers pour authentifier l'utilisateur. Il est utilisé lors de la création de connexions via la AWS Glue console. L'utilisateur qui crée une connexion doit fournir les informations OAuth associées telles que l'ID client et le secret client pour son application HubSpot cliente. La AWS Glue console redirigera l'utilisateur vers l' HubSpot endroit où il doit se connecter et autorisera AWS Glue les autorisations demandées pour accéder à son HubSpot instance.
- Les utilisateurs peuvent toujours choisir de créer leur propre application connectée HubSpot et de fournir leur propre identifiant client et leur propre secret client lors de la création de connexions via la AWS Glue console. Dans ce scénario, ils seront toujours redirigés HubSpot vers pour se connecter et autoriser AWS Glue l'accès à leurs ressources.
- Ce type d'autorisation donne lieu à un jeton d'actualisation et à un jeton d'accès. Le jeton d'accès est de courte durée et peut être actualisé automatiquement sans interaction de l'utilisateur à l'aide du jeton d'actualisation.
- Pour consulter HubSpot la documentation publique sur la création d'une application connectée pour le OAuth flux de code d'autorisation, consultez la section Applications publiques.

Pour configurer une HubSpot connexion, procédez comme suit :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes :
   a. Pour l'application connectée gérée par le client, le secret doit contenir le secret du consommateur de l'application connectée avec `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` comme clé.
   b. Remarque : Vous devez créer un secret pour la connexion dans AWS Glue.
2. Dans AWS Glue Glue Studio, créez une connexion sous Data Connections en suivant les étapes ci-dessous :
   a. Lorsque vous sélectionnez un type de connexion, sélectionnez HubSpot.
   b. Fournissez l' HubSpot environnement.
   c. Sélectionnez le rôle AWS IAM qui AWS Glue peut assumer les actions suivantes et qui dispose des autorisations nécessaires pour les actions suivantes :

      JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

   d. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.
   e. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.
3. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.
4. Dans la configuration de votre AWS Glue tâche, fournissez `connectionName` une connexion réseau supplémentaire.

Lecture à partir d' HubSpot entités

Prérequis

Un HubSpot objet que vous aimeriez lire. Vous aurez besoin du nom de l'objet, comme contact ou task. Le tableau suivant présente les entités prises en charge pour la source de synchronisation.

Entités prises en charge pour la source de synchronisation

Entité	Version de l'API	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Campagnes	v1	Non	Oui	Non	Oui	Non
Companies	v3	Oui	Oui	Oui	Oui	Oui
Contacts	v3	Oui	Oui	Oui	Oui	Oui
Contact Lists	v1	Non	Oui	Non	Oui	Non
Deals	v3	Oui	Oui	Oui	Oui	Oui
CRM Pipeline (Deal Pipelines)	v1	Non	Non	Non	Oui	Non
Événement s d'e-mail	v1	Non	Oui	Non	Oui	Non
Appels	v3	Oui	Oui	Oui	Oui	Oui
Remarques	v3	Oui	Oui	Oui	Oui	Oui
Emails	v3	Oui	Oui	Oui	Oui	Oui
Meetings	v3	Oui	Oui	Oui	Oui	Oui

Entité	Version de l'API	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Tâches	v3	Oui	Oui	Oui	Oui	Oui
Postal Mails	v3	Oui	Oui	Oui	Oui	Oui
Custom Objects	v3	Oui	Oui	Oui	Oui	Oui
Forms	v2	Non	Non	Non	Oui	Non
Owners	v3	Non	Oui	Non	Oui	Non
Produits	v3	Oui	Oui	Oui	Oui	Oui
Tickets	v3	Oui	Oui	Oui	Oui	Oui
Flux de travail	v3	Non	Non	Non	Oui	Non
Associations	v4	Oui	Non	Non	Oui	Non
Associations Labels	v4	Non	Non	Non	Oui	Non

Exemple :

```
hubspot_read = glueContext.create_dynamic_frame.from_options(
    connection_type="hubspot",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "contact",
        "API_VERSION": "v3"
    }
```

Entités prises en charge pour la source asynchrone

Entité	Version de l'API	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Companies	v3	Oui	Non	Oui	Oui	Non
Contacts	v3	Oui	Non	Oui	Oui	Non
Deals	v3	Oui	Non	Oui	Oui	Non
Appels	v3	Oui	Non	Oui	Oui	Non
Remarques	v3	Oui	Non	Oui	Oui	Non
Emails	v3	Oui	Non	Oui	Oui	Non
Meetings	v3	Oui	Non	Oui	Oui	Non
Tâches	v3	Oui	Non	Oui	Oui	Non
Postal Mails	v3	Oui	Non	Oui	Oui	Non
Custom Objects	v3	Oui	Non	Oui	Oui	Non
Produits	v3	Oui	Non	Oui	Oui	Non
Tickets	v3	Oui	Non	Oui	Oui	Non

Exemple :

```
hubspot_read = glueContext.create_dynamic_frame.from_options(
    connection_type="hubspot",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "contact",
```

```
        "API_VERSION": "v3",
        "TRANSFER_MODE": "ASYNC"
    }
```

HubSpot détails de l'entité et du champ :

HubSpot API v4 :

Entité	Version de l'API	Champ	Type de données	Opérateurs pris en charge
Associations Label	v4	category	String	N/A
		typeId (IDtype)	Entier	N/A
		étiquette	String	N/A
Associations		from	Struct	N/A
		id	String	"="
		to	List	N/A

> **Note**
>
> Pour l'`Associations`objet, pour récupérer les associations entre deux objets, vous devez fournir le « from Id » (l'ID du premier objet) via un filtre obligatoire lors de la création d'une tâche. AWS Glue Si vous souhaitez récupérer des associations pour plusieurs IDs dans ce cas, vous devez en fournir plusieurs IDs dans la `where` clause. Par exemple : pour récupérer `Associations` les contacts IDs « 1 » et « 151 », vous devez fournir un filtre sous `where id=1 AND id=151` la forme.

HubSpot API version 3 :

Entité	Champ	Type de données	Opérateurs pris en charge
Propriétaire	firstName	String	N/A
	lastName	String	N/A
	createdAt	DateTime	N/A
	archived	Booléen	N/A
	teams	List	N/A
	id	String	N/A
	userId	Entier	N/A
	e-mail	String	N/A
	updatedAt	DateTime	N/A
Flux de travail	name	String	N/A
	id	Entier	N/A
	type	String	N/A
	enabled	Booléen	N/A
	insertedAt	Long	N/A
	updatedAt	Long	N/A
	contactListIds	Struct	N/A
	personaTagIds	List	N/A

Pour les entités suivantes, HubSpot fournit des points de terminaison permettant de récupérer les métadonnées de manière dynamique, afin que le support des opérateurs soit capturé au niveau du type de données pour chaque entité.

> **Note**
>
> `DML_STATUS` est un champ virtuel ajouté à chaque enregistrement lors de l'exécution afin de déterminer son état (CRÉÉ/MIS À JOUR) en mode de synchronisation. L'opérateur `CONTAINS/LIKE` n'est pas pris en charge en mode asynchrone.

Entité	Type de données	Opérateurs pris en charge
Contact	Entier	"=, !=, <, >, >=, <="
	Long	"=, !=, <, >, >=, <="
	String	« =, !=, LIKE »
	Date	N/A
	DateTime	« between »
	Booléen	"="
	List	N/A
	Struct	N/A
Société	Entier	"=, !=, <, >, >=, <="
	Long	"=, !=, <, >, >=, <="
	String	« =, !=, LIKE »
	Date	N/A
	DateTime	« between »
	Booléen	"="
	List	N/A
	Struct	N/A

Entité	Type de données	Opérateurs pris en charge
Deal	Entier	"=, !=, <, >, >=, <="
	Long	"=, !=, <, >, >=, <="
	String	« =, !=, LIKE »
	Date	N/A
	DateTime	« between »
	Booléen	"="
	List	N/A
	Struct	N/A
Ticket	Entier	"=, !=, <, >, >=, <="
	Long	"=, !=, <, >, >=, <="
	String	« =, !=, LIKE »
	Date	N/A
	DateTime	« between »
	Booléen	"="
	List	N/A
	Struct	N/A
Produit (langue française non garantie)	Entier	"=, !=, <, >, >=, <="
	Long	"=, !=, <, >, >=, <="
	String	« =, !=, LIKE »
	Date	N/A

Entité	Type de données	Opérateurs pris en charge
	DateTime	« between »
	Booléen	"="
	List	N/A
	Struct	N/A
Custom Object	Entier	"=, !=, <, >, >=, <="
	Long	"=, !=, <, >, >=, <="
	String	« =, !=, LIKE »
	Date	N/A
	DateTime	« between »
	Booléen	"="
	List	N/A
	Struct	N/A
Appelez	Entier	"=, !=, <, >, >=, <="
	Long	"=, !=, <, >, >=, <="
	String	« =, !=, LIKE »
	Date	N/A
	DateTime	« between »
	Booléen	"="
	List	N/A
	Struct	N/A

Entité	Type de données	Opérateurs pris en charge
E-mail	Entier	"=, !=, <, >, >=, <="
	Long	"=, !=, <, >, >=, <="
	String	« =, !=, LIKE »
	Date	N/A
	DateTime	« between »
	Booléen	"="
	List	N/A
	Struct	N/A
Réunion	Entier	"=, !=, <, >, >=, <="
	Long	"=, !=, <, >, >=, <="
	String	« =, !=, LIKE »
	Date	N/A
	DateTime	« between »
	Booléen	"="
	List	N/A
	Struct	N/A
Remarque	Entier	"=, !=, <, >, >=, <="
	Long	"=, !=, <, >, >=, <="
	String	« =, !=, LIKE »
	Date	N/A

Entité	Type de données	Opérateurs pris en charge
	DateTime	« between »
	Booléen	"="
	List	N/A
	Struct	N/A
Sous-tâche	Entier	"=, !=, <, >, >=, <="
	Long	"=, !=, <, >, >=, <="
	String	« =, !=, LIKE »
	Date	N/A
	DateTime	« between »
	Booléen	"="
	List	N/A
	Struct	N/A
Postal Mail	Entier	"=, !=, <, >, >=, <="
	Long	"=, !=, <, >, >=, <="
	String	« =, !=, LIKE »
	Date	N/A
	DateTime	« between »
	Booléen	"="
	List	N/A
	Struct	N/A

HubSpot API v2 :

Entité	Champ	Type de données	Opérateurs pris en charge
Formulaire	portalId	Entier	N/A
	guid	String	N/A
	name	String	N/A
	méthode	String	N/A
	cssClass	String	N/A
	redirect	String	N/A
	submitText	String	N/A
	notifyRecipients	String	N/A
	createdAt	Long	N/A
	updatedAt	Long	N/A
	ignoreCurrentValues	Booléen	N/A
	deletable	Booléen	N/A
	inlineMessage	Booléen	N/A
	captchaEnabled	Booléen	N/A
	cloneable	Booléen	N/A
	formFieldGroups	List	N/A
	editable	Booléen	N/A
	deletedAt	Entier	N/A
	themeName	String	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	parentId	Entier	N/A
	style	String	N/A
	isPublished	Booléen	N/A
	publishAt	Entier	N/A
	unpublishAt	Entier	N/A
	publishedAt	Entier	N/A
	kickbackEmailWorkflowId	String	N/A
	kickbackEmailsJson	Entier	N/A
	customUid	String	N/A
	createMarketableContact	Booléen	N/A
	editVersion	Entier	N/A
	thankYouMessageJason	String	N/A
	themeColor	String	N/A
	alwaysCreateNewEnterprise	Booléen	N/A
	internalUpdatedAt	Long	N/A
	businessUnitId	Entier	N/A
	portableKey	String	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	paymentSessionTemplateIdentifiants	List	N/A
	selectedExternalOptions	List	N/A

HubSpot API v1 :

Entité	Champ	Type de données	Opérateurs pris en charge
Campaign	id	Entier	N/A
	appId	Entier	N/A
	appName	String	N/A
	lastUpdatedTime	Long	N/A
Contact_List	dynamic	Booléen	N/A
	name	String	N/A
	portalId	Entier	N/A
	createdAt	Long	N/A
	listId	Entier	N/A
	updatedAt	Long	N/A
	ListType	String	N/A
	filtres	List	N/A
	authorId	Entier	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	metaData	Struct	N/A
	archived	Booléen	N/A
	ilsFilterBranch	String	N/A
	filterIds	List	N/A
	limitExempt	Booléen	N/A
	internal	Booléen	N/A
	readOnly	Booléen	N/A
	parentId	Entier	N/A
Email_Event	id	String	N/A
	type	String	N/A
	destinataire	String	N/A
	portalId	Entier	N/A
	appId	Entier	N/A
	appName	String	N/A
	emailCampaignId	Long	N/A
	attempt	Entier	N/A
	créé	Long	N/A
	sentBy	Struct	N/A
	smtpId	String	N/A
	response	String	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	subject	String	N/A
	cc	List	N/A
	bcc	List	N/A
	replyTo	List	N/A
	from	String	N/A
	dropReason	String	N/A
	dropMessage	String	N/A
	browser	Struct	N/A
	userAgent	String	N/A
	duration	Long	N/A
	location	Struct	N/A
	filteredEvent	Booléen	N/A
	deviceType	String	N/A
	suppressedReason	String	N/A
	suppressedMessage	String	N/A
CRM_Pipeline	pipelineId	String	N/A
	createdAt	Long	N/A
	updatedAt	Long	N/A
	objectType	String	N/A
	étiquette	String	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	displayOrder	Entier	N/A
	actif	Booléen	N/A
	étapes	List	N/A
	objectTypeId	String	N/A
	default	Booléen	N/A

Requêtes de partitionnement

Vous pouvez indiquer les options Spark supplémentaires `PARTITION_FIELD`, `LOWER_BOUND`, `UPPER_BOUND` et `NUM_PARTITIONS` si vous souhaitez utiliser la simultanéité dans Spark. Avec ces paramètres, la requête d'origine serait divisée en `NUM_PARTITIONS` nombres de sous-requêtes pouvant être exécutées simultanément par les tâches Spark.

- `PARTITION_FIELD` : le nom du champ à utiliser pour partitionner la requête.
- `LOWER_BOUND` : une valeur limite inférieure inclusive du champ de partition choisi.

  Pour le DateTime champ, nous acceptons la valeur au format ISO.

  Exemples de valeurs valides :

  ```
  "2024-01-01T10:00:00.115Z"
  ```

- `UPPER_BOUND` : une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS` : le nombre de partitions.

Le tableau suivant décrit les détails de prise en charge dans les champs de partitionnement d'entités :

Nom de l'entité	Champs de partitionnement	Type de données
contact	hs_object_id	Long

Nom de l'entité	Champs de partitionnement	Type de données
	createdate, lastmodifieddate	DateTime
company	hs_object_id	Long
	createdate, hs_lastmodifieddate	DateTime
deal	hs_object_id	Long
	createdate, hs_createdate, hs_lastmodifieddate	DateTime
ticket	hs_object_id	Long
	createdate, hs_lastmodifieddate	DateTime
product	hs_object_id	Long
	createdate, hs_lastmodifieddate	DateTime
custom_object	hs_object_id	Long
	createdate, hs_lastmodifieddate	DateTime
appel	hs_object_id	Long
	createdate, hs_lastmodifieddate	DateTime
e-mail	hs_object_id	Long
	createdate, hs_lastmodifieddate	DateTime
meeting	hs_object_id	Long

Nom de l'entité	Champs de partitionnement	Type de données
	createdate, hs_lastmodifieddate	DateTime
note	hs_object_id	Long
	createdate, hs_lastmodifieddate	DateTime
tâche	hs_object_id	Long
	createdate, hs_lastmodifieddate	DateTime
postal_mail	hs_object_id	Long
	createdate, hs_lastmodifieddate	DateTime

Exemple :

```
hubspot_read = glueContext.create_dynamic_frame.from_options(
    connection_type="hubspot",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "company",
        "API_VERSION": "v3",
        "PARTITION_FIELD": "hs_object_id"
        "LOWER_BOUND": "50"
        "UPPER_BOUND": "16726619290"
        "NUM_PARTITIONS": "10"
    }
```

Écrire à des HubSpot entités

Conditions préalables

- Un HubSpot objet sur lequel vous souhaitez écrire. Vous aurez besoin du nom de l'objet, comme contact ou ticket.

- Le HubSpot connecteur prend en charge les opérations d'écriture suivantes :
  - INSERT
  - UPDATE
- Lorsque vous utilisez l'opération d'écriture `UPDATE`, vous devez proposer l'option `ID_FIELD_NAMES` afin de spécifier le champ d'ID pour les enregistrements.

Entités prises en charge pour la destination de la synchronisation

Entité	Version de l'API	Sera pris en charge en tant que connecteur de destination	Peut être inséré	Peut être mis à jour
Companies	v3	Oui	Oui (Unique, en bloc)	Oui (Unique, en bloc)
Contacts	v3	Oui	Oui (Unique, en bloc)	Oui (Unique, en bloc)
Deals	v3	Oui	Oui (Unique, en bloc)	Oui (Unique, en bloc)
Produits	v3	Oui	Oui (Unique, en bloc)	Oui (Unique, en bloc)
Appels	v3	Oui	Oui (Unique, en bloc)	Oui (Unique, en bloc)
Meetings	v3	Oui	Oui (Unique, en bloc)	Oui (Unique, en bloc)
Remarques	v3	Oui	Oui (Unique, en bloc)	Oui (Unique, en bloc)
Emails	v3	Oui	Oui (Unique, en bloc)	Oui (Unique, en bloc)

Entité	Version de l'API	Sera pris en charge en tant que connecteur de destination	Peut être inséré	Peut être mis à jour
Tâches	v3	Oui	Oui (Unique, en bloc)	Oui (Unique, en bloc)
Postal Mails	v3	Oui	Oui (Unique, en bloc)	Oui (Unique, en bloc)
Custom Objects	v3	Oui	Oui (Unique, en bloc)	Oui (Unique, en bloc)
Tickets	v3	Oui	Oui (Unique, en bloc)	Oui (Unique, en bloc)
Associations	v4	Oui	Oui (Unique, en bloc)	Non
Associations Labels	v4	Oui	Oui (Unique, en bloc)	Oui (Unique, en bloc)

Exemples :

Opération INSERT

```
hubspot_write = glueContext.write_dynamic_frame.from_options(
    frame=frameToWrite,
    connection_type="hubspot",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "contact",
        "API_VERSION": "v3",
        "WRITE_OPERATION": "INSERT"
    }
)
```

Opération UPDATE

```
hubspot_write = glueContext.write_dynamic_frame.from_options(
    frame=frameToWrite,
    connection_type="hubspot",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "deal",
        "API_VERSION": "v3",
        "WRITE_OPERATION": "UPDATE",
        "ID_FIELD_NAMES": "hs_object_id"
    }
)
```

HubSpot options de connexion

Les options de connexion suivantes sont disponibles pour HubSpot :

- `ENTITY_NAME`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Le nom de votre objet dans HubSpot.
- `API_VERSION`(Chaîne) - (Obligatoire) Utilisé pour la lecture. HubSpot Version de l'API Rest que vous souhaitez utiliser. Par exemple : v1, v2, v3, v4.
- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.
- `PARTITION_FIELD`(Chaîne) - Utilisé pour la lecture. Champ à utiliser pour partitionner la requête.
- `LOWER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite inférieure inclusive du champ de partition choisi.
- `UPPER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS`(Entier) - Par défaut : 1. Utilisé pour la lecture. Nombre de partitions pour la lecture.
- `TRANSFER_MODE`(Chaîne) - Utilisé pour indiquer si la requête doit être exécutée en mode asynchrone.

- `WRITE_OPERATION`(Chaîne) - Par défaut : INSERT. Utilisé pour l'écriture La valeur doit être INSERT ou UPDATE.
- `ID_FIELD_NAMES`(Chaîne) - Valeur par défaut : null. Nécessaire pour UPDATE.

Limitations et remarques relatives au HubSpot connecteur

Les limitations ou remarques suivantes concernent le HubSpot connecteur :

- Les points de terminaison de recherche sont limités à 10 000 résultats au total pour une requête donnée. Toute partition contenant plus de 10 000 enregistrements provoquera une erreur 400.
- Les autres restrictions importantes du connecteur sont décrites dans Restrictions.
- Un maximum de trois instructions de filtrage sont acceptées par HubSpot.
- Actuellement, HubSpot prend en charge les associations entre HubSpot des objets standard (par exemple, contact, entreprise, offre ou ticket) et des objets personnalisés.
  - Pour un compte gratuit : vous ne pouvez créer que 10 types d'associations entre chaque appariement d'objets (par exemple, contacts et companies).
  - Pour un compte de superadministrateur : vous ne pouvez créer que 50 types d'associations au maximum entre chaque appariement d'objets.
  - Pour plus d'informations, consultez Associations v4 et Create and use association labels.
- Les objets « Quote » et « Communications » ne sont pas présents pour Associations, car ils ne sont actuellement pas pris en charge dans le connecteur.
- Pour le mode asynchrone, SaaS trie les valeurs par ordre croissant uniquement.
- Pour l'entité `Ticket`, SaaS ne renvoie pas le champ `hs_object_id` en mode asynchrone.

Connexion à Instagram Ads

Instagram est une application populaire de partage de photos qui vous permet d'interagir avec des marques, des célébrités, des leaders d'opinion, des amis, des membres de votre famille, etc. Il s'agit d'un service de partage de photos et de réseaux sociaux. Les utilisateurs peuvent prendre des photos ou de courtes vidéos et les partager avec leurs followers. Les publicités Instagram sont des publications pour lesquelles les entreprises peuvent payer afin qu'elles soient diffusées aux utilisateurs de l'application.

Rubriques

- AWS Glue support pour Instagram Ads

- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration d'Instagram Ads
- Configuration des connexions Instagram Ads
- Lecture à partir d'entités Instagram Ads
- Options de connexion Instagram Ads
- Restrictions et remarques relatives au connecteur Instagram Ads

AWS Glue support pour Instagram Ads

AWS Glue prend en charge les publicités Instagram comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser les tâches AWS Glue ETL pour interroger les données d'Instagram Ads.

Prise en charge en tant que cible ?

Non.

Versions de l'API Instagram Ads prises en charge

Les versions suivantes de l'API Instagram Ads sont prises en charge :

- v17.0
- v18.0
- v19.0
- v20.0

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les autorisations AWS IAM requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
```

```
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Si vous ne voulez pas utiliser la méthode ci-dessus, vous pouvez également utiliser les politiques IAM gérées suivantes :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration d'Instagram Ads

Avant de pouvoir AWS Glue transférer des données depuis Instagram Ads, vous devez remplir les conditions suivantes :

Configuration requise

Les exigences minimales sont les suivantes :

- Les comptes Instagram standard sont accessibles indirectement via Facebook.
- L'authentification de l'utilisateur est nécessaire pour générer le jeton d'accès.
- Le connecteur du SDK Instagram Ads implémentera le OAuth flux de jetons d'accès utilisateur.
- Nous utilisons OAuth2 .0 pour authentifier nos demandes d'API auprès d'Instagram Ads. Cette authentification Web s'inscrit dans le cadre de l'architecture d'authentification multifactorielle (MFA), qui est un sur-ensemble de l'authentification à deux facteurs.
- L'utilisateur doit accorder des autorisations pour accéder aux points de terminaison. Pour accéder aux données de l'utilisateur, l'autorisation du point de terminaison est gérée par le biais d'autorisations et de fonctionnalités.

Obtenir des informations d'identification OAuth 2.0

Pour obtenir des informations d'identification d'API afin de pouvoir effectuer des appels authentifiés vers votre instance, consultez API de graphe.

Configuration des connexions Instagram Ads

Instagram Ads prend en charge le type de subvention AUTHORIZATION_CODE pour. OAuth2

- Ce type de subvention est considéré comme tridimensionnel OAuth car il repose sur la redirection des utilisateurs vers le serveur d'autorisation tiers pour authentifier l'utilisateur. Il est utilisé lors de la création de connexions via la AWS Glue console.
- Les utilisateurs peuvent choisir de créer leur propre application connectée dans Instagram Ads et de fournir leur propre identifiant client et leur secret client lors de la création de connexions via la AWS Glue console. Dans ce scénario, ils seront toujours redirigés vers Instagram Ads pour se connecter et autoriser AWS Glue l'accès à leurs ressources.
- Ce type d'autorisation donne lieu à un jeton d'accès. Un jeton d'utilisateur système qui arrive à expiration est valide pendant 60 jours à compter de la date de génération ou d'actualisation. Pour assurer la continuité, le développeur doit actualiser le jeton d'accès dans les 60 jours. À défaut, cela entraîne la perte du jeton d'accès et oblige le développeur à en obtenir un nouveau pour retrouver l'accès à l'API. Consultez Actualiser un token d'accès.

Pour configurer une connexion à Instagram Ads :

1. Dans AWS Glue Glue Studio, créez une connexion sous Data Connections en suivant les étapes ci-dessous :

a. Lorsque vous sélectionnez un type de connexion, sélectionnez Instagram Ads.

b. Sélectionnez le rôle AWS IAM qui AWS Glue peut assumer les actions suivantes et qui dispose des autorisations nécessaires à cet effet :

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

c. Indiquez l'ID client de l'application du client gérée par l'utilisateur.

d. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons. Le secret sélectionné doit comporter une clé `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` dont la valeur est le secret client de l'application connectée.

e. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.

2. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.

Lecture à partir d'entités Instagram Ads

Prérequis

Un objet Instagram Ads à partir duquel vous souhaitez lire. Vous aurez besoin du nom de l'objet. Les tableaux suivants présentent les entités prises en charge.

Entités prises en charge pour la source :

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Campaign	Oui	Oui	Non	Oui	Oui
Ad Set	Oui	Oui	Non	Oui	Oui
Ads	Oui	Oui	Non	Oui	Oui
Ad Creative	Non	Oui	Non	Oui	Non
Insights - Account	Non	Oui	Non	Oui	Non
Ad Image	Oui	Oui	Non	Oui	Non
Insights - Ad	Oui	Oui	Non	Oui	Oui
Perspectives - AdSet	Oui	Oui	Non	Oui	Oui
Insights - Campaign	Oui	Oui	Non	Oui	Oui

Exemple :

```
instagramAds_read = glueContext.create_dynamic_frame.from_options(
    connection_type="instagramads",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "API_VERSION": "v20.0"
    }
```

Détails des entités et des champs Instagram Ads

Pour plus d'informations sur les entités et de détails sur les champs, consultez :

- Campaign

- Ad Set
- Ad
- Ad Creative
- Ad Account Insight
- Ad Image
- Ad Insights
- AdSets Perspectives
- Campaigns Insights

Pour plus d'informations, consultez API Marketing.

> **Note**
>
> Les types de données Struct et List sont convertis en type de données String dans la réponse des connecteurs.

Requêtes de partitionnement

Vous pouvez indiquer les options Spark supplémentaires `PARTITION_FIELD`, `LOWER_BOUND`, `UPPER_BOUND` et `NUM_PARTITIONS` si vous souhaitez utiliser la simultanéité dans Spark. Avec ces paramètres, la requête d'origine serait divisée en `NUM_PARTITIONS` nombres de sous-requêtes pouvant être exécutées simultanément par les tâches Spark.

- `PARTITION_FIELD` : le nom du champ à utiliser pour partitionner la requête.
- `LOWER_BOUND` : une valeur limite inférieure inclusive du champ de partition choisi.

  Pour le DateTime champ, nous acceptons le format d'horodatage Spark utilisé dans les requêtes SQL Spark.

  Exemple de valeur valide :

  ```
  "2022-01-01T00:00:00.000Z"
  ```

- `UPPER_BOUND` : une valeur limite supérieure exclusive du champ de partition choisi.

  Exemple de valeur valide :

```
"2024-01-02T00:00:00.000Z"
```

- NUM_PARTITIONS : le nombre de partitions.

Exemple :

```
instagramADs_read = glueContext.create_dynamic_frame.from_options(
    connection_type="instagramads",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "API_VERSION": "v20.0",
        "PARTITION_FIELD": "created_time"
        "LOWER_BOUND": "2022-01-01T00:00:00.000Z"
        "UPPER_BOUND": "2024-01-02T00:00:00.000Z"
        "NUM_PARTITIONS": "10"
    }
```

Options de connexion Instagram Ads

Les options de connexion pour Instagram Ads sont les suivantes :

- ENTITY_NAME(Chaîne) - (Obligatoire) Utilisé pour la lecture. Le nom de votre objet dans Instagram Ads.
- API_VERSION(Chaîne) - (Obligatoire) Utilisé pour la lecture. Version de l'API de graphe Instagram Ads que vous souhaitez utiliser. Par exemple : v21.
- SELECTED_FIELDS(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- FILTER_PREDICATE(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- QUERY(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.
- PARTITION_FIELD(Chaîne) - Utilisé pour la lecture. Champ à utiliser pour partitionner la requête.
- LOWER_BOUND(Chaîne) - Utilisé pour la lecture. Une valeur limite inférieure inclusive du champ de partition choisi.
- UPPER_BOUND(Chaîne) - Utilisé pour la lecture. Une valeur limite supérieure exclusive du champ de partition choisi.

- `NUM_PARTITIONS`(Entier) - Par défaut : 1. Utilisé pour la lecture. Nombre de partitions pour la lecture.

Restrictions et remarques relatives au connecteur Instagram Ads

Voici les restrictions ou les remarques relatives au connecteur Instagram Ads :

- Le nombre d'appels d'une application est le nombre d'appels qu'un utilisateur peut passer pendant une période d'une heure continue, 200 multiplié par le nombre d'utilisateurs. Pour plus de détails sur les plafonds, consultez Plafonds et Plafonds Business Use Case.

Connexion à Intercom dans AWS Glue Studio

Intercom est le système d'exploitation d'Engagement, un canal ouvert entre votre entreprise et vos clients, sur le plan du produit, sur le moment et selon leurs conditions, qui crée un dialogue permanent qui vous permet de tirer le meilleur parti de chaque engagement tout au long du parcours client.

Rubriques

- AWS Glue support pour Intercom
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration d'Intercom
- Configuration de connexions Intercom
- Lecture à partir d'entités Intercom
- Options de connexion Intercom
- Limitations
- Création d'un compte Intercom et configuration de l'application client

AWS Glue support pour Intercom

AWS Glue prend en charge Intercom comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser les tâches AWS Glue ETL pour interroger les données d'Intercom.

Prise en charge en tant que cible ?

Non.

Versions d'API Intercom prises en charge

v2.5. Pour la prise en charge des entités par version spécifique, consultez Lecture à partir d'entités Intercom.

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les autorisations IAM requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Vous pouvez également utiliser les politiques IAM gérées suivantes pour autoriser l'accès :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM et Amazon EC2. Amazon CloudWatch Logs Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.

- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration d'Intercom

Avant de pouvoir AWS Glue effectuer un transfert depuis Intercom, vous devez satisfaire aux exigences suivantes :

Configuration requise

- Vous disposez d'un compte Intercom. Pour de plus amples informations, veuillez consulter Création d'un compte Intercom et configuration de l'application client.
- Votre compte Intercom est activé pour accéder à l'API.
- Vous devez disposer d'une application sous le compte développeur Intercom qui fournit les informations d'identification du client qui permettent AWS Glue d'accéder à vos données en toute sécurité lorsqu'il passe des appels authentifiés vers votre compte. Pour plus d'informations, consultez Intercom - Étapes de création d'un compte et d'une application client.

Si vous répondez à ces exigences, vous êtes prêt à vous connecter AWS Glue à votre compte Intercom.

Configuration de connexions Intercom

Intercom prend en charge le type de `AUTHORIZATION_CODE` subvention pour OAuth 2.

Ce type de subvention est considéré comme « à trois étapes » OAuth car il repose sur la redirection des utilisateurs vers le serveur d'autorisation tiers pour authentifier l'utilisateur. Il est utilisé lors de la création de connexions via la AWS Glue console. La AWS Glue console redirigera l'utilisateur vers Google Ads où il devra se connecter et accorder AWS Glue les autorisations demandées pour accéder à son instance d'intercom.

Les utilisateurs doivent fournir leur propre identifiant client et leur propre secret client lors de la création de connexions via la AWS Glue console. Dans ce scénario, ils seront toujours redirigés vers Intercom pour se connecter et autoriser l'accès AWS Glue à leurs ressources.

Ce type d'autorisation donne lieu à un jeton d'actualisation et à un jeton d'accès. Le jeton d'accès est de courte durée et peut être actualisé automatiquement sans interaction de l'utilisateur à l'aide du jeton d'actualisation.

Pour plus d'informations sur la création d'une application connectée pour le OAuth flux de code d'autorisation, consultez la section API des annonces.

Pour configurer une connexion Intercom :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes. Il est nécessaire de créer un secret pour chaque connexion dans AWS Glue.
   a. Pour les applications connectées gérées par le client : le secret doit contenir le jeton d'accès à l'application connectée, le jeton d'actualisation, ainsi que les valeurs client_id et client_secret.
2. Dans AWS Glue Glue Studio, créez une connexion sous Data Connections en suivant les étapes ci-dessous :
   a. Lorsque vous sélectionnez un type de connexion, sélectionnez Intercom.
   b. Indiquez l'environnement Intercom.
   c. Sélectionnez le rôle IAM qui AWS Glue peut assumer et dispose des autorisations nécessaires pour les actions suivantes :

      JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

d. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.

e. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.

3. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.

Lecture à partir d'entités Intercom

Conditions préalables

- Un objet Intercom à partir duquel vous souhaitez lire. Reportez-vous au tableau des entités prises en charge ci-dessous pour vérifier les entités disponibles.

Entités prises en charge

Entité	API_Vversion	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partitionnement pris en charge
Admins	v2.5	Non	Non	Non	Oui	Non
Companies	v2.5	Non	Oui	Non	Oui	Non
Conversations	v2.5	Oui	Oui	Oui	Oui	Oui
Attributs des données	v2.5	Non	Non	Non	Oui	Non
Contacts	v2.5	Oui	Oui	Oui	Oui	Oui
Segments	v2.5	Non	Non	Non	Oui	Non
Étiquettes	v2.5	Non	Non	Non	Oui	Non
Équipes	v2.5	Non	Non	Non	Oui	Non

Exemple

```
Intercom_read = glueContext.create_dynamic_frame.from_options(
    connection_type="Intercom",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "company",
        "API_VERSION": "V2.5"
    }
)
```

Détails des entités et des champs Intercom

Entité	Champ	Type de données	Opérateurs pris en charge
Admins	type	String	NA
Admins	id	String	NA
Admins	avatar	Struct	NA
Admins	name	String	NA
Admins	e-mail	String	NA
Admins	away_mode_enabled	Booléen	NA
Admins	away_mode_reassign	Booléen	NA
Admins	has_inbox_seat	Booléen	NA
Admins	teams_ids	List	NA
Admins	job_title	String	NA
Companies	type	String	NA
Companies	id	String	NA
Companies	app_id	String	NA

Entité	Champ	Type de données	Opérateurs pris en charge
Companies	created_at	DateTime	NA
Companies	remote_created_at	DateTime	NA
Companies	updated_at	DateTime	NA
Companies	last_request_at	DateTime	NA
Companies	plan	Struct	NA
Companies	company_id	String	NA
Companies	name	String	NA
Companies	custom_attributes	Struct	NA
Companies	session_count	Entier	NA
Companies	monthly_spend	Entier	NA
Companies	user_count	Entier	NA
Companies	industry	String	NA
Companies	size	Entier	NA
Companies	website	String	NA
Companies	tags	Struct	NA
Companies	segments	Struct	NA
Contacts	id	String	EQUAL_TO. NOT_EQUAL_TO
Contacts	type	String	NA
Contacts	workspace_id	String	NA

Entité	Champ	Type de données	Opérateurs pris en charge
Contacts	external_id	String	CONTAINS, EQUAL_TO, NOT_EQUAL_TO
Contacts	rôle	String	EQUAL_TO. NOT_EQUAL_TO
Contacts	e-mail	String	CONTAINS, EQUAL_TO, NOT_EQUAL_TO
Contacts	phone	String	CONTAINS, EQUAL_TO, NOT_EQUAL_TO
Contacts	name	String	CONTAINS, EQUAL_TO, NOT_EQUAL_TO
Contacts	avatar	String	NA
Contacts	owner_id	Entier	EQUAL_TO, NOT_EQUAL_TO, GREATER_THAN, LESS_THAN
Contacts	social_profiles	Struct	NA
Contacts	has_hard_bounced	Booléen	EQUAL_TO
Contacts	marked_email_as_spam	Booléen	EQUAL_TO
Contacts	unsubscribed_from_emails	Booléen	EQUAL_TO

Entité	Champ	Type de données	Opérateurs pris en charge
Contacts	created_at	DateTime	EQUAL_TO, GREATER_THAN, LESS_THAN
Contacts	updated_at	DateTime	EQUAL_TO, GREATER_THAN, LESS_THAN
Contacts	signed_up_at	DateTime	EQUAL_TO, GREATER_THAN, LESS_THAN
Contacts	last_seen_at	DateTime	EQUAL_TO, GREATER_THAN, LESS_THAN
Contacts	last_replied_at	DateTime	EQUAL_TO, GREATER_THAN, LESS_THAN
Contacts	last_contacted_at	DateTime	EQUAL_TO, GREATER_THAN, LESS_THAN
Contacts	last_email_opened_at	DateTime	EQUAL_TO, GREATER_THAN, LESS_THAN
Contacts	last_email_clicked_at	DateTime	EQUAL_TO, GREATER_THAN, LESS_THAN
Contacts	language_override	String	CONTAINS, EQUAL_TO, NOT_EQUAL_TO

Entité	Champ	Type de données	Opérateurs pris en charge
Contacts	browser	String	CONTAINS, EQUAL_TO, NOT_EQUAL_TO
Contacts	browser_version	String	CONTAINS, EQUAL_TO, NOT_EQUAL_TO
Contacts	browser_language	String	CONTAINS, EQUAL_TO, NOT_EQUAL_TO
Contacts	os	String	CONTAINS, EQUAL_TO, NOT_EQUAL_TO
Contacts	location	Struct	NA
Contacts	location_country	String	CONTAINS, EQUAL_TO, NOT_EQUAL_TO
Contacts	location_region	String	CONTAINS, EQUAL_TO, NOT_EQUAL_TO
Contacts	location_city	String	CONTAINS, EQUAL_TO, NOT_EQUAL_TO
Contacts	android_app_name	String	CONTAINS, EQUAL_TO, NOT_EQUAL_TO
Contacts	android_app_version	String	NA

Entité	Champ	Type de données	Opérateurs pris en charge
Contacts	android_device	String	NA
Contacts	android_os_version	String	NA
Contacts	android_sdk_version	String	CONTAINS, EQUAL_TO, NOT_EQUAL_TO
Contacts	android_last_seen_at	Date	NA
Contacts	ios_app_name	String	CONTAINS, EQUAL_TO, NOT_EQUAL_TO
Contacts	ios_app_version	String	NA
Contacts	ios_device	String	NA
Contacts	ios_os_version	String	CONTAINS, EQUAL_TO, NOT_EQUAL_TO
Contacts	ios_sdk_version	String	CONTAINS, EQUAL_TO, NOT_EQUAL_TO
Contacts	ios_last_seen_at	DateTime	NA
Contacts	custom_attributes	Struct	NA
Contacts	tags	Struct	NA
Contacts	notes	Struct	NA
Contacts	companies	Struct	NA

Entité	Champ	Type de données	Opérateurs pris en charge
Contacts	unsubscribed_from_sms	Booléen	NA
Contacts	sms_consent	Booléen	NA
Contacts	opted_out_subscription_types	Struct	NA
Contacts	referrer	String	NA
Contacts	utm_campaign	String	NA
Contacts	utm_content	String	NA
Contacts	utm_medium	String	NA
Contacts	utm_source	String	NA
Contacts	utm_term	String	NA
Conversations	type	String	NA
Conversations	id	Entier	EQUAL_TO, NOT_EQUAL_TO, GREATER_THAN, LESS_THAN
Conversations	created_at	DateTime	EQUAL_TO, NOT_EQUAL_TO, GREATER_THAN, LESS_THAN
Conversations	updated_at	DateTime	EQUAL_TO, NOT_EQUAL_TO, GREATER_THAN, LESS_THAN

Entité	Champ	Type de données	Opérateurs pris en charge
Conversations	source	Struct	NA
Conversations	source_id	String	EQUAL_TO, NOT_EQUAL_TO
Conversations	source_type	String	EQUAL_TO, NOT_EQUAL_TO,
Conversations	source_delivered_as	String	EQUAL_TO, NOT_EQUAL_TO,
Conversations	source_subject	String	CONTAINS, EQUAL_TO, NOT_EQUAL_TO
Conversations	source_body	String	CONTAINS, EQUAL_TO, NOT_EQUAL_TO
Conversations	source_author_id	String	CONTAINS, EQUAL_TO, NOT_EQUAL_TO
Conversations	source_author_type	String	CONTAINS, EQUAL_TO, NOT_EQUAL_TO
Conversations	source_author_name	String	CONTAINS, EQUAL_TO, NOT_EQUAL_TO
Conversations	source_author_email	String	CONTAINS, EQUAL_TO, NOT_EQUAL_TO

Entité	Champ	Type de données	Opérateurs pris en charge
Conversations	source_url	String	CONTAINS, EQUAL_TO, NOT_EQUAL_TO
Conversations	contacts	Struct	NA
Conversations	teammates	Struct	NA
Conversations	title	String	NA
Conversations	admin_assignee_id	Entier	EQUAL_TO, NOT_EQUAL_TO, GREATER_THAN, LESS_THAN
Conversations	team_assignee_id	Entier	CONTAINS, EQUAL_TO, NOT_EQUAL_TO
Conversations	custom_attributes	Struct	NA
Conversations	ouvrir	Booléen	EQUAL_TO
Conversations	state	String	CONTAINS, EQUAL_TO, NOT_EQUAL_TO
Conversations	lire	Booléen	EQUAL_TO
Conversations	waiting_since	DateTime	EQUAL_TO, NOT_EQUAL_TO, GREATER_THAN, LESS_THAN

Entité	Champ	Type de données	Opérateurs pris en charge
Conversations	snoozed_until	DateTime	EQUAL_TO, NOT_EQUAL_TO, GREATER_THAN, LESS_THAN
Conversations	tags	Struct	NA
Conversations	first_contact_reply	Struct	NA
Conversations	priority	String	EQUAL_TO, NOT_EQUAL_TO
Conversations	topics	Struct	NA
Conversations	sla_applied	Struct	NA
Conversations	conversation_rating	Struct	NA
Conversations	conversation_rating_requested_at	DateTime	EQUAL_TO, NOT_EQUAL_TO, GREATER_THAN, LESS_THAN
Conversations	conversation_rating_replied_at	DateTime	EQUAL_TO, NOT_EQUAL_TO, GREATER_THAN, LESS_THAN
Conversations	conversation_rating_score	Entier	EQUAL_TO, NOT_EQUAL_TO, GREATER_THAN, LESS_THAN
Conversations	conversation_rating_remark	String	CONTAINS, EQUAL_TO, NOT_EQUAL_TO

Entité	Champ	Type de données	Opérateurs pris en charge
Conversations	conversation_rating_contact_id	String	CONTAINS, EQUAL_TO, NOT_EQUAL_TO
Conversations	conversation_rating_admin_id	String	CONTAINS, EQUAL_TO, NOT_EQUAL_TO
Conversations	statistics	Struct	NA
Conversations	statistics_time_to_assignment	Entier	EQUAL_TO, NOT_EQUAL_TO, GREATER_THAN, LESS_THAN
Conversations	statistics_time_to_admin_reply	Entier	EQUAL_TO, NOT_EQUAL_TO, GREATER_THAN, LESS_THAN
Conversations	statistics_time_to_first_close	Entier	EQUAL_TO, NOT_EQUAL_TO, GREATER_THAN, LESS_THAN
Conversations	statistics_time_to_last_close	Entier	EQUAL_TO, NOT_EQUAL_TO, GREATER_THAN, LESS_THAN
Conversations	statistics_median_time_to_reply	Entier	EQUAL_TO, NOT_EQUAL_TO, GREATER_THAN, LESS_THAN

Entité	Champ	Type de données	Opérateurs pris en charge
Conversations	statistics_first_contact_reply_at	DateTime	EQUAL_TO, NOT_EQUAL_TO, GREATER_THAN, LESS_THAN
Conversations	statistics_first_assignment_at	DateTime	EQUAL_TO, NOT_EQUAL_TO, GREATER_THAN, LESS_THAN
Conversations	statistics_first_admin_reply_at	DateTime	EQUAL_TO, NOT_EQUAL_TO, GREATER_THAN, LESS_THAN
Conversations	statistics_first_close_at	DateTime	EQUAL_TO, NOT_EQUAL_TO, GREATER_THAN, LESS_THAN
Conversations	statistics_last_assignment_at	DateTime	EQUAL_TO, NOT_EQUAL_TO, GREATER_THAN, LESS_THAN
Conversations	statistics_last_assignment_admin_reply_at	DateTime	EQUAL_TO, NOT_EQUAL_TO, GREATER_THAN, LESS_THAN
Conversations	statistics_last_contact_reply_at	DateTime	EQUAL_TO, NOT_EQUAL_TO, GREATER_THAN, LESS_THAN

Entité	Champ	Type de données	Opérateurs pris en charge
Conversations	statistics_last_admin_reply_at	DateTime	EQUAL_TO, NOT_EQUAL_TO, GREATER_THAN, LESS_THAN
Conversations	statistics_last_close_at	DateTime	EQUAL_TO, NOT_EQUAL_TO, GREATER_THAN, LESS_THAN
Conversations	statistics_last_closed_by_id	String	CONTAINS, EQUAL_TO, NOT_EQUAL_TO
Conversations	statistics_count_reopens	Entier	EQUAL_TO, NOT_EQUAL_TO, GREATER_THAN, LESS_THAN
Conversations	statistics_count_assignments	Entier	EQUAL_TO, NOT_EQUAL_TO, GREATER_THAN, LESS_THAN
Conversations	statistics_count_conversation_parts	Entier	EQUAL_TO, NOT_EQUAL_TO, GREATER_THAN, LESS_THAN
Conversations	conversation_parts	List	NA
Attributs des données	id	Entier	NA
Attributs des données	type	String	NA
Attributs des données	modèle	String	NA

Entité	Champ	Type de données	Opérateurs pris en charge
Attributs des données	name	String	NA
Attributs des données	full_name	String	NA
Attributs des données	étiquette	String	NA
Attributs des données	description	String	NA
Attributs des données	data_type	String	NA
Attributs des données	options	List	NA
Attributs des données	api_writable	Booléen	NA
Attributs des données	ui_writable	Booléen	NA
Attributs des données	personnalisé	Booléen	NA
Attributs des données	archived	Booléen	NA
Attributs des données	created_at	Booléen	NA
Attributs des données	updated_at	DateTime	NA
Attributs des données	admin_id	String	NA
Segments	type	String	NA
Segments	id	String	NA
Segments	name	String	NA
Segments	created_at	DateTime	NA
Segments	updated_at	DateTime	NA
Segments	person_type	String	NA
Segments	count	Entier	NA

Entité	Champ	Type de données	Opérateurs pris en charge
Étiquettes	type	String	NA
Étiquettes	id	String	NA
Étiquettes	name	String	NA
Équipes	type	String	NA
Équipes	id	String	NA
Équipes	name	String	NA
Équipes	admin_ids	List	NA

Requêtes de partitionnement

Les options Spark supplémentaires `PARTITION_FIELD`, `LOWER_BOUND`, `UPPER_BOUND` et `NUM_PARTITIONS` peuvent être indiquées si vous souhaitez utiliser la simultanéité dans Spark. Avec ces paramètres, la requête d'origine serait divisée en `NUM_PARTITIONS` nombres de sous-requêtes pouvant être exécutées simultanément par les tâches Spark.

- `PARTITION_FIELD` : le nom du champ à utiliser pour partitionner la requête.
- `LOWER_BOUND` : une valeur limite inférieure inclusive du champ de partition choisi.

  Pour le champ Date, nous acceptons le format de date Spark utilisé dans les requêtes SQL Spark. Exemples de valeurs valides : `"2024-02-06"`.
- `UPPER_BOUND` : une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS` : nombre de partitions.

Les détails relatifs à la prise en charge des champs de partitionnement par entité sont présentés dans le tableau suivant.

Nom de l'entité	Champ de partitionnement	Type de données
Contacts	created_at, updated_at,last_seen_at	DateTime
Conversations	id	Entier
Conversations	created_at, updated_at	DateTime

Exemple

```
Intercom_read = glueContext.create_dynamic_frame.from_options(
    connection_type="Intercom",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "conversation",
        "API_VERSION": "V2.5",
        "PARTITION_FIELD": "created_at"
        "LOWER_BOUND": "2022-07-13T07:55:27.065Z"
        "UPPER_BOUND": "2022-08-12T07:55:27.065Z"
        "NUM_PARTITIONS": "2"
    }
)
```

Options de connexion Intercom

Les options de connexion pour Intercom sont les suivantes :

- `ENTITY_NAME`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Le nom de votre objet dans Intercom.
- `API_VERSION`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Version d'API Rest Intercom que vous souhaitez utiliser. Exemple : v2.5.
- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.
- `PARTITION_FIELD`(Chaîne) - Utilisé pour la lecture. Champ à utiliser pour partitionner la requête.

- `LOWER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite inférieure inclusive du champ de partition choisi.
- `UPPER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS`(Entier) - Par défaut : 1. Utilisé pour la lecture. Nombre de partitions pour la lecture.
- `INSTANCE_URL`(Chaîne) - URL de l'instance sur laquelle l'utilisateur souhaite exécuter les opérations. Par exemple : https://api.intercom.io.

Limitations

Les restrictions du connecteur Intercom sont les suivantes :

- Lorsque vous utilisez l'entité Company, il existe une limite de 10 000 entreprises pouvant être renvoyées. Pour plus d'informations, consultez List all companies API.
- Lors de l'application de la clause Order by, le filtre est obligatoire pour les entités Contact et Conversation.
- MCA est pris en charge par le fournisseur de SaaS. Cependant, sur la base des limites de débit des API mentionnées dans la documentation, nous n'hébergerons pas MCA sur d'autres charges de travail AWS Glue car cela pourrait avoir un impact sur d'autres charges de travail et entraîner des problèmes de performances en raison de la contention des ressources.

Création d'un compte Intercom et configuration de l'application client

Création d'un compte Intercom

1. Choisissez l'URL Intercom et choisissez Commencer mon essai gratuit dans le coin supérieur droit de la page.
2. Cliquez sur le bouton Essayer gratuitement dans le coin supérieur droit de la page.
3. Choisissez le type d'activité dont vous avez besoin.
4. Sur la page, saisissez toutes les informations demandées.
5. Après avoir saisi toutes les informations, choisissez Enregistrer.

Création d'une application de développeur Intercom

Pour obtenir l'ID client et le secret client, vous devez créer un compte développeur.

1. Accédez à https://app.intercom.com/.
2. Saisissez l'adresse e-mail et le mot de passe/Ouvrez une session à l'aide de Google et connectez-vous.
3. Choisissez le profil utilisateur dans le coin inférieur gauche et choisissez les paramètres.
4. Choisissez Applications et intégration.
5. Choisissez l'onglet Hub développeur sous Applications et intégration.
6. Choisissez Nouvelle application et créez l'application ici.
7. Indiquez le nom de l'application et choisissez Créer l'application.
8. Dans l'application, accédez à la section Authentification.
9. Choisissez l'édition et ajoutez une redirection URIs. Ajoutez l'URL de redirection spécifique à votre région en tant que `https://<aws-region>.console.aws.amazon.com/gluestudio/oauth`. Par exemple, ajoutez `https://us-east-1.console.aws.amazon.com/gluestudio/oauth for the us-east-1 region`.
10. Obtenez l'ID client et le secret client générés dans la section Informations de base.

Connexion à Jira Cloud

Jira Cloud est une plateforme développée par Atlassian. La plateforme inclut des produits de suivi des problèmes qui aident les équipes à planifier et à suivre leurs projets Agile. En tant qu'utilisateur de Jira Cloud, votre compte contient des données relatives à vos projets, telles que des problèmes, des flux de travail et des événements. Vous pouvez l'utiliser AWS Glue pour transférer vos données Jira Cloud vers certains AWS services ou d'autres applications compatibles.

Rubriques

- AWS Glue prise en charge de Jira Cloud
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration de Jira Cloud
- Configuration des connexions Jira Cloud
- Lecture depuis les entités Jira Cloud

- Options de connexion Jira Cloud
- Restrictions et remarques relatives au connecteur Jira Cloud

AWS Glue prise en charge de Jira Cloud

AWS Glue prend en charge Jira Cloud comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser des tâches AWS Glue ETL pour interroger des données depuis Jira Cloud.

Prise en charge en tant que cible ?

Non.

Versions d'API Jira Cloud prises en charge

Les versions d'API Jira Cloud suivantes sont prises en charge :

- v3

Pour la prise en charge des entités par version spécifique, consultez Lecture depuis les entités Jira Cloud.

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les autorisations AWS IAM requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
```

```
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Si vous ne voulez pas utiliser la méthode ci-dessus, vous pouvez également utiliser les politiques IAM gérées suivantes :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration de Jira Cloud

Avant de pouvoir AWS Glue transférer des données depuis Jira Cloud vers des destinations prises en charge, vous devez satisfaire aux exigences suivantes :

Configuration requise

Les exigences minimales sont les suivantes :

- Vous disposez d'un compte Atlassian dans lequel vous pouvez utiliser le produit logiciel Jira dans Jira Cloud. Pour de plus amples informations, veuillez consulter Création d'un compte Jira Cloud.
- Vous devez avoir créé un AWS compte avec l'accès au service à AWS Glue.
- Cette application fournit les informations d'identification du client qui AWS Glue permettent d'accéder à vos données en toute sécurité lorsqu'elle passe des appels authentifiés vers votre compte. Pour plus d'informations, consultez Enabling OAuth 2.0 (3LO) dans la documentation Atlassian Developer.

Si vous répondez à ces exigences, vous êtes prêt à vous connecter AWS Glue à votre compte Jira Cloud.

Création d'un compte Jira Cloud

Pour créer un compte Jira Cloud :

1. Accédez à URL d'inscription Atlassian.
2. Saisissez votre adresse e-mail professionnelle et votre nom, puis choisissez Accepter. Vous recevez un e-mail de vérification.
3. Après avoir vérifié votre adresse e-mail, vous pouvez créer un mot de passe et choisir S'inscrire.
4. Saisissez votre nom d'utilisateur et votre mot de passe, puis choisissez Se connecter.
5. Vous êtes redirigé vers une page sur laquelle vous devez accéder à votre site. Saisissez le nom du site et choisissez Accepter.

Une fois votre site Atlassian Cloud démarré, vous pouvez configurer votre Jira en répondant à quelques questions selon vos préférences en matière de type de projet.

Pour vous connecter à un compte existant :

1. Accédez à l'URL de connexion Atlassian et saisissez vos informations d'identification.
2. Saisissez votre e-mail et votre mot de passe, puis cliquez sur Se connecter. Vous êtes redirigé vers le tableau de bord Jira.

Création d'une application dans Jira Cloud

Pour créer une application dans Jira Cloud et obtenir l'ID client et le secret client auprès de l'application client gérée :

1. Accédez à l'URL Jira Cloud et saisissez vos informations d'identification.
2. Choisissez Create et sélectionnez l'option d'intégration OAuth 2.0.
3. Saisissez le nom de l'application, consultez les conditions générales et choisissez Créer.
4. Accédez à la section Distribution dans le menu de gauche et choisissez Modifier.
5. Dans la section Modifier les contrôles de distribution :
   a. Pour ÉTAT DE DISTRIBUTION, sélectionnez Partage.
   b. Saisissez le nom du fournisseur.

c. Saisissez l'URL de votre politique de confidentialité. Par exemple, https:// docs.aws.amazon.com/glue/ latest/dg/security -iam-awsmanpol.html

d. Saisissez l'URL de vos conditions d'utilisation (facultatif).

e. Saisissez l'URL de votre contact du service client (facultatif).

f. Sélectionnez dans la DÉCLARATION Yes/No DE DONNÉES PERSONNELLES et choisissez Enregistrer les modifications.

6. Accédez à Autorisations dans le menu de gauche de l'application correspondante.
7. Pour API Jira, choisissez Ajouter. Une fois ajoutée, choisissez l'option Configuration.
8. Dans la section Portées classique > API REST de la plateforme Jira, choisissez Modifier les portées et cochez toutes les portées. Cliquez sur Sauvegarder
9. Sous Portées granulaires, choisissez Modifier les portées et sélectionnez les portées suivantes :
10. Faites défiler la page vers le bas et trouvez des portées. Il existe deux types de portées que vous devez sélectionner sous les rubriques « CRM » et « Standard ».
11. Ajoutez les portées suivantes :

```
read:application-role:jira
read:audit-log:jira
read:avatar:jira
read:field:jira
read:group:jira
read:instance-configuration:jira
read:issue-details:jira
read:issue-event:jira
read:issue-link-type:jira
read:issue-meta:jira
read:issue-security-level:jira
read:issue-security-scheme:jira
read:issue-type-scheme:jira
read:issue-type-screen-scheme:jira
read:issue-type:jira
read:issue.time-tracking:jira
read:label:jira
read:notification-scheme:jira
read:permission:jira
read:priority:jira
read:project:jira
read:project-category:jira
read:project-role:jira
```

```
read:project-type:jira
read:project-version:jira
read:project.component:jir
read:project.property:jira
read:resolution:jira
read:screen:jira
read:status:jira
read:user:jira
read:workflow-scheme:jira
read:workflow:jira
read:field-configuration:jira
read:issue-type-hierarchy:jira
read:webhook:jira
```

12 Accédez à Authentification dans le menu de gauche, puis choisissez Ajouter.

13 Entrez une URL de rappel telle que oauth https://us-east-1.console.aws.amazon.com/gluestudio/

14 Accédez à Paramètres dans le menu de gauche et faites défiler la page vers le bas pour les détails de l'authentification. Notez l'ID client et le secret client.

Configuration des connexions Jira Cloud

Jira Cloud prend en charge le type de subvention AUTHORIZATION_CODE pour. OAuth2

- Ce type de subvention est considéré comme « à trois étapes » OAuth car il repose sur la redirection des utilisateurs vers un serveur d'autorisation tiers pour authentifier l'utilisateur. Il est utilisé lors de la création de connexions via la AWS Glue console. La AWS Glue console redirige l'utilisateur vers Jira Cloud où il doit se connecter et autoriser AWS Glue les autorisations demandées pour accéder à son instance Jira Cloud.
- Les utilisateurs peuvent toujours choisir de créer leur propre application connectée dans Jira Cloud et de fournir leur propre identifiant client et leur secret client lors de la création de connexions via la AWS Glue console. Dans ce scénario, ils seront toujours redirigés vers Jira Cloud pour se connecter et autoriser l'accès AWS Glue à leurs ressources.
- Ce type d'autorisation donne lieu à un jeton d'actualisation et à un jeton d'accès. Le jeton d'accès est de courte durée et peut être actualisé automatiquement sans interaction de l'utilisateur à l'aide du jeton d'actualisation.
- Pour consulter la documentation publique de Jira Cloud sur la création d'une application connectée pour le OAuth flux de code d'autorisation, consultez Enabling OAuth 2.0 (3LO).

Pour configurer une connexion Jira Cloud :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes :
   a. Pour l'application connectée gérée par le client, le secret doit contenir le secret du consommateur de l'application connectée avec `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` comme clé.
   b. Remarque : Vous devez créer un secret pour la connexion dans AWS Glue.
2. Dans AWS Glue Glue Studio, créez une connexion sous Data Connections en suivant les étapes ci-dessous :
   a. Lorsque vous sélectionnez un type de connexion, sélectionnez Jira Cloud.
   b. Indiquez l'environnement Jira Cloud.
   c. Sélectionnez le rôle AWS IAM qui AWS Glue peut assumer et dispose des autorisations nécessaires pour les actions suivantes :

      JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

   d. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.
   e. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.
3. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.

Lecture depuis les entités Jira Cloud

Prérequis

Un objet Jira Cloud à partir duquel vous souhaitez lire. Vous aurez besoin du nom de l'objet, comme Audit Record ou Issue. Le tableau suivant présente les entités prises en charge.

Entités prises en charge pour la source :

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Audit Record	Oui	Oui	Non	Oui	Oui
Problème	Oui	Oui	Non	Oui	Oui
Issue Field	Non	Non	Non	Oui	Non
Issue Field Configuration	Oui	Oui	Non	Oui	Oui
Issue Link Type	Non	Non	Non	Oui	Non
Issue Notification Scheme	Oui	Oui	Non	Oui	Oui
Issue Security Scheme	Non	Non	Non	Oui	Non
Issue Type Scheme	Oui	Oui	Oui	Oui	Oui
Issue Type Screen Scheme	Oui	Oui	Oui	Oui	Oui

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Issue Type	Non	Non	Non	Oui	Non
Jira Setting	Oui	Non	Non	Oui	Non
Jira Setting Advanced	Non	Non	Non	Oui	Non
Jira Setting Global	Non	Non	Non	Oui	Non
Étiquette	Non	Non	Non	Oui	Oui
Myself	Oui	Non	Non	Oui	Non
Autorisations	Non	Non	Non	Oui	Non.
Project	Oui	Oui	Oui	Oui	Oui
Project Category	Non	Non	Non	Oui	Non
Type de projet	Non	Non	Non	Oui	Non
Server Info	Non	Non	Non	Oui	Non
Utilisateurs	Non	Non	Non.	Oui	Non
Flux de travail	Oui	Oui	Oui	Oui	Oui
Workflow Scheme	Non	Oui	Non	Oui	Oui

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Workflow Scheme Project Association	Oui	Non	Non	Oui	Non
Workflow Status	Non	Non	Non	Oui	Non
Workflow Status Category	Non	Non	Non	Oui	Non

Exemple :

```
jiracloud_read = glueContext.create_dynamic_frame.from_options(
    connection_type="JiraCloud",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "audit-record",
        "API_VERSION": "v3"
    }
```

Détails des entités et des champs Jira Cloud :

Objet	Champ	Type de données	Opérateurs de filtre pris en charge
Audit Record	filtre	String	"="
	from	DateTime	"="
	to	DateTime	"="
	id	Entier	N/A

Objet	Champ	Type de données	Opérateurs de filtre pris en charge
	récapitulatif	String	N/A
	remoteAddress	String	N/A
	authorAccountId	String	N/A
	créé	String	N/A
	category	String	N/A
	eventSource	String	N/A
	description	String	N/A
	objectItem	Struct	N/A
	changedValues	List	N/A
	associatedItems	List	N/A
Groupes	groupName	List	"="
	name	String	N/A
	groupId	String	"="
Problème	affectedVersion	String	"=, !="
	assignee	String	"=, !="
	category	String	"=, !="
	component	String	"=, !="
	creator	String	"=, !="
	due	DateTime	N/A
	epic_link	String	"=, !="

Objet	Champ	Type de données	Opérateurs de filtre pris en charge
	filtre	String	"=, !="
	fixVersion	String	"=, !="
	hierarchyLevel	Entier	"=, !="
	issueKey	String	"=, !=, >, <, >=, <="
	issueLink	String	"=, !="
	issueLinkType	String	"=, !="
	labels	String	"=, !="
	lastViewed	DateTime	"=, >, <, >=, <=, between"
	level	String	"=, !="
	parent	String	"=, !="
	priority	String	"=, !="
	project	String	"=, !="
	projectType	String	"=, !="
	reporter	String	"=, !="
	resolution	String	"=, !="
	resolved	DateTime	"=, >, <, >=, <=, between"
	sprint	String	"=, !="
	status	String	"=, !="
	type	String	"=, !="

Objet	Champ	Type de données	Opérateurs de filtre pris en charge
	updated	DateTime	"=, >, <, >=, <=, between"
	voter	String	"=, !="
	votes	Entier	"=, !=, <, >, <=, >=, between"
	watcher	String	"=, !="
	watchers	Entier	"=, !=, <, >, <=, >=, between"
	workRatio	Entier	"=, !=, <, >, <=, >=, between"
	validateQuery	String	"="
	expand	String	"="
	fieldByKeys	Booléen	"="
	id	String	N/A
	self	String	N/A
	key	String	N/A
	renderedFields	Struct	N/A
	propriétés	List	"="
	names	Struct	N/A
	schéma	Struct	N/A
	transitions	List	N/A

Objet	Champ	Type de données	Opérateurs de filtre pris en charge
	opérations	Struct	N/A
	editmeta	Struct	N/A
	changelog	Struct	N/A
	versionedRepresentations	Struct	N/A
	fields	List	"="
	fieldsToInclude	Struct	N/A
	warningMessages	List	N/A
	créé	DateTime	N/A
	worklogDate	DateTime	N/A
IssueEvents	id	Entier	N/A
	name	String	N/A
Issue Fields	id	String	N/A
	key	String	N/A
	name	String	N/A
	personnalisé	Booléen	N/A
	orderable	Booléen	N/A
	navigable	Booléen	N/A
	searchable	Booléen	N/A
	clauseNames	List	N/A

Objet	Champ	Type de données	Opérateurs de filtre pris en charge
	scope	Struct	N/A
	schéma	Struct	N/A
Issue Field Configurations	isDefault	Booléen	"="
	query	String	"="
	id	Entier	"="
	name	String	N/A
	description	String	N/A
Issue Link Type	id	String	N/A
	name	String	N/A
	inward	String	N/A
	outward	String	N/A
	self	String	N/A
Issue Notification Schemes	expand	String	"="
	self	String	N/A
	id	Entier	N/A
	name	String	N/A
	description	String	N/A
	notificationSchemeEvents	List	N/A
	scope	Struct	N/A

Objet	Champ	Type de données	Opérateurs de filtre pris en charge
Issue Priority	self	String	N/A
	statusColor	String	N/A
	description	String	N/A
	iconUrl	String	N/A
	name	String	N/A
	id	String	N/A
	isDefault	Booléen	N/A
Issue Resolutions	self	String	N/A
	id	String	N/A
	description	String	N/A
	name	String	N/A
Issue Security Scheme	self	String	N/A
	id	Entier	N/A
	name	String	N/A
	description	String	N/A
	defaultSecurityLevelId	Entier	N/A
	levels	List	N/A
Issue Type	self	String	N/A
	id	String	N/A
	description	String	N/A

Objet	Champ	Type de données	Opérateurs de filtre pris en charge
	iconUrl	String	N/A
	name	String	N/A
	subtask	Booléen	N/A
	avatarId	Entier	N/A
	entityId	String	N/A
	hierarchyLevel	Entier	N/A
	scope	Struct	N/A
Issue Type Scheme	orderBy	String	"="
	expand	String	"="
	queryString	String	"="
	id	String	N/A
	name	String	N/A
	description	String	N/A
	defaultIssueTypeId	String	N/A
	isDefault	Booléen	N/A
Issue Type Screen Scheme	queryString	String	"="
	orderBy	String	"="
	expand	String	"="
	id	String	"="
	name	String	N/A

Objet	Champ	Type de données	Opérateurs de filtre pris en charge
	description	String	N/A
Jira Settings	key	String	N/A
	keyFilter	String	"="
	id	String	N/A
	value	String	N/A
	name	String	N/A
	desc	String	N/A
	type	String	N/A
	defaultValue	String	N/A
	exemple	String	N/A
	allowedValues	List	N/A
Jira Settings Advanced	id	String	N/A
	key	String	N/A
	value	String	N/A
	name	String	N/A
	desc	String	N/A
	type	String	N/A
	defaultValue	String	N/A
	exemple	String	N/A
	allowedValues	List	N/A

Objet	Champ	Type de données	Opérateurs de filtre pris en charge
Jira Settings Global	votingEnabled	Booléen	N/A
	watchingEnabled	Booléen	N/A
	unassignedIssuesAllowed	Booléen	N/A
	subTasksEnabled	Booléen	N/A
	issueLinkingEnabled	Booléen	N/A
	timeTrackingEnabled	Booléen	N/A
	attachmentsEnabled	Booléen	N/A
	timeTrackingConfiguration	Struct	N/A
Étiquette	values	List	N/A
Myself	expand	String	"="
	self	String	N/A
	accountId	String	N/A
	accountType	String	N/A
	emailAddress	String	N/A
	avatarUrls	String	N/A
	displayName	String	N/A
	actif	Booléen	N/A
	timeZone	String	N/A
	locale	String	N/A

Objet	Champ	Type de données	Opérateurs de filtre pris en charge
	groups	Struct	N/A
	applicationRoles	Struct	N/A
Autorisations	id	String	N/A
	key	String	N/A
	name	String	N/A
	type	String	N/A
	description	String	N/A
	havePermission	Booléen	N/A
	deprecatedKey	Booléen	N/A
Project	orderBy	String	"="
	clés	List	"="
	query	String	"="
	typeKey	String	"="
	categoryId	Entier	"="
	action	String	"="
	expand	String	"="
	status	List	"="
	self	String	N/A
	id	Entier	"="
	key	String	N/A

Objet	Champ	Type de données	Opérateurs de filtre pris en charge
	description	String	N/A
	lead	Struct	N/A
	components	List	N/A
	issueTypes	List	N/A
	url	String	N/A
	e-mail	String	N/A
	assigneeType	String	N/A
	versions	List	N/A
	name	String	N/A
	roles	Struct	N/A
	avatarUrls	Struct	N/A
	projectCategory	Struct	N/A
	projectTypeKey	String	N/A
	simplified	Booléen	N/A
	style	String	N/A
	favourite	Booléen	N/A
	isPrivate	Booléen	N/A
	issueTypeHierarchy	Struct	N/A
	des autorisations	Struct	N/A
	propriétés	List	"="

Objet	Champ	Type de données	Opérateurs de filtre pris en charge
	uuid	String	N/A
	insight	Struct	N/A
	deleted	Booléen	N/A
	retentionTillDate	String	N/A
	deletedDate	String	N/A
	deletedBy	Struct	N/A
	archived	Booléen	N/A
	archivedDate	String	N/A
	archivedBy	Struct	N/A
	landedPageInfo	Struct	N/A
Project Category	self	String	N/A
	id	String	N/A
	name	String	N/A
	description	String	N/A
Type de projet	key	String	N/A
	formattedKey	String	N/A
	description	String	N/A
	descriptionI18nKey	String	N/A
	icon	String	N/A
	color	String	N/A

Objet	Champ	Type de données	Opérateurs de filtre pris en charge
Server Info	baseUrl	String	N/A
	version	String	N/A
	versionNumbers	List	N/A
	deploymentType	String	N/A
	buildNumber	Entier	N/A
	buildDate	DateTime	N/A
	serverTime	DateTime	N/A
	scmInfo	String	N/A
	serverTitle	String	N/A
	healthChecks	List	N/A
Utilisateurs	self	String	N/A
	accountId	String	N/A
	accountType	String	N/A
	emailAddress	String	N/A
	avatarUrls	Struct	N/A
	displayName	String	N/A
	actif	Booléen	N/A
	timeZone	String	N/A
	locale	String	N/A
	groups	Struct	N/A

Objet	Champ	Type de données	Opérateurs de filtre pris en charge
	applicationRoles	Struct	N/A
	expand	String	N/A
Flux de travail	workflowName	String	"="
	expand	String	"="
	queryString	String	"="
	orderBy	String	"="
	isActive	Booléen	"="
	id	Struct	N/A
	description	String	N/A
	transitions	List	N/A
	statuses	List	N/A
	isDefault	Booléen	N/A
	schemes	List	N/A
	projects	List	N/A
	hasDraftWorkflow	Booléen	N/A
	opérations	Struct	N/A
	créé	String	N/A
	updated	String	N/A
Workflow Scheme	self	String	N/A
	id	Entier	N/A

Objet	Champ	Type de données	Opérateurs de filtre pris en charge
	name	String	N/A
	description	String	N/A
	defaultWorkflow	String	N/A
	issueTypeMappings	Struct	N/A
	originalDefaultWorkflow	String	N/A
	originalIssueTypeMappages	Struct	N/A
	draft	Booléen	N/A
	lastModifiedUser	Struct	N/A
	lastModified	String	N/A
	updateDraftIfNécessaire	Booléen	N/A
	issueTypes	Struct	N/A
Workflow Scheme Project Association	projectId	Entier	"="
	projectIds	List	N/A
	workflowScheme	Struct	N/A
Workflow Status	self	String	N/A
	description	String	N/A
	iconUrl	String	N/A
	name	String	N/A

Objet	Champ	Type de données	Opérateurs de filtre pris en charge
	id	String	N/A
	StatusCategory	Struct	N/A
Workflow Status Category	self	String	N/A
	id	String	N/A
	key	String	N/A
	colorName	String	N/A
	name	String	N/A

Requêtes de partitionnement

Vous pouvez indiquer l'option Spark supplémentaire `NUM_PARTITIONS` si vous souhaitez utiliser la simultanéité dans Spark. Avec ce paramètre, la requête d'origine serait divisée en `NUM_PARTITIONS` nombres de sous-requêtes pouvant être exécutées simultanément par les tâches Spark.

- `NUM_PARTITIONS` : le nombre de partitions.

Exemple :

```
jiraCloud_read = glueContext.create_dynamic_frame.from_options(
    connection_type="JiraCloud",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "issue",
        "API_VERSION": "v3",
        "NUM_PARTITIONS": "10"
    }
```

Options de connexion Jira Cloud

Les options de connexion pour Jira Cloud sont les suivantes :

- `ENTITY_NAME`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Le nom de votre objet dans Jira Cloud.
- `API_VERSION`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Version d'API Rest Jira Cloud que vous souhaitez utiliser. Par exemple : v3.
- `DOMAIN_URL`(Chaîne) - (Obligatoire) L'ID Jira Cloud que vous souhaitez utiliser.
- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.
- `NUM_PARTITIONS`(Entier) - Par défaut : 1. Utilisé pour la lecture. Nombre de partitions pour la lecture.

Restrictions et remarques relatives au connecteur Jira Cloud

Voici les restrictions ou les remarques relatives au connecteur Jira Cloud :

- L'`Contains`opérateur ne travaille pas avec le `resourceName` champ, qui est de type `String` donnée.
- Par défaut, si aucun filtre explicite n'est appliqué, seuls les problèmes des 30 derniers jours seront analysés. Les utilisateurs ont la possibilité de remplacer ce filtre par défaut en spécifiant un filtre personnalisé.

Connexion à Kustomer

Kustomer est une puissante plateforme d'expérience client qui réunit tout ce dont vous avez besoin pour mieux servir vos clients dans un seul easy-to-use outil.

Rubriques

- AWS Glue support pour Kustomer
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration de Kustomer
- Configuration de connexions Kustomer
- Lecture à partir d'entités Kustomer

- Options de connexion Kustomer
- Restrictions de Kustomer

AWS Glue support pour Kustomer

AWS Glue prend en charge Kustomer comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser les tâches AWS Glue ETL pour interroger les données de Kustomer.

Prise en charge en tant que cible ?

Non.

Versions de l'API Kustomer prises en charge

Les versions d'API Kustomer suivantes sont prises en charge :

- v1

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les autorisations AWS IAM requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
```

```
      ],
      "Resource": "*"
    }
  ]
}
```

Si vous ne voulez pas utiliser la méthode ci-dessus, vous pouvez également utiliser les politiques IAM gérées suivantes :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration de Kustomer

Avant de pouvoir AWS Glue transférer des données de Kustomer vers des destinations prises en charge, vous devez répondre aux exigences suivantes :

Configuration requise

Les exigences minimales sont les suivantes :

- Vous disposez d'un compte Kustomer qui contient les données que vous souhaitez transférer.
- Dans les paramètres utilisateur de votre compte, vous avez créé une clé d'API. Pour de plus amples informations, veuillez consulter Création d'une clé d'API.
- Vous fournissez la clé d'API AWS Glue lors de la création de la connexion.

Si vous remplissez ces conditions, vous êtes prêt à vous connecter AWS Glue à votre compte Kustomer.

Création d'une clé d'API

Pour créer une clé d'API que vous utiliserez pour créer une connexion pour le connecteur Kustomer dans AWS Glue Studio :

1. Connectez-vous au tableau de bord Kustomer à l'aide de vos informations d'identification.
2. Sélectionnez l'icône Paramètres dans le menu de gauche.
3. Développez le menu déroulant Sécurité et sélectionnez Clés d'API.
4. Sur la page de création d'une clé d'API, sélectionnez Ajouter une clé d'API dans le coin supérieur droit.
5. Renseignez les entrées obligatoires pour la clé d'API en cours de création.
   - Nom : n'importe quel nom à donner à votre clé d'API.
   - Rôles : « org » doit être sélectionné pour que le APIs Kustomer fonctionne.
   - Expire (en jours) : le nombre de jours pendant lesquels vous souhaitez que la clé d'API soit valide. Vous pouvez conserver la valeur N'expire jamais, si cela convient à votre cas d'utilisation.
6. Choisissez Créer.
7. Stockez la valeur de la clé API (jeton) pour une utilisation ultérieure afin de créer une connexion pour le connecteur Kustomer dans AWS Glue Studio.

Configuration de connexions Kustomer

Pour configurer une connexion Kustomer :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes :
   a. Pour l'application connectée gérée par le client, le secret doit contenir le secret du consommateur de l'application connectée avec `apiKey` comme clé.
   b. Remarque : vous devez créer un secret pour vos connexions dans AWS Glue.

1. Dans AWS Glue Glue Studio, créez une connexion sous Data Connections en suivant les étapes ci-dessous :
   a. Sous Connexions, choisissez Créer une connexion.
   b. Lorsque vous sélectionnez une source de données, sélectionnez Kustomer.

c. Sélectionnez le rôle AWS IAM qui AWS Glue peut assumer et dispose des autorisations nécessaires pour les actions suivantes :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

d. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.

e. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.

2. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.

Lecture à partir d'entités Kustomer

Prérequis

Un objet Kustomer à partir duquel vous souhaitez lire. Vous aurez besoin du nom de l'objet, comme Brands ou Cards. Le tableau suivant présente les entités prises en charge.

Entités prises en charge pour la source :

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Brands	Non	Oui	Non	Oui	Non
Cards	Non	Oui	Non	Oui	Non
Chat Settings	Non	Non	Non	Oui	Non
Companies	Oui	Oui	Oui	Oui	Oui
Conversat ions	Oui	Oui	Oui	Oui	Oui
Customers	Oui	Oui	Oui	Oui	Oui
Customer Searches Pinned	Non	Oui	Non	Oui	Non
Customer Searches Position	Non	Non	Non	Oui	Non
Email Hooks	Non	Oui	Non	Oui	Non
Web Hooks	Non	Oui	Non	Oui	Non
KB Articles	Non	Oui	Non	Oui	Non
KB Categorie s	Non	Oui	Non	Oui	Non
KB Forms	Non	Oui	Non	Oui	Non
KB Routes	Non	Oui	Non	Oui	Non
KB Tags	Non	Oui	Non	Oui	Non

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
KB Templates	Non	Oui	Non	Oui	Non
KB Themes	Non	Oui	Non	Oui	Non
Klasses	Non	Oui	Non	Oui	Non
KViews	Non	Oui	Non	Oui	Non
Messages	Oui	Oui	Oui	Oui	Oui
Remarques	Oui	Oui	Oui	Oui	Oui
Notifications	Non	Oui	Non	Oui	Non

Exemple :

```
Kustomer_read = glueContext.create_dynamic_frame.from_options(
    connection_type="kustomer",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "brands",
        "API_VERSION": "v1"
    }
```

Détails des entités et des champs Kustomer

Pour plus d'informations sur les entités et de détails sur les champs, consultez :

- Brands
- Cards
- Chat Settings
- Companies
- Conversations
- Customers

- Customers Searches Pinned
- Customer Searches Positions
- Hooks Email
- Hooks Web
- KB Articles
- KB Categories
- KB Forms
- KB Routes
- KB Tags
- KB Templates
- KB Themes
- Klasses
- KViews
- Messages
- Remarques
- Notifications

API Kustomer v1

Entité	Champ	Type de données	Opérateurs pris en charge
Brands	id	String	N/A
	name	String	N/A
	iconUrl	String	N/A
	createdAt	DateTime	N/A
	updatedAt	DateTime	N/A
	modifiedAt	DateTime	N/A
	default	Booléen	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
Cards	id	String	N/A
	name	String	N/A
	createdAt	DateTime	N/A
	updatedAt	DateTime	N/A
	modifiedAt	DateTime	N/A
	description	String	N/A
	url	String	N/A
	contexts	List	N/A
Chat Settings	id	String	N/A
	settingsVersion	Entier	N/A
	widgetType	String	N/A
	version	Entier	N/A
	teamName	String	N/A
	greeting	String	N/A
	autoreply	String	N/A
	embedIconUrl	String	N/A
	embedIconColor	String	N/A
	fallbackEmailSubject	String	N/A
	fallbackEmailIntroduction	String	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	enabled	Booléen	N/A
	outboundChatEnabled	Booléen	N/A
	updatedAt	DateTime	N/A
	modifiedAt	DateTime	N/A
	offhoursMessage	String	N/A
	offhoursImageUrl	String	N/A
	closableChat	Booléen	N/A
	noHistory	Booléen	N/A
	disableAttachments	Booléen	N/A
	volumeControl	Struct	N/A
	singleSessonChat	Booléen	N/A
	showTypingIndicatorWeb	Booléen	N/A
Companies	id	String	N/A
	name	String	=, !=, CONTAINS
	createdAt	DateTime	N/A
	updatedAt	DateTime	N/A
	modifiedAt	DateTime	=, !=, <, <=, >, >=, BETWEEN
	tags	List	N/A
	domains	List	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	emails	List	N/A
	phones	List	N/A
	whatsapps	List	N/A
	socials	List	N/A
	urls	List	N/A
	locations	List	N/A
	roleGroupVersions	List	N/A
	rev	Entier	N/A
Conversations	id	String	N/A
	name	String	=, !=, CONTAINS
	preview	String	N/A
	canaux	List	N/A
	status	String	=, !=, CONTAINS
	messageCount	Entier	=, !=, >, >=, <, <=
	noteCount	Entier	=, !=, >, >=, <, <=
	satisfaction	Entier	=, !=, >, >=, <, <=
	satisfactionLevel	Struct	N/A
	createdAt	DateTime	=, !=, <, <=, >, >=, BETWEEN
	updatedAt	DateTime	=, !=, <, <=, >, >=, BETWEEN

Entité	Champ	Type de données	Opérateurs pris en charge
	modifiedAt	DateTime	=, !=, <, <=, >, >=, BETWEEN
	lastActivityAt	DateTime	N/A
	spam	Booléen	N/A
	ended	Booléen	=, !=
	endedAt	DateTime	=, !=, <, <=, >, >=, BETWEEN
	endedReason	String	CONTAINS
	endedByType	String	N/A
	importedAt	String	N/A
	tags	List	N/A
	suggestedTags	List	N/A
	sentiment	String	N/A
	prédictions	List	N/A
	suggestedShortcuts	List	N/A
	firstMessageIn	Struct	N/A
	firstMessageOut	Struct	N/A
	lastMessageIn	Struct	N/A
	lastMessageOut	Struct	N/A
	lastMessageAt	DateTime	=, !=, <, <=, >, >=, BETWEEN

Entité	Champ	Type de données	Opérateurs pris en charge
	lastMessageUnrespondedÀ	Struct	N/A
	lastMessageUnrespondedToSinceLastDone	Struct	N/A
	assignedUsers	List	N/A
	assignedTeams	List	N/A
	firstResponse	Struct	N/A
	firstResponseSinceLastDone	Struct	N/A
	lastResponse	Struct	N/A
	firstDone	Struct	N/A
	lastDone	Struct	N/A
	direction	String	=, !=, CONTAINS
	lastMessageDirection	String	N/A
	outboundMessageCount	Entier	N/A
	inboundMessageCount	Entier	N/A
	rev	Entier	N/A
	priority	Entier	=, !=, >, >=, <, <=
	roleGroupVersions	List	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	accessOverride	List	N/A
	assistant	Struct	N/A
	phase	String	N/A
	Skills	List	N/A
	matchedTi meBasedRègles	List	N/A
Customers	id	String	N/A
	name	String	=, !=, CONTAINS
	displayName	String	N/A
	displayColor	String	N/A
	displayIcon	String	N/A
	externalId	String	=, !=, CONTAINS
	externalIds	List	N/A
	sharedExternalIds	List	N/A
	emails	List	N/A
	sharedEmails	List	N/A
	phones	List	N/A
	sharedPhones	List	N/A
	whatsapps	List	N/A
	facebookIds	List	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	instagramIds	List	N/A
	socials	List	N/A
	sharedSocials	List	N/A
	urls	List	N/A
	locations	List	N/A
	activeUsers	List	N/A
	watchers	List	N/A
	recentLocation	Struct	N/A
	locale	String	=, !=, CONTAINS
	timeZone	String	N/A
	gender	String	=, !=, CONTAINS
	createdAt	DateTime	=, !=, <, <=, >, >=, BETWEEN
	updatedAt	DateTime	=, !=, <, <=, >, >=, BETWEEN
	modifiedAt	DateTime	=, !=, <, <=, >, >=, BETWEEN
	lastActivityAt	DateTime	N/A
	deleted	Booléen	N/A
	lastConversation	Struct	N/A
	conversationCounts	Struct	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	preview	Struct	N/A
	tags	List	N/A
	progressiveStatus	String	=, !=, CONTAINS
	verified	Booléen	N/A
	rev	Entier	N/A
	recentItems	List	N/A
	defaultLang	String	=, !=, CONTAINS
	satisfactionLevel	Struct	N/A
	roleGroupVersions	List	N/A
	accessOverride	List	N/A
	companyName	String	N/A
	firstName	String	N/A
	lastName	String	N/A
Customer Searches Pinned	id	String	N/A
	search	String	N/A
	createdAt	DateTime	N/A
Customer Searches Positions	id	String	N/A
	positions	List	N/A
	children	List	N/A
	createdAt	DateTime	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	updatedAt	DateTime	N/A
	modifiedAt	DateTime	N/A
	rev	Entier	N/A
Email Hooks	id	String	N/A
	description	String	N/A
	debug	Booléen	N/A
	e-mail	String	N/A
	eventName	String	N/A
	title	String	N/A
	hachage	String	N/A
	key	String	N/A
	createdAt	DateTime	N/A
	modifiedAt	DateTime	N/A
	updatedAt	DateTime	N/A
Web Hooks	id	String	N/A
	description	String	N/A
	eventName	String	N/A
	hachage	String	N/A
	url	String	N/A
	createdAt	DateTime	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	modifiedAt	DateTime	N/A
	updatedAt	DateTime	N/A
	title	String	N/A
	version	Entier	N/A
	debug	Booléen	N/A
KB Articles	id	String	N/A
	hachage	String	N/A
	title	String	N/A
	source	String	N/A
	status	String	N/A
	scope	String	N/A
	createdAt	DateTime	N/A
	updatedAt	DateTime	N/A
	deleted	Booléen	N/A
	deletedAt	DateTime	N/A
	modifiedAt	DateTime	N/A
	publishedAt	DateTime	N/A
	tags	List	N/A
	categories	List	N/A
	knowledgeBases	List	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	metaTitle	String	N/A
	metaDescription	String	N/A
	metaKeywords	List	N/A
	langVersions	Struct	N/A
	latestLangs	Struct	N/A
KB Categories	id	String	N/A
	hachage	String	N/A
	createdAt	DateTime	N/A
	modifiedAt	DateTime	N/A
	updatedAt	DateTime	N/A
	published	Booléen	N/A
	positions	List	N/A
	categoryPositions	List	N/A
	racine	Booléen	N/A
	langs	Struct	N/A
KB Forms	id	String	N/A
	name	String	N/A
	slug	String	N/A
	hachage	String	N/A
	body	String	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	layout	List	N/A
	layoutV2	List	N/A
	componentsV2	Struct	N/A
	conditions	Struct	N/A
	advanced	Booléen	N/A
	createdAt	DateTime	N/A
	updatedAt	DateTime	N/A
	publishedAt	DateTime	N/A
	modifiedAt	String	N/A
	published	Booléen	N/A
	snippets	List	N/A
	recaptcha	Booléen	N/A
	klass	String	N/A
	channel	String	N/A
	deflection	Booléen	N/A
	formHookEnabled	Booléen	N/A
	replyFrom	String	N/A
	wcag	Booléen	N/A
KB Routes	id	String	N/A
	url	String	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	routableType	String	N/A
	routableId	String	N/A
	createdAt	DateTime	N/A
	updatedAt	DateTime	N/A
	modifiedAt	DateTime	N/A
KB Tags	id	String	N/A
	name	String	N/A
	createdAt	DateTime	N/A
	updatedAt	DateTime	N/A
	modifiedAt	DateTime	N/A
KB Templates	id	String	N/A
	title	String	N/A
	description	String	N/A
	beta	Booléen	N/A
	manifeste	Struct	N/A
	jsxSnippets	List	N/A
	images	List	N/A
	version	String	N/A
	createdAt	DateTime	N/A
	updatedAt	DateTime	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
KB Themes	id	String	N/A
	name	String	N/A
	actif	Booléen	N/A
	default	Booléen	N/A
	lastfileUpdatedAt	DateTime	N/A
	personnalisé	Booléen	N/A
	status	String	N/A
	templateVersionId	String	N/A
	templateTitle	String	N/A
	templateVersion	String	N/A
	manifeste	Struct	N/A
	configSnippets	List	N/A
	jsxSnippets	List	N/A
	createdAt	DateTime	N/A
	updatedAt	DateTime	N/A
	modifiedAt	DateTime	N/A
	rev	Entier	N/A
Klasses	id	String	N/A
	name	String	N/A
	icon	String	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	color	String	N/A
	appDisabled	Booléen	N/A
	status	String	N/A
	updatedAt	DateTime	N/A
	createdAt	DateTime	N/A
	s3 DataUrl	String	N/A
KViews	id	String	N/A
	ressource	String	N/A
	modèle	String	N/A
	contexte	String	N/A
	meta	Struct	N/A
	appDisabled	Booléen	N/A
	enabled	Booléen	N/A
	advanced	Booléen	N/A
	layout	List	N/A
	components	Struct	N/A
	conditions	Struct	N/A
	rev	Entier	N/A
	createdAt	DateTime	N/A
	modifiedAt	DateTime	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	updatedAt	DateTime	N/A
Notifications	id	String	N/A
	name	String	N/A
	status	String	N/A
	event	Struct	N/A
	createdAt	DateTime	N/A
	updatedAt	DateTime	N/A
Messages	id	String	N/A
	externalId	String	N/A
	channel	String	=, !=, CONTAINS
	app	String	N/A
	size	Entier	=, !=, >, >=, <, <=
	direction	String	=, !=, CONTAINS
	preview	String	N/A
	subject	String	N/A
	meta	Struct	N/A
	status	String	=, !=, CONTAINS
	directionType	String	=, !=, CONTAINS
	assignedTeams	List	N/A
	assignedUsers	List	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	errorAt	DateTime	=, !=, <, <=, >, >=, BETWEEN
	auto	Booléen	=, !=
	sentAt	DateTime	=, !=, <, <=, >, >=, BETWEEN
	createdAt	DateTime	=, !=, <, <=, >, >=, BETWEEN
	updatedAt	DateTime	N/A
	modifiedAt	DateTime	N/A
	redacted	Booléen	N/A
	createdByTeams	List	N/A
	rev	Entier	N/A
	reactions	List	N/A
	intentDetections	List	N/A
Remarques	id	String	N/A
	body	String	CONTAINS
	createdAt	DateTime	=, !=, <, <=, >, >=, BETWEEN
	updatedAt	DateTime	=, !=, <, <=, >, >=, BETWEEN
	modifiedAt	DateTime	=, !=, <, <=, >, >=, BETWEEN

Entité	Champ	Type de données	Opérateurs pris en charge
	createdByTeams	List	N/A

Requêtes de partitionnement

Partitionnement basé sur les champs

Vous pouvez indiquer les options Spark supplémentaires `PARTITION_FIELD`, `LOWER_BOUND`, `UPPER_BOUND` et `NUM_PARTITIONS` si vous souhaitez utiliser la simultanéité dans Spark. Avec ces paramètres, la requête d'origine serait divisée en `NUM_PARTITIONS` nombres de sous-requêtes pouvant être exécutées simultanément par les tâches Spark.

- `PARTITION_FIELD` : le nom du champ à utiliser pour partitionner la requête.
- `LOWER_BOUND` : une valeur limite inférieure inclusive du champ de partition choisi.

  Pour le DateTime champ, nous acceptons la valeur au format ISO.

  Exemple de valeur valide :

  ```
  "2023-01-15T11:18:39.205Z"
  ```
- `UPPER_BOUND` : une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS` : le nombre de partitions.

Les détails relatifs à la prise en charge des champs de partitionnement par entité sont présentés dans le tableau suivant :

Nom de l'entité	Champs de partitionnement	Type de données
Companies	modifiedAt	DateTime
Conversations	CreatedAt, UpdatedAt , ModifiedAt, EndedAt, lastMessageAt	DateTime
	messageCount, noteCount	BigInteger

Nom de l'entité	Champs de partitionnement	Type de données
	priority	Entier
Customers	createdAt, updatedAt, modifiedAt	DateTime
Messages	errorAt, sentAt, createdAt	DateTime
	size	BigInteger
Remarques	createdAt, updatedAt, modifiedAt	DateTime

Exemple :

```
Kustomer_read = glueContext.create_dynamic_frame.from_options(
    connection_type="kustomer",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "conversation",
        "API_VERSION": "v1",
        "PARTITION_FIELD": "createdAt"
        "LOWER_BOUND": "2023-01-15T11:18:39.205Z"
        "UPPER_BOUND": "2023-02-15T11:18:39.205Z"
        "NUM_PARTITIONS": "2"
    }
```

Options de connexion Kustomer

Les options de connexion pour Kustomer sont les suivantes :

- `ENTITY_NAME`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Le nom de votre objet dans Kustomer.
- `API_VERSION`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Version d'API Rest Kustomer que vous souhaitez utiliser.
- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.

- QUERY(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.
- PARTITION_FIELD(Chaîne) - Utilisé pour la lecture. Champ à utiliser pour partitionner la requête.
- LOWER_BOUND(Chaîne) - Utilisé pour la lecture. Une valeur limite inférieure inclusive du champ de partition choisi.
- UPPER_BOUND(Chaîne) - Utilisé pour la lecture. Une valeur limite supérieure exclusive du champ de partition choisi.
- NUM_PARTITIONS(Entier) - Par défaut : 1. Utilisé pour la lecture. Nombre de partitions pour la lecture.
- INSTANCE_URL(Chaîne) - (Obligatoire) Utilisé pour la lecture. URL de l'instance Kustomer.

Restrictions de Kustomer

Les restrictions ou remarques suivantes concernent Kustomer :

- L'entité `Customer Searches` n'est pas prise en charge, car la documentation de l'API Kustomer n'a déclaré aucun point de terminaison pour elle.
- La prise en charge de la filtration et du transfert incrémentiel sur l'entité `Klasses` n'est pas prise en charge.
- Order by peut être prise en charge sur plusieurs champs applicables dans une seule demande.

  Cependant, il a été observé sur plusieurs champs que la fonctionnalité Order by se comportait de manière incohérente du côté de SaaS pour certaines combinaisons. Elle est imprévisible, car il peut y avoir « n » combinaisons susceptibles d'afficher des résultats de tri incorrects. Par exemple :

  Pour l'entité `Customers`, Order by `progressiveStatus desc, name asc` ne donne pas le résultat de tri correct. Le tri se fait uniquement en fonction de l'ordre `progressiveStatus`. Si un tel comportement est observé, vous pouvez utiliser un seul champ pour effectuer le tri.
- La fonctionnalité Order by sur le champ « id » n'est prise en charge que par les entités `Conversations` et `Messages` en tant que paramètre de requête. Par exemple : https://api.kustomerapp.com/v1/ des conversations ? sort=desc (Ceci trie les résultats par « id » dans l'ordre décroissant.)

  De plus, tout autre filtre ou commande sur un autre champ est traduit dans un corps de requête POST dont le point de terminaison de l'API est POST https://api.kustomerapp.com/v1/ clients/search. Pour permettre la prise en charge de la commande par « identifiant » dans

`Conversations` et`Messages`, soit seul l'ordre par identifiant doit être présent, soit tout autre filtre and/or commandé par un autre champ applicable.

- Kustomer permet de récupérer un maximum de 10 000 enregistrements, qu'il s'agisse d'une demande filtrée ou non filtrée. En raison de cette restriction, toute entité détenant plus de 10 000 enregistrements subira une perte de données. Il existe deux solutions possibles pour atténuer partiellement ce problème :
  - Appliquez des filtres pour récupérer un ensemble d'enregistrements spécifique.
  - Si un filtre est appliqué à plus de 10 000 enregistrements, appliquez une valeur de filtre successive dans une nouvelle demande ultérieure ou appliquez des plages dans les filtres. Par exemple :

    filterExpression de la première demande : `modifiedAt >= 2022-03-15T05:26:23.000Z and modifiedAt < 2023-03-15T05:26:23.000Z`

    Supposons que la limite de 10 000 enregistrements est dépassée.

    Une autre requête peut être déclenchée avec filterExpression : `modifiedAt >= 2023-03-15T05:26:23.000Z`
- En tant que comportement SaaS, l'opérateur `CONTAINS` de Kustomer prend en charge la mise en correspondance uniquement sur des mots complets et non sur des correspondances partielles au sein d'un mot. Par exemple : « body CONTAINS 'test record' » correspondra à un enregistrement ayant « test » dans le champ « body ». Cependant, « body CONTAINS 'test' » ne correspondra pas à un enregistrement contenant testAnotherRecord « » dans le champ « body ».

Connexion à LinkedIn

LinkedIn est un outil de marketing payant qui permet d'accéder aux réseaux LinkedIn sociaux par le biais de diverses publications sponsorisées et d'autres méthodes. LinkedIn est un puissant outil marketing permettant aux entreprises B2B de créer des prospects, de se faire reconnaître en ligne, de partager du contenu, etc.

Rubriques

- AWS Glue support pour LinkedIn
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration LinkedIn
- Configuration des LinkedIn connexions

- Lecture à partir d' LinkedIn entités
- LinkedIn options de connexion
- Création d'un LinkedIn compte
- Limitations

AWS Glue support pour LinkedIn

AWS Glue prend en charge LinkedIn les éléments suivants :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser des tâches AWS Glue ETL pour interroger des données à partir de LinkedIn.

Prise en charge en tant que cible ?

Non.

Versions LinkedIn d'API prises en charge

202406 (Juin 2024)

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les AWS autorisations requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
```

```
      "Resource": "*"
    }
  ]
}
```

Si vous ne voulez pas utiliser la méthode ci-dessus, vous pouvez également utiliser les politiques IAM gérées suivantes :

- AWSGlueServiceRole — Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess — Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration LinkedIn

Avant de pouvoir AWS Glue utiliser le transfert depuis LinkedIn, vous devez satisfaire aux exigences suivantes :

Configuration requise

- Tu as un LinkedIn compte. Pour plus d'informations sur la création d'un compte, consultez la section Création d'un LinkedIn compte.
- Votre LinkedIn compte est activé pour accéder à l'API.
- Vous avez créé une `OAuth2 API` intégration dans votre LinkedIn compte. Cette intégration fournit les informations d'identification du client qui AWS Glue permettent d'accéder à vos données en toute sécurité lorsqu'il effectue des appels authentifiés vers votre compte. Pour de plus amples informations, veuillez consulter the section called “Création d'un LinkedIn compte”.

Si vous remplissez ces conditions, vous êtes prêt à vous connecter AWS Glue à votre LinkedIn compte. Pour les connexions classiques, vous n'avez rien d'autre à faire dans LinkedIn.

Configuration des LinkedIn connexions

LinkedIn prend en charge le type de `AUTHORIZATION_CODE` subvention pour OAuth2.

Ce type d'autorisation est considéré comme `OAuth` tridimensionnel, car il repose sur la redirection des utilisateurs vers le serveur d'autorisation tiers pour authentifier l'utilisateur. Les utilisateurs peuvent choisir de créer leur propre application connectée LinkedIn et de fournir leur propre identifiant client et leur propre secret client lors de la création de connexions via la AWS Glue console. Dans ce scénario, ils seront toujours redirigés LinkedIn vers pour se connecter et autoriser AWS Glue l'accès à leurs ressources.

Ce type d'autorisation donne lieu à un jeton d'actualisation et à un jeton d'accès. Le jeton d'accès expire 60 jours après sa création. Un nouveau jeton d'accès peut être obtenu à l'aide du jeton d'actualisation.

Pour consulter LinkedIn la documentation publique sur la création d'une application connectée pour `Authorization Code OAuth` Flow, consultez Authorization Code Flow (à 3 étapes). OAuth

Configuration d'une LinkedIn connexion

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes :

   - Pour l'application connectée gérée par le client : le secret doit contenir le secret du consommateur de l'application connectée avec `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` comme clé.
   - Pour l'application connectée AWS gérée : secret vide ou secret avec une valeur temporaire.

   > **Note**
   >
   > Il est indispensable de créer un secret par connexion dans AWS Glue.

2. Dans le AWS Glue Studio, créez une connexion sous Connexions de données en suivant les étapes ci-dessous :

   1. Lorsque vous sélectionnez un type de connexion, sélectionnez LinkedIn.
   2. Fournissez l' LinkedIn environnement.
   3. Sélectionnez le rôle IAM pour lequel AWS Glue vous pouvez assumer et qui dispose des autorisations nécessaires pour effectuer les actions suivantes :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

4. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.
5. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.

3. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.

Lecture à partir d' LinkedIn entités

Conditions préalables

Un LinkedIn objet que vous aimeriez lire. Reportez-vous au tableau des entités prises en charge ci-dessous pour vérifier les entités disponibles.

Entités prises en charge

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Ad Accounts	Oui	Oui	Oui	Oui	Non

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Campagnes	Oui	Oui	Oui	Oui	Non
Campaign Groups	Oui	Oui	Oui	Oui	Non
Creatives	Oui	Oui	Oui	Oui	Non
Ad Analytics	Oui	Non	Non	Oui	Non
Toutes les analyses publicitaires AdAcocunts	Oui	Non	Non	Oui	Non
Ad Analytics All Campaigns	Oui	Non	Non	Oui	Non
Toutes les analyses publicitaires CampaignG roups	Oui	Non	Non	Oui	Non
Toutes les analyses publicitaires AdCreatives	Oui	Non	Non	Oui	Non
Share Statistics	Oui	Non	Non	Oui	Non
Page Statistic s	Oui	Non	Non	Oui	Non

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Follower Statistics	Oui	Non	Non	Oui	Non

Exemple

```
netsuiteerp_read = glueContext.create_dynamic_frame.from_options(
    connection_type="linkedin",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "adaccounts",
        "API_VERSION": "202406"
    }
)
```

LinkedIn détails de l'entité et du champ

Type de données de champs	Opérateurs de filtre pris en charge
String	=
DateTime	BETWEEN, =
Numérique	=
Booléen	=

LinkedIn options de connexion

Les options de connexion suivantes sont disponibles pour LinkedIn :

- `ENTITY_NAME`(Chaîne) - (Obligatoire) Utilisé pour la lecture/l'écriture. Le nom de votre objet dans LinkedIn. Par exemple, adAccounts.
- `API_VERSION`(Chaîne) — (Obligatoire) Utilisé pour la lecture/écriture. LinkedIn Version de l'API Rest que vous souhaitez utiliser. La valeur sera 202406, car seule la version 202406 LinkedIn est actuellement prise en charge.

- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'entité sélectionnée.
- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.

Création d'un LinkedIn compte

Création d'une LinkedIn application et d' OAuth informations d'identification

1. Accédez à la page de votre réseau de LinkedIn développeurs et connectez-vous à l'aide des informations d'identification de votre LinkedIn compte.
2. Accédez à la page Mes applications et choisissez Créer une application pour créer une nouvelle LinkedIn application.
3. Saisissez les informations suivantes dans le formulaire d'inscription à l'application :
   - Nom de l'entreprise : sélectionnez une entreprise existante ou créez-en une.
   - Nom : saisissez le nom de l'application.
   - Description : saisissez la description de l'application.
   - Logo de l'application : sélectionnez un fichier image comme logo de votre application.
   - Utilisation de l'application : sélectionnez l'utilisation de votre application.
   - URL du site Web : saisissez l'URL du site Web qui contient des informations détaillées sur votre application.
   - Adresse e-mail professionnelle : saisissez votre adresse e-mail professionnelle.
   - Numéro de téléphone professionnel : saisissez votre numéro de téléphone professionnel.
   - LinkedIn Conditions d'utilisation de l'API — Lisez et acceptez.
4. Après avoir rempli le formulaire d'inscription à l'application, choisissez Soumettre.

   Vous serez redirigé vers la page Authentification, où les clés d'authentification (ID client et secret client) et d'autres informations pertinentes seront affichées.
5. Si votre application Web nécessite l'accès à l'adresse e-mail de l'utilisateur depuis son LinkedIn compte, sélectionnez l'`r_emailaddress`autorisation. En outre, vous pouvez spécifier la redirection autorisée URLs pour votre LinkedIn application.

Création d'une page dans le LinkedIn compte

1. Accédez aux produits pour LinkedIn développeurs.
2. Dans le coin supérieur droit de la page Produits pour LinkedIn développeurs, sélectionnez Mes applications.
3. Dans le coin supérieur droit de la page Mes applications, sélectionnez Créer une application.
4. Sur la page Créer une application, dans le champ Nom de l'application, saisissez un nom pour votre application.
5. Dans le champ LinkedIn Page, saisissez le nom ou l'URL de la page de votre entreprise.

   > **Note**
   >
   > Si vous n'avez pas de LinkedIn page, vous pouvez en créer une en sélectionnant Créer une nouvelle LinkedIn.

6. Dans le champ URL de la politique de confidentialité, saisissez l'URL de votre politique de confidentialité.
7. Choisissez Charger un logo pour charger une image à afficher aux utilisateurs lorsqu'ils se connectent à votre application.
8. Dans la section Accord juridique, sélectionnez J'ai lu et j'accepte les présentes conditions.
9. Sélectionnez Créer une application.

   Votre application sera créée et sera disponible sous l'onglet Mes applications.

Publier des annonces de campagne dans LinkedIn

1. Connectez-vous à Gestionnaire de campagnes.
2. Sélectionnez un groupe de campagnes existant ou choisissez Créer pour en créer un.
3. Sélectionnez votre objectif.
4. Sélectionnez votre groupe, votre budget et votre calendrier.
5. Développez votre public cible.
6. Sélectionnez le format de votre annonce.
7. Sélectionnez votre budget et votre calendrier.
8. Configurez vos annonces.
9. Procédez à la vérification et au lancement.

Limitations

Pour les champs Analytique `ad_analytics_all_adAccounts`, `ad_analytics_all_campaigns`, `ad_analytics_all_campaign_groups` et `ad_analytics_all_adCreatives`, un filtre est obligatoire pour récupérer les enregistrements.

Connexion à Mailchimp.

Mailchimp est une plateforme all-in-one marketing qui vous aide à gérer et à communiquer avec vos clients, clients et autres parties intéressées. Leur approche du marketing met l'accent sur des pratiques saines de gestion des contacts, des e-mails très bien conçus, des flux de travail automatisés uniques et une analyse de données performante. Si vous utilisez Mailchimp, vous pouvez vous connecter AWS Glue à votre compte Mailchimp. Vous pouvez ensuite utiliser Mailchimp comme source de données dans vos tâches ETL. Exécutez ces tâches pour transférer des données entre Mailchimp et des AWS services ou d'autres applications prises en charge.

Rubriques

- AWS Glue support pour Mailchimp
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration de Mailchimp
- Configuration de connexions Mailchimp
- Lecture à partir d'entités Mailchimp
- Options de connexion Mailchimp
- Création d'un compte Mailchimp
- Limitations

AWS Glue support pour Mailchimp

AWS Glue supporte Mailchimp comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser les tâches AWS Glue ETL pour interroger les données de Mailchimp.

Prise en charge en tant que cible ?

Non.

Versions de l'API Mailchimp prises en charge

3.0

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les AWS autorisations requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Si vous ne voulez pas utiliser la méthode ci-dessus, vous pouvez également utiliser les politiques IAM gérées suivantes :

- AWSGlueServiceRole — Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess — Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la

convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration de Mailchimp

Avant de pouvoir AWS Glue effectuer un transfert depuis Mailchimp, vous devez remplir les conditions suivantes :

Configuration requise

- Vous disposez d'un compte Mailchimp avec adresse e-mail et mot de passe. Pour plus d'informations sur la création d'un compte, consultez Creating a Mailchimp account.
- Vous devez avoir créé AWS un compte avec l'accès au service à AWS Glue.
- Assurez-vous d'avoir créé l'une des ressources suivantes. Ces ressources fournissent des informations d'identification qui AWS Glue permettent d'accéder en toute sécurité à vos données lorsque vous passez des appels authentifiés vers votre compte :
  - Une application pour développeurs qui prend en charge l'authentification OAuth 2.0. Pour plus d'informations sur la création d'une application pour développeur, consultez Creating a Mailchimp account.

Si vous remplissez ces conditions, vous êtes prêt à vous connecter AWS Glue à votre compte Mailchimp. Pour les connexions classiques, vous n'avez rien d'autre à faire dans Mailchimp.

Configuration de connexions Mailchimp

Mailchimp prend en charge les deux types de mécanismes d'authentification suivants :

- Mailchimp prend en charge le type d'autorisation `AUTHORIZATION_CODE`.
  - Ce type d'autorisation est considéré comme `OAuth` tridimensionnel, car il repose sur la redirection des utilisateurs vers le serveur d'autorisation tiers pour authentifier l'utilisateur. Il est utilisé lors de la création de connexions via la console AWS Glue . L'utilisateur qui crée une connexion peut par défaut s'appuyer sur une application appartenant à AWS Glue connectée dans laquelle il n'a pas besoin de fournir d'informations relatives à `OAuth`, à l'exception de son ID client et de son secret client Mailchimp. La AWS Glue console redirige l'utilisateur vers Mailchimp où il doit se connecter et autoriser AWS Glue les autorisations demandées pour accéder à son instance Mailchimp.

  - Les utilisateurs peuvent toujours choisir de créer leur propre application connectée dans Mailchimp et de fournir leur propre ID client et leur propre secret client lors de la création de connexions via la AWS Glue console. Dans ce scénario, ils seront toujours redirigés vers Mailchimp pour se connecter et autoriser l'accès AWS Glue à leurs ressources.
  - Pour consulter la documentation publique de Mailchimp sur la création d'une application connectée pour `AUTHORIZATION_CODE OAuth` Flow, voir Accès aux données pour le compte d'autres utilisateurs avec OAuth 2.
- Authentification personnalisée : pour consulter la documentation publique de Mailchimp sur la génération des clés d'API requises pour une autorisation personnalisée, consultez About API Keys.

Pour configurer une connexion Mailchimp :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes :
   - Authentification `OAuth` : pour l'application connectée gérée par le client, le secret doit contenir le secret du consommateur de l'application connectée avec `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` comme clé.
   - Authentification personnalisée : pour l'application connectée gérée par le client, le secret doit contenir le secret du consommateur de l'application connectée avec « api_key » comme clé.

   > **Note**
   >
   > Il est indispensable de créer un secret par connexion dans AWS Glue.

2. Dans AWS Glue Studio, créez une connexion sous Connexions de données en suivant les étapes ci-dessous :
   a. Sous Connexions, sélectionnez Créer une connexion.
   b. Lorsque vous sélectionnez une source de données, sélectionnez Mailchimp.
   c. Indiquez l'`instanceUrl` Mailchimp.
   d. Sélectionnez le rôle IAM pour lequel AWS Glue vous pouvez assumer et qui dispose des autorisations nécessaires pour effectuer les actions suivantes :

      JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
```

```
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

e. Sélectionnez le type d'authentification pour vous connecter à Mailchimp :

- Pour `0Auth` l'authentification : fournissez l'URL du jeton, l'application cliente gérée par l'utilisateur ClientId du Mailchimp auquel vous souhaitez vous connecter.
- Pour une authentification personnalisée : sélectionnez le type d'authentification CUSTOM pour vous connecter à Mailchimp.

f. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.

g. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.

3. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.
4. Dans la configuration de votre AWS Glue tâche, fournissez `connectionName` une connexion réseau supplémentaire.

Lecture à partir d'entités Mailchimp

Conditions préalables

Un objet Mailchimp à partir duquel vous souhaitez lire. Reportez-vous au tableau des entités prises en charge ci-dessous pour vérifier les entités disponibles.

Entités prises en charge

- Abuse-reports
- Automation
- Campagnes

- Click-details
- Lists
- Membres
- Open-details
- Segments
- Stores
- Unsubscribed

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Automatis ation	Oui	Oui	Oui	Oui	Oui
Campagnes	Non	Non	Non	Non	Non
Lists	Oui	Oui	Non	Oui	Oui
Reports Abuse	Non	Oui	Non	Oui	Oui
Reports Open	Non	Oui	Non	Oui	Oui
Reports Click	Oui	Oui	Non	Oui	Oui
Reports Unsubscribe	Non	Oui	Non	Oui	Oui
Segment	Non	Oui	Non	Oui	Oui
Segment Members	Oui	Oui	Non	Oui	Non
Stores	Oui	Oui	Oui	Oui	Non

Exemple

```
mailchimp_read = glueContext.create_dynamic_frame.from_options(
            connection_type="mailchimp",
            connection_options={

                  "connectionName": "connectionName",

                  "ENTITY_NAME": "stores",
"INSTANCE_URL": "https://us14.api.mailchimp.com",
                  "API_VERSION": "3.0"
               })
```

Détails des entités et des champs Mailchimp

- Abuse-reports
- Automation
- Campagnes
- Click-details
- Lists
- Membres
- Open-details
- Segments
- Stores
- Unsubscribed

Requêtes de partitionnement

Vous pouvez indiquer les options Spark supplémentaires `PARTITION_FIELD`, `LOWER_BOUND`, `UPPER_BOUND` et `NUM_PARTITIONS` si vous souhaitez utiliser la simultanéité dans Spark. Avec ces paramètres, la requête d'origine serait divisée en `NUM_PARTITIONS` nombres de sous-requêtes pouvant être exécutées simultanément par les tâches Spark.

- `PARTITION_FIELD` : le nom du champ à utiliser pour partitionner la requête.
- `LOWER_BOUND` : une valeur limite inférieure inclusive du champ de partition choisi.

  Pour le DateTime champ, nous acceptons la valeur au format ISO.

Exemple de valeur valide :

```
"2024-07-01T00:00:00.000Z"
```

- UPPER_BOUND : une valeur limite supérieure exclusive du champ de partition choisi.
- NUM_PARTITIONS : le nombre de partitions.

Le tableau suivant décrit les détails de prise en charge dans les champs de partitionnement d'entités :

Nom de l'entité	Champs de partitionnement	Type de données

Exemple :

```
read_read = glueContext.create_dynamic_frame.from_options(
    connection_type="mailchimp",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "automations",
        "API_VERSION": "3.0",
        "INSTANCE_URL": "https://us14.api.mailchimp.com",
        "PARTITION_FIELD": "create_time",
        "LOWER_BOUND": "2024-02-05T14:09:30.115Z",
        "UPPER_BOUND": "2024-06-07T13:30:00.134Z",
        "NUM_PARTITIONS": "3"
    }
```

Options de connexion Mailchimp

Les options de connexion pour Mailchimp sont les suivantes :

- ENTITY_NAME(Chaîne) - (Obligatoire) Utilisé pour la lecture/l'écriture. Le nom de votre objet dans Mailchimp.

- `INSTANCE_URL`(Chaîne) - (Obligatoire) Une URL d'instance Mailchimp valide.
- `API_VERSION`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Version d'API Rest de Mailchimp Engage que vous souhaitez utiliser. Par exemple : 3.0.
- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.
- `PARTITION_FIELD`(Chaîne) - Utilisé pour la lecture. Champ à utiliser pour partitionner la requête.
- `LOWER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite inférieure inclusive du champ de partition choisi.
- `UPPER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS`(Entier) - Par défaut : 1. Utilisé pour la lecture. Nombre de partitions pour la lecture.

Création d'un compte Mailchimp

1. Accédez à la page de connexion à Mailchimp, saisissez votre adresse e-mail et votre mot de passe, puis choisissez S'inscrire.
2. Ouvrez l'e-mail de confirmation envoyé par Mailchimp et cliquez sur le lien de confirmation pour vérifier votre compte.

> **Note**
>
> Le délai de réception de l'e-mail d'activation peut varier. Si vous n'avez pas reçu l'e-mail d'activation, consultez votre dossier de courrier indésirable et lisez nos conseils de résolution des problèmes liés à l'e-mail d'activation. Mailchimp bloque les inscriptions provenant d'adresses e-mail basées sur des rôles, telles que admin@pottedplanter.com ou security@example.com.

La première fois que vous vous connectez à votre compte, Mailchimp vous demande les informations requises. Mailchimp utilise ces informations pour s'assurer que votre compte est

conforme à ses Conditions d'utilisation et pour fournir des conseils adaptés à vos besoins et à ceux de votre entreprise.

3. Saisissez vos informations, suivez les instructions pour terminer le processus d'activation et lancez-vous dans votre nouveau compte Mailchimp.

Enregistrement d'une application **`OAuth2.0`**

1. Accédez à la page de connexion à Mailchimp, saisissez votre adresse e-mail et votre mot de passe, puis choisissez Se connecter.
2. Sélectionnez l'icône Utilisateur dans le coin supérieur droit, puis choisissez Compte et facturation dans le menu déroulant.
3. Sélectionnez Extras et choisissez Applications enregistrées dans le menu déroulant.
4. Recherchez et choisissez Enregistrer une application.
5. Entrez les détails suivants :
   - Nom de l'application : nom de l'application.
   - Entreprise/organisation : nom de votre entreprise ou organisation.
   - Site Web de l'application : site Web de l'application.
   - URI de redirection : un modèle d'URI de redirection est un chemin d'URI (ou une liste de chemins séparés par des virgules) vers lequel Mailchimp peut rediriger (si demandé) une fois le flux de connexion terminé. Par exemple, `https://ap-southeast-2\\.console\\.aws\\.amazon\\.com`
6. Choisissez Créer.
7. L'ID client et le secret client seront désormais visibles. Copiez-les et enregistrez-les dans un endroit sûr. Ensuite, choisissez Terminé.

> **Note**
>
> Vos chaînes d'ID client et de secret client sont des informations d'identification utilisées pour établir une connexion avec ce connecteur lors de l'utilisation de AppFlow ou AWS Glue.

Génération d'une clé d'API

1. Accédez à la page de connexion à Mailchimp, saisissez votre adresse e-mail et votre mot de passe, puis choisissez Se connecter.
2. Sélectionnez l'icône Utilisateur dans le coin supérieur droit, puis choisissez Compte et facturation dans le menu déroulant.
3. Sélectionnez Extras et choisissez Clés d'API dans le menu déroulant.
4. Sélectionnez Créer une clé.
5. Saisissez un nom pour la clé, puis choisissez Générer une clé.

   La page suivante affiche la clé d'API générée.
6. Copiez votre clé, stockez-la en toute sécurité et choisissez OK.

Limitations

Les restrictions du connecteur Mailchimp sont les suivantes :

- La filtration n'est prise en charge que par les entités `Campaigns`, `Automations`, `Lists`, `Open Details`, `Members` et `Segments`.
- Lorsque vous utilisez un filtre sur un champ de type de données `DateTime`, vous devez transmettre des valeurs au format suivant : `yyyy-mm-ddThh:MM:ssZ`.

Connexion à Microsoft Dynamics 365 CRM

Microsoft Dynamics 365 est une gamme de produits d'applications métier intelligentes de planification des ressources d'entreprise et de gestion de la relation client.

Rubriques

- AWS Glue support pour Microsoft Dynamics 365
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration de Microsoft Dynamics 365 CRM
- Configuration des connexions Microsoft Dynamics 365 CRM
- Lecture à partir d'entités Microsoft Dynamics 365 CRM
- Référence des options de connexion Microsoft Dynamics 365 CRM

- Limitations

AWS Glue support pour Microsoft Dynamics 365

AWS Glue prend en charge Microsoft Dynamics 365 comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser des tâches AWS Glue ETL pour interroger les données de Microsoft Dynamics 365.

Prise en charge en tant que cible ?

Non.

Versions de l'API Microsoft Dynamics 365 CRM prises en charge

v9.2.

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les AWS autorisations requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
```

```
}
```

Vous pouvez également utiliser les politiques IAM gérées suivantes pour autoriser l'accès :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration de Microsoft Dynamics 365 CRM

Avant de pouvoir AWS Glue transférer des données depuis Microsoft Dynamics 365 CRM, vous devez satisfaire aux exigences suivantes :

Configuration requise

- Vous disposez d'un compte de développeur Microsoft Dynamics 365 CRM avec ClientId et Secret.
- Votre compte Microsoft Dynamics 365 CRM dispose d'un accès à l'API avec une licence valide.

Si vous répondez à ces exigences, vous êtes prêt à vous connecter AWS Glue à votre compte Microsoft Dynamics 365 CRM. Pour les connexions classiques, vous n'avez rien d'autre à faire dans Microsoft Dynamics 365 CRM.

Configuration des connexions Microsoft Dynamics 365 CRM

Type d'autorisation AUTHORIZATION_CODE

- Ce type de subvention est considéré comme « à trois étapes » OAuth car il repose sur la redirection des utilisateurs vers le serveur d'autorisation tiers pour authentifier l'utilisateur. Il est utilisé lors de la création de connexions via la AWS Glue console. La AWS Glue console redirige

l'utilisateur vers Microsoft Dynamics 365 CRM où il doit se connecter et accorder AWS Glue les autorisations demandées pour accéder à son instance Microsoft Dynamics 365 CRM.

- Les utilisateurs peuvent choisir de créer leur propre application connectée dans Microsoft Dynamics 365 CRM et de fournir leur propre identifiant client et leur propre secret client lors de la création de connexions via la AWS Glue console. Dans ce scénario, ils seront toujours redirigés vers Microsoft Dynamics 365 CRM pour se connecter et autoriser AWS Glue l'accès à leurs ressources.
- Ce type d'autorisation donne lieu à un jeton d'actualisation et à un jeton d'accès. Le jeton d'accès est de courte durée et peut être actualisé automatiquement sans interaction de l'utilisateur à l'aide du jeton d'actualisation.
- Pour consulter la documentation publique de Microsoft Dynamics 365 CRM sur la création d'une application connectée pour le OAuth flux de code d'autorisation, voir | Microsoft Learn. Enregistrement des applications Microsoft.

Microsoft Dynamics 365 CRM prend en charge l'authentification OAuth2 2.0.

Pour configurer une connexion Microsoft Dynamics 365 CRM :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes. Il est nécessaire de créer un secret pour chaque connexion dans AWS Glue.
   - Pour le type de AuthorizationCode subvention :

     Pour l'application connectée gérée par le client : le secret doit contenir le secret du client de l'application connectée avec `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` comme clé.
2. Dans AWS Glue Studio, créez une connexion sous Connexions de données en suivant les étapes ci-dessous :
   a. Lorsque vous sélectionnez une source de données, sélectionnez Microsoft Dynamics 365 CRM.
   b. Sélectionnez la valeur INSTANCE_URL de l'instance Microsoft Dynamics 365 CRM à laquelle vous souhaitez vous connecter.
   c. Sélectionnez le rôle IAM qui AWS Glue peut assumer et qui dispose des autorisations nécessaires pour les actions suivantes :

      JSON

```
{
  "Version":"2012-10-17",
```

```
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

d. Sélectionnez URL du jeton et URL du code d'autorisation pour accéder à votre espace de travail Microsoft Dynamics 365 CRM.

e. Fournissez une application client gérée par l'utilisateur ClientId de votre application Microsoft Dynamics 365 CRM.

f. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.

g. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.

3. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`. Choisissez Suivant.
4. Dans la configuration de votre AWS Glue tâche, fournissez `connectionName` une connexion réseau supplémentaire.

Lecture à partir d'entités Microsoft Dynamics 365 CRM

Conditions préalables

- Un objet Microsoft Dynamics 365 CRM à partir duquel vous souhaitez lire. Vous aurez besoin du nom de l'objet, comme contacts ou accounts. Le tableau suivant présente les entités prises en charge.

Entités prises en charge

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Entité dynamique	Oui	Oui	Oui	Oui	Oui

Exemple

```
dynamics365_read = glueContext.create_dynamic_frame.from_options(
    connection_type="microsoftdynamics365crm",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "dynamic_entity",
        "API_VERSION": "v9.2",
        "INSTANCE_URL": "https://{tenantID}.api.crm.dynamics.com"
    }
```

Détails des entités et des champs Microsoft Dynamics 365 CRM

Entités avec métadonnées dynamiques :

Microsoft Dynamics 365 CRM fournit des points de terminaison pour récupérer les métadonnées de manière dynamique. Par conséquent, pour les entités dynamiques, la prise en charge des opérateurs est capturée au niveau du type de données.

Entité	Type de données	Opérateurs pris en charge
Entité dynamique	DateTime	=, <, <=, >, >=, BETWEEN
	Date	=, <, <=, >, >=
	String	=, !=
	Double	=, <, <=, >, >=
	Entier	=, <, <=, >, >=
	Décimal	=, <, <=, >, >=

Entité	Type de données	Opérateurs pris en charge
	Long	=, <, <=, >, >=
	BigInteger	=, <, <=, >, >=
	List	NA
	Struct	NA
	Map	NA

Requêtes de partitionnement

Microsoft Dynamics 365 CRM prend uniquement en charge le partitionnement basé sur les champs.

Les options Spark supplémentaires `PARTITION_FIELD`, `LOWER_BOUND`, `UPPER_BOUND` et `NUM_PARTITIONS` peuvent être fournies si vous souhaitez utiliser la simultanéité dans Spark. Avec ces paramètres, la requête d'origine serait divisée en `NUM_PARTITIONS` nombres de sous-requêtes pouvant être exécutées simultanément par les tâches Spark.

- `PARTITION_FIELD` : le nom du champ à utiliser pour partitionner la requête.
- `LOWER_BOUND` : une valeur limite inférieure inclusive du champ de partition choisi.

  Pour le champ Datetime, nous acceptons le format d'horodatage Spark utilisé dans les requêtes SQL Spark. Exemples de valeurs valides : `"2024-01-30T06:47:51.000Z"`.
- `UPPER_BOUND` : une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS` : nombre de partitions.

Les détails relatifs à la prise en charge des champs de partitionnement par entité sont présentés dans le tableau ci-dessous :

Nom de l'entité	Champs de partitionnement	DataType
Entité dynamique (entité standard)	DateTime Champs dynamique s interrogeables	createdon, modifiedon

Nom de l'entité	Champs de partitionnement	DataType
Entité dynamique (entité personnalisée)	createdon, modifiedon	createdon, modifiedon

Exemple

```
dynamics365_read = glueContext.create_dynamic_frame.from_options(
    connection_type="microsoftdynamics365crm",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "dynamic_entity",
        "API_VERSION": "v9.2",
        "instanceUrl": "https://{tenantID}.api.crm.dynamics.com"
        "PARTITION_FIELD": "createdon"
        "LOWER_BOUND": "2024-01-30T06:47:51.000Z"
        "UPPER_BOUND": "2024-06-30T06:47:51.000Z"
        "NUM_PARTITIONS": "10"
    }
```

Référence des options de connexion Microsoft Dynamics 365 CRM

Les options de connexion pour Microsoft Dynamics 365 CRM sont les suivantes :

- `ENTITY_NAME`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Le nom de votre objet dans Microsoft Dynamics 365 CRM.
- `API_VERSION`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Version de l'API Rest Microsoft Dynamics 365 CRM que vous souhaitez utiliser.
- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.
- `INSTANCE_URL`(Chaîne) - (Obligatoire) Une URL d'instance Microsoft Dynamics 365 CRM valide au format : `https://{tenantID}.api.crm.dynamics.com`
- `NUM_PARTITIONS`(Entier) - Par défaut : 1. Utilisé pour la lecture. Nombre de partitions pour la lecture.

- `PARTITION_FIELD`(Chaîne) - Utilisé pour la lecture. Champ à utiliser pour partitionner la requête.
- `LOWER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite inférieure inclusive du champ de partition choisi. Exemple: `2024-01-30T06:47:51.000Z`.
- `UPPER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite supérieure exclusive du champ de partition choisi. Exemple: `2024-06-30T06:47:51.000Z`.

Limitations

Les restrictions pour Microsoft Dynamics 365 CRM sont les suivantes :

- Dans Microsoft Dynamics 365 CRM, le partitionnement basé sur les enregistrements n'est pas pris en charge, car il ne prend pas en charge de paramètre de décalage.
- La pagination est fixée à un maximum de 500 enregistrements par page afin d'éviter les exceptions liées au serveur interne du SaaS en raison de la combinaison de limites de taille des données et de débit.
  - Documentation SaaS sur la pagination
  - Documentation SaaS sur les limites de débit
- Microsoft Dynamics 365 CRM prend `order by` en charge uniquement sur les champs parent pour toutes les entités. `order by` n'est pas pris en charge dans les sous-champs.
  - Les directions ASC et DESC sont prises en charge.
  - `order by` sur plusieurs champs est pris en charge.
- Le filtrage sur le champ « createddatetime » de l'entité standard `aadusers` génère une erreur de requête incorrecte de la part de SaaS, même s'il prend en charge la filtration. Il n'existe aucune identification spécifique d'une autre entité rencontrant un problème similaire en raison de la nature dynamique des métadonnées et la cause première n'est pas connue non plus. Elle ne peut donc pas être traitée.
- Les types d'objets complexes, tels que Struct, List et Map, ne prennent pas en charge le filtrage.
- Dans de nombreux champs pouvant être extraits d'une réponse, `isRetrievable` est marqué comme `false` dans une réponse de métadonnées dynamiques. Afin d'éviter toute perte de données, `isRetrievable` est défini sur `true` pour tous les champs.
- Le partitionnement basé sur les champs est pris en charge sur toutes les entités s'il répond aux critères suivants :
  - DateTime les champs interrogeables doivent être présents dans les entités standard ou `createdon` les `modifiedon` champs (générés par le système) dans les entités personnalisées.

- Il n'existe aucune identification exclusive des champs générés par le système ou de la propriété nullable à partir des métadonnées SaaS APIs, mais il est courant que seuls les champs disponibles par défaut soient filtrables et non nullables. Par conséquent, le critère de sélection de champ ci-dessus est considéré comme protégé des valeurs Null et s'il est filtrable, il sera éligible au partitionnement.

Connexion à Microsoft Teams

Microsoft Teams est un espace de travail collaboratif au sein de Microsoft 365 qui sert de plateforme centrale pour les conversations sur le lieu de travail, le travail d'équipe collaboratif, les chats vidéo et le partage de documents, le tout conçu pour améliorer la productivité des employés grâce à une suite d'outils unifiée.

Rubriques

- AWS Glue support pour Microsoft Teams
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration de Microsoft Teams
- Configuration des connexions Microsoft Teams
- Lecture à partir d'entités Microsoft Teams
- Référence des options de connexion Microsoft Teams
- Limitations
- Créez un compte Microsoft Teams :

AWS Glue support pour Microsoft Teams

AWS Glue prend en charge Microsoft Teams comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser les tâches AWS Glue ETL pour interroger les données de Microsoft Teams.

Prise en charge en tant que cible ?

Non.

Versions de l'API Microsoft Teams prises en charge

v1. Pour connaître la prise en charge des entités par version, consultez la section Entités prises en charge pour Source.

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les AWS autorisations requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Vous pouvez également utiliser les politiques IAM gérées suivantes pour autoriser l'accès :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs

bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration de Microsoft Teams

Avant de pouvoir AWS Glue transférer des données depuis Microsoft Teams, vous devez satisfaire aux exigences suivantes :

Configuration requise

- Vous disposez d'un compte de développeur Microsoft Teams avec e-mail et mot de passe. Pour de plus amples informations, veuillez consulter Créez un compte Microsoft Teams :.
- Vous devez avoir configuré une OAuth2 application dans votre compte Microsoft qui fournit l'identifiant client et les informations d'identification secrètes qui permettent d' AWS Glue accéder à vos données en toute sécurité lorsqu'elle effectue des appels authentifiés vers votre compte. Pour de plus amples informations, veuillez consulter Créez un compte Microsoft Teams :.

Si vous répondez à ces exigences, vous êtes prêt à vous connecter AWS Glue à votre compte Microsoft Teams. Pour les connexions classiques, vous n'avez rien d'autre à faire dans Microsoft Teams.

Configuration des connexions Microsoft Teams

Microsoft Teams prend en charge les deux types de mécanismes d'authentification suivants :

1. OAuth Auth : Microsoft Teams prend en charge le type de subvention AUTHORIZATION_CODE pour. OAuth2
   - Ce type de subvention est considéré comme « à trois étapes » OAuth car il repose sur la redirection des utilisateurs vers le serveur d'autorisation tiers pour authentifier l'utilisateur. Il est utilisé lors de la création de connexions via la AWS Glue console. L'utilisateur qui crée une connexion peut par défaut s'appuyer sur une application connectée AWS Glue détenue par lui, dans laquelle il n'a pas besoin de fournir d'informations OAuth connexes, à l'exception de l'URL de l'instance Microsoft Teams. La AWS Glue console redirige l'utilisateur vers Microsoft Teams où il doit se connecter et accorder AWS Glue les autorisations demandées pour accéder à son instance Microsoft Teams.
   - Les utilisateurs peuvent choisir de créer leur propre application connectée dans Microsoft Teams et de fournir leur propre identifiant client et leur propre secret client lors de la création

de connexions via la AWS Glue console. Dans ce scénario, ils seront toujours redirigés vers Microsoft Teams pour se connecter et autoriser AWS Glue l'accès à leurs ressources.

- Ce type d'autorisation donne lieu à un jeton d'actualisation et à un jeton d'accès. Le jeton d'accès est actif pendant une heure et pourra être actualisé automatiquement sans interaction de l'utilisateur à l'aide du jeton d'actualisation.
- Pour consulter la documentation publique de Microsoft Teams sur la création d'une application connectée pour le OAuth flux de code d'autorisation, voir | Microsoft Learn. Enregistrez une application auprès de la plateforme d'identité Microsoft : Microsoft Graph.

Pour configurer une connexion Microsoft Teams :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes. Il est nécessaire de créer un secret pour chaque connexion dans AWS Glue.
   a. Pour l' OAuth authentification :
      - Pour l'application connectée gérée par le client : le secret doit contenir le secret du consommateur de l'application connectée avec USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET comme clé.
2. Dans AWS Glue Studio, créez une connexion sous Connexions de données en suivant les étapes ci-dessous :
   a. Sous Connexions de données, choisissez Créer une connexion.
   b. Lorsque vous sélectionnez une source de données, sélectionnez Microsoft Teams.
   c. Indiquez votre identifiant de locataire Microsoft Teams.
   d. Sélectionnez le rôle IAM qui AWS Glue peut assumer les actions suivantes et qui dispose des autorisations nécessaires pour celles-ci :

      JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
```

```
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

e. Fournir une application client gérée par l'utilisateur ClientId de l'application Microsoft Teams.

f. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.

g. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.

3. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`. Choisissez Suivant.
4. Dans la configuration de votre AWS Glue tâche, fournissez `connectionName` une connexion réseau supplémentaire.

Lecture à partir d'entités Microsoft Teams

Conditions préalables

- Un objet Microsoft Teams à partir duquel vous souhaitez lire. Vous aurez besoin du nom de l'objet, comme team ou channel-message. Le tableau suivant présente les entités prises en charge.

Entités prises en charge pour la source

Toutes les entités sont prises en charge par la version 1.0 de l'API.

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Équipes	Non	Non	Non	Oui	Non
Team Members	Oui	Oui	Non	Oui	Oui
Groupes	Oui	Oui	Oui	Oui	Oui

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Group Members	Oui	Oui	Non	Oui	Non
Canaux	Oui	Non	Non	Oui	Oui
Channel Messages	Non	Oui	Non	Oui	Non
Channel Message Replies	Non	Oui	Non	Oui	Non
Channel Tabs	Oui	Non	Non	Oui	Non
Chats	Oui	Oui	Oui	Oui	Oui
Calendar Events	Oui	Oui	Oui	Oui	Oui

Exemple

```
MicrosoftTeams_read = glueContext.create_dynamic_frame.from_options(
    connection_type="MicrosoftTeams",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "company",
        "API_VERSION": "v1.0"
    }
```

Détails des entités et des champs Microsoft Teams

Liste des entités :

- Équipe : https://docs.microsoft.com/en-us/graph/api/user-list-joinedteams ? view=graph-rest-1.0

- Membre de l'équipe : -list-members ? https://docs.microsoft.com/en-us/ graph/api/team view=graph-rest-1.0
- Groupe : https://docs.microsoft.com/en-us/graph/api/group-list ? view=graph-rest-1.0
- Membre du groupe : -list-members ? https://docs.microsoft.com/en-us/ graph/api/group view=graph-rest-1.0
- Chaîne : https://docs.microsoft.com/en-us/graph/api/channel-list ? view=graph-rest-1.0
- Message du canal : -list-messages ? https://docs.microsoft.com/en-us/ graph/api/channel view=graph-rest-1.0
- Channel-Message-Reply : -list-replies ? https://docs.microsoft.com/en-us/ graph/api/chatmessage view=graph-rest-1.0
- Channel-Tab : https://docs.microsoft.com/en-us/ graph/api/channel -list-tabs ? view=graph-rest-1.0
- Chat : https://docs.microsoft.com/en-us/graph/api/chat-liste ? view=graph-rest-1.0
- Calendrier-événement : -liste-événements ? https://docs.microsoft.com/en-us/ graph/api/group view=graph-rest-1.0

Requêtes de partitionnement

Les options Spark supplémentaires `PARTITION_FIELD`, `LOWER_BOUND`, `UPPER_BOUND` et `NUM_PARTITIONS` peuvent être indiquées si vous souhaitez utiliser la simultanéité dans Spark. Avec ces paramètres, la requête d'origine serait divisée en `NUM_PARTITIONS` nombres de sous-requêtes pouvant être exécutées simultanément par les tâches Spark.

- `PARTITION_FIELD` : le nom du champ à utiliser pour partitionner la requête.
- `LOWER_BOUND` : une valeur limite inférieure inclusive du champ de partition choisi.

  Pour le champ Date, nous acceptons le format de date Spark utilisé dans les requêtes SQL Spark. Exemples de valeurs valides : `"2024-02-06"`.
- `UPPER_BOUND` : une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS` : nombre de partitions.

Les détails relatifs à la prise en charge des champs de partitionnement par entité sont présentés dans le tableau ci-dessous :

Nom de l'entité	Champs de partitionnement	Type de données
Team Members	visibleHistoryStartDateTime	DateTime
Groupes	createdDateTime	DateTime
Canaux	createdDateTime	DateTime
Chats	createdDateTime, lastModifiedDate Heure	DateTime
Calendar Events	createdDateTime, lastModifiedDate Heure, Début initial	DateTime

Exemple

```
microsoftteams_read = glueContext.create_dynamic_frame.from_options(
    connection_type="MicrosoftTeams",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "group",
        "API_VERSION": "v1.0",
        "PARTITION_FIELD": "createdDateTime"
        "LOWER_BOUND": "2022-07-13T07:55:27.065Z"
        "UPPER_BOUND": "2022-08-12T07:55:27.065Z"
        "NUM_PARTITIONS": "2"
    }
```

Référence des options de connexion Microsoft Teams

Les options de connexion pour Microsoft Teams sont les suivantes :

- `ENTITY_NAME`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Le nom de votre objet dans Microsoft Teams.
- `API_VERSION`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Version de l'API Rest Microsoft Teams que vous souhaitez utiliser. Exemple : v1.0.
- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.

- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.
- `PARTITION_FIELD`(Chaîne) - Utilisé pour la lecture. Champ à utiliser pour partitionner la requête.
- `LOWER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite inférieure inclusive du champ de partition choisi.
- `UPPER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS`(Entier) - Par défaut : 1. Utilisé pour la lecture. Nombre de partitions pour la lecture.

Limitations

Les restrictions pour le connecteur Microsoft Teams sont les suivantes :

- L'API Microsoft Teams renvoie un nombre d'enregistrements inférieur à celui spécifié pour les entités Chat et Team Member. Ce problème a été signalé au support Microsoft Teams et fait actuellement l'objet d'une enquête.

Créez un compte Microsoft Teams :

1. Accédez à la page d'accueil de Microsoft Teams, https://account.microsoft.com/account/puis choisissez Se connecter.
2. Sélectionnez Créer un compte.
3. Entrez les informations requises pour la création du compte et créez un compte.
4. Accédez au site Web de Microsoft Teams à l'adresse https://www.microsoft.com/en-in/microsoft-teams/log-in.
5. Inscrivez-vous à l'aide du compte Microsoft que vous venez de créer.
6. Une fois votre inscription réussie sur Teams, accédez à https://account.microsft.com/services.
7. Choisissez Essayer Microsoft 365.
8. Activez l'un des abonnements Microsoft 365 ou Microsoft Teams ci-dessous pour accéder à toutes les fonctionnalités requises du connecteur Microsoft Teams :

    - Microsoft Teams Essentials

- Microsoft 365 Business
- Microsoft 365 Business Basic
- Microsoft 365 Business Standard
- Microsoft 365 Business Premium

Créez une application client gérée :

1. Pour créer une application gérée, vous devez enregistrer une nouvelle OAuth application sur Microsoft Entra (anciennement Azure Active Directory) :
2. Connectez-vous au centre d'administration Microsoft Entra.
3. Si vous avez accès à plusieurs locataires, utilisez l'icône Paramètres dans le menu supérieur pour passer au client dans lequel vous souhaitez enregistrer l'application depuis le menu Répertoires + abonnements.
4. Accédez à Identité > Applications > Enregistrements d'applications et sélectionnez Nouvel enregistrement.
5. Saisissez un nom d'affichage pour votre application.
6. Indiquez qui peut utiliser l'application dans la section Types de comptes pris en charge. Pour que cette application soit globale, sélectionnez « Comptes dans n'importe quel répertoire d'organisation » ou « Comptes dans n'importe quel répertoire d'organisation et comptes Microsoft personnels ».
7. Entrez l'URI de redirection `https://{region}.console.aws.amazon.com/appflow/oauth`. Par exemple, pour la `us-west-2 region`, ajoutez `https://us-west-2.console.aws.amazon.com/appflow/oauth`. Vous pouvez en ajouter plusieurs URLs pour les différentes régions que vous souhaitez utiliser.
8. Enregistrez l'application.
9. Notez l'ID client pour un usage ultérieur.
10. Choisissez Ajouter un certificat ou un secret dans la section Essentials.
11. Choisissez Nouveau secret client.
12. Entrez la description et la durée d'expiration.
13. Copiez et enregistrez le secret client pour une utilisation future.
14. Dans la liste du menu de gauche, sélectionnez Autorisations d'API.
15. Choisissez Ajouter une autorisation.
16. Sélectionnez « Microsoft Graph ».

17. Sélectionnez « Autorisations déléguées ».
18. Vérifiez toutes les autorisations suivantes :

    - User.Read
    - Offline_access
    - User.Read.All
    - Utilisateur. ReadWrite.Tout
    - TeamsTab.ReadWriteForTeam
    - TeamsTab.ReadWriteForChat
    - TeamsTab. ReadWrite.Tout
    - TeamsTab. Lisez. Tout
    - TeamSettings. ReadWrite.Tout
    - TeamSettings. Lisez. Tout
    - TeamMember. ReadWrite.Tout
    - TeamMember. Lisez. Tout
    - L'équipe. ReadBasic.Tout
    - GroupMember. ReadWrite.Tout
    - GroupMember. Lisez. Tout
    - Groupe. ReadWrite.Tout
    - Group.Read.All
    - Annuaire. ReadWrite.Tout
    - Directory.Read.All
    - Annuaire. AccessAsUser.Tout
    - Discuter. ReadWrite
    - Discuter. ReadBasic
    - Chat.Read
    - ChannelSettings. ReadWrite.Tout
    - ChannelSettings. Lisez. Tout
    - ChannelMessage. Lisez. Tout
    - Canal. ReadBasic.Tout

19. Choisissez Ajouter des autorisations. Votre application est maintenant correctement configurée. Vous pouvez utiliser l'ID et le secret client pour créer une connexion. Pour plus d'informations, voir https://learn.microsoft.com/en-us/graph/ auth-register-app-v 2.

Connexion à Mixpanel

Mixpanel est une plateforme analytique en temps réel performante qui aide les entreprises à mesurer et à optimiser l'engagement des utilisateurs. Mixpanel est une application utilisée pour suivre le comportement des clients. Cela vous permet de suivre la manière dont les utilisateurs interagissent avec votre produit et d'analyser ces données à l'aide de rapports interactifs qui vous permettent de consulter et de visualiser les résultats en quelques clics. En tant qu'utilisateur de Mixpanel, vous pouvez vous connecter AWS Glue à votre compte Mixpanel. Vous pouvez ensuite utiliser Mixpanel comme source de données dans vos tâches ETL. Exécutez ces tâches pour transférer des données entre Mixpanel et AWS des services ou d'autres applications prises en charge.

Rubriques

- AWS Glue support pour Mixpanel
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration de Mixpanel
- Configuration de connexions Mixpanel
- Lecture à partir d'entités Mixpanel
- Options de connexion Mixpanel
- Création d'un compte Mixpanel et configuration de l'application client
- Limitations

AWS Glue support pour Mixpanel

AWS Glue supporte Mixpanel comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser les tâches AWS Glue ETL pour interroger les données de Mixpanel.

Prise en charge en tant que cible ?

Non.

Versions de l'API Mixpanel prises en charge

2.0

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les AWS autorisations requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Si vous ne voulez pas utiliser la méthode ci-dessus, vous pouvez également utiliser les politiques IAM gérées suivantes :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la

convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration de Mixpanel

Avant de pouvoir AWS Glue effectuer un transfert depuis Mixpanel, vous devez satisfaire aux exigences suivantes :

Configuration requise

- Vous disposez d'un compte Mixpanel. Pour plus d'informations sur la création d'un compte, consultez Creating a Mixpanel account.
- Votre compte Mixpanel est activé pour accéder à l'API. L'accès à l'API est activé par défaut pour les éditions Enterprise, Unlimited, Developer et Performance.

Si vous remplissez ces conditions, vous êtes prêt à vous connecter AWS Glue à votre compte Mixpanel. Pour les connexions classiques, vous n'avez rien d'autre à faire dans Mixpanel.

Configuration de connexions Mixpanel

Mixpanel prend en charge le nom d'utilisateur et le mot de passe pour `BasicAuth`. L'authentification de base est une méthode d'authentification simple dans laquelle les clients fournissent des informations d'identification directement pour accéder aux ressources protégées. AWS Glue est capable d'utiliser le nom d'utilisateur et le mot de passe pour authentifier APIs Mixpanel.

Pour découvrir la documentation publique de Mixpanel sur le flux `BasicAuth`, consultez Mixpanel Service Accounts.

Pour configurer une connexion Mixpanel :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes :
   - Pour l'authentification de base, le secret doit contenir le secret du consommateur de l'application connectée avec `USERNAME` et `PASSWORD` comme clé.

     > **Note**
     >
     > Il est indispensable de créer un secret par connexion dans AWS Glue.

2. Dans le AWS Glue Studio, créez une connexion sous Connexions de données en suivant les étapes ci-dessous :
   a. Lorsque vous sélectionnez un type de connexion, sélectionnez Mixpanel.
   b. Indiquez l'`INSTANCE_URL` de Mixpanel à laquelle vous souhaitez vous connecter.
   c. Sélectionnez le rôle IAM pour lequel AWS Glue vous pouvez assumer et qui dispose des autorisations nécessaires pour les actions suivantes :

      JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

   d. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.
   e. Sélectionnez Options réseau si vous souhaitez utiliser votre réseau.
3. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.

Lecture à partir d'entités Mixpanel

Conditions préalables

Vous devez disposer d'un objet Mixpanel, tel que `Funnels`, `Retention` ou `Retention Funnels`, à partir duquel vous souhaitez lire des données. En outre, vous devez connaître le nom de l'objet.

Entités prises en charge

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Funnels	Oui	Non	Non	Oui	Non
Retention	Oui	Non	Non	Oui	Non
Segmentation	Oui	Non	Non	Oui	Non
Segmentation Sum	Oui	Non	Non	Oui	Non
Segmentation Average	Oui	Non	Non	Oui	Non
Cohorts	Oui	Non	Non	Oui	Non
Engage	Non	Oui	Non	Oui	Non
Événements	Oui	Non	Non	Oui	Non
Events Top	Oui	Non	Non	Oui	Non
Events Names	Oui	Non	Non	Oui	Non
Events Properties	Oui	Non	Non	Oui	Non
Events Properties Top	Oui	Non	Non	Oui	Non
Events Properties Values	Oui	Non	Non	Oui	Non
Annotations	Oui	Non	Non	Oui	Non

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Profile Event Activity	Oui	Non	Non	Oui	Non

Exemple

```
mixpanel_read = glueContext.create_dynamic_frame.from_options(
    connection_type="mixpanel",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "/cohorts/list?project_id=2603353",
        "API_VERSION": "2.0",
        "INSTANCE_URL": "https://www.mixpanel.com/api/app/me"
    }
```

Détails des entités et des champs Mixpanel

Entité	Champ	Type de données	Opérateurs pris en charge
Funnel (Synthèse)	funnel_id	Entier	'='
	workspace_id	Entier	'='
	from_date	Date	'='
	to_date	Date	'=’
	longueur	Entier	'='
	length_unit	String	'='
	interval	Entier	'='
	unité	String	'='

Entité	Champ	Type de données	Opérateurs pris en charge
	limite	Entier	'='
	data	Struct	
	meta	Struct	
Retention	workspace_id	Entier	'='
	unité	String	'='
	addiction_unit	String	'='
	from_date	Date	'='
	to_date	Date	'='
	event	String	'='
	limite	Entier	'='
	data	Struct	
Segmentation	workspace_id	Entier	'='
	event	String	'='
	from_date	Date	'='
	to_date	Date	'='
	unité	String	'='
	interval	Entier	'='
	limite	Entier	'='
	type	String	'='
	series	List	

Entité	Champ	Type de données	Opérateurs pris en charge
	values	Struct	
	data	Struct	
Segmentation Numeric	workspace_id	Entier	'='
	event	String	'='
	on	String	'='
	from_date	Date	'='
	to_date	Date	'='
	unité	String	'='
	type	String	'='
	series	List	
	values	Struct	
Segmentation Sum	workspace_id	Entier	'='
	event	String	'='
	on	String	'='
	from_date	Date	'='
	to_date	Date	'='
	unité	String	'='
	metadata	Struct	
	results	Struct	

Entité	Champ	Type de données	Opérateurs pris en charge
Segmentation Average	workspace_id	Entier	'='
	event	String	'='
	on	String	'='
	from_date	Date	'='
	to_date	Date	'='
	unité	String	'='
	metadata	Struct	
	results	Struct	
Cohorts	count	Entier	
	is_visible	Entier	
	description	String	
	créé	DateTime	
	project_id	Entier	
	id	BigInteger	
	name	String	
	data_group_id	String	
Engage	distinct_id	String	
		propriétés	Struct
Événement	espace de travail	Entier	'='
	event	String	'='

Entité	Champ	Type de données	Opérateurs pris en charge
	type	String	'='
	unité	String	'='
	interval	Entier	'='
	from_date	Date	'='
	to_date	Date	'='
	series	List	
	values	Struct	
Events Top	type	String	'='
	workspace_id	Entier	'='
	limite	Entier	'='
	montant	Entier	
	event	String	
	percent_change	Float	
Event Name	data	List	
	workspace_id	Entier	'='
	type	String	'='
	limite	Entier	'='
Events Properties	workspace_id	Entier	'='
	event	String	'='
	name	String	'='

Entité	Champ	Type de données	Opérateurs pris en charge
	type	String	'='
	unité	String	'='
	interval	Entier	'='
	from_date	Date	'='
	to_date	Date	'='
	limite	Entier	'='
	data	Struct	
	series	List	
	values	Struct	
Event Properties Top	workspace_id	Entier	'='
	event	String	'='
	limite	Entier	'='
	data	Struct	
Event Properties Value	workspace_id	Entier	'='
	event	String	'='
	limite	Entier	'='
	name	String	'='
	data	List	
Annotation	workspace_id	Entier	
	date	DateTime	

Entité	Champ	Type de données	Opérateurs pris en charge
	project_id	Entier	
	id	BigInteger	
	description	String	
	from_date	Date	BETWEEN
Profile Event Activity	workspace_id	Entier	'='
	distinct_ids	String	'='
	from_date	Date	'='
	to_date	Date	'='
	event	String	
	propriétés	Struct	

Options de connexion Mixpanel

Les options de connexion pour Mixpanel sont les suivantes :

- `ENTITY_NAME` (Chaîne) - (Obligatoire) Utilisé pour la lecture/l'écriture. Le nom de votre objet dans Mixpanel.
- `API_VERSION` (Chaîne) - (Obligatoire) Utilisé pour la lecture/l'écriture. Version d'API Rest Mixpanel que vous souhaitez utiliser. Par exemple : v2.0.
- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.

Création d'un compte Mixpanel et configuration de l'application client

Création d'un compte Mixpanel

1. Accédez à la page d'accueil de Mixpanel.
2. Sur la page d'accueil de Mixpanel, choisissez S'inscrire dans le coin supérieur droit de la page.
3. Sur la page Démarrer, effectuez les actions suivantes :

   - Dans la zone prévue à cet effet, saisissez votre adresse e-mail.
   - Cochez la case requise pour accepter les conditions.
   - Choisissez Démarrer pour continuer.

   Une fois l'opération terminée, vous recevrez un e-mail de vérification.
4. Vérifiez la présence d'un message de vérification dans votre boîte de réception, ouvrez l'e-mail et suivez les instructions pour vérifier votre adresse e-mail.
5. Sur la page de vérification, choisissez Vérifier l'adresse e-mail pour terminer la vérification de votre adresse e-mail.
6. Sur la page Nommer votre organisation, saisissez le nom de votre organisation et choisissez Suivant.
7. Sur la page Votre premier projet, entrez les détails de votre projet et choisissez Créer.
8. Sur la page suivante, choisissez Démarrer pour terminer la création de votre compte.

Connexion à un compte Mixpanel

1. Accédez à la page de connexion de Mixpanel.
2. Entrez votre adresse e-mail, puis choisissez Continuer.
3. Vérifiez la présence d'un message de vérification dans votre boîte de réception, ouvrez l'e-mail et suivez les instructions pour vérifier votre adresse e-mail.
4. Sur la page suivante, cliquez sur le bouton Se connecter pour vous connecter à votre compte.

Achat d'une offre Mixpanel

1. Sur la page Mixpanel, sélectionnez l'icône Paramètres dans l'angle supérieur droit.
2. Dans la liste des options, sélectionnez Détails de l'offre et facturation.
3. Sur la page Détails de l'offre et facturation, sélectionnez Mettre à niveau ou Modifier.

4. Sur la page suivante, sélectionnez l'offre que vous souhaitez acheter.

   Ceci termine le processus de création de compte et d'achat de l'offre.

Création d'un nom d'utilisateur et d'un secret client (pour enregistrer votre application)

1. Sur la page Mixpanel, sélectionnez l'icône Paramètres dans le coin supérieur droit.
2. Dans la liste des options, sélectionnez Paramètres du projet.
3. Sur la page Paramètres du projet, sélectionnez Comptes de service, puis Ajouter un compte de service.
4. Dans la liste déroulante Comptes de service, sélectionnez le compte de service ou entrez le nom à créer, ajoutez le rôle du projet, spécifiez les dates d'expiration et sélectionnez Ajouter.

> **Important**
>
> Une fois l'étape précédente terminée, la page suivante affiche la clé secrète du compte de service. Assurez-vous de sauvegarder la clé secrète du compte de service. Passé ce point, vous ne pourrez pas y accéder à nouveau.

Limitations

Les restrictions du connecteur Mixpanel sont les suivantes :

- Pour l'entité `Segmentation Numeric`, l'API Mixpanel génère une erreur `400 (Bad Request)` si aucune donnée numérique n'est trouvée pour les filtres obligatoires. Nous traitons cela comme une réponse 0K visant à prévenir les pannes de débit.
- Le champ interrogeable `limit` a été supprimé des entités prises en charge pour les raisons suivantes :
  - Il provoquait des erreurs, car il était interprété comme la fonctionnalité de limite du kit SDK.
  - Le filtre n'avait aucune utilité pratique.
  - Une fonctionnalité équivalente est désormais couverte par l'implémentation de la fonctionnalité de limite.
- Le partitionnement basé sur les champs ne peut pas être pris en charge en raison de l'absence d'opérateurs requis (>=, <=, <, >, `between`) pour le partitionnement depuis la plateforme SaaS. Bien qu'il prenne en charge l'opérateur `between`, les champs pour lesquels il prend en charge cet

opérateur ne sont pas récupérables. Par conséquent, les critères de partitionnement basé sur les champs ne sont pas satisfaits.

- Comme il n'existe aucune disposition permettant d'obtenir une valeur de « décalage » pour les entités prenant en charge la pagination, il n'est pas possible de prendre en charge le partitionnement basé sur les enregistrements pour Mixpanel.
- L'entité `Cohorts` ne prend en charge que le champ `CreatedDate/Time` et il n'y a aucun champ pour identifier `UpdatedDate/Time`. Par conséquent, `DML_Status` ne peut pas être identifié. De plus, il n'existe aucun point de terminaison pour identifier les enregistrements supprimés. Par conséquent, la pris en charge de CDC n'est pas possible.
- Pour exécuter une AWS Glue tâche pour les entités mentionnées ci-dessous, des filtres obligatoires sont requis. Reportez-vous au tableau ci-dessous pour obtenir les noms d'entités et leurs filtres requis.

  Nom de l'entité et filtres requis

Nom de l'entité	Filtres obligatoires
Annotations	from_date, to_date
Cohorts	Aucune
Engage	Aucune
Event	event, type, unit, from_date, to_date, on
Events Name	type
Events Properties	event, name, type, unit, from_date, to_date
Events Properties Top	event
Events Properties Values	event, name
Events Top	type
Funnels	funnel_id, from_date, to_date
Profile Event Activity	distinct_ids, from_date, to_date
Retention	from_date, to_date, unit, addiction_unit

Nom de l'entité	Filtres obligatoires
Segmentation	event, from_date, to_date
Segmentation Average	event, from_date, to_date, on
Segmentation Numeric	event, from_date, to_date, on
Segmentation Sum	event, from_date, to_date, on

Connexion à Monday

Monday.com est un système d'exploitation de travail polyvalent qui rationalise la gestion de projet et la collaboration en équipe. Il propose des flux de travail personnalisables, des tableaux de bord visuels et des outils d'automatisation pour améliorer la productivité. Les utilisateurs peuvent suivre les tâches, gérer les ressources et communiquer efficacement sur une plateforme intégrée.

Rubriques

- AWS Glue support pour le lundi
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration de Monday
- Configuration de connexions Monday
- Lecture à partir d'entités Monday
- Référence des options de connexion Monday
- Limitations
- Créer un compte Monday :

AWS Glue support pour le lundi

AWS Glue prend en charge le lundi comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser les tâches AWS Glue ETL pour interroger les données à partir du lundi.

Prise en charge en tant que cible ?

Non.

Versions d'API Monday prises en charge

v2.

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les AWS autorisations requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Vous pouvez également utiliser les politiques IAM gérées suivantes pour autoriser l'accès :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la

convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration de Monday

Avant de pouvoir transférer AWS Glue des données à partir du lundi, vous devez remplir les conditions suivantes :

Configuration requise

- Vous disposez d'un compte de développeur Monday avec e-mail et mot de passe. Pour de plus amples informations, veuillez consulter Créer un compte Monday :.
- Votre compte de développeur Monday est activé pour accéder à l'API. Toute utilisation du lundi APIs est disponible sans frais supplémentaires pendant la période d'essai. Une fois la période d'essai terminée, vous devez acheter un abonnement pour créer des données et y accéder. Pour plus d'informations, consultez la page des licences de Monday.

Si vous remplissez ces conditions, vous êtes prêt à vous connecter AWS Glue à votre compte Monday. Pour les connexions classiques, vous n'avez rien d'autre à faire dans Monday.

Configuration de connexions Monday

Monday prend en charge les deux types de mécanismes d'authentification suivants :

1. OAuth Auth : Monday prend en charge le type de subvention AUTHORIZATION_CODE pour. OAuth2
   - Ce type de subvention est considéré comme « à trois étapes » OAuth car il repose sur la redirection des utilisateurs vers le serveur d'autorisation tiers pour authentifier l'utilisateur. Il est utilisé lors de la création de connexions via la AWS Glue console. L'utilisateur qui crée une connexion peut par défaut s'appuyer sur une application connectée AWS Glue détenue par lui, dans laquelle il n'a pas besoin de fournir d'informations OAuth connexes, à l'exception de l'URL de l'instance du lundi. La AWS Glue console redirigera l'utilisateur vers Monday, où il devra se connecter et autoriser AWS Glue les autorisations demandées pour accéder à son instance Monday.
   - Les utilisateurs doivent choisir de créer leur propre application connectée le lundi et de fournir leur propre identifiant client et leur propre secret client lors de la création de connexions via

la AWS Glue console. Dans ce scénario, ils seront toujours redirigés vers le lundi pour se connecter et autoriser AWS Glue l'accès à leurs ressources.

- Ce type d'autorisation donne lieu à un jeton d'actualisation et à un jeton d'accès. Le jeton d'accès est actif pendant une heure et pourra être actualisé automatiquement sans interaction de l'utilisateur à l'aide du jeton d'actualisation.
- Pour plus d'informations, consultez la documentation sur la création d'une application connectée pour le flux AUTHORIZATION_CODE OAuth .

2. Authentification personnalisée :
   - Pour consulter la documentation publique du lundi sur la génération des clés d'API requises pour une autorisation personnalisée, voir https://developer.monday.com/api-reference/api-token-permissionsdocs/authentication#.

Pour configurer une connexion Monday :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes. Il est nécessaire de créer un secret pour chaque connexion dans AWS Glue.
   a. Pour l' OAuth authentification :
      - Pour l'application connectée gérée par le client : le secret doit contenir le secret du consommateur de l'application connectée avec USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET comme clé.
   b. Pour l'authentification personnalisée :
      - Pour l'application connectée gérée par le client : le secret doit contenir le secret du consommateur de l'application connectée avec `personalAccessToken` comme clé.
2. Dans AWS Glue Studio, créez une connexion sous Connexions de données en suivant les étapes ci-dessous :
   a. Sous Connexions de données, choisissez Créer une connexion.
   b. Lorsque vous sélectionnez une source de données, sélectionnez Monday.
   c. Indiquez l'URL de votre instance Monday.
   d. Sélectionnez le rôle IAM qui AWS Glue peut assumer les actions suivantes et qui dispose des autorisations nécessaires pour celles-ci :

      JSON

```
{
  "Version":"2012-10-17",
```

```
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

e. Sélectionnez le type d'authentification pour vous connecter à Monday
   - Pour OAuth l'authentification : indiquez l'URL du jeton et l'application client gérée par l'utilisateur ClientId du lundi auquel vous souhaitez vous connecter.
   - Pour une authentification personnalisée : sélectionnez le type d'authentification CUSTOM pour vous connecter à Monday.

f. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.

g. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.

3. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`. Choisissez Suivant.
4. Dans la configuration de votre AWS Glue tâche, fournissez `connectionName` une connexion réseau supplémentaire.

Lecture à partir d'entités Monday

Conditions préalables

- Un objet Monday à partir duquel vous souhaitez lire. Reportez-vous au tableau des entités prises en charge ci-dessous pour vérifier les entités disponibles.

Entités prises en charge pour la source

Liste des entités :

- Compte : https://developer.monday.com/api-reference/docs/account #queries
- Forum : https://developer.monday.com/api-reference/docs/boards #queries
- Colonne : https://developer.monday.com/api-reference/docs/columns #queries
- Documents : https://developer.monday.com/api-reference/docs/docs #queries
- Bloc de documents : https://developer.monday.com/api-reference/docs/blocks #queries
- Fichiers : https://developer.monday.com/api-reference/docs/files #queries
- Dossiers : https://developer.monday.com/api-reference/docs/folders #queries
- Groupes : https://developer.monday.com/api-reference/docs/groups #queries
- Numéro d'objet : https://developer.monday.com/api-reference/docs/items #queries
- Sous-éléments : https://developer.monday.com/api-reference/docs/subitems #queries
- Mots clés : https://developer.monday.com/api-reference/docs/tags-queries #queries
- Équipes : https://developer.monday.com/api-reference/docs/teams #queries
- Mises à jour : https://developer.monday.com/api-reference/docs/updates #queries
- Utilisateurs : https://developer.monday.com/api-reference/docs/users #queries
- Espaces de travail : https://developer.monday.com/api-reference/docs/espaces de travail #queries

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Compte	Non	Non	Non	Oui	Non
Boards	Oui	Oui	Non	Oui	Non
Columns	Non	Non	Non	Oui	Non
Docs	Oui	Oui	Non	Oui	Non
Document Blocks	Non	Oui	Non	Oui	Non
Fichiers	Oui	Non	Non	Oui	Non
Groupes	Non	Non	Non	Oui	Non

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Élément	Oui	Oui	Non	Oui	Non
Subitems	Non	Non	Non	Oui	Non
Étiquettes	Oui	Non	Non	Oui	Oui
Équipes	Oui	Non	Non	Oui	Non
Mises à jour	Non	Oui	Non	Oui	Non
Utilisateurs	Oui	Oui	Non	Oui	Non
Espaces de travail	Oui	Oui	Non	Oui	Non
Dossiers	Oui	Oui	Non	Oui	Non

Exemple

```
monday_read = glueContext.create_dynamic_frame.from_options(
     connection_type="monday",
     connection_options={
         "connectionName": "connectionName",
         "ENTITY_NAME": "account",
         "API_VERSION": "v2"
     }
```

Référence des options de connexion Monday

Les options de connexion pour Monday sont les suivantes :

- `ENTITY_NAME`(Chaîne) - (Obligatoire) Utilisé pour la lecture/l'écriture. Le nom de votre objet dans Monday.
- `API_VERSION`(Chaîne) - (Obligatoire) Utilisé pour la lecture/l'écriture. Version d'API Rest Monday que vous souhaitez utiliser. Exemple : v2.

- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.

Limitations

Les restrictions du connecteur Monday sont les suivantes :

- La réponse de métadonnées dynamiques présente certains conflits avec la documentation, comme indiqué ci-dessous :
  - L'entité Group, Column prend en charge les opérations de filtrage, mais elle n'est pas présente dans le point de terminaison des métadonnées dynamiques et est donc conservée en tant qu'entité non filtrable.
  - Le point de terminaison dynamique comprend environ 15 000 lignes et renvoie les métadonnées de toutes les entités en une seule réponse. De ce fait, le chargement des champs prend en moyenne 10 secondes, ce qui nécessiterait un certain temps supplémentaire lors de l'exécution d'une tâche.
  - Consultez le tableau ci-dessous pour connaître la limite de débit de Monday. La taille importante des données de réponse de l'entité dynamique entraîne un retard notable, les champs nécessitant en moyenne 10 secondes pour se charger.

Limite de complexité	5 000 000 points de complexité
Limite d'appels quotidiens	10 000 pour le plan Pro
Limite par minute	500 requêtes par minute
Limite de simultanéité	100 demandes simultanées maximum pour le plan Pro

Créer un compte Monday :

1. Accédez à la page d'accueil de Monday, https://monday.com/et choisissez Login.
2. Vous serez redirigé vers la page de connexion. Au bas de la page, cliquez sur S'inscrire.

3. Entrez votre adresse e-mail, puis choisissez Continuer. Vous pouvez également vous connecter avec Google.
4. Saisissez les détails requis, puis choisissez Continuer.
5. Répondez aux questions de l’enquête et suivez les étapes pour terminer le processus de création de compte.

Enregistrez une OAuth application :

1. Connectez-vous à votre compte monday.com. Cliquez sur votre avatar (icône d’image) dans le coin inférieur gauche de votre écran.
2. Choisissez Développeur.
3. Sélectionnez Créer une application.
4. Renseignez les champs obligatoires pour le nom et la description.
5. Accédez à la section « OAuth » présente sur le côté droit, ajoutez les portées et choisissez « Enregistrer la fonctionnalité ».
6. Accédez à l’onglet « URL de redirection » à côté de la portée, ajoutez l’URL de redirection et choisissez « Enregistrer la fonctionnalité ».
7. Dans l' URLsonglet Redirection, indiquez l'URL de votre application. Cela devrait être https://{region-code}.console.aws.amazon.com/appflow/ OAuth. Par exemple, si vous utilisez `us-east-1` , vous pouvez ajouter `https://us-east-1.console.aws.amazon.com/appflow/oauth`.
8. L’application est maintenant prête à être utilisée. Vous trouverez vos informations d’identification dans la section « Informations de base ». Notez vos chaînes d’ID client et de secret client. Ces chaînes sont utilisées pour établir une connexion avec cette application à l'aide d'un AppFlow connecteur.

Générer un jeton d’accès personnel :

Actuellement, monday.com ne propose que nos jetons API V2, qui sont tous des jetons personnels. Pour accéder à vos jetons d’API, vous pouvez utiliser l’une des deux méthodes suivantes en fonction de votre niveau d’utilisateur. Les utilisateurs administrateurs peuvent utiliser les deux méthodes pour acquérir leurs jetons d’API. Les utilisateurs membres peuvent accéder à leurs jetons d’API depuis leurs onglets Développeur.

Administrateurs : si vous êtes un utilisateur administrateur de votre compte monday.com, vous pouvez accéder à vos jetons d'API depuis l'onglet « Admin » en procédant comme suit :

1. Connectez-vous à votre compte monday.com. Cliquez sur votre avatar (icône d'image) dans le coin inférieur gauche de votre écran.
2. Sélectionnez « Administration » dans le menu qui s'affiche (vous devez disposer des autorisations d'administrateur).
3. Accédez à la section « API » et générez un « jeton d'API V2 ». Vous pouvez copier votre jeton et l'utiliser.

Développeur : si vous êtes un utilisateur membre de votre compte monday.com, vous pouvez accéder à vos jetons d'API depuis l'onglet Développeur en procédant comme suit :

1. Connectez-vous à votre compte monday.com. Cliquez sur votre avatar (icône d'image) dans le coin inférieur gauche de votre écran.
2. Sélectionnez « Développeurs » dans le menu qui s'affiche.
3. Dans le menu supérieur, choisissez le menu déroulant « Développeur ». Sélectionnez la première option dans le menu déroulant intitulé « Mes jetons d'accès ».

Connexion à MongoDB dans AWS Glue Studio

AWS Glue fournit une prise en charge intégrée pour MongoDB. AWS Glue Studio fournit une interface visuelle pour se connecter à MongoDB, créer des tâches d'intégration de données et les exécuter sur l'exécution Spark AWS Glue Studio sans serveur.

Rubriques

- Création d'une connexion MongoDB
- Création d'un nœud source MongoDB
- Création d'un nœud cible MongoDB
- Options avancées

Création d'une connexion MongoDB

Prérequis :

- Si votre instance MongoDB se trouve dans un Amazon VPC, configurez Amazon VPC pour permettre à votre AWS Glue tâche de communiquer avec l'instance MongoDB sans que le trafic ne transite par l'Internet public.

  Dans Amazon VPC, identifiez ou créez un VPC, un sous-réseau et un groupe de sécurité qui AWS Glue seront utilisés lors de l'exécution de la tâche. En outre, vous devez vous assurer qu'Amazon VPC est configuré pour autoriser le trafic réseau entre votre instance MongoDB et cet emplacement. Selon la configuration de votre réseau, cela peut nécessiter des modifications des règles du groupe de sécurité, du réseau ACLs, des passerelles NAT et des connexions d'appairage.

Pour configurer une connexion à MongoDB :

1. Vous pouvez éventuellement créer un secret à AWS Secrets Manager l'aide de vos informations d'identification MongoDB. Pour créer un secret dans Secrets Manager, suivez le didacticiel disponible dans la section Créer un AWS Secrets Manager secret dans la AWS Secrets Manager documentation. Après avoir créé le secret, conservez le nom du secret *secretName* pour l'étape suivante.

   - Lorsque vous sélectionnez des paires clé/valeur, créez une paire pour la clé `username` contenant la valeur. *mongodbUser*

     Lorsque vous sélectionnez des paires clé/valeur, créez une paire pour la clé `password` contenant la valeur. *mongodbPass*

2. Dans la AWS Glue console, créez une connexion en suivant les étapes décrites dansthe section called "Ajouter une AWS Glue connexion". Après avoir créé la connexion, conservez le nom de la connexion pour une utilisation future dans AWS Glue. *connectionName*

   - Lorsque vous sélectionnez un type de connexion, sélectionnez MongoDB ou MongoDB Atlas.
   - Lorsque vous sélectionnez une URL MongoDB ou une URL MongoDB Atlas, indiquez le nom d'hôte de votre instance MongoDB.

     Une URL MongoDB est fournie au format `mongodb://`*mongoHost*`:`*mongoPort*`/`*mongoDBname*.

     Une URL MongoDB Atlas est fournie au format `mongodb +srv://`*mongoHost*`/`*mongoDBname*.

- Si vous avez choisi de créer un secret Secrets Manager, choisissez le type AWS Secrets Manager d'identifiant.

  Ensuite, dans AWS Secret, fournissez*secretName*.

- Si vous choisissez de fournir un nom d'utilisateur et un mot de passe, fournissez *mongodbUser* et*mongodbPass*.

3. Dans les situations suivantes, vous pouvez avoir besoin d'une configuration supplémentaire :

   - Pour les instances MongoDB hébergées AWS dans un Amazon VPC
     - Vous devrez fournir les informations de connexion Amazon VPC à la AWS Glue connexion qui définit vos informations d'identification de sécurité MongoDB. Lorsque vous créez ou mettez à jour votre connexion, définissez le VPC, le sous-réseau et les groupes de sécurité dans les options réseau.

Après avoir créé une connexion AWS Glue MongoDB, vous devez effectuer les étapes suivantes avant d'exécuter votre AWS Glue tâche :

- Lorsque vous travaillez avec des AWS Glue tâches dans l'éditeur visuel, vous devez fournir les informations de connexion Amazon VPC pour que votre tâche se connecte à MongoDB. Identifiez un emplacement approprié dans Amazon VPC et fournissez-le à votre connexion AWS Glue MongoDB.
- Si vous avez choisi de créer un secret Secrets Manager, accordez au rôle IAM associé à votre AWS Glue tâche l'autorisation de lecture*secretName*.

Création d'un nœud source MongoDB

Prérequis

- Une connexion AWS Glue MongoDB, comme décrit dans la section précédente,. the section called "Création d'une connexion MongoDB"
- Si vous choisissez de créer un secret Secrets Manager, allouez des autorisations pour votre tâche pour lire le secret utilisé par la connexion.
- Une collection MongoDB à partir de laquelle vous souhaitez lire. Vous aurez besoin des informations d'identification pour la collection.

Une collection MongoDB est identifiée par un nom de base de données et un nom de collection,,*mongodbName*. *mongodbCollection*

Ajout d'une source de données MongoDB

Pour ajouter un nœud Source de données – MongoDB :

1. Choisissez la connexion pour votre source de données MongoDB. Puisque vous l'avez créé, il devrait être disponible dans le menu déroulant. Si vous devez créer une connexion, choisissez Créer une connexion MongoDB. Pour de plus amples informations, veuillez consulter la section précédente the section called "Création d'une connexion MongoDB".

   Une fois que vous avez choisi une connexion, vous pouvez afficher ses propriétés en cliquant sur Afficher les propriétés.
2. Choisissez une Base de données. Saisissez *mongodbName*.
3. Choisissez une Collection. Saisissez *mongodbCollection*.
4. Choisissez votre Partitionneur, la Taille de partition (Mo) et la Clé de partition. Pour plus d'informations sur les paramètres de partition, consultez the section called "« connectionType »: « mongodb » comme source".
5. Dans les Propriétés MongoDB personnalisées, saisissez les paramètres et les valeurs nécessaires.

Création d'un nœud cible MongoDB

Prérequis

- Une connexion AWS Glue MongoDB, configurée avec un AWS Secrets Manager secret, comme décrit dans la section précédente,. the section called "Création d'une connexion MongoDB"
- Les autorisations appropriées sur votre tâche pour lire le secret utilisé par la connexion.
- Une table MongoDB dans laquelle vous souhaitez écrire,. *tableName*

Ajout d'une cible de données MongoDB

Pour ajouter un nœud Cible de données – MongoDB :

1. Choisissez la connexion pour votre source de données MongoDB. Puisque vous l'avez créé, il devrait être disponible dans le menu déroulant. Si vous devez créer une connexion, choisissez Créer une connexion MongoDB. Pour de plus amples informations, veuillez consulter la section précédente the section called “Création d'une connexion MongoDB”.

   Une fois que vous avez choisi une connexion, vous pouvez afficher ses propriétés en cliquant sur Afficher les propriétés.
2. Choisissez une Base de données. Saisissez *`mongodbName`*.
3. Choisissez une Collection. Saisissez *`mongodbCollection`*.
4. Choisissez votre Partitionneur, la Taille de partition (Mo) et la Clé de partition. Pour plus d'informations sur les paramètres de partition, consultez the section called “« connectionType »: « mongodb » comme source”.
5. Choisissez Réessayer les écritures si vous le souhaitez.
6. Dans les Propriétés MongoDB personnalisées, saisissez les paramètres et les valeurs nécessaires.

Options avancées

Vous pouvez fournir des options avancées lors de la création d'un nœud MongoDB. Ces options sont les mêmes que celles disponibles lors de la programmation AWS Glue de scripts Spark.

Consultez the section called “Connexion MongoDB”.

Connexion à Oracle NetSuite

Oracle NetSuite est une solution de gestion d'entreprise all-in-one dans le cloud qui aide les entreprises à fonctionner plus efficacement en automatisant les processus de base et en fournissant une visibilité en temps réel sur les performances opérationnelles et financières. Grâce à une suite unique et intégrée d'applications pour la gestion de la comptabilité, du traitement des commandes, de la gestion des stocks, de la production, de la chaîne d'approvisionnement et des opérations d'entrepôt, Oracle NetSuite donne aux entreprises une visibilité claire sur leurs données et un contrôle plus strict de leurs activités.

Rubriques

- AWS Glue support pour Oracle NetSuite
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration d'Oracle NetSuite
- Configuration des NetSuite connexions Oracle
- Lecture à partir d' NetSuite entités Oracle
- Options de NetSuite connexion Oracle
- Limitations et remarques relatives au NetSuite connecteur Oracle

AWS Glue support pour Oracle NetSuite

AWS Glue prend en charge Oracle NetSuite comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser des tâches AWS Glue ETL pour interroger les données d'Oracle NetSuite.

Prise en charge en tant que cible ?

Non.

Versions d' NetSuite API Oracle prises en charge

Les versions suivantes de NetSuite l'API Oracle sont prises en charge :

- v1

Pour connaître la prise en charge des entités par version, consultez la section Entités prises en charge pour Source.

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les autorisations AWS IAM requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
```

```
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Si vous ne voulez pas utiliser la méthode ci-dessus, vous pouvez également utiliser les politiques IAM gérées suivantes :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration d'Oracle NetSuite

Avant de pouvoir AWS Glue transférer des données depuis Oracle NetSuite, vous devez satisfaire aux exigences suivantes :

Configuration requise

Les exigences minimales sont les suivantes :

- Vous avez un NetSuite compte Oracle. Pour de plus amples informations, veuillez consulter Création d'un NetSuite compte Oracle.
- Votre NetSuite compte Oracle est activé pour accéder à l'API.
- Vous avez créé une intégration d'API OAuth 2.0 dans votre compte de NetSuite développeur Oracle. Cette intégration fournit les informations d'identification du client qui AWS Glue permettent d'accéder à vos données en toute sécurité lorsqu'il effectue des appels authentifiés vers votre compte. Pour de plus amples informations, veuillez consulter Création d'une application NetSuite cliente Oracle et d'informations d'identification OAuth 2.0.

Si vous répondez à ces exigences, vous êtes prêt à vous connecter AWS Glue à votre NetSuite compte Oracle.

Création d'un NetSuite compte Oracle

Accédez à Oracle NetSuite, puis choisissez Free Product Tour. Renseignez les informations requises pour bénéficier d'une présentation guidée gratuite du produit, grâce à laquelle vous pourrez contacter un fournisseur. La procédure pour obtenir un compte est la suivante :

- L'achat d'un NetSuite compte se fait par l'intermédiaire d'un fournisseur, qui fournit un compte form/quote qui doit être légalement révisé.
- Le compte à acheter pour le NetSuite connecteur Oracle est un compte Standard Cloud Service.
- Ce compte est créé par le fournisseur et les informations d'identification temporaires sont partagées par celui-ci. Vous recevrez un e-mail de bienvenue NetSuite < billing@notification.netsuite.com > < system@sent-via.netsuite.com > contenant des informations telles que votre nom d'utilisateur et un lien pour définir votre mot de passe.
- Utilisez le lien Définir votre mot de passe pour définir le mot de passe du nom d'utilisateur indiqué par le fournisseur.

Création d'une application NetSuite cliente Oracle et d'informations d'identification OAuth 2.0

Pour obtenir l'ID client et le secret du client, vous devez créer une application NetSuite client Oracle :

1. Connectez-vous à votre NetSuite compte via l'identifiant NetSuite client.
2. Choisissez Configuration > Entreprise > Activer les fonctionnalités.
3. Accédez à la SuiteCloudsection et cochez la case REST WEB SERVICES sous SuiteTalk (Web Services).

4. Cochez la case OAUTH 2.0 sous Gérer l'authentification. Cliquez sur Sauvegarder
5. Accédez à Configuration > Intégration > Gérer les intégrations et choisissez Nouveau pour créer une application OAuth2 .0.
6. Saisissez le nom de votre choix et conservez le paramètre ÉTAT Activé.
7. Si cette case est cochée, décochez les cases TBA : FLUX D'AUTORISATION et AUTHENTIFICATION BASÉE SUR LES JETONS affichées sous Authentification basée sur les jetons.
8. Cochez les cases AUTHORIZATION CODE GRANT et PUBLIC CLIENT sous OAuth 2.0.
9. Sous Auth, notez l'ID client et le secret client.
10. Saisissez un URI DE REDIRECTION. Par exemple, https://us-east-1.console.aws.amazon.com/gluestudio/ oauth
11. Cochez la case SERVICES WEB REST sous PORTÉE.
12. Cochez la case INFORMATIONS D'IDENTIFICATION DE L'UTILISATEUR sous Informations d'identification de l'utilisateur. Choisissez Enregistrer.
13. Notez l' KEY/CLIENT ID DU CONSOMMATEUR et le SECRET/CLIENT SECRET DU CONSOMMATEUR sous Identifiants du client. Ces valeurs ne sont affichées qu'une seule fois.
14. Créez un rôle ADMINISTRATEUR au besoin en accédant à Utilisateur/Rôles > Gérer les rôles > Nouveau.
15. Lors de la création d'un rôle personnalisé, ajoutez un accès complet sous l'onglet Autorisations pour les entités/fonctionnalités suivantes :
    - « Dépôt », « Articles », « Livraison des articles », « Inscription au journal », « Bon de commande », « Filiales », « Fournisseurs », « Factures », « Autorisation de retour du fournisseur », « Suivi du temps », « Paiement client », « Entrées d'enregistrement personnalisées », « Types d'enregistrements personnalisés », « Gestion des applications autorisées OAuth 2.0 », « Champs d'entité personnalisés », « Connectez-vous à l'aide de jetons d'accès OAuth 2.0 ».

Pour plus d'informations, consultez la OAuth version 2.0 dans la documentation d' NetSuite Applications Suite.

Configuration des NetSuite connexions Oracle

Oracle NetSuite prend en charge le type de subvention AUTHORIZATION_CODE pour. OAuth2
Le type de subvention détermine le AWS Glue mode de communication avec Oracle NetSuite pour demander l'accès à vos données.

- Ce type de subvention est considéré comme « à trois étapes » OAuth car il repose sur la redirection des utilisateurs vers un serveur d'autorisation tiers pour authentifier l'utilisateur. Il est utilisé lors de la création de connexions via la AWS Glue console. L'utilisateur qui crée une connexion peut par défaut s'appuyer sur une application connectée lui AWS Glue appartenant (application client AWS Glue gérée) sur laquelle il n'a pas besoin de fournir d'informations OAuth connexes, à l'exception de l'URL de son NetSuite instance Oracle. La AWS Glue console redirige l'utilisateur vers Oracle NetSuite où il doit se connecter et accorder AWS Glue les autorisations demandées pour accéder à son NetSuite instance Oracle.
- Les utilisateurs peuvent toujours choisir de créer leur propre application connectée dans Oracle NetSuite et de fournir leur propre identifiant client et leur propre secret client lors de la création de connexions via la AWS Glue console. Dans ce scénario, ils seront toujours redirigés vers Oracle pour NetSuite se connecter et autoriser AWS Glue l'accès à leurs ressources.
- Ce type d'autorisation donne lieu à un jeton d'actualisation et à un jeton d'accès. Le jeton d'accès est de courte durée et peut être actualisé automatiquement sans interaction de l'utilisateur à l'aide du jeton d'actualisation.
- Pour consulter NetSuite la documentation Oracle publique sur la création d'une application connectée pour le OAuth flux de code d'autorisation, voir Applications publiques.

Pour configurer une NetSuite connexion Oracle, procédez comme suit :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes :
   a. Pour l'application connectée gérée par le client, le secret doit contenir le secret du consommateur de l'application connectée avec `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` comme clé.
   b. Remarque : Vous devez créer un secret pour votre connexion dans AWS Glue.
2. Dans AWS Glue Glue Studio, créez une connexion sous Data Connections en suivant les étapes ci-dessous :
   a. Lorsque vous sélectionnez un type de connexion, sélectionnez Oracle NetSuite.
   b. Fournissez l' NetSuite environnement Oracle.
   c. Sélectionnez le rôle AWS IAM qui AWS Glue peut assumer et dispose des autorisations nécessaires pour les actions suivantes :

      JSON

```
{
  "Version":"2012-10-17",
```

```
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

d. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.

e. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.

3. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.

Lecture à partir d' NetSuite entités Oracle

Prérequis

Un NetSuite objet Oracle dont vous souhaitez lire un extrait. Vous aurez besoin du nom de l'objet, comme `deposit` ou `timebill`. Le tableau suivant présente les entités prises en charge.

Entités prises en charge pour la source :

Entité	Peut être filtré	Order by prise en charge	Limit prise en charge	SELECT prise en charge*	Partition nement pris en charge
Deposit	Oui	Non	Oui	Oui	Oui
Description Item	Oui	Non	Oui	Oui	Oui

Entité	Peut être filtré	Order by prise en charge	Limit prise en charge	SELECT prise en charge*	Partition nement pris en charge
Inventory Item	Oui	Non	Oui	Oui	Oui
Item Fulfillme nt	Oui	Non	Oui	Oui	Oui
Item Group	Oui	Non	Oui	Oui	Oui
Journal Entry	Oui	Non	Oui	Oui	Oui
Non-Inven tory Purchase Item	Oui	Non	Oui	Oui	Oui
Non-Inven tory Resale Item	Oui	Non	Oui	Oui	Oui
Non-Inven tory Sale Item	Oui	Non	Oui	Oui	Oui
Bon de commande	Oui	Non	Oui	Oui	Oui
Subsidiary	Oui	Non	Oui	Oui	Oui
Vendor	Oui	Non	Oui	Oui	Oui
Vendor Bill	Oui	Non	Oui	Oui	Oui
Vendor Return Authorization	Oui	Non	Oui	Oui	Oui
Time Bill	Oui	Non	Oui	Oui	Oui

Entité	Peut être filtré	Order by prise en charge	Limit prise en charge	SELECT prise en charge*	Partition nement pris en charge
Customer Payment	Oui	Non	Oui	Oui	Oui
Fulfillment Request	Oui	Non	Oui	Oui	Oui
Élément	Oui	Oui	Oui	Oui	Oui
Transaction Line	Oui	Oui	Oui	Oui	Oui
Transaction Accounting Line	Oui	Oui	Oui	Oui	Oui
Custom Record Types (Dynamic)	Oui	Oui	Oui	Oui	Oui

Exemple :

```
netsuiteerp_read = glueContext.create_dynamic_frame.from_options(
    connection_type="netsuiteerp",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "deposit",
        "API_VERSION": "v1"
    }
)
```

Détails de NetSuite l'entité et du champ Oracle :

Oracle charge NetSuite dynamiquement les champs disponibles sous l'entité sélectionnée. Les opérateurs de filtrage suivants sont pris en charge en fonction du type de données du champ.

Type de données de champs	Opérateurs de filtre pris en charge
String	LIKE, =, !=
Date	BETWEEN, =, <, >, <=, >=
DateTime	BETWEEN, <, <=, >, >=
Numérique	=, !=, <, <=, >, >=
Booléen	=, !=

Format de saisie attendu pour les valeurs booléennes dans Expression de filtre :

Entité	Format de valeur booléenne « true »	Format de valeur booléenne « false »	Exemple
Entités Item, Transaction Line, Transaction Accounting Line, and Custom Record Type	T ou t	F ou f	isinactive = "T" ou isinactive = "t"
Toutes les autres entités	vrai	false	isinactive = true

Requêtes de partitionnement

Partitionnement basé sur les champs

Le NetSuite connecteur Oracle possède des métadonnées dynamiques afin que les champs pris en charge pour le partitionnement basé sur les champs soient choisis de manière dynamique. Le partitionnement basé sur les champs est pris en charge sur les champs de type Integer BigInteger, Date ou DateTime.

Vous pouvez indiquer les options Spark supplémentaires `PARTITION_FIELD`, `LOWER_BOUND`, `UPPER_BOUND` et `NUM_PARTITIONS` si vous souhaitez utiliser la simultanéité dans Spark. Avec ces paramètres, la requête d'origine serait divisée en `NUM_PARTITIONS` nombres de sous-requêtes pouvant être exécutées simultanément par les tâches Spark.

- PARTITION_FIELD : le nom du champ à utiliser pour partitionner la requête.
- LOWER_BOUND : une valeur limite inférieure inclusive du champ de partition choisi.

  Pour le champ Timestamp, nous acceptons le format d'horodatage Spark utilisé dans les requêtes SQL Spark.

  Exemples de valeurs valides :

  ```
  "TIMESTAMP \"1707256978123\""
  "TIMESTAMP \"1702600882\""
  "TIMESTAMP '2024-02-06T22:00:00:00.000Z'"
  "TIMESTAMP '2024-02-06T22:00:00:00Z'"
  "TIMESTAMP '2024-02-06'"
  ```

- UPPER_BOUND : une valeur limite supérieure exclusive du champ de partition choisi.
- NUM_PARTITIONS : le nombre de partitions.

Exemple :

```
netsuiteerp_read = glueContext.create_dynamic_frame.from_options(
    connection_type="netsuiteerp",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "deposit",
        "API_VERSION": "v1",
        "PARTITION_FIELD": "id",
        "LOWER_BOUND": "1",
        "UPPER_BOUND": "10000",
        "NUM_PARTITIONS": "10"
    }
```

Partitionnement basé sur les enregistrements

Vous pouvez indiquer l'option Spark supplémentaire NUM_PARTITIONS si vous souhaitez utiliser la simultanéité dans Spark. Avec ce paramètre, la requête d'origine serait divisée en NUM_PARTITIONS nombres de sous-requêtes pouvant être exécutées simultanément par les tâches Spark.

Dans le partitionnement basé sur les enregistrements, le nombre total d'enregistrements présents est demandé à partir de NetSuite l'API Oracle, puis divisé par le NUM_PARTITIONS nombre fourni, le nombre d'enregistrements qui en résulte est ensuite extrait simultanément par chaque sous-requête.

- NUM_PARTITIONS : le nombre de partitions.

Exemple :

```
netsuiteerp_read = glueContext.create_dynamic_frame.from_options(
    connection_type="netsuiteerp",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "deposit",
        "API_VERSION": "v1",
        "NUM_PARTITIONS": "3"
    }
```

Options de NetSuite connexion Oracle

Les options de connexion pour Oracle sont les NetSuite suivantes :

- ENTITY_NAME(Chaîne) - (Obligatoire) Utilisé pour la lecture. Nom de l' NetSuite entité Oracle. Exemple : deposit.
- API_VERSION(Chaîne) - (Obligatoire) Utilisé pour la lecture. Version de l' NetSuite API Oracle Rest que vous souhaitez utiliser. La valeur sera v1, car Oracle ne prend NetSuite actuellement en charge que la version v1.
- SELECTED_FIELDS(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Liste de colonnes séparées par des virgules que vous souhaitez sélectionner pour l'entité sélectionnée.
- FILTER_PREDICATE(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- QUERY(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.
- PARTITION_FIELD(Chaîne) - Utilisé pour la lecture. Champ à utiliser pour partitionner la requête (partitionnement basé sur les champs).
- LOWER_BOUND(Chaîne) - Utilisé pour la lecture. Une valeur limite inférieure inclusive du champ de partition choisi, utilisée dans le partitionnement basé sur des champs.
- UPPER_BOUND(Chaîne) - Utilisé pour la lecture. Une valeur limite supérieure exclusive du champ de partition choisi, utilisée dans le partitionnement basé sur des champs.

- `NUM_PARTITIONS`(Entier) - Par défaut : 1. Utilisé pour la lecture. Nombre de partitions pour la lecture. Utilisée à la fois dans le partitionnement basé sur des champs et dans le partitionnement basé sur des enregistrements.
- `INSTANCEE_URL`(String) - Une URL d' NetSuite instance valide avec un format https://{account-id}.suitetalk.api.netsuite.com.

Limitations et remarques relatives au NetSuite connecteur Oracle

Les limitations ou remarques suivantes concernent le NetSuite connecteur Oracle :

- Les valeurs des paramètres access_token et refresh_token sont au format JSON Web Token (JWT). Le jeton d'accès est valide pendant 60 minutes tandis que la valeur refresh_token est valide pendant sept jours.
- Lors de la génération de l'ID et du secret client, si vous sélectionnez « CLIENT PUBLIC » en plus d'« OCTROI DU CODE D'AUTORISATION », le jeton d'actualisation n'est valide que pendant trois heures et ne peut être utilisé qu'une seule fois.
- Vous pouvez récupérer au maximum 1 000 000 enregistrements à l'aide du connecteur. Pour plus d'informations, consultez Executing SuiteQL Queries Through REST Web Services.
- Les partitions sont créées de manière à ce que chaque partition récupère les enregistrements par multiples de 1 000, sauf peut-être la dernière qui récupère les enregistrements restants.
- Pour les objets Item, Transaction Line et Transaction Accounting Line, le connecteur ne prendra pas en charge certains opérateurs pour les raisons suivantes :
  - L'application des opérateurs de filtre `EQUAL_TO` et `NOT_EQUAL_TO` aux champs de type Date donne des résultats peu fiables.
  - L'application de l'opérateur de filtre `LESS_THAN_OR_EQUAL_TO` aux champs de type Date donne des résultats peu fiables et se comporte de la même manière que l'opérateur `LESS_THAN`.
  - L'application de l'opérateur de filtre `GREATER_THAN` aux champs de type Date= donne des résultats peu fiables et se comporte de la même manière que l'opérateur `GREATER_THAN_OR_EQUAL_TO`.
- Pour les objets Item, Transaction Line, Transaction Accounting Line et Custom Record Type, les valeurs booléennes sont présentées au format T/F plutôt qu'au format standard true/false. The connector maps the t/f values to true/false afin de garantir la cohérence des données.

Connexion au OpenSearch service dans AWS Glue Studio

AWS Gluefournit un support intégré pour Amazon OpenSearch Service. AWS Glue Studiofournit une interface visuelle pour se connecter à Amazon OpenSearch Service, créer des tâches d'intégration de données et les exécuter sur le runtime Spark AWS Glue Studio sans serveur. Cette fonctionnalité n'est pas compatible avec le OpenSearch Service serverless.

AWS Glue Studio crée une connexion unifiée pour Amazon OpenSearch Service. Pour de plus amples informations, veuillez consulter Considérations.

Rubriques

- Création d'une connexion OpenSearch de service
- Création d'un nœud OpenSearch de source de service
- Création d'un nœud cible de OpenSearch service
- Options avancées

Création d'une connexion OpenSearch de service

Prérequis :

- Identifiez le point de terminaison *aosEndpoint* et le port du domaine que *aosPort* vous souhaitez consulter, ou créez la ressource en suivant les instructions de la documentation Amazon OpenSearch Service. Pour plus d'informations sur la création d'un domaine, consultez la section Création et gestion OpenSearch de domaines Amazon Service dans la documentation Amazon OpenSearch Service.

  Un point de terminaison de domaine Amazon OpenSearch Service aura le formulaire par défaut suivant, https://search - *domainName* -*unstructuredIdContent*. *region*.es.amazonaws.com. Pour plus d'informations sur l'identification du point de terminaison de votre domaine, consultez la section Création et gestion des domaines Amazon OpenSearch Service dans la documentation Amazon OpenSearch Service.

  Identifiez ou générez des informations d'authentification HTTP de base, *aosUser* et *aosPassword* pour votre domaine.

Pour configurer une connexion au OpenSearch service, procédez comme suit :

1. Dans AWS Secrets Manager, créez un secret à l'aide de vos informations d'identification OpenSearch de service. Pour créer un secret dans Secrets Manager, suivez le didacticiel disponible dans la section Créer un AWS Secrets Manager secret dans la AWS Secrets Manager documentation. Après avoir créé le secret, conservez le nom du secret *secretName* pour l'étape suivante.

   - Lorsque vous sélectionnez des paires clé/valeur, créez une paire pour la clé USERNAME contenant la valeur. *aosUser*
   - Lorsque vous sélectionnez des paires clé/valeur, créez une paire pour la clé PASSWORD contenant la valeur. *aosPassword*
2. Dans la AWS Glue console, créez une connexion en suivant les étapes décrites dansthe section called "Ajouter une AWS Glue connexion". Après avoir créé la connexion, conservez le nom de la connexion pour une utilisation future dans AWS Glue. *connectionName*

   - Lorsque vous sélectionnez un type de connexion, sélectionnez OpenSearch Service.
   - Lorsque vous sélectionnez un point de terminaison de domaine, fournissez*aosEndpoint*.
   - Lorsque vous sélectionnez un port, indiquez*aosPort*.
   - Lorsque vous sélectionnez un AWS secret, fournissez*secretName*.

Création d'un nœud OpenSearch de source de service

Prérequis

- Une connexion AWS Glue OpenSearch de service, configurée avec un AWS Secrets Manager secret, comme décrit dans la section précédente,the section called "Création d'une connexion OpenSearch de service".
- Les autorisations appropriées sur votre tâche pour lire le secret utilisé par la connexion.
- Un index des OpenSearch services que vous aimeriez lire,*aosIndex*.

Ajouter une source OpenSearch de données de service

Pour ajouter une source de données — nœud OpenSearch de service :

1. Choisissez la connexion pour votre source OpenSearch de données de service. Puisque vous l'avez créé, il devrait être disponible dans le menu déroulant. Si vous devez créer une connexion,

choisissez Create OpenSearch Service connection. Pour de plus amples informations, veuillez consulter la section précédente the section called "Création d'une connexion OpenSearch de service".

Une fois que vous avez choisi une connexion, vous pouvez afficher ses propriétés en cliquant sur Afficher les propriétés.

2. Fournissez Index, l'index que vous souhaitez lire.
3. Vous pouvez éventuellement fournir Query, une OpenSearch requête permettant de fournir des résultats plus spécifiques. Pour plus d'informations sur la rédaction de OpenSearch requêtes, consultezthe section called "Lire depuis le OpenSearch service".
4. Dans Propriétés du OpenSearch service personnalisé, entrez les paramètres et les valeurs nécessaires.

Création d'un nœud cible de OpenSearch service

Prérequis

- Une connexion AWS Glue OpenSearch de service, configurée avec un AWS Secrets Manager secret, comme décrit dans la section précédente,the section called "Création d'une connexion OpenSearch de service".
- Les autorisations appropriées sur votre tâche pour lire le secret utilisé par la connexion.
- Un index des OpenSearch services dans lequel vous souhaitez écrire,*aosIndex*.

Ajouter une cible OpenSearch de données de service

Pour ajouter une cible de données — nœud OpenSearch de service :

1. Choisissez la connexion pour votre source OpenSearch de données de service. Puisque vous l'avez créé, il devrait être disponible dans le menu déroulant. Si vous devez créer une connexion, choisissez Create OpenSearch Service connection. Pour de plus amples informations, veuillez consulter la section précédente the section called "Création d'une connexion OpenSearch de service".

   Une fois que vous avez choisi une connexion, vous pouvez afficher ses propriétés en cliquant sur Afficher les propriétés.
2. Fournissez Index, l'index que vous souhaitez lire.

3. Dans Propriétés du OpenSearch service personnalisé, entrez les paramètres et les valeurs nécessaires.

Options avancées

Vous pouvez fournir des options avancées lors de la création d'un nœud OpenSearch de service. Ces options sont les mêmes que celles disponibles lors de la programmation AWS Glue de scripts Spark.

Consultez the section called "OpenSearch Connexions de service".

Connexion à Okta

L'API Okta est l'interface programmatique d'Okta, utilisée pour gérer des comptes et des campagnes Okta volumineux ou complexes. Si vous êtes un utilisateur d'Okta, vous pouvez vous connecter AWS Glue à votre compte Okta. Vous pouvez ensuite utiliser Okta comme source de données dans vos tâches ETL. Exécutez ces tâches pour transférer des données entre Okta et AWS des services ou d'autres applications prises en charge.

Rubriques

- AWS Glue support pour Okta
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration d'Okta
- Configuration de connexions Okta
- Lecture à partir d'entités Okta
- Référence des options de connexion Okta
- Étapes de création d'un compte Okta et d'une application pour développeurs
- Limitations

AWS Glue support pour Okta

AWS Glue prend en charge Okta comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser les tâches AWS Glue ETL pour interroger les données d'Okta.

Prise en charge en tant que cible ?

Non.

Versions de l'API Okta prises en charge

v1.

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les AWS autorisations requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Vous pouvez également utiliser les politiques IAM gérées suivantes pour autoriser l'accès :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la

convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration d'Okta

Avant de pouvoir transférer des données AWS Glue vers ou depuis Okta, vous devez satisfaire aux exigences suivantes :

Configuration requise

- Vous disposez d'un compte Okta. Pour plus d'informations sur la création d'un compte, consultez Étapes de création d'un compte Okta et d'une application pour développeurs.
- Votre compte Okta est activé pour accéder à l'API.
- Vous avez créé une intégration d' OAuth2 API dans votre compte Okta. Cette intégration fournit les informations d'identification du client qui AWS Glue permettent d'accéder à vos données en toute sécurité lorsqu'il effectue des appels authentifiés vers votre compte. Pour plus d'informations, reportez-vous aux étapes de création d'une application client et aux informations d'identification OAuth2 2.0 : étapes de création d'un nouveau compte Okta et d'une application pour développeurs
- Vous avez un compte Okta avec un OktaApiToken. Reportez-vous à la documentation Okta.

Si vous répondez à ces exigences, vous êtes prêt à vous connecter AWS Glue à votre compte Okta. Pour les connexions classiques, vous n'avez rien d'autre à faire dans Okta.

Configuration de connexions Okta

Okta prend en charge deux types de mécanismes d'authentification :

- OAuth auth : Okta prend en charge le type de `AUTHORIZATION_CODE` subvention.
  - Ce type de subvention est considéré comme « à trois étapes » OAuth car il repose sur la redirection des utilisateurs vers le serveur d'autorisation tiers pour authentifier l'utilisateur. Il est utilisé lors de la création de connexions via la AWS Glue console. La AWS Glue console redirige l'utilisateur vers Okta où il doit se connecter et accorder AWS Glue les autorisations demandées pour accéder à son instance Okta.
  - Les utilisateurs peuvent choisir de créer leur propre application connectée dans Okta et de fournir leur propre identifiant client et leur propre secret client lors de la création de connexions

via la AWS Glue console. Dans ce scénario, ils seront toujours redirigés vers Okta pour se connecter et autoriser l'accès AWS Glue à leurs ressources.

- Ce type d'autorisation donne lieu à un jeton d'actualisation et à un jeton d'accès. Le jeton d'accès est de courte durée et peut être actualisé automatiquement sans interaction de l'utilisateur à l'aide du jeton d'actualisation.
- Pour plus d'informations, consultez la documentation publique d'Okta sur la création d'une application connectée pour le OAuth flux de code d'autorisation.

- Authentification personnalisée :
  - Pour consulter la documentation publique d'Okta sur la génération des clés d'API requises pour une autorisation personnalisée, consultez la documentation Okta.

Pour configurer une connexion Okta :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes. Il est nécessaire de créer un secret pour chaque connexion dans AWS Glue.
   a. Pour l' OAuth authentification :
      - Pour l'application connectée gérée par le client : le secret doit contenir le secret du consommateur de l'application connectée avec `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` comme clé.
   b. Pour l'authentification personnalisée :
      - Pour l'application connectée gérée par le client : le secret doit contenir le secret du consommateur de l'application connectée avec `OktaApiToken` comme clé.
2. Dans AWS Glue Studio, créez une connexion sous Connexions de données en suivant les étapes ci-dessous :
   a. Sous Connexions, choisissez Créer une connexion.
   b. Lorsque vous sélectionnez une source de données, sélectionnez Okta.
   c. Indiquez votre sous-domaine Okta.
   d. Sélectionnez l'URL du domaine Okta de votre compte Okta.
   e. Sélectionnez le rôle IAM qui AWS Glue peut assumer et dispose des autorisations nécessaires pour les actions suivantes :

      JSON

```
{
  "Version":"2012-10-17",
```

```
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

f. Sélectionnez le type d'authentification pour vous connecter à une source de données.

g. Pour le type OAuth2 d'authentification, fournissez l'application client gérée par l'utilisateur ClientId de l'application Okta.

h. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.

i. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.

3. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.

4. Dans la configuration de votre AWS Glue tâche, fournissez `connectionName` une connexion réseau supplémentaire.

Lecture à partir d'entités Okta

Conditions préalables

- Un objet Okta à partir duquel vous souhaitez lire. Reportez-vous au tableau des entités prises en charge ci-dessous pour vérifier les entités disponibles.

Entités prises en charge

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Applications	Oui	Oui	Non	Oui	Non
Devices	Oui	Oui	Non	Oui	Oui
Groupes	Oui	Oui	Oui	Oui	Oui
Utilisateurs	Oui	Oui	Oui	Oui	Oui
Types d'utilisateurs	Non	Non	Non	Oui	Non

Exemple

```
okta_read = glueContext.create_dynamic_frame.from_options(
    connection_type="Okta",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "applications",
        "API_VERSION": "v1"
    }
```

Détails des entités et des champs Okta

Liste des entités :

- Application : https://developer.okta.com/docs/api/openapi/okta-management/management/tag/Application/
- Appareil : https://developer.okta.com/docs/api/openapi/okta-management/management/tag/Device/
- Groupe : https://developer.okta.com/docs/api/openapi/okta-management/management/tag/Group/
- Utilisateur : https://developer.okta.com/docs/api/openapi/okta-management/management/tag/User/
- Type d'utilisateur : https://developer.okta.com/docs/api/openapi/okta-management/management/tag/UserType/

Requêtes de partitionnement

Les options Spark supplémentaires PARTITION_FIELD, LOWER_BOUND, UPPER_BOUND et NUM_PARTITIONS peuvent être indiquées si vous souhaitez utiliser la simultanéité dans Spark. Avec ces paramètres, la requête d'origine serait divisée en NUM_PARTITIONS nombres de sous-requêtes pouvant être exécutées simultanément par les tâches Spark.

- PARTITION_FIELD : le nom du champ à utiliser pour partitionner la requête.
- LOWER_BOUND : une valeur limite inférieure inclusive du champ de partition choisi.

  Pour le champ Date, nous acceptons le format de date Spark utilisé dans les requêtes SQL Spark. Exemples de valeurs valides : "2024-02-06".
- UPPER_BOUND : une valeur limite supérieure exclusive du champ de partition choisi.
- NUM_PARTITIONS : nombre de partitions.

Exemple

```
okta_read = glueContext.create_dynamic_frame.from_options(
    connection_type="okta",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "lastUpdated",
        "API_VERSION": "v1",
        "PARTITION_FIELD": "lastMembershipUpdated"
        "LOWER_BOUND": "2022-08-10T10:28:46.000Z"
        "UPPER_BOUND": "2024-08-10T10:28:46.000Z"
        "NUM_PARTITIONS": "10"
    }
```

Référence des options de connexion Okta

Les options de connexion pour Okta sont les suivantes :

- ENTITY_NAME(Chaîne) - (Obligatoire) Utilisé pour la lecture/l'écriture. Le nom de votre objet dans Okta.
- API_VERSION(Chaîne) - (Obligatoire) Utilisé pour la lecture/l'écriture. Version d'API Rest Okta que vous souhaitez utiliser. Exemple : v1.
- SELECTED_FIELDS(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.

- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.
- `PARTITION_FIELD`(Chaîne) - Utilisé pour la lecture. Champ à utiliser pour partitionner la requête.
- `LOWER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite inférieure inclusive du champ de partition choisi.
- `UPPER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS`(Entier) - Par défaut : 1. Utilisé pour la lecture. Nombre de partitions pour la lecture.

Étapes de création d'un compte Okta et d'une application pour développeurs

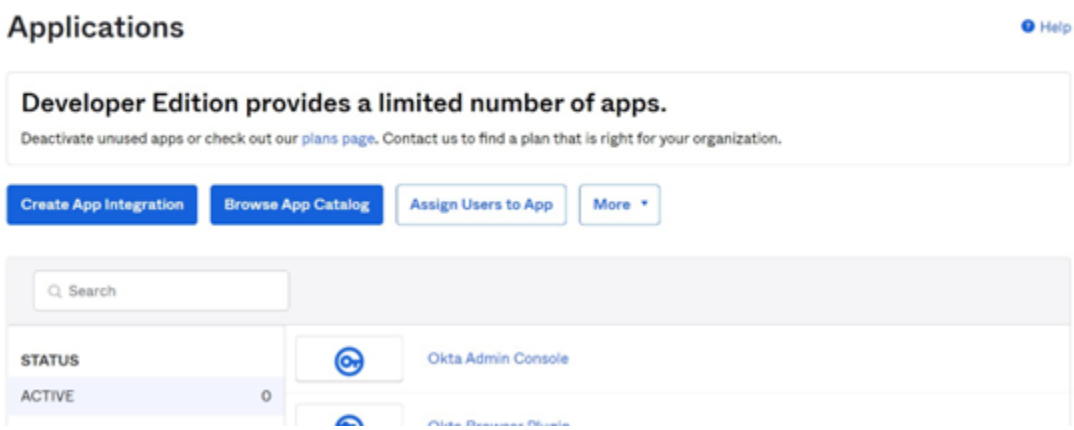
Créez un compte de développeur sur Okta pour accéder à l'API Okta. Un compte de développeur Okta gratuit donne accès à la plupart des fonctionnalités de développement clés requises pour accéder à l'API Okta.

Pour créer un compte de développeur sur Okta

1. Accédez à https://developer.okta.com/signup/.
2. Saisissez les informations du compte (e-mail, prénom, nom de famille et pays/région). Choisissez Je ne suis pas un robot, puis S'inscrire.
3. Un e-mail de vérification est envoyé à votre adresse e-mail enregistrée. Vous recevrez un lien dans votre messagerie pour activer votre compte de développeur Okta. Choisissez Activer.
4. Vous serez redirigé vers la page de réinitialisation du mot de passe. Saisissez le nouveau mot de passe à deux reprises et choisissez Réinitialiser le mot de passe.
5. Vous serez redirigé vers le tableau de bord de votre compte de développeur Okta.

Pour créer une application cliente et des informations d'identification OAuth 2.0

1. Dans le tableau de bord du développeur, choisissez Créer une intégration d'application.



2. La fenêtre Créer une intégration d'application apparaîtra et présentera différentes méthodes de connexion. Sélectionnez OIDC — OpenID Connect.
3. Faites défiler la page jusqu'à la section Type d'application. Sélectionnez Application Web, puis cliquez sur Suivant.
4. Sur l'écran « Nouvelle intégration d'application Web », renseignez les informations suivantes :
   - Nom de l'intégration d'application : saisissez le nom de l'application.
   - Type d'autorisation : choisissez le code d'autorisation et le jeton d'actualisation dans la liste.
   - Redirection de connexion URIs : choisissez Ajouter une URI et **`https://{regioncode}.console.aws.amazon.com/appflow/oauth`** ajoutez-la. Par exemple, si vous utilisez `us-west-2 (Oregon)`, vous pouvez ajouter `https://us-east-1.console.aws.amazon.com/appflow/oauth`.
   - Accès contrôlé : assignez l'application à vos groupes d'utilisateurs selon vos besoins et choisissez Enregistrer.
5. Votre ID client et votre secret client sont générés.

Limitations

Les limites du connecteur Okta sont les suivantes :

- Pour l'entité « Applications », un seul filtre peut être appliqué. Si plus d'un filtre est appliqué, l'erreur 400 Bad Request est renvoyée avec le résumé : « Critères de recherche non valides ».
- La commande Order By ne peut être prise en charge que par des requêtes de recherche. Par exemple, `http://dev-15940405.okta.com/api/v1/groups?search=type e.q. "OKTA_GROUP"&sortBy=lastUpdated&sortOrder=asc` .

Connexion à PayPal

PayPal est un système de paiement qui facilite les transferts d'argent en ligne entre les parties, tels que les transferts entre clients et fournisseurs en ligne. Si vous êtes un PayPal utilisateur, votre compte contient des données relatives à vos transactions, telles que leurs payeurs, leurs dates et leurs statuts. Vous pouvez l'utiliser AWS Glue pour transférer des données depuis certains AWS services ou PayPal vers d'autres applications prises en charge.

Rubriques

- AWS Glue support pour PayPal
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration PayPal
- Configuration des PayPal connexions
- Lecture à partir d' PayPal entités
- PayPal options de connexion
- Limitations et remarques relatives au PayPal connecteur

AWS Glue support pour PayPal

AWS Glue prend en charge PayPal les éléments suivants :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser des tâches AWS Glue ETL pour interroger des données à partir de PayPal.

Prise en charge en tant que cible ?

Non.

Versions PayPal d'API prises en charge

Les versions PayPal d'API suivantes sont prises en charge :

- v1

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les autorisations AWS IAM requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Si vous ne voulez pas utiliser la méthode ci-dessus, vous pouvez également utiliser les politiques IAM gérées suivantes :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration PayPal

Avant de pouvoir AWS Glue transférer des données depuis PayPal, vous devez satisfaire aux exigences suivantes :

Configuration requise

Les exigences minimales sont les suivantes :

- Vous avez un PayPal compte avec des informations d'identification client.
- Votre PayPal compte dispose d'un accès à l'API avec une licence valide.

Si vous remplissez ces conditions, vous êtes prêt à vous connecter AWS Glue à votre PayPal compte. Pour les connexions classiques, vous n'avez rien d'autre à faire dans PayPal.

Configuration des PayPal connexions

PayPal prend en charge le type de subvention CLIENT CREDENTIALS pour OAuth2.

- Ce type de subvention est considéré comme bidimensionnel OAuth 2.0 car il est utilisé par les clients pour obtenir un jeton d'accès en dehors du contexte d'un utilisateur. AWS Glue est capable d'utiliser l'ID client et le secret du client pour authentifier PayPal APIs ceux fournis par les services personnalisés que vous définissez.
- Chaque service personnalisé appartient à un utilisateur régi uniquement par l'API, qui dispose d'un ensemble de rôles et d'autorisations autorisant le service à effectuer des actions spécifiques. Un jeton d'accès est associé à un seul service personnalisé.
- Ce type d'autorisation donne lieu à un jeton d'accès de courte durée qui peut être renouvelé en appelant à nouveau le point de terminaison `/v2/oauth2/token`.
- Pour la PayPal documentation publique de la OAuth version 2.0 avec les informations d'identification du client, voir Authentification.

Pour configurer une PayPal connexion :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes :
   a. Pour l'application connectée gérée par le client, le secret doit contenir le secret du consommateur de l'application connectée avec `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` comme clé.
   b. Remarque : vous devez créer un secret pour vos connexions dans AWS Glue.

1. Dans AWS Glue Glue Studio, créez une connexion sous Data Connections en suivant les étapes ci-dessous :

a. Lorsque vous sélectionnez un type de connexion, sélectionnez PayPal.

b. Indiquez `INSTANCE_URL` l' PayPal instance à laquelle vous souhaitez vous connecter.

c. Sélectionnez le rôle AWS IAM qui AWS Glue peut assumer et dispose des autorisations nécessaires pour les actions suivantes :

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

d. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.

e. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.

2. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.

Obtenir des informations d'identification OAuth 2.0

Pour appeler l'API Rest, vous devez échanger votre ID client et votre secret client contre un jeton d'accès. Pour plus d'informations, consultez Commencer avec PayPal REST APIs.

Lecture à partir d' PayPal entités

Prérequis

Un PayPal objet que vous aimeriez lire. Vous aurez besoin du nom de l'objet, `transaction`.

Entités prises en charge pour la source :

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
transaction	Oui	Oui	Non	Oui	Oui

Exemple :

```
paypal_read = glueContext.create_dynamic_frame.from_options(
    connection_type="paypal",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "transaction",
        "API_VERSION": "v1",
        "INSTANCE_URL": "https://api-m.paypal.com"
    }
```

PayPal détails de l'entité et du champ :

Entités avec métadonnées statiques :

Entité	Champ	Type de données	Opérateurs pris en charge
transaction	transaction_initiation_date	DateTime	Entre
	last_refreshed_datetime	String	N/A
	payment_instrument_type	String	=
	balance_affecting_records_only	String	=
	store_id	String	=

Entité	Champ	Type de données	Opérateurs pris en charge
	terminal_id	String	=
	transaction_currency	String	=
	transaction_id	String	N/A
	transaction_status	String	N/A
	transaction_type	String	N/A
	transaction_info	Struct	N/A
	payer_info	Struct	N/A
	shipping_info	Struct	N/A
	cart_info	Struct	N/A
	store_info	Struct	N/A
	auction_info	Struct	N/A
	incentive_info	Struct	N/A

Requêtes de partitionnement

Vous pouvez indiquer les options Spark supplémentaires `PARTITION_FIELD`, `LOWER_BOUND`, `UPPER_BOUND` et `NUM_PARTITIONS` si vous souhaitez utiliser la simultanéité dans Spark. Avec ces paramètres, la requête d'origine serait divisée en `NUM_PARTITIONS` nombres de sous-requêtes pouvant être exécutées simultanément par les tâches Spark.

- `PARTITION_FIELD` : le nom du champ à utiliser pour partitionner la requête.
- `LOWER_BOUND` : une valeur limite inférieure inclusive du champ de partition choisi.

  Pour le champ Datetime, nous acceptons la valeur au format ISO.

  Exemples de valeurs valides :

```
"2024-07-01T00:00:00.000Z"
```

- UPPER_BOUND : une valeur limite supérieure exclusive du champ de partition choisi.
- NUM_PARTITIONS : le nombre de partitions.

Le champ suivant est pris en charge pour le partitionnement par entité :

Nom de l'entité	Champs de partitionnement	Type de données
transaction	transaction_initiation_date	DateTime

Exemple :

```
paypal_read = glueContext.create_dynamic_frame.from_options(
    connection_type="paypal",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "transaction",
        "API_VERSION": "v1",
        "PARTITION_FIELD": "transaction_initiation_date"
        "LOWER_BOUND": "2024-07-01T00:00:00.000Z"
        "UPPER_BOUND": "2024-07-02T00:00:00.000Z"
        "NUM_PARTITIONS": "10"
    }
```

PayPal options de connexion

Les options de connexion suivantes sont disponibles pour PayPal :

- ENTITY_NAME(Chaîne) - (Obligatoire) Utilisé pour la lecture. Le nom de votre objet dans PayPal.
- API_VERSION(Chaîne) - (Obligatoire) Utilisé pour la lecture. PayPal Version de l'API Rest que vous souhaitez utiliser.
- SELECTED_FIELDS(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- FILTER_PREDICATE(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.

- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.
- `PARTITION_FIELD`(Chaîne) - Utilisé pour la lecture. Champ à utiliser pour partitionner la requête.
- `LOWER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite inférieure inclusive du champ de partition choisi.
- `UPPER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS`(Entier) - Par défaut : 1. Utilisé pour la lecture. Nombre de partitions pour la lecture.

Limitations et remarques relatives au PayPal connecteur

Les limitations ou remarques suivantes concernent le PayPal connecteur :

- La documentation PayPal des transactions indique qu'il faut au maximum trois heures pour que les transactions exécutées apparaissent dans l'appel de transactions de liste. Cependant, il a été observé que cela prenait plus de temps que cela, en fonction de la valeur `last_refreshed_datetime`. `last_refreshed_datetime`Voici la durée pendant laquelle vous avez des données disponibles auprès du APIs.
- Si la valeur `last_refreshed_datetime` est inférieure à la valeur `end_date` demandée, `end_date` devient égale à `last_refreshed_datetime`, car nous n'avons que des données jusqu'à ce moment-là.
- Le champ `transaction_initiation_date` est un filtre obligatoire à fournir pour l'entité `transaction` et la plage de dates maximale prise en charge pour ce champ est de 31 jours.
- Lorsque vous appelez la demande d'API d'entité `transaction` avec des filtres (paramètres de requête) autres que le champ `transaction_initiation_date`, il est prévu que la valeur du champ `ending_balance` ne soit pas extraite dans la réponse.

Connexion à Pendo

Pendo fournit un entrepôt de données enrichi pour les données d'interaction avec les utilisateurs. Les clients transféreront ces données AWS afin de les associer à d'autres données sur le produit, d'effectuer des analyses et des tableaux de bord supplémentaires et de définir des alertes s'ils le souhaitent.

Rubriques

- AWS Glue support pour Pendo
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration de Pendo
- Configuration de connexions Pendo
- Lecture à partir d'entités Pendo
- Options de connexion Pendo
- Limitations

AWS Glue support pour Pendo

AWS Glue prend en charge Pendo comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser les tâches AWS Glue ETL pour interroger les données de Pendo.

Prise en charge en tant que cible ?

Non.

Versions de l'API Pendo prises en charge

v1

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les AWS autorisations requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
```

```
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Si vous ne voulez pas utiliser la méthode ci-dessus, vous pouvez également utiliser les politiques IAM gérées suivantes :

- AWSGlueServiceRole — Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess — Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration de Pendo

Avant de pouvoir AWS Glue effectuer un transfert depuis Pendo, vous devez satisfaire aux exigences suivantes :

Configuration requise

- Vous possédez un compte Pendo avec une `apiKey` avec `write access` activé.
- Votre compte Pendo dispose d'un accès à l'API avec une licence valide.

Si vous répondez à ces exigences, vous êtes prêt à vous connecter AWS Glue à votre compte Pendo. Pour les connexions classiques, vous n'avez rien d'autre à faire dans Pendo.

Configuration de connexions Pendo

Pendo prend en charge l'authentification personnalisée.

Pour consulter la documentation publique de Pendo sur la génération des clés d'API requises pour une autorisation personnalisée, consultez Authentication – Pendo REST API Documentation

Pour configurer une connexion Pendo :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes :
   - Pour l'application connectée gérée par le client, le secret doit contenir le secret du consommateur de l'application connectée avec `apiKey` comme clé.

   > **Note**
   >
   > Il est indispensable de créer un secret par connexion dans AWS Glue.

2. Dans AWS Glue Studio, créez une connexion sous Connexions de données en suivant les étapes ci-dessous :
   a. Lorsque vous sélectionnez une source de données, sélectionnez Pendo.
   b. Indiquez l'`instanceUrl` de l'instance Pendo à laquelle vous souhaitez vous connecter.
   c. Sélectionnez le rôle IAM pour lequel AWS Glue vous pouvez assumer et qui dispose des autorisations nécessaires pour effectuer les actions suivantes :

      JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
```

```
  ]
}
```

d. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.

e. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.

3. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.
4. Dans la configuration de votre AWS Glue tâche, fournissez `connectionName` une connexion réseau supplémentaire.

Lecture à partir d'entités Pendo

Conditions préalables

Un objet Pendo à partir duquel vous souhaitez lire. Reportez-vous au tableau des entités prises en charge ci-dessous pour vérifier les entités disponibles.

Entités prises en charge

- Fonctionnalité
- Guide
- Page
- Rapport
- Report Data
- Visitor
- Compte
- Event
- Feature Event
- Guide Event
- Page Event
- Poll Event
- Track Event

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Fonctionn alité	Non	Non	Non	Oui	Non
Guide	Non	Non	Non	Oui	Non
Page	Non	Non	Non	Oui	Non
Rapport	Non	Non	Non	Oui	Non
Report Data	Non	Non	Non	Oui	Non
Visitor (Aggregation API)	Oui	Non	Oui	Oui	Non
Account (Aggregation API)	Oui	Non	Oui	Oui	Non
Event (Aggregation API)	Oui	Non	Oui	Oui	Non
Feature Event (Aggregation API)	Oui	Non	Oui	Oui	Oui
Guide Event (Aggregation API)	Oui	Non	Oui	Oui	Oui
Account (Aggregation API)	Oui	Non	Oui	Oui	Oui

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Page Event (Aggregation API)	Oui	Non	Oui	Oui	Oui
Poll Event (Aggregation API)	Oui	Non	Oui	Oui	Oui
Track Event (Aggregation API)	Oui	Non	Oui	Oui	Oui

Exemple

```
Pendo_read = glueContext.create_dynamic_frame.from_options(
    connection_type="glue.spark.Pendo",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "feature",
        "API_VERSION": "v1",
        "INSTANCE_URL": "instanceUrl"
    }
```

Requêtes de partitionnement

Vous pouvez indiquer les options Spark supplémentaires `PARTITION_FIELD`, `LOWER_BOUND`, `UPPER_BOUND` et `NUM_PARTITIONS` si vous souhaitez utiliser la simultanéité dans Spark. Avec ces paramètres, la requête d'origine serait divisée en `NUM_PARTITIONS` nombres de sous-requêtes pouvant être exécutées simultanément par les tâches Spark.

- `PARTITION_FIELD` : le nom du champ à utiliser pour partitionner la requête.
- `LOWER_BOUND` : une valeur limite inférieure inclusive du champ de partition choisi.

  Pour le DateTime champ, nous acceptons la valeur au format ISO.

Exemple de valeur valide :

```
"2024-07-01T00:00:00.000Z"
```

- UPPER_BOUND : une valeur limite supérieure exclusive du champ de partition choisi.
- NUM_PARTITIONS : le nombre de partitions.

Le tableau suivant décrit les détails de prise en charge dans les champs de partitionnement d'entités :

Nom de l'entité
Event
Feature Event
Guide Event
Page Event
Poll Event
Track Event

Exemple :

```
pendo_read = glueContext.create_dynamic_frame.from_options(
    connection_type="glue.spark.pendo",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "event",
        "API_VERSION": "v1",
        "INSTANCE_URL": "instanceUrl"
        "NUM_PARTITIONS": "10",
        "PARTITION_FIELD": "appId"
        "LOWER_BOUND": "4656"
        "UPPER_BOUND": "7788"
    }
```

Options de connexion Pendo

Les options de connexion pour Pendo sont les suivantes :

- `ENTITY_NAME`(Chaîne) - (Obligatoire) Utilisé pour la lecture/l'écriture. Le nom de votre objet dans Pendo.
- `INSTANCE_URL`(Chaîne) - (Obligatoire) Une URL d'instance Pendo valide avec les valeurs autorisées suivantes :
  - Par défaut
  - L'Europe
  - US1
- `API_VERSION`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Version d'API Rest de Pendo Engage que vous souhaitez utiliser. Par exemple : 3.0.
- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.
- `PARTITION_FIELD`(Chaîne) - Utilisé pour la lecture. Champ à utiliser pour partitionner la requête.
- `LOWER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite inférieure inclusive du champ de partition choisi.
- `UPPER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS`(Entier) - Par défaut : 1. Utilisé pour la lecture. Nombre de partitions pour la lecture.

Limitations

Les restrictions du connecteur Pendo sont les suivantes :

- La pagination n'est pas prise en charge dans Pendo.
- La filtration est prise en charge uniquement par les objets de l'API Aggregate (`Account`, `Event`, `Feature Event`, `Guide Events`, `Page Event`, `Poll Event`, `Track Event` et `Visitor`).

- DateTimeRange est un paramètre de filtre obligatoire pour les objets de l'API Aggregate (`EventFeature Event`,`Guide Events`,`Page Event`,, `Poll Event`,`Track Event`)
- La période dayRange sera arrondie au début de la période dans le fuseau horaire. Par exemple, si le filtre fourni est `2023-01-12T07:55:27.065Z`, cette période sera arrondie au début de la période, c'est-à-dire `2023-01-12T00:00:00Z`.

Connexion à Pipedrive

Pipedrive est un CRM de pipeline de vente conçu pour aider les petites entreprises à gérer les prospects, à suivre les activités commerciales et à conclure davantage de ventes. Pipedrive permet aux équipes commerciales des petites entreprises de rationaliser les processus et de consolider les données de vente dans un seul outil de vente CRM unifié. Si vous êtes un utilisateur de Pipedrive, vous pouvez vous connecter AWS Glue à votre compte Pipedrive. Vous pouvez ensuite utiliser Pipedrive comme source de données dans vos tâches ETL. Exécutez ces tâches pour transférer des données entre Pipedrive et des AWS services ou d'autres applications prises en charge.

Rubriques

- AWS Glue support pour Pipedrive
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration de Pipedrive
- Configuration des connexions Pipedrive
- Lecture à partir d'entités Pipedrive
- Référence des options de connexion Pipedrive
- Limitations

AWS Glue support pour Pipedrive

AWS Glue prend en charge Pipedrive comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser les tâches AWS Glue ETL pour interroger les données de Pipedrive.

Prise en charge en tant que cible ?

Non.

Versions de l'API Pipedrive prises en charge

v1.

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les AWS autorisations requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Vous pouvez également utiliser les politiques IAM gérées suivantes pour autoriser l'accès :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs

bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration de Pipedrive

Avant de pouvoir AWS Glue transférer des données depuis Pipedrive, vous devez satisfaire aux exigences suivantes :

Configuration requise

- Vous disposez d'un compte Pipedrive.
- Votre compte Pipedrive est activé pour accéder à l'API.

Si vous répondez à ces exigences, vous êtes prêt à vous connecter AWS Glue à votre compte Pipedrive. Pour les connexions classiques, vous n'avez rien d'autre à faire dans Pipedrive.

Configuration des connexions Pipedrive

Pipedrive prend en charge le type de subvention AUTHORIZATION_CODE pour. OAuth2

- Ce type de subvention est considéré comme « à trois étapes » OAuth car il repose sur la redirection des utilisateurs vers le serveur d'autorisation tiers pour authentifier l'utilisateur. Il est utilisé lors de la création de connexions via la AWS Glue console. L'utilisateur qui crée une connexion peut par défaut s'appuyer sur une application connectée qu'il AWS Glue possède et dans laquelle il n'a pas besoin de fournir d'informations OAuth connexes, à l'exception de l'instanceurl de Pipedrive. La AWS Glue console redirigera l'utilisateur vers Pipedrive où il devra se connecter et accorder AWS Glue les autorisations demandées pour accéder à son instance Pipedrive.
- Les utilisateurs doivent choisir de créer leur propre application connectée dans Pipedrive et de fournir leur propre identifiant client et leur propre secret client lors de la création de connexions via la AWS Glue console. Dans ce scénario, ils seront toujours redirigés vers Pipedrive pour se connecter et autoriser l'accès AWS Glue à leurs ressources.
- Ce type d'autorisation donne lieu à un jeton d'actualisation et à un jeton d'accès. Le jeton d'accès est actif pendant une heure et pourra être actualisé automatiquement sans interaction de l'utilisateur à l'aide du jeton d'actualisation.
- Pour plus d'informations, consultez la documentation sur la création d'une application connectée pour le flux AUTHORIZATION_CODE OAuth .

Pour configurer une connexion Pipedrive :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes. Il est nécessaire de créer un secret pour chaque connexion dans AWS Glue.
   a. Pour l'application connectée gérée par le client : le secret doit contenir le secret du consommateur de l'application connectée avec USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET comme clé.
2. Dans AWS Glue Studio, créez une connexion sous Connexions de données en suivant les étapes ci-dessous :
   a. Sous Connexions de données, choisissez Créer une connexion.
   b. Lorsque vous sélectionnez une source de données, sélectionnez Pipedrive.
   c. Indiquez votre URL d'instance Pipedrive.
   d. Sélectionnez le rôle IAM qui AWS Glue peut assumer les actions suivantes et qui dispose des autorisations nécessaires pour les actions suivantes :

      JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

   e. Fournissez l'application client gérée par l'utilisateur ClientId du Pipedrive auquel vous souhaitez vous connecter.
   f. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.

g. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.

3. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`. Choisissez Suivant.
4. Indiquez la valeur connectionName et choisissez Suivant.
5. Sur la page suivante, choisissez Créer une connexion. Il vous sera demandé de vous connecter à Pipedrive. Indiquez votre nom d'utilisateur et votre mot de passe, puis choisissez Se connecter.
6. Une fois connecté, choisissez Continuer vers l'application. Votre connexion est maintenant prête à être utilisée.
7. Dans la configuration de votre AWS Glue tâche, fournissez `connectionName` une connexion réseau supplémentaire.

Lecture à partir d'entités Pipedrive

Conditions préalables

- Un objet Pipedrive à partir duquel vous souhaitez lire. Reportez-vous au tableau des entités prises en charge ci-dessous pour vérifier les entités disponibles.

Entités prises en charge

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Activités	Oui	Oui	Non	Oui	Oui
Type d'activité	Non	Non	Non	Oui	Non
Call Logs	Non	Non	Non	Oui	Non
Currencies	Oui	Oui	Non	Oui	Non
Deals	Oui	Oui	Oui	Oui	Oui
Leads	Oui	Oui	Oui	Oui	Non

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Lead Sources	Non	Oui	Non	Oui	Non
Lead Labels	Non	Non	Non	Non	Non
Remarques	Oui	Oui	Oui	Oui	Oui
Organisation	Oui	Oui	Non	Oui	Oui
Jeux d'autorisations	Oui	Non	Non	Oui	Non
Personnes	Oui	Oui	Oui	Oui	Oui
Pipelines	Non	Oui	Non	Oui	Non
Produits	Oui	Oui	Non	Oui	Oui
Roles	Non	Oui	Non	Oui	Non
Stages	Oui	Oui	Non	Oui	Non
Utilisateurs	Non	Non	Non	Oui	Non

Exemple

```
pipedrive_read= glueContext.create_dynamic_frame.from_options(
    connection_type="PIPEDRIVE",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "activites",
        "API_VERSION": "v1"
    }
```

Détails des entités et des champs Pipedrive

Liste des entités :

- Activités : https://developers.pipedrive.com/docs/api/v1/Activities
- Type d'activité : https://developers.pipedrive.com/docs/api/v1/ActivityTypes
- Journaux d'appels : https://developers.pipedrive.com/docs/api/v1/CallLogs
- Devises : https://developers.pipedrive.com/docs/api/v1/Currencies
- Offres : https://developers.pipedrive.com/docs/api/v1/Deals
- Prospects : https://developers.pipedrive.com/docs/api/v1/Leads
- Sources de plomb : https://developers.pipedrive.com/docs/api/v1/LeadSources
- Étiquettes en plomb : https://developers.pipedrive.com/docs/api/v1/LeadLabels
- Remarques : https://developers.pipedrive.com/docs/api/v1/Notes
- Organisations : https://developers.pipedrive.com/docs/api/v1/Organizations
- Ensembles d'autorisations : https://developers.pipedrive.com/docs/api/v1/PermissionSets
- Personnes : https://developers.pipedrive.com/docs/api/v1/Persons
- Canalisations : https://developers.pipedrive.com/docs/api/v1/Pipelines
- Produits : https://developers.pipedrive.com/docs/api/v1/Products
- Rôles : https://developers.pipedrive.com/docs/api/v1/Roles
- Étapes : https://developers.pipedrive.com/docs/api/v1/Stages
- Utilisateurs : https://developers.pipedrive.com/docs/api/v1/Users

Entité	Type de données	Opérateurs pris en charge
Activities, Deals, Notes, Organization, Persons et Products.	Date	'='
	Entier	'='
	String	'='
	Booléen	'='

Requêtes de partitionnement

Dans Pipedrive, un seul champ (due_date) de l'entité Activities prend en charge le partitionnement basé sur les champs. Il s'agit d'un champ Date.

Les options Spark supplémentaires `PARTITION_FIELD`, `LOWER_BOUND`, `UPPER_BOUND` et `NUM_PARTITIONS` peuvent être indiquées si vous souhaitez utiliser la simultanéité dans Spark. Avec ces paramètres, la requête d'origine serait divisée en `NUM_PARTITIONS` nombres de sous-requêtes pouvant être exécutées simultanément par les tâches Spark.

- `PARTITION_FIELD` : le nom du champ à utiliser pour partitionner la requête.
- `LOWER_BOUND` : une valeur limite inférieure inclusive du champ de partition choisi.

  Pour le champ Date, nous acceptons le format de date Spark utilisé dans les requêtes SQL Spark. Exemples de valeurs valides : `"2024-02-06"`.
- `UPPER_BOUND` : une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS` : nombre de partitions.

Exemple

```
pipedrive_read = glueContext.create_dynamic_frame.from_options(
    connection_type="PIPEDRIVE",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "activites",
        "API_VERSION": "v1",
        "PARTITION_FIELD": "due_date"
        "LOWER_BOUND": "2023-09-07T02:03:00.000Z"
        "UPPER_BOUND": "2024-05-07T02:03:00.000Z"
        "NUM_PARTITIONS": "10"
    }
```

Référence des options de connexion Pipedrive

Les options de connexion pour Pipedrive sont les suivantes :

- `ENTITY_NAME`(Chaîne) - (Obligatoire) Utilisé pour la lecture/l'écriture. Le nom de votre objet dans Pipedrive.

- `API_VERSION`(Chaîne) - (Obligatoire) Utilisé pour la lecture/l'écriture. Version d'API Rest Pipedrive que vous souhaitez utiliser. Exemple : v1.
- `INSTANCE_URL`(Chaîne) - (Obligatoire) URL de l'instance sur laquelle l'utilisateur souhaite exécuter les opérations. Exemple : v1.
- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.
- `PARTITION_FIELD`(Chaîne) - Utilisé pour la lecture. Champ à utiliser pour partitionner la requête.
- `LOWER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite inférieure inclusive du champ de partition choisi.
- `UPPER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS`(Entier) - Par défaut : 1. Utilisé pour la lecture. Nombre de partitions pour la lecture.

Limitations

Les restrictions du connecteur Pipedrive sont les suivantes :

- Pipedrive prend en charge le partitionnement basé sur les champs pour une seule entité (Activities).
- Pipedrive prend en charge le partitionnement basé sur les enregistrements pour les entités Activities, Deals, Notes, Persons, Organizations et Products.
- Dans l'entité Deals, le champ de statut en tant que filtre renverra tous les enregistrements si une valeur de filtre de valeur non valide est utilisée.
- Dans l'entité Deals, la commande avec plusieurs champs n'est pas prise en charge.
- Pour obtenir des données de performance, nous utilisons un AWS compte local. Cependant, en raison de la limitation de l'actualisation locale du jeton d'accès, la AWS Glue tâche de traitement de 1 Go de données échoue. Par conséquent, nous avons optimisé le test de performance avec 179 Mo de données, et les résultats ci-dessus sont basés sur cette optimisation. Néanmoins, nous avons observé qu'avec l'augmentation du nombre de partitions, le point de terminaison SaaS prend plus de temps qu'une partition unique. Nous avons consulté l'équipe d'assistance de Pipedrive

à propos de ce comportement, qui nous a informés que Pipedrive limitait silencieusement les demandes et retardait la réponse. Par conséquent, lorsque vous exécutez la AWS Glue tâche avec de grands ensembles de données ou que vous appelez plusieurs fois le même point de terminaison d'API, cela peut entraîner un problème de délai d'attente en raison de l'implémentation de l'API Pipedrive. Cependant, les temps de réponse du connecteur et du shim diminuent comme prévu en raison de l'augmentation du nombre de partitions.

Connexion à Productboard

Productboard est le système de gestion des produits qui aide les équipes produit à commercialiser les bons produits plus rapidement. Plus de 3 000 entreprises modernes axées sur les produits, comme Zendesk UiPath et Microsoft, utilisent Productboard pour comprendre ce dont les utilisateurs ont réellement besoin, hiérarchiser les prochaines étapes et rallier tout le monde autour de leur feuille de route.

Rubriques

- AWS Glue support pour Productboard
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration de Productboard
- Configuration de connexions Productboard
- Lecture à partir d'entités Productboard
- Options de connexion Productboard
- Création d'un compte Productboard
- Limitations

AWS Glue support pour Productboard

AWS Glue prend en charge Productboard comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser les tâches AWS Glue ETL pour interroger les données de Productboard.

Prise en charge en tant que cible ?

Non.

Versions de l'API Productboard prises en charge

v1

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les AWS autorisations requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Si vous ne voulez pas utiliser la méthode ci-dessus, vous pouvez également utiliser les politiques IAM gérées suivantes :

- AWSGlueServiceRole — Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess — Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la

convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration de Productboard

Avant de pouvoir AWS Glue utiliser le transfert depuis Productboard, vous devez satisfaire aux exigences suivantes :

Configuration requise

- Vous disposez d'un compte Productboard avec adresse e-mail et mot de passe. Pour plus d'informations sur la création d'un compte, consultez Creating a Productboard account.
- Vous devez avoir créé AWS un compte avec l'accès au service à AWS Glue.
- Vous disposez des informations d'authentification d'un compte Productboard : soit un jeton JWT si vous souhaitez utiliser une authentification personnalisée, soit un identifiant client et un code secret si vous souhaitez utiliser .0. OAuth2
- Si l'utilisateur souhaite l'utiliser`OAuth2.0`, enregistrez votre application auprès de Productboard et configurez-la en suivant les instructions de la section Comment intégrer Productboard via OAuth2 la documentation destinée aux développeurs.

Si vous répondez à ces exigences, vous êtes prêt à vous connecter AWS Glue à votre compte Productboard. Pour les connexions classiques, vous n'avez rien d'autre à faire dans Productboard.

Configuration de connexions Productboard

Productboard prend en charge l'authentification personnalisée et `OAuth2.0`. Pour `OAuth2.0`, Productboard prend en charge le type d'autorisation `AUTHORIZATION_CODE`.

- Ce type d'autorisation est considéré comme `OAuth` tridimensionnel, car il repose sur la redirection des utilisateurs vers le serveur d'autorisation tiers pour authentifier l'utilisateur. Il est utilisé lors de la création de connexions via la AWS Glue console. L'utilisateur qui crée une connexion peut par défaut s'appuyer sur une application connectée AWS Glue dont il est propriétaire et où il n'a pas besoin de fournir d'informations `OAuth` connexes, à l'exception de son identifiant client Productboard et de son secret client. La AWS Glue console redirige l'utilisateur vers Productboard où il doit se connecter et autoriser AWS Glue les autorisations demandées pour accéder à son instance Productboard.

- Les utilisateurs peuvent toujours choisir de créer leur propre application connectée dans Productboard et de fournir leur propre identifiant client et leur propre secret client lors de la création de connexions via la AWS Glue console. Dans ce scénario, ils seront toujours redirigés vers Productboard pour se connecter et autoriser l'accès AWS Glue à leurs ressources.
- Ce type d'autorisation donne lieu à un jeton d'actualisation et à un jeton d'accès. Le jeton d'accès est de courte durée et peut être actualisé automatiquement sans interaction de l'utilisateur à l'aide du jeton d'actualisation.
- Pour consulter la documentation publique de Productboard sur la création d'une application connectée pour `AUTHORIZATION_CODE OAuth` Flow, consultez Comment intégrer à Productboard via OAuth2 la documentation destinée aux développeurs.

Pour configurer une connexion Productboard :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes :
   - Pour l'authentification `OAuth` : pour l'application connectée gérée par le client, le secret doit contenir le secret du consommateur de l'application connectée avec `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` comme clé.
   - Pour `Custom auth` : pour l'application connectée gérée par le client, le secret doit contenir le `JWT token` de l'application connectée avec `access_token` comme clé.

   > **Note**
   >
   > Il est indispensable de créer un secret par connexion dans AWS Glue.

2. Dans AWS Glue Studio, créez une connexion sous Connexions de données en suivant les étapes ci-dessous :
   a. Lorsque vous sélectionnez une source de données, sélectionnez Productboard.
   b. Sélectionnez le rôle IAM pour lequel AWS Glue vous pouvez assumer et qui dispose des autorisations nécessaires pour effectuer les actions suivantes :

      JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
```

```
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

c. Sélectionnez le type d'authentification pour vous connecter à une source de données :

- Pour l'authentification `OAuth` : indiquez le `Token URL` et l'`User Managed Client Application ClientId` de l'application Productboard.

d. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.

e. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.

3. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.
4. Dans la configuration de votre AWS Glue tâche, fournissez `connectionName` une connexion réseau supplémentaire.

Lecture à partir d'entités Productboard

Conditions préalables

Un objet Productboard à partir duquel vous souhaitez lire. Reportez-vous au tableau des entités prises en charge ci-dessous pour vérifier les entités disponibles.

Entités prises en charge

- Abuse-reports
- Automation
- Campagnes
- Click-details
- Lists
- Membres

- Open-details
- Segments
- Stores
- Unsubscribed

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Caractéri stiques	Oui	Oui	Non	Oui	Oui
Éléments	Non	Oui	Non	Oui	Non
Produits	Non	Oui	Non	Oui	Non
Statuts des fonctionn alités	Non	Oui	Non	Oui	Oui
Définitions de champs personnalisés	Non	Oui	Non	Oui	Non
Valeurs de champs personnalisés	Oui	Oui	Non	Oui	Non

Exemple

```
Productboard_read = glueContext.create_dynamic_frame.from_options(
    connection_type="Productboard",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "feature",
        "API_VERSION": "1"
    }
```

Détails des entités et des champs Productboard

- Fonctions
- Composants
- Statuts des fonctionnalités
- Produits
- Définitions de champs personnalisés
- Valeurs de champs personnalisés

Options de connexion Productboard

Les options de connexion pour Productboard sont les suivantes :

- `ENTITY_NAME`(Chaîne) - (Obligatoire) Utilisé pour la lecture/l'écriture. Le nom de votre objet dans Productboard.
- `API_VERSION`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Version d'API Rest de Productboard Engage que vous souhaitez utiliser. Par exemple : 3.0.
- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.

Création d'un compte Productboard

1. Accédez à la page de connexion à Productboard, saisissez votre adresse e-mail et votre mot de passe, puis choisissez M'inscrire.
2. Dans le champ Nom du compte, saisissez le nom de votre compte Productboard, puis cochez la case J'accepte la politique de confidentialité.
3. Sur la page Créez à présent votre espace de travail, dans le champ URL de l'espace de travail, entrez l'URL de votre nouvel espace de travail. Choisissez ensuite Continuer pour passer à la page suivante et fournir les informations restantes.

   Cela permet de créer votre compte d'essai. Le compte d'essai est gratuit pendant 15 jours. Une fois votre période d'essai expirée, vous pouvez acheter un forfait payant. Notez votre

adresse e-mail, votre mot de passe et l'URL de votre espace de travail. Vous aurez besoin de ces informations pour accéder à votre compte à l'avenir.

Enregistrement d'une application **`OAuth2.0`**

1. Accédez à la page de connexion à Productboard, saisissez votre adresse e-mail et votre mot de passe, puis choisissez Se connecter.
2. Sélectionnez l'icône Utilisateur dans le coin supérieur droit, puis choisissez Compte et facturation dans le menu déroulant.
3. Sélectionnez Extras et choisissez Applications enregistrées dans le menu déroulant.
4. Recherchez et choisissez Enregistrer une application.
5. Entrez les détails suivants :
   - Nom de l'application : nom de l'application.
   - Entreprise/organisation : nom de votre entreprise ou organisation.
   - Site Web de l'application : site Web de l'application.
   - URI de redirection : un modèle d'URI de redirection est un chemin d'URI (ou une liste de chemins séparés par des virgules) vers lequel Productboard peut rediriger (si demandé) une fois le flux de connexion terminé. Par exemple, `https://ap-southeast-2\\.console\\.aws\\.amazon\\.com`
6. Choisissez Créer.
7. L'ID client et le secret client seront désormais visibles. Copiez-les et enregistrez-les dans un endroit sûr. Ensuite, choisissez Terminé.

> **Note**
>
> Vos chaînes d'ID client et de secret client sont des informations d'identification utilisées pour établir une connexion avec ce connecteur lors de l'utilisation de AppFlow ou AWS Glue.

Récupération des informations d'identification CustomAuth

1. Accédez à la page de connexion à Productboard, saisissez votre adresse e-mail et votre mot de passe, puis choisissez Se connecter.

Vous serez ensuite redirigé vers la page d'accueil.

2. Sur la page d'accueil, accédez à Paramètres de l'espace de travail > Intégrations > Public APIs > Jeton d'accès.

> **Note**
>
> Si la APIs section publique n'est pas visible, votre compte est peut-être associé au forfait Essentials. L'accès aux jetons d'API nécessite au minimum un forfait Pro. Les fonctionnalités et les noms des forfaits sont susceptibles d'être modifiés. Pour plus d'informations sur les packages, consultez Productboard pricing.

3. Choisissez + pour générer un jeton et assurez-vous de le stocker en toute sécurité pour une référence future.

Création d'informations d'identification **`OAuth2.0`**

Pour utiliser l'authentification `OAuth2.0` avec le connecteur Productboard, vous devez enregistrer votre application sur la plateforme Productboard et générer un `Client ID` et un `Client Secret`.

1. Accédez à la page de connexion à Productboard, saisissez votre adresse e-mail et votre mot de passe, puis choisissez Se connecter.
2. Pour enregistrer une nouvelle OAuth2 application avec votre compte Productboard, rendez-vous sur la page Producboard.
3. Complétez les champs obligatoires et sélectionnez les portées nécessaires pour chaque entité à laquelle vous souhaitez accéder.

> **Note**
>
> Vous avez choisi les quatre portées suivantes, qui sont obligatoires pour les six entités prises en charge.

4. L'URL de redirection doit être au format suivant : `https://ap-southeast-2\\.console\\.aws\\.amazon\\.com`.

> **Note**
>
> Les redirections Appflow URLs sont susceptibles d'être modifiées. Une fois disponible, veuillez mettre à jour la redirection URLs pour la AWS Glue plateforme.

5. L'ID client et le secret client seront désormais visibles. Copiez-les et enregistrez-les dans un endroit sûr.
6. Vous pouvez configurer et vérifier `OAuth2` en suivant les étapes décrites dans la section Comment intégrer à Productboard via la documentation destinée aux OAuth2 développeurs.

Limitations

Les restrictions du connecteur Productboard sont les suivantes :

- Productboard ne prend en charge ni le partitionnement basé sur les champs, ni le partitionnement basé sur les enregistrements.

Connexion à QuickBooks

QuickBooks est une application comptable de premier plan pour les petites et moyennes entreprises. QuickBooks les applications comptables remontent aux années 1980 en tant que l'un des premiers produits d'Intuit. Il s'agissait donc à l'origine d'un logiciel de bureau. Aujourd'hui, QuickBooks propose plusieurs applications comptables et financières commerciales sous forme de logiciels installables et de logiciels SaaS basés sur le cloud. En tant qu' QuickBooks utilisateur, vous pouvez vous connecter AWS Glue à votre QuickBooks compte. Vous pouvez ensuite l'utiliser QuickBooks comme source de données dans vos tâches ETL. Exécutez ces tâches pour transférer des données entre QuickBooks des AWS services ou d'autres applications prises en charge.

Rubriques

- AWS Glue support pour QuickBooks
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration QuickBooks
- Configuration des QuickBooks connexions
- Lecture à partir d' QuickBooks entités
- QuickBooks options de connexion

- Limitations et remarques relatives au QuickBooks connecteur

AWS Glue support pour QuickBooks

AWS Glue prend en charge QuickBooks les éléments suivants :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser des tâches AWS Glue ETL pour interroger des données à partir de QuickBooks.

Prise en charge en tant que cible ?

Non.

Versions QuickBooks d'API prises en charge

Les versions QuickBooks d'API suivantes sont prises en charge :

- v3

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les autorisations AWS IAM requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
```

```
  ]
}
```

Si vous ne voulez pas utiliser la méthode ci-dessus, vous pouvez également utiliser les politiques IAM gérées suivantes :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration QuickBooks

Avant de pouvoir AWS Glue transférer des données depuis QuickBooks, vous devez satisfaire aux exigences suivantes :

Configuration requise

Les exigences minimales sont les suivantes :

- Tu as un QuickBooks compte.
- Votre QuickBooks compte est activé pour accéder à l'API.

Pour plus d'informations, consultez les rubriques suivantes de la QuickBooks documentation :

- Créer un compte Intuit
- Créer et commencer à développer votre application

Si vous remplissez ces conditions, vous êtes prêt à vous connecter AWS Glue à votre QuickBooks compte. Pour les connexions classiques, vous n'avez rien d'autre à faire dans QuickBooks.

Configuration des QuickBooks connexions

QuickBooks prend en charge le type de subvention AUTHORIZATION_CODE pour. OAuth2 Le type de subvention détermine le mode de AWS Glue communication avec QuickBooks lequel vous pouvez demander l'accès à vos données.

- Ce type de subvention est considéré comme « à trois étapes » OAuth car il repose sur la redirection des utilisateurs vers un serveur d'autorisation tiers pour authentifier l'utilisateur. Il est utilisé lors de la création de connexions via la AWS Glue console.
- Les utilisateurs peuvent toujours choisir de créer leur propre application connectée QuickBooks et de fournir leur propre identifiant client et leur propre secret client lors de la création de connexions via la AWS Glue console. Dans ce scénario, ils seront toujours redirigés QuickBooks vers pour se connecter et autoriser AWS Glue l'accès à leurs ressources.
- Ce type d'autorisation donne lieu à un jeton d'actualisation et à un jeton d'accès. Le jeton d'accès est de courte durée et peut être actualisé automatiquement sans interaction de l'utilisateur à l'aide du jeton d'actualisation.
- Pour consulter QuickBooks la documentation publique sur la création d'une application connectée pour le OAuth flux de code d'autorisation, voir Configuration OAuth 2.0.

Pour configurer une QuickBooks connexion :

1. Dans AWS Glue Glue Studio, créez une connexion sous Data Connections en suivant les étapes ci-dessous :
   a. Lorsque vous sélectionnez un type de connexion, sélectionnez QuickBooks.
   b. Fournissez l'URL de l'instance et l'ID d'entreprise de l' QuickBooks instance à laquelle vous souhaitez vous connecter.
   c. Sélectionnez le rôle AWS IAM qui AWS Glue peut assumer et dispose des autorisations nécessaires pour les actions suivantes :

      JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
```

```
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

d. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.

e. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.

2. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.

Lecture à partir d' QuickBooks entités

Prérequis

Un QuickBooks objet que vous aimeriez lire.

Entités prises en charge pour la source :

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Compte	Oui	Oui	Oui	Oui	Oui
Bill	Oui	Oui	Oui	Oui	Oui
Company Info	Non	Non	Non	Oui	Non
Client	Oui	Oui	Oui	Oui	Oui
Employee	Oui	Oui	Oui	Oui	Oui
Estimate	Oui	Oui	Oui	Oui	Oui

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Invoice	Oui	Oui	Oui	Oui	Oui
Élément	Oui	Oui	Oui	Oui	Oui
Payment	Oui	Oui	Oui	Oui	Oui
Préférences	Non	Non	Non	Oui	Non
Profit and Loss	Oui	Non	Non	Oui	Non
Tax Agency	Oui	Oui	Oui	Oui	Oui
Vendors	Oui	Oui	Oui	Oui	Oui

Exemple :

```
QuickBooks_read = glueContext.create_dynamic_frame.from_options(
    connection_type="quickbooks",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "Account",
        "API_VERSION": "v3"
    }
```

QuickBooks détails de l'entité et du champ :

Pour plus d'informations sur les entités et de détails sur les champs, consultez :

- Compte
- Bill
- CompanyInfo
- Client
- Employee
- Estimate

- Invoice
- Elément
- Payment
- Préférences
- ProfitAndLoss
- TaxAgency
- Fournisseur

Requêtes de partitionnement

Partitionnement basé sur les champs :

Dans QuickBooks, les champs Integer et DateTime datatype prennent en charge le partitionnement basé sur les champs.

Vous pouvez indiquer les options Spark supplémentaires `PARTITION_FIELD`, `LOWER_BOUND`, `UPPER_BOUND` et `NUM_PARTITIONS` si vous souhaitez utiliser la simultanéité dans Spark. Avec ces paramètres, la requête d'origine serait divisée en `NUM_PARTITIONS` nombres de sous-requêtes pouvant être exécutées simultanément par les tâches Spark.

- `PARTITION_FIELD` : le nom du champ à utiliser pour partitionner la requête.
- `LOWER_BOUND` : une valeur limite inférieure inclusive du champ de partition choisi.

  Pour le champ Datetime, nous acceptons le format d'horodatage Spark utilisé dans les requêtes SQL Spark.

  Exemples de valeurs valides :

  ```
  "2024-05-07T02:03:00.00Z"
  ```

- `UPPER_BOUND` : une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS` : le nombre de partitions.

Exemple :

```
QuickBooks_read = glueContext.create_dynamic_frame.from_options(
```

```
    connection_type="quickbooks",
    connection_options={
        "connectionName": "connectionName",
        "REALMID": "12345678690123456789",
        "ENTITY_NAME": "Account",
        "API_VERSION": "v3",
        "PARTITION_FIELD": "MetaData_CreateTime"
        "LOWER_BOUND": "2023-09-07T02:03:00.000Z"
        "UPPER_BOUND": "2024-05-07T02:03:00.000Z"
        "NUM_PARTITIONS": "10"
    }
```

Partitionnement basé sur des enregistrements :

La requête d'origine est divisée en NUM_PARTITIONS nombres de sous-requêtes qui peuvent être exécutées simultanément par les tâches Spark :

- NUM_PARTITIONS : le nombre de partitions.

Exemple :

```
QuickBooks_read = glueContext.create_dynamic_frame.from_options(
    connection_type="quickbooks",
    connection_options={
        "connectionName": "connectionName",
        "REALMID": "1234567890123456789",
        "ENTITY_NAME": "Bill",
        "API_VERSION": "v3",
        "NUM_PARTITIONS": "10"
    }
```

QuickBooks options de connexion

Les options de connexion suivantes sont disponibles pour QuickBooks :

- ENTITY_NAME(Chaîne) - (Obligatoire) Utilisé pour la lecture. Le nom de votre objet dans QuickBooks.
- INSTANCE_URL(String) - (Obligatoire) Une URL d' QuickBooks instance valide.
- API_VERSION(Chaîne) - (Obligatoire) Utilisé pour la lecture. QuickBooks Version de l'API Rest que vous souhaitez utiliser.

- `REALM_ID`(Chaîne) - Un identifiant identifiant une entreprise QuickBooks en ligne individuelle à laquelle vous envoyez des demandes.
- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.
- `PARTITION_FIELD`(Chaîne) - Utilisé pour la lecture. Champ à utiliser pour partitionner la requête.
- `LOWER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite inférieure inclusive du champ de partition choisi.
- `UPPER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS`(Entier) - Par défaut : 1. Utilisé pour la lecture. Nombre de partitions pour la lecture.

Limitations et remarques relatives au QuickBooks connecteur

Les limitations ou remarques suivantes concernent le QuickBooks connecteur :

- Dans l'API `taxAgency`, le filtrage Order by ne fonctionne pas comme prévu.

Connexion à une API REST

AWS Glue vous permet de configurer un AWS Glue ConnectionType qui peut être utilisé pour vous connecter AWS Glue à n'importe quelle source de données basée sur l'API REST. Cela peut être utilisé comme source de données dans vos tâches ETL. Vous pouvez exécuter ces tâches pour transférer des données entre la source de données basée sur l'API REST et les AWS services ou autres applications prises en charge.

Rubriques

- AWS Glue support pour l'API REST
- Politiques contenant les opérations d'API pour l'enregistrement des types de connexion et des creating/using connexions
- Configuration d'une API REST ConnectionType
- Configuration d'une connexion à l'API REST

- Tutoriel : Création d'une API REST ConnectionType et d'une connexion
- Limitations

AWS Glue support pour l'API REST

AWS Glue prend en charge l'API REST comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser des tâches AWS Glue ETL pour interroger des données à partir d'une source de données basée sur l'API REST.

Prise en charge en tant que cible ?

Non.

Politiques contenant les opérations d'API pour l'enregistrement des types de connexion et des creating/using connexions

L'exemple de politique IAM suivant décrit les autorisations requises pour l'enregistrement, la création, la gestion et l'utilisation des connexions à l'API REST dans les tâches AWS Glue ETL. Si vous créez un rôle, créez une politique contenant les éléments suivants :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:RegisterConnectionType",
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:CreateConnection",
                "glue:RefreshOAuth2Tokens",
                "glue:ListEntities",
                "glue:DescribeEntity"
            ],
            "Resource": "*"
        }
    ]
}
```

Vous pouvez également utiliser les politiques IAM suivantes pour autoriser l'accès :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Si vous fournissez des options réseau lors de la création d'une connexion à l'API REST, les actions suivantes doivent également être incluses dans le rôle IAM :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DeleteNetworkInterface"
            ],
            "Resource": "*"
        }
    ]
}
```

Configuration d'une API REST ConnectionType

Avant de pouvoir AWS Glue transférer des données depuis la source de données basée sur l'API REST, vous devez satisfaire aux exigences suivantes :

Configuration requise

Les exigences minimales sont les suivantes :

- Vous avez configuré et enregistré un type de connexion à l' AWS Glue API REST. Consultez la section Connexion à REST APIs.
- Si vous utilisez les informations d'identification du OAuth2 client, le code d'autorisation ou JWT, configurez l'application client en conséquence.

Si vous répondez à ces exigences, vous êtes prêt à vous connecter AWS Glue à votre source de données basée sur l'API REST. En général, aucune autre configuration n'est nécessaire du côté de l'API REST.

Configuration d'une connexion à l'API REST

Pour configurer un connecteur d' AWS Glue API REST, vous devez configurer un type de AWS Glue connexion. Ce type de connexion contient des détails sur les propriétés du fonctionnement de la source de données REST et interprète des éléments tels que l'authentification, les demandes, les réponses, la pagination, les validations et les entités/métadonnées. Pour obtenir la liste complète des propriétés requises pour un type de connexion AWS Glue REST, consultez l' RegisterConnectionTypeAPI et les étapes de connexion à REST APIs.

Lors de la création du connecteur d'API REST, la politique suivante est nécessaire pour autoriser les actions pertinentes :

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:RegisterConnectionType",
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:CreateConnection",
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DeleteNetworkInterface"
```

```
            ],
            "Resource": "*"
        }
    ]
}
```

Tutoriel : Création d'une API REST ConnectionType et d'une connexion

Connexion à l'API REST Foo

Nous allons créer une AWS Glue API REST ConnectionType et une AWS Glue connexion correspondante pour l'API Foo REST. Cette API possède les propriétés suivantes (qui peuvent être extraites de la documentation de l'API REST).

- URL de l'instance : https://foo.cloud.com/rest/ v1.
- Type d'authentification : OAuth2 (informations d'identification du client).
- Méthode REST : GET.
- Type de pagination : Offset avec les propriétés « limit » et « offset » placées dans le paramètre de requête de la requête.
- Entités prises en charge :
  - Barre : chemin relatif [/bar.json].
  - Base : chemin relatif [/baz.json].

Une fois tous les détails obtenus, nous pouvons commencer à créer la AWS Glue connexion à l'API Foo REST.

Pour créer une connexion à l'API REST :

1. Créez le type de connexion à l'API REST en AWS Glue appelant l' RegisterConnectionType API à l'aide de l' AWS API, de la CLI ou du SDK. Cela créera une nouvelle ConnectionType ressource dans AWS Glue.

   ```
   {
       "ConnectionType": "REST-FOO-CONNECTOR",
       "IntegrationType": "REST",
       "Description": "AWS Glue Connection Type for the FOO REST API",
       "ConnectionProperties": {
           "Url": {
               "Name": "Url",
   ```

```
            "Required": true,
            "DefaultValue": "https://foo.cloud.com/rest/v1",
            "PropertyType": "USER_INPUT"
        }
    },
    "ConnectorAuthenticationConfiguration": {
        "AuthenticationTypes": ["OAUTH2"],
        "OAuth2Properties": {
            "OAuth2GrantType": "CLIENT_CREDENTIALS"
        }
    },
    "RestConfiguration": {
        "GlobalSourceConfiguration": {
        "RequestMethod": "GET",
        "ResponseConfiguration": {
            "ResultPath": "$.result",
            "ErrorPath": "$.error.message"
        },
        "PaginationConfiguration": {
            "OffsetConfiguration": {
                "OffsetParameter": {
                    "Key": "offset",
                    "PropertyLocation": "QUERY_PARAM"
                },
                "LimitParameter": {
                    "Key": "limit",
                    "PropertyLocation": "QUERY_PARAM",
                    "DefaultValue": "50"
                }
            }
        }
    },
    "ValidationEndpointConfiguration": {
        "RequestMethod": "GET",
        "RequestPath": "/bar.json?offset=1&limit=10"
    },
    "EntityConfigurations": {
        "bar": {
            "SourceConfiguration": {
                "RequestMethod": "GET",
                "RequestPath": "/bar.json",
                "ResponseConfiguration": {
                    "ResultPath": "$.result",
                    "ErrorPath": "$.error.message"
```

```
                }
            },
            "Schema": {
                "name": {
                    "Name": "name",
                    "FieldDataType": "STRING"
                },
                "description": {
                    "Name": "description",
                    "FieldDataType": "STRING"
                },
                "id": {
                    "Name": "id",
                    "FieldDataType": "STRING"
                },
                "status": {
                    "Name": "status",
                    "FieldDataType": "STRING"
                }
            }
        }
    }
}
}
```

2. Dans AWS Secrets Manager, créez un secret. Le secret doit contenir le secret du consommateur de l'application connectée avec `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` comme clé.

> **Note**
>
> Vous devez créer un secret par connexion dans AWS Glue

3. Créez la AWS Glue connexion en appelant l' CreateConnection API à l'aide de l' AWS API, de la CLI ou du SDK.
   a. Référencez le nom du type de connexion REST indiqué à l'étape 1 sous la forme « ConnectionType ».
   b. Indiquez le InstanceUrl et tout autre ConnectionProperties élément défini au cours du processus AWS Glue ConnectionType d'enregistrement.
   c. Choisissez parmi les types d'authentification configurés. L'API REST utilisée par Foo OAuth2 avec le type de ClientCredentials subvention.

d. Fournissez le SecretArnet les autres AuthenticationPropertiesqui sont configurés. Par exemple, nous avons configuré 0AUTH2 comme suit : AuthenticationType nous allons définir les « OAuth2 Propriétés » dans le CreateConnectionInput. Cela nécessitera des propriétés telles que « OAuth2 GrantType », « TokenUrl » et « OAuth2 ClientApplication ».

4. Faites la CreateConnection demande qui créera la AWS Glue connexion.

```
{
    "ConnectionInput": {
        "Name": "ConnectionFooREST",
        "ConnectionType": "REST-FOO-CONNECTOR",
        "ConnectionProperties": {},
        "ValidateCredentials": true,
        "AuthenticationConfiguration": {
            "AuthenticationType": "OAUTH2",
            "SecretArn":
 "arn:aws:secretsmanager:<region>:<accountId>:secret:<secretId>",
            "OAuth2Properties": {
                "OAuth2GrantType": "CLIENT_CREDENTIALS",
                "TokenUrl": "https://foo.cloud.com/oauth/token",
                "OAuth2ClientApplication": {
                    "UserManagedClientApplicationClientId": "your-managed-client-id"
                }
            }
        }
    }
}
```

Limitations

Les limites du connecteur d'API REST sont les suivantes :

- Le connecteur d'API REST est uniquement disponible via l' AWS API, la CLI ou le SDK. Vous ne pouvez pas configurer les connecteurs REST via la console.
- Le AWS Glue REST ne ConnectionType peut être configuré que pour LIRE les données de la source de données basée sur l'API REST. La connexion ne peut être utilisée que comme SOURCE dans les tâches AWS Glue ETL.
- Le filtrage et le partitionnement ne sont pas pris en charge.
- La sélection de champs n'est pas prise en charge.

Connexion à Salesforce

Salesforce fournit un logiciel de gestion de la relation client (CRM) qui vous aide dans les domaines des ventes, du service client, du commerce électronique, etc. Si vous êtes un utilisateur de Salesforce, vous pouvez vous connecter AWS Glue à votre compte Salesforce. Vous pouvez ensuite utiliser Salesforce comme source de données ou comme destination dans vos tâches ETL. Exécutez ces tâches pour transférer des données entre Salesforce et les AWS services ou d'autres applications prises en charge.

Rubriques

- AWS Glue support pour Salesforce
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration de Salesforce
- Application du profil Administrateur système
- Configuration des connexions Salesforce
- Lecture à partir de Salesforce
- Écriture vers Salesforce
- Options de connexion à Salesforce
- Restrictions du connecteur Salesforce
- Configuration du flux de code d'autorisation pour Salesforce
- Configurer le OAuth flux de porteurs JWT pour Salesforce

AWS Glue support pour Salesforce

AWS Glue prend en charge Salesforce comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser les tâches AWS Glue ETL pour interroger les données de Salesforce.

Prise en charge en tant que cible ?

Oui. Vous pouvez utiliser des tâches AWS Glue ETL pour écrire des enregistrements dans Salesforce.

Versions de l'API Salesforce prises en charge

Les versions suivantes de l'API Salesforce sont prises en charge :

- v58.0
- v59.0
- v60.0

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique IAM suivant décrit les autorisations requises pour créer, gérer et utiliser des connexions Salesforce dans les tâches AWS Glue ETL. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Vous pouvez également utiliser les politiques IAM suivantes pour autoriser l'accès :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources

spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.

- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Si vous indiquez des options réseau lors de la création d'une connexion Salesforce, les actions suivantes doivent également être incluses dans le rôle IAM :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

Pour les connexions Salesforce zéro ETL, consultez Zero-ETL prerequisites.

Pour les connexions Salesforce zéro ETL, consultez Zero-ETL prerequisites.

Configuration de Salesforce

Avant de pouvoir transférer des données AWS Glue vers ou depuis Salesforce, vous devez répondre aux exigences suivantes :

Configuration requise

Les exigences minimales sont les suivantes :

- Vous disposez d’un compte Salesforce.
- Votre compte Salesforce est activé pour accéder à l’API. L’accès à l’API est activé par défaut pour les éditions Enterprise, Unlimited, Developer et Performance.

Si vous répondez à ces exigences, vous êtes prêt à vous connecter AWS Glue à votre compte Salesforce. AWS Glue gère les autres exigences avec l'application connectée AWS gérée.

L'application connectée AWS gérée pour Salesforce

L'application connectée AWS gérée vous permet de créer une connexion Salesforce en moins d'étapes. Dans Salesforce, une application connectée est un framework qui autorise des applications externes, par exemple AWS Glue, à accéder à vos données Salesforce à l'aide de la OAuth version 2.0. Pour utiliser l'application connectée AWS gérée, créez une connexion Salesforce à l'aide de la AWS Glue console. Lorsque vous configurez la connexion, définissez le OAuth type d'autorisation sur Code d'autorisation et laissez la case Utiliser une application client AWS gérée cochée.

Lorsque vous enregistrez la connexion, vous serez redirigé vers Salesforce pour vous connecter et approuver AWS Glue l'accès à votre compte Salesforce.

Application du profil Administrateur système

Dans Salesforce, suivez les étapes pour appliquer le profil Administrateur système :

1. Dans Salesforce, accédez à Paramètres **>** Applications connectées **>** OAuth Utilisation des applications connectées.
2. Dans la liste des applications connectées, recherchez AWS Glue et choisissez Installer. Si nécessaire, choisissez Débloquer.
3. Accédez à Paramètres **>** Gérer les applications connectées, puis choisissez AWS Glue. Sous OAuth Politiques, choisissez Administrateur, les utilisateurs approuvés sont préautorisés et sélectionnez le profil d'administrateur système. Cette action restreint l'accès aux AWS Glue seuls utilisateurs possédant le profil d'administrateur système.

Application du profil Administrateur système

Dans Salesforce, suivez les étapes pour appliquer le profil Administrateur système :

1. Dans Salesforce, accédez à Paramètres > Applications connectées > OAuth Utilisation des applications connectées.
2. Dans la liste des applications connectées, recherchez AWS Glue et choisissez Installer. Si nécessaire, choisissez Débloquer.
3. Accédez à Paramètres > Gérer les applications connectées, puis choisissez AWS Glue. Sous OAuth Politiques, choisissez Administrateur, les utilisateurs approuvés sont préautorisés et sélectionnez le profil d'administrateur système. Cette action restreint l'accès aux AWS Glue seuls utilisateurs possédant le profil d'administrateur système.

Configuration des connexions Salesforce

Pour configurer une connexion Salesforce :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes :
   a. Pour le type d'autorisation JWT_TOKEN, le secret doit contenir la clé JWT_TOKEN avec sa valeur.
   b. Pour le type de AuthorizationCode subvention :
      i. Pour une application connectée AWS gérée, un secret vide ou un secret avec une valeur temporaire doit être fourni.
      ii. Pour une application connectée gérée par le client, le secret doit contenir l'application connectée `Consumer Secret` avec `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` comme clé.
   c. Remarque : Vous devez créer un secret pour votre connexion dans AWS Glue.
2. Dans AWS Glue Glue Studio, créez une connexion sous Data Connections en suivant les étapes ci-dessous :
   a. Lorsque vous sélectionnez un type de connexion, sélectionnez Salesforce.
   b. Indiquez la valeur INSTANCE_URL de l'instance Salesforce à laquelle vous souhaitez vous connecter.
   c. Indiquez l'environnement Salesforce.
   d. Sélectionnez le rôle AWS IAM qui AWS Glue peut assumer et dispose des autorisations nécessaires pour les actions suivantes :

      JSON

```
{
```

```
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

e. Sélectionnez le type de OAuth2 subvention que vous souhaitez utiliser pour les connexions. Le type de subvention détermine le mode AWS Glue de communication avec Salesforce pour demander l'accès à vos données. Votre choix influe sur les exigences que vous devez respecter avant de créer la connexion. Vous pouvez choisir l'un des types suivants :

- Type d'autorisation JWT_BEARER : ce type d'autorisation fonctionne bien pour les scénarios d'automatisation, car il permet de créer un jeton Web JSON (JWT) dès le départ avec les autorisations d'un utilisateur particulier dans l'instance Salesforce. Le créateur a le contrôle de la durée de validité du JWT. AWS Glue est capable d'utiliser le JWT pour obtenir un jeton d'accès qui est utilisé pour appeler Salesforce APIs.

  Ce flux nécessite que l'utilisateur ait créé une application connectée dans son instance Salesforce qui permet d'émettre des jetons d'accès basés sur JWT pour les utilisateurs.

  Pour plus d'informations sur la création d'une application connectée pour le OAuth flux porteur JWT, voir OAuth 2.0 JWT bearer flow pour l'intégration. server-to-server Pour configurer le flux de porteurs JWT avec l'application connectée Salesforce, consultez Configurer le OAuth flux de porteurs JWT pour Salesforce.

- Type de subvention AUTHORIZATION_CODE : Ce type de subvention est considéré comme « à trois étapes » OAuth car il repose sur la redirection des utilisateurs vers le serveur d'autorisation tiers pour authentifier l'utilisateur. Il est utilisé lors de la création de connexions via la AWS Glue console. L'utilisateur qui crée une connexion peut par défaut s'appuyer sur une application AWS Glue connectée (application client AWS Glue gérée) dans laquelle il n'a pas besoin de fournir d'informations OAuth connexes, à l'exception de l'URL de son instance

Salesforce. La AWS Glue console redirige l'utilisateur vers Salesforce où il doit se connecter et autoriser AWS Glue les autorisations demandées pour accéder à son instance Salesforce.

Les utilisateurs peuvent toujours choisir de créer leur propre application connectée dans Salesforce et de fournir leur propre identifiant client et leur secret client lors de la création de connexions via la AWS Glue console. Dans ce scénario, ils seront toujours redirigés vers Salesforce pour se connecter et autoriser AWS Glue l'accès à leurs ressources.

Ce type d'autorisation donne lieu à un jeton d'actualisation et à un jeton d'accès. Le jeton d'accès est de courte durée et peut être actualisé automatiquement sans interaction de l'utilisateur à l'aide du jeton d'actualisation.

Pour plus d'informations sur la création d'une application connectée pour le OAuth flux de code d'autorisation, consultezthe section called "Configuration du flux de code d'autorisation pour Salesforce".

f. Sélectionnez `secretName` celui que vous souhaitez utiliser pour cette connexion AWS Glue afin de stocker les jetons OAuth 2.0.

g. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.

3. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.

4. Si vous indiquez des options réseau, accordez également au rôle IAM les autorisations suivantes :

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

Configuration des connexions Salesforce avec la AWS CLI

Vous pouvez créer des connexions Salesforce à l'aide de la AWS CLI :

```
aws glue create-connection --connection-input \
"{\"Name\": \"salesforce-conn1\",\"ConnectionType\": \"SALESFORCE\",
\"ConnectionProperties\": {\"ROLE_ARN\": \"arn:aws:iam::123456789012:role/glue-role\",
\"INSTANCE_URL\": \"https://example.my.salesforce.com\"},\"ValidateCredentials\": true,
\"AuthenticationConfiguration\": {\"AuthenticationType\": \"OAUTH2\",\"SecretArn\":
 \"arn:aws:secretsmanager:us-east-1:123456789012:secret:salesforce-conn1-secret-IAmcdk
\",\"OAuth2Properties\": {\"OAuth2GrantType\": \"JWT_BEARER\",\"TokenUrl\": \"https://
login.salesforce.com/services/oauth2/token\"}}}" \
--endpoint-url https://glue.us-east-1.amazonaws.com \
--region us-east-1
```

Lecture à partir de Salesforce

Prérequis

Un objet Salesforce à partir duquel vous souhaitez lire. Vous aurez besoin du nom de l'objet, comme `Account`, `Case` ou `Opportunity`.

Exemple :

```
salesforce_read = glueContext.create_dynamic_frame.from_options(
    connection_type="salesforce",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "Account",
        "API_VERSION": "v60.0"
    }
)
```

Requêtes de partitionnement

Vous pouvez indiquer les options Spark supplémentaires `PARTITION_FIELD`, `LOWER_BOUND`, `UPPER_BOUND` et `NUM_PARTITIONS` si vous souhaitez utiliser la simultanéité dans Spark. Avec ces paramètres, la requête d'origine serait divisée en `NUM_PARTITIONS` nombres de sous-requêtes pouvant être exécutées simultanément par les tâches Spark.

- `PARTITION_FIELD` : le nom du champ à utiliser pour partitionner la requête.

- LOWER_BOUND : une valeur limite inférieure inclusive du champ de partition choisi.

  Pour les champs Date ou Timestamp, le connecteur accepte le format d'horodatage Spark utilisé dans les requêtes SQL Spark.

  Exemples de valeurs valides :

  ```
  "TIMESTAMP \"1707256978123\""
  "TIMESTAMP '2018-01-01 00:00:00.000 UTC'"
  "TIMESTAMP \"2018-01-01 00:00:00 Pacific/Tahiti\""
  "TIMESTAMP \"2018-01-01 00:00:00\""
  "TIMESTAMP \"-123456789\" Pacific/Tahiti"
  "TIMESTAMP \"1702600882\""
  ```

- UPPER_BOUND : une valeur limite supérieure exclusive du champ de partition choisi.
- NUM_PARTITIONS : le nombre de partitions.
- TRANSFER_MODE : prend en charge deux modes : SYNC et ASYNC. La valeur par défaut est SYNC. Lorsque cette option est définie sur ASYNC, la requête Bulk API 2.0 sera utilisée pour le traitement.

Exemple :

```
salesforce_read = glueContext.create_dynamic_frame.from_options(
    connection_type="salesforce",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "Account",
        "API_VERSION": "v60.0",
        "PARTITION_FIELD": "SystemModstamp",
        "LOWER_BOUND": "TIMESTAMP '2021-01-01 00:00:00 Pacific/Tahiti'",
        "UPPER_BOUND": "TIMESTAMP '2023-01-10 00:00:00 Pacific/Tahiti'",
        "NUM_PARTITIONS": "10",
        "TRANSFER_MODE": "ASYNC"
    }
)
```

Option FILTER_PREDICATE

FILTER_PREDICATE : il s'agit d'un paramètre facultatif. Cette option est utilisée pour le filtre de requêtes.

Exemples de FILTER_PREDICATE :

```
    Case 1: FILTER_PREDICATE with single criterion
    Examples:
      LastModifiedDate >= TIMESTAMP '2025-04-01 00:00:00 Pacific/Tahiti'
      LastModifiedDate <= TIMESTAMP "2025-04-01 00:00:00"
      LastModifiedDate >= TIMESTAMP '2018-01-01 00:00:00.000 UTC'
      LastModifiedDate <= TIMESTAMP "-123456789 Pacific/Tahiti"
      LastModifiedDate <= TIMESTAMP "1702600882"

    Case 2: FILTER_PREDICATE with multiple criteria
    Examples:
      LastModifiedDate >= TIMESTAMP '2025-04-01 00:00:00 Pacific/Tahiti' AND Id =
"0012w00001CotGiAAJ"
      LastModifiedDate >= TIMESTAMP "1702600882" AND Id = "001gL000002i26MQAQ"

    Case 3: FILTER_PREDICATE single criterion with LIMIT
    Examples:
      LastModifiedDate >= TIMESTAMP "1702600882" LIMIT 2

    Case 4: FILTER_PREDICATE with LIMIT
    Examples:
      LIMIT 2
```

Écriture vers Salesforce

Conditions préalables

Un objet Salesforce auquel vous souhaitez écrire. Vous aurez besoin du nom de l'objet, comme `Account`, `Case` ou `Opportunity`.

Le connecteur Salesforce prend en charge quatre opérations d'écriture :

- INSERT
- UPSERT
- UPDATE
- DELETE

Lorsque vous utilisez l'opération d'écriture `UPSERT`, vous devez proposer l'option `ID_FIELD_NAMES` afin de spécifier le champ d'ID externe pour les enregistrements.

Vous pouvez également ajouter d'autres options de connexion :

- TRANSFER_MODE : prend en charge deux modes : SYNC et ASYNC. La valeur par défaut est SYNC. Lorsque cette option est définie sur ASYNC, Bulk API 2.0 Ingest sera utilisé pour le traitement.
- FAIL_ON_FIRST_ERROR: La valeur par défaut estFALSE, ce qui signifie que la AWS Glue tâche continuera à traiter toutes les données même si certains enregistrements d'écriture ont échoué. Lorsqu'elle est définie surTRUE, la AWS Glue tâche échouera en cas d'échec des enregistrements d'écriture et le traitement ne sera pas poursuivi.

Exemple

```
salesforce_write = glueContext.write_dynamic_frame.from_options(
    frame=frameToWrite,
    connection_type="salesforce",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "Account",
        "API_VERSION": "v60.0",
        "WRITE_OPERATION": "INSERT",
        "TRANSFER_MODE": "ASYNC",
        "FAIL_ON_FIRST_ERROR": "true"
    }
)
```

Options de connexion à Salesforce

Les options de connexion suivantes sont prises en charge pour le connecteur Salesforce :

- ENTITY_NAME(Chaîne) - (Obligatoire) Utilisé pour la lecture/l'écriture. Le nom de votre objet dans Salesforce.
- API_VERSION(Chaîne) - (Obligatoire) Utilisé pour la lecture/l'écriture. Version d'API Rest de Salesforce que vous souhaitez utiliser.
- SELECTED_FIELDS(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- FILTER_PREDICATE(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.

  Lorsque vous indiquez un prédicat de filtre, seul l'opérateur AND est pris en charge. D'autres opérateurs tels que OR et IN ne sont pas pris en charge actuellement.
- QUERY(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.

- PARTITION_FIELD(Chaîne) - Utilisé pour la lecture. Champ à utiliser pour partitionner la requête.
- LOWER_BOUND(Chaîne) - Utilisé pour la lecture. Une valeur limite inférieure inclusive du champ de partition choisi.
- UPPER_BOUND(Chaîne) - Utilisé pour la lecture. Une valeur limite supérieure exclusive du champ de partition choisi.
- NUM_PARTITIONS(Entier) - Par défaut : 1. Utilisé pour la lecture. Nombre de partitions pour la lecture.
- IMPORT_DELETED_RECORDS(Chaîne) - Par défaut : FALSE. Utilisé pour la lecture. Pour obtenir les enregistrements supprimés lors de la requête.
- WRITE_OPERATION(Chaîne) - Par défaut : INSERT. Utilisé pour l'écriture La valeur doit être INSERT, UPDATE, UPSERT, DELETE.
- ID_FIELD_NAMES(Chaîne) - Valeur par défaut : null. Nécessaire pour UPDATE et UPSERT.

Restrictions du connecteur Salesforce

Les restrictions du connecteur Salesforce sont les suivantes :

- Nous prenons uniquement en charge Spark SQL, tandis que Salesforce SOQL n'est pas pris en charge.
- Les signets de tâche ne sont pas pris en charge.
- Les noms de champs Salesforce sont sensibles à la casse. Lorsque vous écrivez dans Salesforce, les données doivent correspondre à la casse des champs définis dans Salesforce.

Configuration du flux de code d'autorisation pour Salesforce

Reportez-vous à la documentation publique de Salesforce pour activer le flux de code d'autorisation OAuth 2.0.

Pour configurer l'application connectée :

1. Cochez la case Activer OAuth les paramètres.
2. Dans le champ de texte URL de rappel, entrez une ou plusieurs redirections URLs pour AWS Glue.

   URLs Les redirections ont le format suivant :

https ://*region*.console.aws.amazon. com/gluestudio/oauth

Dans cette URL, la région est le code de la AWS région que vous utilisez AWS Glue pour transférer des données depuis Salesforce. Par exemple, le code pour la région USA Est (Virginie du Nord) est `us-east-1`. Pour cette région, l'URL est la suivante :

https://us-east-1.console.aws.amazon.com/gluestudio/oauth

Pour les AWS régions qui les prennent AWS Glue en charge et leurs codes, voir les AWS Glue points de terminaison et les quotas dans la référence AWS générale.

3. Cochez la case Exiger le secret pour le flux du serveur Web.
4. Dans la liste OAuth des étendues disponibles, ajoutez les étendues suivantes :
   - Gérez les données utilisateur via APIs (api)
   - Accéder aux autorisations personnalisées (custom_permissions)
   - Accéder au service d'URL d'identité (identifiant, profil, e-mail, adresse, téléphone)
   - Accéder à des identifiants utilisateur uniques (openid)
   - Exécuter des demandes à tout moment (refresh_token, offline_access)
5. Définissez la politique de jeton d'actualisation pour l'application connectée sur Le jeton d'actualisation est valide jusqu'à sa révocation. Dans le cas contraire, vos tâches échoueront à l'expiration de votre jeton d'actualisation. Pour plus d'informations sur la façon de vérifier et de modifier la politique des jetons d'actualisation, consultez la section Gérer les politiques d' OAuth accès pour une application connectée dans la documentation Salesforce.

Configurer le OAuth flux de porteurs JWT pour Salesforce

Reportez-vous à la documentation publique de Salesforce pour activer server-to-server l'intégration avec les jetons Web JSON OAuth 2.0.

Une fois que vous avez créé un JWT et correctement configuré l'application connectée dans Salesforce, vous pouvez créer une connexion Salesforce avec la clé `JWT_TOKEN` définie dans votre secret Secrets Manager. Définissez le type de OAuth subvention sur JWT Bearer Token lors de la création de la connexion.

Connexion à Salesforce Marketing Cloud

Salesforce Marketing Cloud est un fournisseur de logiciels d'automatisation et d'analytique marketing pour le marketing par e-mail, mobile, réseaux sociaux et en ligne. Il propose également des services de conseil et de mise en œuvre. En tant qu'utilisateur de Salesforce Marketing Cloud, vous pouvez vous connecter AWS Glue à votre compte Salesforce Marketing Cloud. Vous pouvez ensuite utiliser Salesforce Marketing Cloud comme source de données ou comme destination dans vos tâches ETL. Exécutez ces tâches pour transférer des données entre Salesforce Marketing Cloud et les AWS services ou autres applications prises en charge.

Rubriques

- AWS Glue support pour Salesforce Marketing Cloud
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration de Salesforce Marketing Cloud
- Configuration des connexions Salesforce Marketing Cloud
- Lecture à partir des entités Salesforce Marketing Cloud
- Écriture destinée aux entités Salesforce Marketing Cloud
- Options de connexion à Salesforce Marketing Cloud
- Restrictions et remarques relatives au connecteur Salesforce Marketing Cloud

AWS Glue support pour Salesforce Marketing Cloud

AWS Glue prend en charge Salesforce Marketing Cloud comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser des tâches AWS Glue ETL pour interroger des données depuis Salesforce Marketing Cloud.

Prise en charge en tant que cible ?

Non.

Versions de l'API Salesforce Marketing Cloud prises en charge

Les versions suivantes de l'API Salesforce Marketing Cloud sont prises en charge :

- v1

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les autorisations AWS IAM requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Si vous ne voulez pas utiliser la méthode ci-dessus, vous pouvez également utiliser les politiques IAM gérées suivantes :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs

bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration de Salesforce Marketing Cloud

Avant de pouvoir AWS Glue transférer des données depuis Salesforce Marketing Cloud, vous devez satisfaire aux exigences suivantes :

Configuration requise

Les exigences minimales sont les suivantes :

- Vous disposez d'un compte Salesforce Marketing Cloud. Pour de plus amples informations, veuillez consulter Création d'un compte Salesforce Marketing Cloud.
- Votre compte Salesforce Marketing Cloud est activé pour accéder à l'API. L'accès à l'API est activé par défaut pour les éditions Enterprise, Unlimited, Developer et Performance.

Si vous répondez à ces exigences, vous êtes prêt à vous connecter AWS Glue à votre compte Salesforce Marketing Cloud. Pour les connexions classiques, vous n'avez rien d'autre à faire dans Salesforce Marketing Cloud.

Création d'un compte Salesforce Marketing Cloud

Pour Salesforce Marketing Cloud, vous devez contacter le fournisseur pour créer un compte. Si vous ou votre entreprise êtes associé à Salesforce, contactez votre responsable de compte Salesforce pour demander une licence Salesforce Marketing Cloud. Sinon, vous pouvez demander à contacter un représentant Salesforce comme suit :

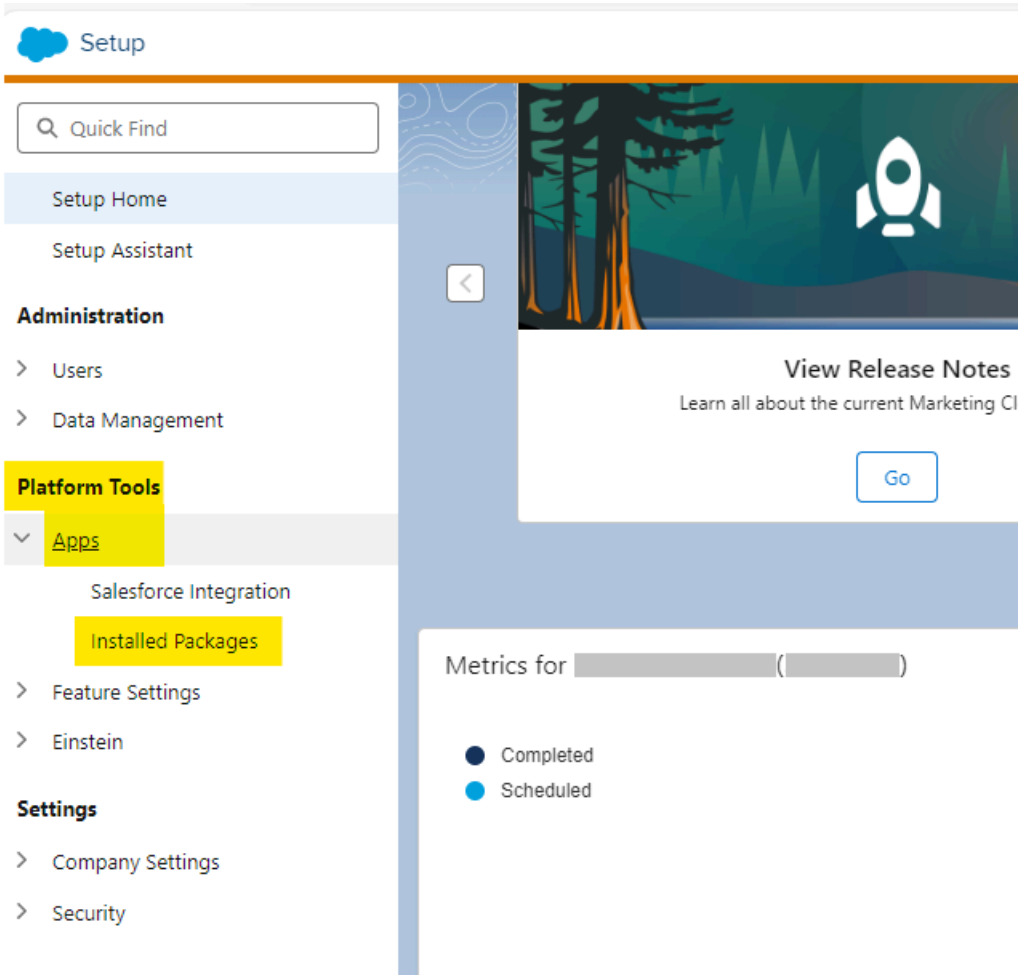
1. Accédez à https://www.salesforce.com/in/products/marketing-cloud/overview/et choisissez S'inscrire.
2. Cliquez sur le lien Contactez-nous en haut à droite de la page.
3. Saisissez les informations requises dans le formulaire et choisissez Me contacter.

Un représentant de Salesforce vous contactera pour discuter de vos besoins.

Création d'un projet et informations d'identification OAuth 2.0

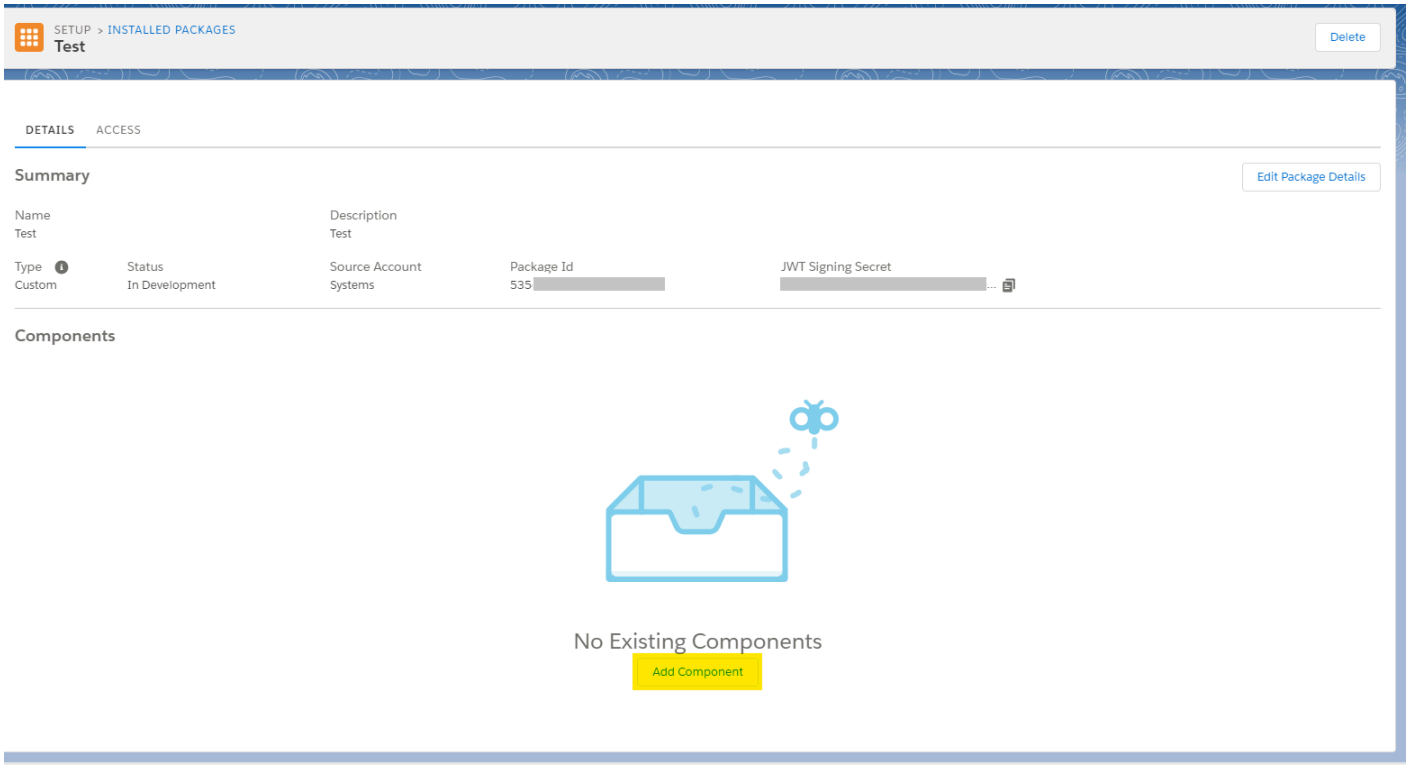
Pour obtenir un projet et des informations d'identification OAuth 2.0 :

1. Connectez-vous à votre instance Salesforce Marketing Cloud à l'aide de votre nom d'utilisateur et de votre mot de passe et authentifiez-vous à l'aide de votre numéro de téléphone portable enregistré.
2. Cliquez sur votre profil en haut à droite, puis sur Configurer.
3. Sous Outils de plateforme, choisissez Applications, puis Packages installés.



4. Sur la page Packages installés, cliquez sur Nouveau dans le coin supérieur droit. Saisissez le nom et la description du package.

   Enregistrez le package. Une fois le package enregistré, vous pouvez consulter les détails du package.
5. Sur la page Détails du package, dans la section Composant, choisissez Ajouter un composant.



6. Sélectionnez le type de composant « Intégration d'API » et cliquez sur Suivant.
7. Sélectionnez le type d'intégration « Server-à-serveur » (qui possède le type d' OAuth autorisation des informations d'identification du client) et cliquez sur Suivant.
8. Ajoutez les portées en fonction de vos besoins et cliquez sur Enregistrer.

Configuration des connexions Salesforce Marketing Cloud

Salesforce Marketing Cloud prend en charge le type de subvention CLIENT CREDENTIALS pour OAuth2.

- Ce type de subvention est considéré comme bidimensionnel OAuth 2.0 car il est utilisé par les clients pour obtenir un jeton d'accès en dehors du contexte d'un utilisateur. AWS Glue est capable d'utiliser l'ID client et le secret du client pour authentifier le Salesforce Marketing Cloud, APIs qui sont fournis par les services personnalisés que vous définissez.
- Chaque service personnalisé appartient à un utilisateur régi uniquement par l'API, qui dispose d'un ensemble de rôles et d'autorisations autorisant le service à effectuer des actions spécifiques. Un jeton d'accès est associé à un seul service personnalisé.
- Ce type d'autorisation donne lieu à un jeton d'accès de courte durée qui peut être renouvelé en appelant un point de terminaison d'identité.

- Pour consulter la documentation publique de Salesforce Marketing Cloud pour OAuth 2.0 avec les informations d'identification du client, voir Configurer votre environnement de développement pour les packages améliorés.

Pour configurer une connexion Salesforce Marketing Cloud :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes :
   a. Pour l'application connectée gérée par le client, le secret doit contenir le secret du consommateur de l'application connectée avec `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` comme clé.
   b. Remarque : Vous devez créer un secret pour vos connexions dans AWS Glue.
2. Dans AWS Glue Glue Studio, créez une connexion sous Data Connections en suivant les étapes ci-dessous :
   a. Lorsque vous sélectionnez un type de connexion, sélectionnez Salesforce Marketing Cloud.
   b. Indiquez le `Subdomain Endpoint` du Salesforce Marketing Cloud auquel vous souhaitez vous connecter.
   c. Sélectionnez le rôle AWS IAM qui AWS Glue peut assumer et dispose des autorisations nécessaires pour les actions suivantes :

      JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

d. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.

e. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.

3. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.

Lecture à partir des entités Salesforce Marketing Cloud

Prérequis

Un objet Salesforce Marketing Cloud à partir duquel vous souhaitez lire. Vous aurez besoin du nom de l’objet, comme `Activity` ou `Campaigns`. Le tableau suivant présente les entités prises en charge.

Entités prises en charge pour la source :

Entité	Interface	Peut être filtré	Limit prise en charge	Order by prise en charge	SELECT prise en charge*	Partitionnement pris en charge
Event Notification Callback	REST	Non	Non	Non	Oui	Non
Seed-List	REST	Non	Oui	Non	Oui	Non
Configuration	REST	Oui	Oui	Non	Oui	Non
Vérification de domaine	REST	Oui	Oui	Oui	Oui	Non
Objects Nested Tags	REST	Oui	Non	Non	Oui	Non
Contact	REST	Non	Oui	Non	Oui	Non

Entité	Interface	Peut être filtré	Limit prise en charge	Order by prise en charge	SELECT prise en charge*	Partition nement pris en charge
Event Notification Subscript ion	REST	Non	Non	Non	Oui	Non
Messageri e	REST	Non	Oui	Non	Oui	Non
Activité	SOAP	Non	Non	Non	Oui	Oui
Bounce Event	SOAP	Non	Non	Non	Oui	Oui
Click Event	SOAP	Non	Non	Non	Oui	Oui
Content Area	SOAP	Non	Non	Non	Oui	Oui
Data Extension	SOAP	Non	Oui	Non	Oui	Oui
E-mail	SOAP	Non	Oui	Non	Oui	Oui
Forwarded Email Event	SOAP	Non	Oui	Non	Oui	Oui
Transfére r un e-mail OptInEvent	SOAP	Non	Oui	Non	Oui	Oui
Lien	SOAP	Non	Oui	Non	Oui	Oui
Link Send	SOAP	Non	Oui	Non	Oui	Oui

Entité	Interface	Peut être filtré	Limit prise en charge	Order by prise en charge	SELECT prise en charge*	Partition nement pris en charge
List	SOAP	Non	Oui	Non	Oui	Oui
List Subscriber	SOAP	Non	Oui	Non	Oui	Oui
Not Sent Event	SOAP	Non	Oui	Non	Oui	Oui
Open Event	SOAP	Non	Oui	Non	Oui	Oui
Envoyer	SOAP	Non	Oui	Non	Oui	Oui
Sent Event	SOAP	Non	Oui	Non	Oui	Oui
Subscriber	SOAP	Non	Oui	Non	Oui	Oui
Survey Event	SOAP	Non	Oui	Non	Oui	Oui
Unsub Event	SOAP	Non	Oui	Non	Oui	Oui
Audit Events	REST	Non	Oui	Oui	Oui	Non
Campagnes	REST	Non	Oui	Oui	Oui	Non
Interacti ons	REST	Non	Oui	Oui	Oui	Non
Content Assets	REST	Non	Oui	Oui	Oui	Non

Exemple pour REST :

```
salesforcemarketingcloud_read = glueContext.create_dynamic_frame.from_options(
    connection_type="salesforcemarketingcloud",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "Campaigns",
        "API_VERSION": "v1",
        "INSTANCE_URL": "https://**********.rest.marketingcloudapis.com"
    }
)
```

Exemple pour SOAP :

```
salesforcemarketingcloud_read = glueContext.create_dynamic_frame.from_options(
    connection_type="salesforcemarketingcloud",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "Activity",
        "API_VERSION": "v1",
        "INSTANCE_URL": "https://**********.soap.marketingcloudapis.com"
    }
)
```

Détails des entités et des champs Salesforce Marketing Cloud :

Les tableaux ci-après décrivent les entités Salesforce Marketing Cloud. Il existe des entités REST avec des métadonnées statiques et des entités SOAP avec des métadonnées dynamiques.

Entités REST avec métadonnées statiques :

Entité	Champ	Type de données	Opérateurs pris en charge
Event Notification Callback	callbackId	String	
	callbackName	String	
	url	String	
	maxBatchSize	Entier	

Entité	Champ	Type de données	Opérateurs pris en charge
	status	String	
	statusReason	String	
Seed-List	id	String	
	name	String	
	description	String	
	activeSeedCount	Entier	
Configuration	customerKey	String	
	name	String	
	description	String	
	locationType	String	'='
	awsFileTransferLocation	Struct	
Domain verification	enterpriseId	Entier	
	status	String	'='
	domainType	String	'='
	memberId	Entier	
	emailSendTime	DateTime	
	domaine	String	
	isSendable	Booléen	
Objects Nested Tags	id	Entier	

Entité	Champ	Type de données	Opérateurs pris en charge
	modifiedDate	DateTime	
	tags	List	
	name	String	
	description	String	
	parentId	Entier	
Contact	values	List	
Event Notification Subscription	subscriptionName	String	
	callbackId	String	
	callbackName	String	
	eventCategoryTypes	List	
	filtres	List	
	url	String	
	maxBatchSize	Entier	
	subscriptionId	String	
	status	String	
	statusReason	String	
Messagerie	deliveryTime	DateTime	
	id	String	
	messageId	String	
	status	String	

Entité	Champ	Type de données	Opérateurs pris en charge
	to	Struct	
Interactions	status	String	'='
	id	String	
	key	String	
	name	String	
	lastPublishedDate	DateTime	
	description	String	
	version	Entier	
	workflowApiVersion	Entier	
	createdDate	DateTime	
	modifiedDate	DateTime	
	goals	Struct	
	stats	Struct	
	entryMode	String	
	defaults	Struct	
	executionMode	Struct	
	definitionId	String	
Content Assets	id	Entier	
	customerKey	String	
	objectId	String	

Entité	Champ	Type de données	Opérateurs pris en charge
	contentType	String	
	assetType	Struct	
	name	String	
	description	String	
	owner	Struct	
	createdDate	DateTime	
	createdBy	Struct	
	modifiedDate	DateTime	
	modifiedBy	Struct	
	thumbnail	Struct	
	category	Struct	
	meta	Struct	
	vues	Struct	
	availableViews	Struct	
	data	Struct	
	legacyData	Struct	
	modelVersion	Entier	
	Version	Entier	
	Locked (Verrouillée)	Booléen	
	FileProperties	Struct	

Entité	Champ	Type de données	Opérateurs pris en charge
	Étiquettes	List	
	Contenu	String	
	Conception	String	
	SuperContent	String	
	CustomFields	Struct	
	Blocs	Struct	
	MinBlocks	Entier	
	MaxBlocks	Entier	
	Canaux	Struct	
	AllowedBlocks	List	
	Emplacements	Struct	
	BusinessUnitAvailability	Struct	
	sharingProperties	Struct	
	sharingProperties.sharedWith	Struct	
	sharingProperties.sharingType	String	
	Modèle	Struct	
	Fichier	String	
	GenerateFrom	String	

Entité	Champ	Type de données	Opérateurs pris en charge
Audit Events	id	Entier	
	createdDate	DateTime	
	memberId	Entier	
	enterpriseId	Entier	
	employee	Struct	
	objectType	Struct	
	fonctionnement	Struct	
	objet	Struct	
	transactionId	String	
Campagnes	id	Entier	
	createdDate	DateTime	
	modifiedDate	DateTime	
	name	String	
	description	String	
	campaignCode	String	
	color	String	
	favorite	Booléen	

Entités SOAP avec métadonnées dynamiques :

Entité	Type de données	Opérateurs pris en charge
Activité	String	LIKE,!=,=
	Struct	
	Entier	!=,=,>=,<=,<,>
	Double	!=,=,>=,<=,<,>
	Booléen	!=,=
	DateTime	>=,<=,<,>,=,BETWEEN
Bounce Event	Entier	!=,=,>=,<=,<,>
	DateTime	>=,<=,<,>,=,BETWEEN
	String	LIKE,!=,=
	Struct	
Click Event	Entier	!=,=,>=,<=,<,>
	DateTime	>=,<=,<,>,=,BETWEEN
	String	LIKE,!=,=
	Struct	
Content Area	Struct	
	String	LIKE,!=,=
	Entier	!=,=,>=,<=,<,>
	DateTime	>=,<=,<,>,=,BETWEEN
	Booléen	!=,=
Data Extension	DateTime	>=,<=,<,>,=,BETWEEN

Entité	Type de données	Opérateurs pris en charge
	String	LIKE,!=,=
E-mail	Entier	!=,=,>=,<=,<,>
	String	LIKE,!=,=
	DateTime	>=,<=,<,>,=,BETWEEN
	Booléen	!=,=
	Struct	
Forwarded Email Event	Entier	!=,=,>=,<=,<,>
	String	LIKE,!=,=
	DateTime	>=,<=,<,>,=,BETWEEN
	Struct	
Courrier électronique transféré OptInEvent	Entier	!=,=,>=,<=,<,>
	String	LIKE,!=,=
	DateTime	>=,<=,<,>,=,BETWEEN
	Struct	
Lien	Entier	!=,=,>=,<=,<,>
Link Send	Entier	!=,=,>=,<=,<,>
	String	LIKE,!=,=
	Double	!=,=,>=,<=,<,>
List	Entier	!=,=,>=,<=,<,>
	String	LIKE,!=,=

Entité	Type de données	Opérateurs pris en charge
	DateTime	>=,<=,<,>,=,BETWEEN
	Struct	
List Subscriber	Entier	!=,=,>=,<=,<,>
	String	LIKE,!=,=
	DateTime	>=,<=,<,>,=,BETWEEN
	Struct	
Not Sent Event	Entier	!=,=,>=,<=,<,>
	String	LIKE,!=,=
	DateTime	>=,<=,<,>,=,BETWEEN
	Struct	
Open Event	Entier	!=,=,>=,<=,<,>
	String	LIKE,!=,=
	DateTime	>=,<=,<,>,=,BETWEEN
	Struct	
Envoyer	Entier	!=,=,>=,<=,<,>
	String	LIKE,!=,=
	DateTime	>=,<=,<,>,=,BETWEEN
	Booléen	!=,=
	Struct	
Sent Event	Entier	!=,=,>=,<=,<,>

Entité	Type de données	Opérateurs pris en charge
	String	LIKE,!=,=
	DateTime	>=,<=,<,>,=,BETWEEN
	Struct	
Subscriber	Entier	!=,=,>=,<=,<,>
	String	LIKE,!=,=
	DateTime	>=,<=,<,>,=,BETWEEN
	Struct	
Survey Event	Entier	!=,=,>=,<=,<,>
	String	LIKE,!=,=
	DateTime	>=,<=,<,>,=,BETWEEN
	Struct	
Unsub Event	Entier	!=,=,>=,<=,<,>
	String	LIKE,!=,=
	DateTime	>=,<=,<,>,=,BETWEEN
	Booléen	!=,=
	Struct	

Requêtes de partitionnement

Dans Salesforce Marketing Cloud, les champs Integer et DateTime Datatype prennent en charge le partitionnement basé sur les champs.

Vous pouvez indiquer les options Spark supplémentaires `PARTITION_FIELD`, `LOWER_BOUND`, `UPPER_BOUND` et `NUM_PARTITIONS` si vous souhaitez utiliser la simultanéité dans Spark. Avec ces

paramètres, la requête d'origine serait divisée en NUM_PARTITIONS nombres de sous-requêtes pouvant être exécutées simultanément par les tâches Spark.

- PARTITION_FIELD : le nom du champ à utiliser pour partitionner la requête.
- LOWER_BOUND : une valeur limite inférieure inclusive du champ de partition choisi.

  Pour le champ Timestamp, nous acceptons le format d'horodatage Spark utilisé dans les requêtes SQL Spark.

  Exemples de valeurs valides :

  ```
  "2024-05-07T02:03:00.00Z"
  ```

- UPPER_BOUND : une valeur limite supérieure exclusive du champ de partition choisi.
- NUM_PARTITIONS : le nombre de partitions.

Exemple :

```
salesforcemarketingcloud_read = glueContext.create_dynamic_frame.from_options(
    connection_type="salesforcemarketingcloud",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "ListSubscriber",
        "API_VERSION": "v1",
        "PARTITION_FIELD": "CreatedDate",
        "LOWER_BOUND": "2023-09-07T02:03:00.000Z",
        "UPPER_BOUND": "2024-05-07T02:03:00.000Z",
        "NUM_PARTITIONS": "10"
    }
)
```

Écriture destinée aux entités Salesforce Marketing Cloud

Conditions préalables

- Un objet Salesforce Marketing auquel vous souhaitez écrire. Vous devrez spécifier le nom de l'objet tel que List ou Campaigns ou l'une des autres entités décrites dans le tableau ci-dessous.
- Le connecteur Salesforce Marketing Cloud prend en charge trois opérations d'écriture :
  - INSERT

- UPSERT
- UPDATE

Lorsque vous utilisez les opérations d'écriture `UPDATE` et `UPSERT`, vous devez proposer l'option `ID_FIELD_NAMES` afin de spécifier le champ d'ID externe pour les enregistrements.

Entités prises en charge pour la destination :

Entité	Priority	Interface (REST, SOAP, etc.)	Peut être inséré	Peut être mis à jour	Peut faire l'objet d'un upsert
Campagnes	P0	REST	Y- Unique	Y- Unique	N
Content Assets	P0	REST	Y- Unique, en bloc	Y- Unique	N
Contact	P1	REST	Y- Unique	Y- Unique	N
Vérification de domaine	P1	REST	Y- Unique	Y- Unique, en bloc	N
Event Notification Callback	P1	REST	Y- Unique	Y- Unique	N
Event Notification Subscription	P1	REST	Y- Unique	Y- Unique	N
Messagerie	P1	REST	Y- Unique	N	N
Object Nested Tag	P2	REST	Y- Unique	Y- Unique	N
Seed-List	P1	REST	Y- Unique	Y- Unique	N
Configuration	P1	REST	Y- Unique	Y- Unique	N

Entité	Priority	Interface (REST, SOAP, etc.)	Peut être inséré	Peut être mis à jour	Peut faire l'objet d'un upsert
Data Extension	P0	SOAP	Y- Unique	Y- Unique	Y- Unique
E-mail	P0	SOAP	Y- Unique	Y- Unique	N
List	P0	SOAP	Y- Unique	Y- Unique	N
Envoyer	P0	SOAP	Y- Unique	N	N
Subscriber	P0	SOAP	Y- Unique	Y- Unique	N

Exemple d'opération INSERT pour REST :

```
salesforcemarketingcloud_write = glueContext.write_dynamic_frame.from_options(
    connection_type="salesforcemarketingcloud",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "Campaigns",
        "API_VERSION": "v1",
        "writeOperation" : "INSERT",
        "INSTANCE_URL": "https://**********************.rest.marketingcloudapis.com"
    }
)
```

Exemple d'opération INSERT pour SOAP :

```
salesforcemarketingcloud_write = glueContext.write_dynamic_frame.from_options(
    connection_type="salesforcemarketingcloud",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "List",
        "API_VERSION": "v1",
        "writeOperation" : "INSERT",
        "INSTANCE_URL": "https://**********************.rest.marketingcloudapis.com"
    }
)
```

Exemple d'opération UPDATE pour REST :

```
salesforcemarketingcloud_write = glueContext.write_dynamic_frame.from_options(
    connection_type="salesforcemarketingcloud",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "Campaigns",
        "API_VERSION": "v1",
        "writeOperation" : "UPDATE",
         "ID_FIELD_NAMES": "id",
        "INSTANCE_URL": "https://**********.rest.marketingcloudapis.com"
    }
)
```

Exemple d'opération UPDATE pour SOAP :

```
salesforcemarketingcloud_write = glueContext.write_dynamic_frame.from_options(
    connection_type="salesforcemarketingcloud",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "List",
        "API_VERSION": "v1",
        "writeOperation" : "UPDATE",
         "ID_FIELD_NAMES": "id",
        "INSTANCE_URL": "https://**********.rest.marketingcloudapis.com"
    }
)
```

Exemple d'opération UPSERT pour SOAP :

```
salesforcemarketingcloud_write = glueContext.write_dynamic_frame.from_options(
    connection_type="salesforcemarketingcloud",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "DataExtension/Insert-***E/6*******3",
        "API_VERSION": "v1",
        "writeOperation" : "UPSERT",
        "INSTANCE_URL": "https://**********.rest.marketingcloudapis.com"
    }
)
```

Options de connexion à Salesforce Marketing Cloud

Les options de connexion pour Salesforce Marketing Cloud sont les suivantes :

- `ENTITY_NAME`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Le nom de votre objet dans Salesforce Marketing Cloud.
- `API_VERSION`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Version de l'API Rest et de SOAP API Salesforce Marketing Cloud que vous souhaitez utiliser.
- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.
- `PARTITION_FIELD`(Chaîne) - Utilisé pour la lecture. Champ à utiliser pour partitionner la requête.
- `LOWER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite inférieure inclusive du champ de partition choisi.
- `UPPER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS`(Entier) - Par défaut : 1. Utilisé pour la lecture. Nombre de partitions pour la lecture.
- `WRITE_OPERATION`(Chaîne) - Par défaut : INSERT. Utilisé pour l'écriture La valeur doit être INSERT, UPDATE, UPSERT.
- `ID_FIELD_NAMES`(Chaîne) - Valeur par défaut : null. Nécessaire pour UPDATE/UPSERT.

Restrictions et remarques relatives au connecteur Salesforce Marketing Cloud

Les restrictions ou les remarques du connecteur Salesforce Marketing Cloud sont les suivantes :

- Lorsque vous utilisez un filtre sur DateTime un champ de type de données, vous devez transmettre la valeur au format « yyyy-mm-dd THH:MM:SSZ ».
- Dans Data Preview, la valeur Boolean Datatype apparaît sous la forme d'une valeur vide.
- Pour les entités SOAP, vous pouvez définir un maximum de deux filtres, et pour les entités REST, vous ne pouvez définir qu'un seul filtre, ce qui limite les tests de partitionnement avec des filtres.
- Plusieurs comportements inattendus ont été observés du côté du SaaS : le champ `Link.Alias` de l'entité `linksend` ne prend pas en charge l'opérateur CONTAINS (par exemple, `Link.Alias`

`CONTAINS "ViewPrivacyPolicy"`), et les opérateurs de filtre pour les entités Data Extension (telles que EQUALS et GREATER THAN) ne renvoient pas les résultats attendus.

- L'API ClickEvent SOAP SFMC prend du temps pour refléter les enregistrements nouvellement créés, de sorte que les enregistrements créés récemment peuvent ne pas être immédiatement disponibles dans la réponse de l'API.

  Exemple : Si vous créez 5 nouveaux ClickEvent enregistrements le 2025-01-10T 14:30:00 et que vous les récupérez immédiatement à l'aide de l'API SOAP, la réponse risque de ne pas inclure les 5 enregistrements. Il peut s'écouler jusqu'à cinq minutes pour que les enregistrements qui viennent d'être créés apparaissent dans la réponse de l'API. Ce délai peut également affecter à la fois la récupération des données et les exécutions planifiées.
- Deux DateTime formats différents : 2025-03-11T 04:46:00 (sans millisecondes) et 2025-03-11T 04:46:00.000 Z sont pris en charge lors de l'exécution d'opérations d'écriture en (en millisecondes). AWS Glue
- Pour l'entité Event Notification Subscription, un abonnement ne peut être créé que pour une URL de rappel vérifiée, et vous pouvez avoir jusqu'à 200 abonnements par rappel.
- Pour l'entité Event Notification Callback, un maximum de 50 enregistrements peuvent être créés par compte.

Connexion à Salesforce Commerce Cloud

L'API B2C Commerce est un ensemble d'instances de commerce B2C RESTful APIs permettant d'interagir avec les instances de commerce B2C. Elle porte plusieurs noms différents : API Salesforce Commerce, l'acronyme SCAPI ou simplement API Commerce.

L'API permet aux développeurs de créer un large éventail d'applications : des vitrines complètes aux outils personnalisés pour les commerçants, afin d'améliorer Business Manager. Pour tous les clients B2C Commerce, l'API est disponible gratuitement.

L'API est divisée en deux groupes principaux APIs : Shopper APIs et Admin APIs. Au sein de chaque groupe, elles sont divisées en familles d'API et en petits groupes axés sur les fonctionnalités associées.

Rubriques

- AWS Glue support pour Salesforce Commerce Cloud
- Politiques contenant les opérations API pour la création et l'utilisation des connexions

- Configuration de Salesforce Commerce Cloud
- Configuration des connexions Salesforce Commerce Cloud
- Lecture à partir d'entités Salesforce Commerce Cloud
- Référence des options de connexion Salesforce Commerce Cloud
- Limitations

AWS Glue support pour Salesforce Commerce Cloud

AWS Glue prend en charge Salesforce Commerce Cloud comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser des tâches AWS Glue ETL pour interroger des données depuis Salesforce Commerce Cloud.

Prise en charge en tant que cible ?

Non.

Versions de l'API Salesforce Commerce Cloud prises en charge

v1.

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les AWS autorisations requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
```

```
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Vous pouvez également utiliser les politiques IAM gérées suivantes pour autoriser l'accès :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration de Salesforce Commerce Cloud

Avant de pouvoir AWS Glue transférer des données depuis Salesforce Commerce Cloud, vous devez répondre aux exigences suivantes :

Configuration requise

- Vous disposez d'une application client Salesforce Commerce Cloud avec un ID client et un secret client.
- Votre compte Salesforce Commerce Cloud est activé pour accéder à l'API.

Si vous répondez à ces exigences, vous êtes prêt à vous connecter AWS Glue à votre compte Salesforce Commerce Cloud. Pour les connexions classiques, vous n'avez rien d'autre à faire dans Salesforce Commerce Cloud.

Configuration des connexions Salesforce Commerce Cloud

Salesforce Commerce Cloud prend en charge le type de subvention CLIENT CREDENTIALS pour OAuth2.

- Ce type de subvention est considéré comme bidimensionnel OAuth 2.0 car il est utilisé par les clients pour obtenir un jeton d'accès en dehors du contexte d'un utilisateur. AWS Glue est capable d'utiliser l'identifiant et le secret du client pour authentifier Salesforce Commerce Cloud, APIs qui sont fournis par des services personnalisés que vous définissez.
- Chaque service personnalisé appartient à un utilisateur régi uniquement par l'API, qui dispose d'un ensemble de rôles et d'autorisations autorisant le service à effectuer des actions spécifiques. Un jeton d'accès est associé à un seul service personnalisé.
- Ce type d'autorisation donne lieu à un jeton d'accès de courte durée qui peut être renouvelé en appelant un point de terminaison d'identité.
- Pour plus d'informations sur la documentation de Salesforce Commerce Cloud relative à la génération des informations d'identification client, consultez la documentation Salesforce.

Pour configurer une connexion Salesforce Commerce Cloud :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes. Il est nécessaire de créer un secret pour chaque connexion dans AWS Glue.
   a. Pour l'application connectée gérée par le client : le secret doit contenir le secret du consommateur de l'application connectée avec USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET comme clé.
2. Dans AWS Glue Studio, créez une connexion sous Connexions de données en suivant les étapes ci-dessous :
   a. Sous Connexions de données, choisissez Créer une connexion.
   b. Lorsque vous sélectionnez une source de données, sélectionnez Salesforce Commerce Cloud.
   c. Entrez votre code abrégé, votre identifiant d'organisation et votre identifiant de site Salesforce Commerce Cloud.
   d. Sélectionnez l'URL du domaine Salesforce Commerce Cloud de votre compte Salesforce Commerce Cloud.
   e. Sélectionnez le rôle IAM qui AWS Glue peut assumer les actions suivantes et qui dispose des autorisations nécessaires pour les actions suivantes :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

f. Indiquez les OAuth champs d'application (facultatif, application client gérée par l'utilisateur) ClientId du Salesforce Commerce Cloud auquel vous souhaitez vous connecter.

g. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.

h. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.

3. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.

4. Dans la configuration de votre AWS Glue tâche, fournissez `connectionName` une connexion réseau supplémentaire.

Lecture à partir d'entités Salesforce Commerce Cloud

Conditions préalables

- Un objet Salesforce Commerce Cloud à partir duquel vous souhaitez lire. Reportez-vous au tableau des entités prises en charge ci-dessous pour vérifier les entités disponibles.

Entités prises en charge

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Missions	Oui	Oui	Oui	Oui	Oui
Campagnes	Oui	Oui	Oui	Oui	Oui
Catalogues	Oui	Oui	Oui	Oui	Oui
Catégories	Oui	Oui	Oui	Oui	Oui
Coupons	Oui	Oui	Oui	Oui	Oui
Gift Certificates	Oui	Oui	Oui	Oui	Oui
Produits	Oui	Oui	Oui	Oui	Oui
Promotions	Oui	Oui	Oui	Oui	Oui
Source Code Groups	Oui	Oui	Oui	Oui	Oui

Exemple

```
salesforce_commerce_cloud_read = glueContext.create_dynamic_frame.from_options(
     connection_type="SalesforceCommerceCloud",
     connection_options={
         "connectionName": "connectionName",
         "ENTITY_NAME": "campaign",
         "API_VERSION": "v1"
     }
)
```

Détails des entités et des champs Salesforce Commerce Cloud

Liste des entités :

- Missions : https://developer.salesforce.com/docs/commerce/commerce-api/references/assignments

- Campagnes : https://developer.salesforce.com/docs/commerce/commerce-api/references/campaigns
- Catalogues : https://developer.salesforce.com/docs/commerce/commerce-api/references/catalogs
- Catégories : https://developer.salesforce.com/docs/commerce/commerce-api/references/catalogs?META = Catégories de recherche
- Chèques-cadeaux : https://developer.salesforce.com/docs/commerce/commerce-api/references/gift-chèques-cadeaux
- Produits : https://developer.salesforce.com/docs/commerce/commerce-api/references/products
- Promotions : https://developer.salesforce.com/docs/commerce/commerce-api/references/promotions
- Groupes de code source : https://developer.salesforce.com/docs/commerce/commerce-api/references/source-code-groups

Requêtes de partitionnement

Les options Spark supplémentaires `PARTITION_FIELD`, `LOWER_BOUND`, `UPPER_BOUND` et `NUM_PARTITIONS` peuvent être indiquées si vous souhaitez utiliser la simultanéité dans Spark. Avec ces paramètres, la requête d'origine serait divisée en `NUM_PARTITIONS` nombres de sous-requêtes pouvant être exécutées simultanément par les tâches Spark.

- `PARTITION_FIELD` : le nom du champ à utiliser pour partitionner la requête.
- `LOWER_BOUND` : une valeur limite inférieure inclusive du champ de partition choisi.

  Pour le champ Date, nous acceptons le format de date Spark utilisé dans les requêtes SQL Spark. Exemples de valeurs valides : `"2024-02-06"`.
- `UPPER_BOUND` : une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS` : nombre de partitions.

Les détails relatifs à la prise en charge des champs de partitionnement par entité sont présentés dans le tableau ci-dessous :

Entité	Champ de partitionnement	DataType
Campagnes	lastModified	DateTime

Entité	Champ de partitionnement	DataType
Campagnes	startDate	DateTime
Campagnes	endDate	DateTime
Catalogues	creationDate	DateTime
Catégories	creationDate	DateTime
Gift Certificates	merchantId	String
Gift Certificates	creationDate	DateTime
Produits	creationDate	DateTime
Produits	lastModified	DateTime
Source Code Groups	creationDate	DateTime
Source Code Groups	startTime	DateTime
Source Code Groups	endTime	DateTime

Exemple

```
salesforceCommerceCloud_read = glueContext.create_dynamic_frame.from_options(
    connection_type="SalesforceCommerceCloud",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "coupons",
        "API_VERSION": "v1",
        "PARTITION_FIELD": "creationDate",
        "LOWER_BOUND": "2020-05-01T20:55:02.000Z",
        "UPPER_BOUND": "2024-07-11T20:55:02.000Z",
        "NUM_PARTITIONS": "10"
    }
)
```

Référence des options de connexion Salesforce Commerce Cloud

Les options de connexion pour Salesforce Commerce Cloud sont les suivantes :

- `ENTITY_NAME`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Le nom de votre objet dans Salesforce Commerce Cloud.
- `API_VERSION`(Chaîne) - (Obligatoire) Utilisé pour la lecture/l'écriture. Version d'API Rest de Salesforce Commerce Cloud que vous souhaitez utiliser. Exemple : v1.
- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.
- `PARTITION_FIELD`(Chaîne) - Utilisé pour la lecture. Champ à utiliser pour partitionner la requête.
- `LOWER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite inférieure inclusive du champ de partition choisi.
- `UPPER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS`(Entier) - Par défaut : 1. Utilisé pour la lecture. Nombre de partitions pour la lecture.

Limitations

Les restrictions du connecteur Salesforce Commerce Cloud sont les suivantes :

- Le filtre Contient ne fonctionne pas comme prévu lors du partitionnement.
- L'entité de CDN Zones ne prend pas en charge les instances d'environnement de test (sandbox) et ne prend en charge que les types d'instances de développement et de production. Pour plus d'informations, consultez https://help.salesforce.com/s/ArticleView ?id=cc.b2c_embedded_cdn_overview.htm.
- Dans Salesforce Commerce Cloud, il n'existe aucun point de terminaison d'API pour récupérer les métadonnées dynamiques. Par conséquent, aucune disposition ne permet de prendre en charge les champs personnalisés dans l'entité Produit et Catégorie.

- L'identifiant du site est un paramètre de requête obligatoire. Vous devez transmettre la valeur de l'identifiant du site via le paramètre de connecteur personnalisé. Pour plus d'informations, consultez Base URL and Request Formation.
- Vous pouvez appliquer des filtres sur un maximum de deux champs (à l'exception de Niveaux, le cas échéant) dans une seule demande d'API en combinant différents opérateurs, comme indiqué dans le tableau ci-dessous :

Critères de filtrage	Est pris en charge ?
Un champ avec l'opérateur CONTAINS dans une seule demande d'API.	Oui
Un champ avec l'opérateur Equals dans une seule demande d'API.	Oui
Un champ avec l'opérateur BETWEEN dans une seule demande d'API.	Oui
Deux champs ou plus avec l'opérateur CONTAINS dans une seule demande d'API.	Non
Deux champs ou plus avec l'opérateur Equals dans une seule demande d'API.	Non
Deux champs ou plus avec l'opérateur BETWEEN dans une seule demande d'API.	Non
Un champ avec Equals et un champ avec l'opérateur CONTAINS dans une seule demande d'API.	Oui
Un champ avec BETWEEN et un champ avec l'opérateur CONTAINS dans une seule demande d'API.	Oui
Un champ avec BETWEEN et un champ avec l'opérateur Equals dans une seule demande d'API.	Oui

Critères de filtrage	Est pris en charge ?
Un champ avec Equals, un champ avec CONTAINS et un champ avec l'opérateur BETWEEN dans une seule demande d'API.	Non
Un champ avec l'opérateur Equals lorsque INCREMENTAL PULL est appliqué dans une seule demande d'API.	Oui
Un champ avec l'opérateur CONTAINS lorsque INCREMENTAL PULL est appliqué dans une seule demande d'API.	Oui
Un champ avec l'opérateur BETWEEN lorsque INCREMENTAL PULL est appliqué dans une seule demande d'API.	Non
Un opérateur Equals et un opérateur CONTAINS lorsque INCREMENTAL PULL est appliqué dans une seule requête d'API.	Non

- Dans certaines entités, le type de données des champs lors de la récupération est différent de celui utilisé comme champs consultables. Par conséquent, aucune fonctionnalité de filtrage n'est prévue pour ces champs. Le tableau suivant fournit des informations détaillées sur ces champs.

N° Sr.	Nom de l'entité	Nom de champ pouvant faire l'objet d'une recherche	Type de données pouvant faire l'objet d'une recherche	Type de données récupérable
1	Catalogue	name	String	Struct
2	Catalogue	description	String	Struct
3	Catégorie	name	String	Struct
4	Catégorie	description	String	Struct

N° Sr.	Nom de l'entité	Nom de champ pouvant faire l'objet d'une recherche	Type de données pouvant faire l'objet d'une recherche	Type de données récupérable
5	Produit (langue française non garantie)	name	String	Struct
6	Produit (langue française non garantie)	searchable	Booléen	Struct
7	Produit (langue française non garantie)	validFrom	DateTime	Struct
8	Produit (langue française non garantie)	validTo	DateTime	Struct
9	Produit (langue française non garantie)	type	String	Struct
10	Produit (langue française non garantie)	onlineFlag	Booléen	Struct
11	Promotion	name	String	Struct

Connexion à Salesforce Marketing Cloud Account Engagement

Salesforce Marketing Cloud Account Engagement est une solution d'automatisation marketing qui aide les entreprises à créer des liens significatifs, à générer plus de pipelines et à permettre au service commercial de conclure davantage de ventes. Si vous êtes un utilisateur de Salesforce Marketing Cloud Account Engagement, vous pouvez vous connecter AWS Glue à votre compte

Salesforce Marketing Cloud Account Engagement. Vous pouvez utiliser Salesforce Marketing Cloud Account Engagement comme source de données dans vos tâches ETL. Exécutez ces tâches pour transférer les données de Salesforce Marketing Cloud Account Engagement vers AWS des services ou d'autres applications prises en charge.

Rubriques

- AWS Glue support pour l'engagement des comptes Salesforce Marketing Cloud
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration de Salesforce Marketing Cloud Account Engagement
- Configuration des connexions Salesforce Marketing Cloud Account Engagement
- Lecture à partir des entités Salesforce Marketing Cloud Account Engagement
- Options de connexion Salesforce Marketing Cloud Account Engagement
- Restrictions et remarques relatives au connecteur Salesforce Marketing Cloud Account Engagement

AWS Glue support pour l'engagement des comptes Salesforce Marketing Cloud

AWS Glue prend en charge l'engagement des comptes Salesforce Marketing Cloud comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser les tâches AWS Glue ETL pour interroger les données de Salesforce Marketing Cloud Account Engagement en mode asynchrone ou synchronisé.

Prise en charge en tant que cible ?

Non.

Versions de l'API Salesforce Marketing Cloud Account Engagement prises en charge

Les versions suivantes de l'API Salesforce Marketing Cloud Account Engagement sont prises en charge :

- v5

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les autorisations AWS IAM requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Si vous ne voulez pas utiliser la méthode ci-dessus, vous pouvez également utiliser les politiques IAM gérées suivantes :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration de Salesforce Marketing Cloud Account Engagement

Avant de pouvoir AWS Glue transférer des données depuis Salesforce Marketing Cloud Account Engagement, vous devez répondre aux exigences suivantes :

Configuration requise

Les exigences minimales sont les suivantes :

- Vous disposez d’un compte Salesforce Marketing.
- Vous disposez d’un forfait Account Engagement sous licence pour le compte Salesforce.
- Vous avez synchronisé l’utilisateur Salesforce avec l’utilisateur Account Engagement.
- Vous avez créé une nouvelle application connectée sous App Manager pour obtenir des OAuth informations d'identification.

Si vous répondez à ces exigences, vous êtes prêt à vous connecter AWS Glue à votre compte Salesforce Marketing Cloud Account Engagement.

Configuration des connexions Salesforce Marketing Cloud Account Engagement

Le type de subvention détermine le mode AWS Glue de communication avec Salesforce Marketing Cloud Account Engagement pour demander l'accès à vos données. Votre choix influe sur les exigences que vous devez respecter avant de créer la connexion. Salesforce Marketing Cloud Account Engagement ne prend en charge que le type de subvention AUTHORIZATION_CODE pour la version 2.0. OAuth

- Ce type de subvention est considéré comme « à trois étapes » OAuth car il repose sur la redirection des utilisateurs vers un serveur d'autorisation tiers pour authentifier l'utilisateur. Il est utilisé lors de la création de connexions via la AWS Glue console.
- Les utilisateurs peuvent toujours choisir de créer leur propre application connectée dans Salesforce Marketing Cloud Account Engagement et de fournir leur propre identifiant client et leur secret client lors de la création de connexions via la AWS Glue console. Dans ce scénario, ils seront toujours redirigés vers Salesforce Marketing Cloud Account Engagement pour se connecter et autoriser AWS Glue l'accès à leurs ressources.
- Ce type d’autorisation donne lieu à un jeton d’actualisation et à un jeton d’accès. Le jeton d’accès est de courte durée et peut être actualisé automatiquement sans interaction de l’utilisateur à l’aide du jeton d’actualisation.

- Pour consulter la documentation publique relative à l'engagement des comptes Salesforce Marketing Cloud sur la création d'une application connectée pour le OAuth flux de code d'autorisation, consultez Authentification.

Pour configurer une connexion Salesforce Marketing Cloud Account Engagement :

1. Dans AWS Glue Glue Studio, créez une connexion sous Data Connections en suivant les étapes ci-dessous :
   a. Lorsque vous sélectionnez un type de connexion, sélectionnez Salesforce Marketing Cloud Account Engagement.
   b. Indiquez l'`INSTANCE_URL` d'une instance Salesforce Marketing Cloud Account Engagement à laquelle vous souhaitez vous connecter.
   c. Indiquez l'`PARDOT_BUSINESS_UNIT_ID` d'une instance Salesforce Marketing Cloud Account Engagement à laquelle vous souhaitez vous connecter.
   d. Sélectionnez l'URL du code d'autorisation approprié dans la liste déroulante.
   e. Sélectionnez l'URL du jeton appropriée dans le menu déroulant.
   f. Sélectionnez le rôle AWS IAM qui AWS Glue peut assumer les actions suivantes et qui dispose des autorisations nécessaires pour les actions suivantes :

      JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

g. Indiquez l'ID client de l'application client gérée par l'utilisateur (l'ID client de l'application connectée).

h. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons. Le secret sélectionné doit comporter une clé `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` dont la valeur est le secret client de l'application connectée.

i. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.

2. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.
3. Dans la configuration de votre AWS Glue tâche, fournissez `connectionName` une connexion réseau supplémentaire.

Lecture à partir des entités Salesforce Marketing Cloud Account Engagement

Prérequis

Un objet Salesforce Marketing Cloud Account Engagement à partir duquel vous souhaitez lire. Vous aurez besoin du nom de l'objet.

Entités prises en charge pour la source de synchronisation :

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Campaign	Oui	Oui	Oui	Oui	Oui
Contenu dynamique	Oui	Oui	Oui	Oui	Oui
E-mail	Oui	Oui	Oui	Oui	Oui
Email Template	Oui	Oui	Oui	Oui	Oui
Engagemen t Studio Program	Oui	Oui	Oui	Oui	Oui

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Folder Contents	Oui	Oui	Oui	Oui	Oui
Landing Page	Oui	Oui	Oui	Oui	Oui
Lifecycle History	Oui	Oui	Oui	Oui	Oui
Lifecycle Stage	Oui	Oui	Oui	Oui	Oui
List	Oui	Oui	Oui	Oui	Oui
List Email	Oui	Oui	Oui	Oui	Oui
List Membership	Oui	Oui	Oui	Oui	Oui
Opportunity	Oui	Oui	Oui	Oui	Oui
Prospect	Oui	Oui	Oui	Oui	Oui
Prospect Account	Oui	Oui	Oui	Oui	Oui
Utilisateur	Oui	Oui	Oui	Oui	Oui

Exemple :

```
salesforcepardot_read = glueContext.create_dynamic_frame.from_options(
    connection_type="SalesforcePardot",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "API_VERSION": "v5"
    }
```

```
)
```

Entités prises en charge pour la source asynchrone :

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Campaign	Oui	Non	Non	Oui	Non
Contenu dynamique	Oui	Non	Non	Oui	Non
Email Template	Oui	Non	Non	Oui	Non
Landing Page	Oui	Non	Non	Oui	Non
Lifecycle History	Oui	Non	Non	Oui	Non
Lifecycle Stage	Oui	Non	Non	Oui	Non
List	Oui	Non	Non	Oui	Non
List Email	Oui	Non	Non	Oui	Non
List Membership	Oui	Non	Non	Oui	Non
Opportunity	Oui	Non	Non	Oui	Non
Prospect	Oui	Non	Non	Oui	Non
Prospect Account	Oui	Non	Non	Oui	Non
Utilisateur	Oui	Non	Non	Oui	Non

Exemple :

```
salesforcepardot_read = glueContext.create_dynamic_frame.from_options(
    connection_type="SalesforcePardot",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "API_VERSION": "v5",
        "TRANSFER_MODE": "ASYNC"
    }
   )
```

Détails des entités et des champs Salesforce Marketing Cloud Account Engagement :

Pour afficher les détails des champs pour les entités suivantes, accédez à API Salesforce Marketing Cloud Account Engagement, choisissez Guides, faites défiler la page vers le bas jusqu'à Open Source API Wrappers, développez Version 5 Docs dans le menu et choisissez une entité.

Liste des entités :

- Campaign
- Contenu dynamique
- E-mail
- Email Template
- Engagement Studio Program
- Folder Content
- Landing Page
- Lifecycle History
- Lifecycle Stage
- List
- List Email
- List Membership
- Opportunity
- Prospect
- Prospect Account
- Utilisateur

Outre les champs mentionnés ci-dessus, le mode asynchrone prend en charge des champs filtrables spécifiques pour chaque entité, comme indiqué dans le tableau ci-dessous.

Entité	Champs filtrables supplémentaires pris en charge en mode asynchrone
Campaign	`createdAfter` , `createdBefore` , `deleted`, `updatedAfter` , `updatedBefore`
Contenu dynamique	`createdAfter` , `createdBefore` , `deleted`, `updatedAfter` , `updatedBefore`
Email Template	`createdAfter` , `createdBefore` , `deleted`, `updatedAfter` , `updatedBefore`
Engagement Studio Program	-
Landing Page	`createdAfter` , `createdBefore` , `deleted`, `updatedAfter` , `updatedBefore`
Lifecycle History	`createdAfter` , `createdBefore`
Lifecycle Stage	`createdAfter` , `createdBefore` , `deleted`, `updatedAfter` , `updatedBefore`
List	`createdAfter` , `createdBefore` , `deleted`, `updatedAfter` , `updatedBefore`
List Email	`createdAfter` , `createdBefore` , `deleted`, `updatedAfter` , `updatedBefore`

Entité	Champs filtrables supplémentaires pris en charge en mode asynchrone
List Membership	`createdAfter` , `createdBefore` , `deleted`, `updatedAfter` , `updatedBefore`
Opportunity	`createdAfter` , `createdBefore` , `deleted`, `updatedAfter` , `updatedBefore`
Prospect	`createdAfter` , `createdBefore` , `deleted`, `updatedAfter` , `updatedBefore`
Prospect Account	`createdAfter` , `createdBefore` , `deleted`
Utilisateur	`createdAfter` , `createdBefore` , `deleted`, `updatedAfter` , `updatedBefore`

Pour plus d'informations sur les champs supplémentaires, consultez API Salesforce Export.

Notez les considérations à prendre en compte pour le connecteur :

- La valeur du champ `delete` dans les entités peut être `false` (par défaut), `true` ou `all`.

Requêtes de partitionnement

Partitionnement basé sur des filtres :

Vous pouvez indiquer les options Spark supplémentaires `PARTITION_FIELD`, `LOWER_BOUND`, `UPPER_BOUND` et `NUM_PARTITIONS` si vous souhaitez utiliser la simultanéité dans Spark. Avec ces paramètres, la requête d'origine serait divisée en `NUM_PARTITIONS` nombres de sous-requêtes pouvant être exécutées simultanément par les tâches Spark.

- `PARTITION_FIELD` : le nom du champ à utiliser pour partitionner la requête.

- LOWER_BOUND : une valeur limite inférieure inclusive du champ de partition choisi.

  Pour le champ Datetime, nous acceptons le format d'horodatage Spark utilisé dans SPark les requêtes SQL.

  Exemples de valeurs valides :

  ```
  "2022-01-01T01:01:01.000Z"
  ```

- UPPER_BOUND : une valeur limite supérieure exclusive du champ de partition choisi.
- NUM_PARTITIONS : le nombre de partitions.
- PARTITION_BY : le type de partitionnement à effectuer. « FIELD » doit être transmis en cas de partitionnement basé sur des champs.

Exemple :

```
salesforcepardot_read = glueContext.create_dynamic_frame.from_options(
    connection_type="salesforcepardot",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "API_VERSION": "v5",
        "PARTITION_FIELD": "createdAt"
        "LOWER_BOUND": "2022-01-01T01:01:01.000Z"
        "UPPER_BOUND": "2024-01-01T01:01:01.000Z"
        "NUM_PARTITIONS": "10",
        "PARTITION_BY": "FIELD"
    }
   )
```

Options de connexion Salesforce Marketing Cloud Account Engagement

Les options de connexion pour Salesforce Marketing Cloud Account Engagement sont les suivantes :

- ENTITY_NAME(Chaîne) - (Obligatoire) Utilisé pour la lecture. Le nom de votre objet dans Salesforce Marketing Cloud Account Engagement.
- PARDOT_BUSINESS_UNIT_ID - (Obligatoire) Utilisé pour créer une connexion. L'ID de l'unité commerciale de l'instance Salesforce Marketing Cloud Account Engagement à laquelle vous souhaitez vous connecter.

- `API_VERSION`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Version de l'API Rest Salesforce Marketing Cloud Account Engagement que vous souhaitez utiliser.
- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- `FILTER_PREDICATE`(Chaîne) -
  - En mode synchronisation - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
  - En mode asynchrone - Par défaut : valeur `DateTime` actuelle (selon le fuseau horaire de l'utilisateur) : 1 an. Utilisé pour la lecture.
- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.
- `PARTITION_FIELD`(Chaîne) - Utilisé pour la lecture. Champ à utiliser pour partitionner la requête.
- `LOWER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite inférieure inclusive du champ de partition choisi.
- `UPPER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS`(Entier) - Par défaut : 1. Utilisé pour la lecture. Nombre de partitions pour la lecture.
- `INSTANCE_URL`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Une URL d'instance Salesforce Marketing Cloud Account Engagement valide.
- `PARTITION_BY`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Le type de partitionnement à effectuer. « FIELD » doit être transmis en cas de partitionnement basé sur des champs.
- `TRANSFER_MODE`(Chaîne) - (Facultatif), Valeur à utiliser pour exécuter une tâche en mode ASYNC. Si cette option n'est pas fournie, la tâche s'exécutera en mode SYNC.

Restrictions et remarques relatives au connecteur Salesforce Marketing Cloud Account Engagement

Les remarques et restrictions suivantes s'appliquent :

- Lorsqu'une limite et un partitionnement sont appliqués, la limite a priorité sur le partitionnement.
- Conformément aux documents de l'API, `SalesforceMarketingCloudEngagement` application d'une RateLimit sur les demandes quotidiennes et simultanées. Pour plus d'informations, consultez Plafonds.

- L'API d'exportation est soumise à la limite quotidienne d'appels de l'API Account Engagement et à la limite d'appels simultanés de l'API Account Engagement pour votre compte.
- À l'instar d'une file d'attente, les appels d' Export/Async API sont exécutés de manière séquentielle pour chaque compte. Les anciennes exportations sont traitées avant les nouvelles exportations.
- La partition n'est pas prise en charge en mode asynchrone.
- Le nombre de champs sélectionnés spécifiés dans les appels Export/Async d'API ne peut pas dépasser 150.
- L'entité Prospect prend en charge plus de 150 champs, mais seuls 150 champs peuvent être sélectionnés à la fois. Si `Select All` est sélectionné, certains champs seront exclus. Pour récupérer les données de ces champs exclus, vous devez les inclure dans l'option `Selected Fields`.

  Voici la liste des champs exclus dans `SELECT_ALL` : `updatedBy.firstName`, `updatedBy.lastName`, `updatedBy.jobTitle`, `updatedBy.roleName`, `updatedBy.salesforceId`, `updatedBy.createdAt`, `updatedBy.updatedAt`, `updatedBy.isDeleted`, `updatedBy.createdById`, `updatedBy.updatedById`, `updatedBy.tagReplacementLanguage`
- Les champs Collection ne peuvent être exportés pour Async. Par exemple, dans List Email, les champs `senderOptions` et `replyToOptions` ne sont pas pris en charge.
- Pour toutes les entités, le filtre est obligatoire. Si aucun filtre n'est fourni, le prédicat de filtre par défaut est défini sur le champ `Created After` avec une valeur de la date/heure actuelle (ajustée à votre fuseau horaire) moins un an.
- Conformément aux restrictions de Salesforce Marketing Cloud Account Engagement, en mode asynchrone, la plage maximale pour récupérer les données est d'un an. Si une requête est fournie pour plus d'un an, la tâche générera une erreur.
- Il existe actuellement un bogue dans Salesforce Pardot. Lorsque la tâche inclut un seul champ qui ne contient aucune donnée, la valeur du champ ne renvoie pas le résultat correct et le nom du champ est alors renvoyé plusieurs fois. L'équipe Salesforce Pardot est consciente du problème et travaille activement à sa résolution.

Connexion à SAP HANA dans AWS Glue Studio

AWS Glue fournit une prise en charge intégrée pour SAP HANA. AWS Glue Studio fournit une interface visuelle pour se connecter à SAP HANA, créer des tâches d'intégration de données et les exécuter sur l'exécution Spark AWS Glue Studio sans serveur.

AWS Glue Studio crée une connexion unifiée pour SAP HANA. Pour de plus amples informations, veuillez consulter Considérations.

Rubriques

- Création d'une connexion SAP HANA
- Création d'un nœud source SAP HANA
- Création d'un nœud cible SAP HANA
- Options avancées

Création d'une connexion SAP HANA

Pour vous connecter à SAP HANA depuis AWS Glue, vous devez créer et stocker vos informations d'identification SAP HANA dans un AWS Secrets Manager secret, puis associer ce secret à une connexion SAP AWS Glue HANA. Vous devrez configurer la connectivité réseau entre votre service SAP HANA et AWS Glue.

Prérequis :

- Si votre service SAP HANA se trouve dans un Amazon VPC, configurez Amazon VPC pour permettre à AWS Glue votre tâche de communiquer avec le service SAP HANA sans que le trafic ne transite par l'Internet public.

  Dans Amazon VPC, identifiez ou créez un VPC, un sous-réseau et un groupe de sécurité qui AWS Glue seront utilisés lors de l'exécution de la tâche. En outre, vous devez vous assurer qu'Amazon VPC est configuré pour autoriser le trafic réseau entre votre point de terminaison SAP HANA et cet emplacement. Votre tâche devra établir une connexion TCP avec votre port JDBC SAP HANA. Pour plus d'informations sur les ports SAP HANA, consultez la documentation SAP HANA. Selon la configuration de votre réseau, cela peut nécessiter des modifications des règles du groupe de sécurité, du réseau ACLs, des passerelles NAT et des connexions d'appairage.

Pour configurer une connexion à SAP HANA :

1. Dans AWS Secrets Manager, créez un secret à l'aide de vos informations d'identification SAP HANA. Pour créer un secret dans Secrets Manager, suivez le didacticiel disponible dans la section Créer un AWS Secrets Manager secret dans la AWS Secrets Manager documentation. Après avoir créé le secret, conservez le nom du secret *`secretName`* pour l'étape suivante.

- Lorsque vous sélectionnez des paires clé/valeur, créez une paire pour la clé `username/USERNAME` contenant la valeur. *saphanaUsername*
- Lorsque vous sélectionnez des paires clé/valeur, créez une paire pour la clé `password/PASSWORD` contenant la valeur. *saphanaPassword*

2. Dans la AWS Glue console, créez une connexion en suivant les étapes décrites dansthe section called "Ajouter une AWS Glue connexion". Après avoir créé la connexion, conservez le nom de la connexion pour une utilisation future dans AWS Glue. *connectionName*

   - Lorsque vous sélectionnez un type de connexion, sélectionnez SAP HANA.
   - Lorsque vous fournissez l'URL SAP HANA, indiquez l'URL de votre instance.

     SAP HANA JDBC URLs sont sous la forme `jdbc:sap://`*saphanaHostname*`:`*saphanaPort*`/?`*databaseName*`=`*saphanaDBname*`,`*Paramete*

     AWS Glue nécessite les paramètres d'URL JDBC suivants :

     - `databaseName` – une base de données par défaut dans SAP HANA à laquelle se connecter.
   - Lorsque vous sélectionnez un AWS secret, fournissez*secretName*.

Après avoir créé une connexion AWS Glue SAP HANA, vous devez effectuer les étapes suivantes avant d'exécuter votre AWS Glue tâche :

- Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire*secretName*.

Création d'un nœud source SAP HANA

Prérequis

- Une connexion AWS Glue SAP HANA, configurée avec un AWS Secrets Manager secret, comme décrit dans la section précédente,the section called "Création d'une connexion SAP HANA".
- Les autorisations appropriées sur votre tâche pour lire le secret utilisé par la connexion.
- Une table SAP HANA que vous souhaitez lire ou interroger*targetQuery*. *tableName*

  Une table peut être spécifiée avec un nom de table SAP HANA et un nom de schéma, sous forme de *schemaName*.*tableName*. Le nom du schéma et le séparateur « . » ne sont pas obligatoires si

la table se trouve dans le schéma par défaut, « public ». Appelle ça*tableIdentifier*. Notez que la base de données est fournie sous forme de paramètre d'URL JDBC dans `connectionName`.

Ajout d'une source de données SAP HANA

Pour ajouter un nœud Source de données – SAP HANA :

1. Choisissez la connexion pour votre source de données SAP HANA. Puisque vous l'avez créé, il devrait être disponible dans le menu déroulant. Si vous devez créer une connexion, choisissez Créer une connexion SAP HANA. Pour de plus amples informations, veuillez consulter la section précédente the section called “Création d'une connexion SAP HANA”.

   Une fois que vous avez choisi une connexion, vous pouvez afficher ses propriétés en cliquant sur Afficher les propriétés.
2. Choisissez une option Source SAP HANA :
   - Choisissez une seule table : accéder à toutes les données à partir d'une seule table.
   - Saisir une requête personnalisée : accéder à un jeu de données à partir de plusieurs tables en fonction de votre requête personnalisée.
3. Si vous avez choisi une seule table, entrez*tableName*.

   Si vous avez choisi Saisir une requête personnalisée, saisissez une requête SQL SELECT.
4. Dans les Propriétés SAP HANA personnalisées, saisissez les paramètres et les valeurs nécessaires.

Création d'un nœud cible SAP HANA

Prérequis

- Une connexion AWS Glue SAP HANA, configurée avec un AWS Secrets Manager secret, comme décrit dans la section précédente,the section called “Création d'une connexion SAP HANA”.
- Les autorisations appropriées sur votre tâche pour lire le secret utilisé par la connexion.
- Une table SAP HANA dans laquelle vous souhaitez écrire,*tableName*.

  Une table peut être spécifiée avec un nom de table SAP HANA et un nom de schéma, sous forme de *schemaName*.*tableName*. Le nom du schéma et le séparateur « . » ne sont pas obligatoires si

la table se trouve dans le schéma par défaut, « public ». Appelle ça*tableIdentifier*. Notez que la base de données est fournie sous forme de paramètre d'URL JDBC dans `connectionName`.

Ajout d'une cible de données SAP HANA

Pour ajouter un nœud Cible de données – SAP HANA :

1. Choisissez la connexion pour votre source de données SAP HANA. Puisque vous l'avez créé, il devrait être disponible dans le menu déroulant. Si vous devez créer une connexion, choisissez Créer une connexion SAP HANA. Pour de plus amples informations, veuillez consulter la section précédente the section called "Création d'une connexion SAP HANA".

   Une fois que vous avez choisi une connexion, vous pouvez afficher ses propriétés en cliquant sur Afficher les propriétés.
2. Configurez le nom de la table en fournissant*tableName*.
3. Dans les Propriétés Teradata personnalisées, saisissez les paramètres et les valeurs nécessaires.

Options avancées

Vous pouvez fournir des options avancées lors de la création d'un nœud SAP HANA. Ces options sont les mêmes que celles disponibles lors de la programmation AWS Glue de scripts Spark.

Consultez the section called "Connexions SAP HANA".

Connexion à SAP OData

SAP OData est un protocole Web standard utilisé pour interroger et mettre à jour les données présentes dans SAP à l'aide de l'ABAP (Advanced Business Application Programming), en appliquant et en s'appuyant sur des technologies Web telles que le HTTP pour fournir un accès aux informations provenant de diverses applications, plateformes et appareils externes. Le produit vous permet d'accéder à tout ce dont vous avez besoin pour vous intégrer parfaitement à votre système, à votre application ou à vos données SAP.

Rubriques

- AWS Glue support pour SAP OData
- Création de connexions

- Création d'un OData job SAP
- Écrire à SAP OData
- Utilisation du script de gestion OData d'état SAP
- Partitionnement pour les entités non ODP
- Options de OData connexion SAP
- Détails de OData l'entité et du champ SAP

AWS Glue support pour SAP OData

AWS Glue prend en charge SAP OData comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser des tâches AWS Glue ETL pour interroger des données auprès de SAP OData.

Prise en charge en tant que cible ?

Oui. Vous pouvez utiliser des tâches AWS Glue ETL pour écrire des enregistrements dans SAP OData.

Versions d' OData API SAP prises en charge

Les versions d' OData API SAP suivantes sont prises en charge :

- 2.0

Sources prises en charge

Les sources suivantes sont prises en charge :

- Sources ODP (Operational Data Provisioning) :
  - Extracteurs BW (1) DataSources
  - Vues CDS
  - SLT
- Sources non ODP, par exemple :
  - Services d'affichage CDS
  - Services basés sur RFC

- Services ABAP personnalisés

Composants SAP pris en charge

Les exigences minimales sont les suivantes :

- Vous devez activer le service de catalogue pour la découverte de service.
  - Configurez les sources de données ODP (Operational Data Provisioning) à extraire dans la passerelle SAP de votre système SAP.
  - OData V2.0 : Activez le ou les services de catalogue OData V2.0 dans votre passerelle SAP via une transaction. `/IWFND/MAINT_SERVICE`
  - Activez les services OData V2.0 dans votre passerelle SAP via une transaction. `/IWFND/MAINT_SERVICE`
  - Votre OData service SAP doit prendre en charge pagination/query les options côté client telles que `$top` et`$skip`. Il doit également prendre en charge l’option de requête système `$count`.
  - Vous devez fournir l'autorisation requise pour que l'utilisateur puisse découvrir les services et extraire les données à l'aide des OData services SAP. Reportez-vous à la documentation de sécurité fournie par SAP.
- Si vous souhaitez utiliser la OAuth version 2.0 comme mécanisme d'autorisation, vous devez activer la OAuth version 2.0 pour le OData service et enregistrer le OAuth client conformément à la documentation SAP.
- Pour générer un OData service basé sur des sources de données ODP, SAP Gateway Foundation doit être installé localement dans votre ERP/BW stack ou dans une configuration de hub.
  - Pour vos ERP/BW applications, la pile SAP NetWeaver AS ABAP doit être 7.50 SP02 ou supérieure.
  - Pour le système de hub (SAP Gateway), le SAP NetWeaver AS ABAP du système de hub doit être 7.50 SP01 ou supérieur pour la configuration du hub distant.
- Pour les sources non ODP, la version de votre NetWeaver stack SAP doit être 7.40 SP02 ou supérieure.

Méthodes d’authentification prises en charge

Les méthodes d’authentification suivantes sont prises en charge :

- Authentification de base

- OAuth 2,0

Conditions préalables

Avant de lancer une AWS Glue tâche d'extraction de données depuis SAP à OData l'aide de la OData connexion SAP, remplissez les conditions préalables suivantes :

- Le OData service SAP correspondant doit être activé dans le système SAP, afin de garantir que la source de données est disponible pour la consommation. Si le OData service n'est pas activé, la tâche Glue ne pourra pas accéder aux données de SAP ni en extraire.
- Les mécanismes d'authentification appropriés tels que l'authentification de base (personnalisée) ou OAuth 2.0 doivent être configurés dans SAP pour garantir que la AWS Glue tâche puisse établir avec succès une connexion avec le OData service SAP.
- Configurez les politiques IAM pour accorder à la AWS Glue tâche les autorisations appropriées pour accéder à SAP, à Secrets Manager et aux autres AWS ressources impliquées dans le processus.
- Si le système SAP est hébergé sur un réseau privé, la connectivité VPC doit être configurée pour garantir que la AWS Glue tâche puisse communiquer en toute sécurité avec SAP sans exposer de données sensibles sur Internet public.

AWS Secrets Manager peut être utilisé pour stocker en toute sécurité des informations sensibles telles que les informations d'identification SAP, que la AWS Glue tâche peut récupérer dynamiquement lors de l'exécution. Cette approche élimine le besoin de coder en dur les informations d'identification, et améliore ainsi la sécurité et la flexibilité.

Les prérequis suivants fournissent des step-by-step conseils sur la façon de configurer chaque composant pour une intégration fluide entre SAP AWS Glue et SAP OData.

Rubriques

- OData Activation de SAP
- politiques IAM
- Connectivité/connexion VPC
- Authentification SAP
- AWS Secrets Manager pour stocker votre secret d'authentification

OData Activation de SAP

Procédez comme suit pour la OData connexion SAP :

Sources ODP

Avant de pouvoir transférer des données depuis un fournisseur ODP, vous devez respecter les conditions requises suivantes :

- Vous disposez d'une instance SAP NetWeaver AS ABAP.
- Votre NetWeaver instance SAP contient un fournisseur ODP à partir duquel vous souhaitez transférer des données. Les fournisseurs ODP incluent :
  - SAP DataSources (code de transaction RSO2)
  - Vues SAP Core Data Services ABAP CDS
  - Systèmes SAP BW ou SAP BW/4HANA (, Object) InfoObject DataStore
  - Réplication en temps réel de tables et de vues de base de données depuis le système source SAP via SAP Landscape Replication Server (SAP SLT)
  - Vues d'informations SAP HANA dans des sources basées sur SAP ABAP
- Votre NetWeaver instance SAP possède le composant SAP Gateway Foundation.
- Vous avez créé un OData service qui extrait les données de votre fournisseur ODP. Pour créer le OData service, vous utilisez le SAP Gateway Service Builder. Pour accéder à vos données ODP, Amazon AppFlow appelle ce service à l'aide de l' OData API. Pour plus d'informations, consultez la section Génération d'un service d'extraction de données ODP OData dans la documentation SAP BW/4HANA.
- Pour générer un OData service basé sur des sources de données ODP, SAP Gateway Foundation doit être installé localement dans votre ERP/BW stack ou dans une configuration de hub.
  - Pour vos ERP/BW applications, la pile SAP NetWeaver AS ABAP doit être 7.50 SP02 ou supérieure.
  - Pour le système de hub (SAP Gateway), le SAP NetWeaver AS ABAP du système de hub doit être 7.50 SP01 ou supérieur pour la configuration du hub distant.

Sources non ODP

- Votre version de SAP NetWeaver Stack doit être 7.40 SP02 ou supérieure.
- Vous devez activer le service de catalogue pour la découverte de service.

  - OData V2.0 : Le ou les services de catalogue OData V2.0 peuvent être activés dans votre SAP Gateway via une transaction `/IWFND/MAINT_SERVICE`
- Votre OData service SAP doit prendre en charge pagination/query les options côté client telles que `$top` et`$skip`. Il doit également prendre en charge l'option de requête système `$count`.
- Pour la OAuth version 2.0, vous devez activer la OAuth version 2.0 pour le OData service, enregistrer le OAuth client conformément à la documentation SAP et définir l'URL de redirection autorisée comme suit :
  - `https://<region>.console.aws.amazon.com/gluestudio/oauth`, en `<region>` remplaçant par la région dans laquelle AWS Glue s'exécute, par exemple : us-east-1.
  - Vous devez activer la configuration sécurisée pour vous connecter via HTTPS.
- Vous devez fournir l'autorisation requise à l'utilisateur dans SAP pour découvrir les services et extraire les données à l'aide des OData services SAP. Reportez-vous à la documentation de sécurité fournie par SAP.

politiques IAM

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les autorisations AWS IAM requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    },
    {
```

```
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue"
      ],
      "Resource": "*"
    }
  ]
}
```

Le rôle doit accorder l'accès à toutes les ressources utilisées par la tâche, par exemple Amazon S3. Si vous ne voulez pas utiliser la méthode ci-dessus, vous pouvez également utiliser les politiques IAM gérées suivantes.

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.
- SecretsManagerReadWrite— Permet read/write d'accéder à AWS Secrets Manager via la console AWS de gestion. Remarque : Cela exclut les actions IAM, alors combinez-les avec `IAMFullAccess` si une configuration de rotation est requise.

IAM Policies/Permissions nécessaire pour configurer le VPC

Les autorisations IAM suivantes sont requises lors de l'utilisation d'une connexion VPC pour AWS Glue créer une connexion. Pour plus de détails, consultez Create an IAM policy for AWS Glue.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:CreateNetworkInterface",
        "ec2:DeleteNetworkInterface",
        "ec2:DescribeNetworkInterfaces"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
```

Connectivité/connexion VPC

Étapes de connexion VPC :

1. Utilisez la connexion VPC existante ou créez-en une autre en suivant la documentation Amazon VPC.
2. Assurez-vous que vous disposez d'une passerelle NAT qui achemine le trafic vers Internet.
3. Choisissez Amazon S3 Gateway comme point de terminaison d'un VPC pour créer une connexion.
4. Activez la résolution DNS et le nom d'hôte DNS pour utiliser les services DNS AWS fournis.
5. Accédez au VPC créé et ajoutez les points de terminaison nécessaires pour différents services tels que STS AWS Glue, Secret Managers.
   a. Choisissez Créer un point de terminaison.
   b. Pour Catégorie de service, sélectionnez AWS Services.
   c. Pour Nom du service, choisissez le service auquel vous vous connectez.
   d. Choisissez VPC et Activer le nom DNS.
   e. Points de terminaison de VCP requis pour la connexion VPC :
      i. STS

ii. AWS Glue

iii. Secrets Manager

Configuration du groupe de sécurité

Le groupe de sécurité doit autoriser le trafic vers son port d'écoute depuis le AWS Glue VPC AWS Glue pour pouvoir s'y connecter. Il est recommandé de limiter autant que possible la plage d’adresses IP source.

AWS Glue nécessite un groupe de sécurité spécial qui autorise tout le trafic entrant à partir de lui-même. Vous pouvez créer une règle de référence circulaire qui autorise tout le trafic provenant du groupe de sécurité. Vous pouvez modifier un groupe de sécurité existant et le spécifier en tant que source.

Ouvrez la communication depuis les ports HTTPS du point de terminaison de l’URL (instance NLB ou SAP).

Options de connectivité

- Connexion HTTPS avec NLB interne et externe, certificat SSL de l’autorité de certification (CA), pas de certificat SSL non auto-signé
- Connexion HTTPS avec certificat SSL d’instance SAP de l’autorité de certification (CA), pas de certificat SSL non auto-signé

Authentification SAP

Le connecteur SAP prend en charge les méthodes d’authentification PERSONNALISÉES (authentification SAP DE BASE) et OAUTH.

Authentification personnalisée

AWS Glue prend en charge l'authentification personnalisée (authentification de base) comme méthode d'établissement de connexions à vos systèmes SAP, permettant l'utilisation d'un nom d'utilisateur et d'un mot de passe pour un accès sécurisé. Ce type d'authentification fonctionne bien pour les scénarios d'automatisation car il permet d'utiliser le nom d'utilisateur et le mot de passe dès le départ avec les autorisations d'un utilisateur particulier dans l' OData instance SAP. AWS Glue est capable d'utiliser le nom d'utilisateur et le mot de passe pour authentifier SAP OData APIs. Dans AWS Glue, l'autorisation de base est implémentée en tant qu'autorisation personnalisée.

Pour consulter la OData documentation SAP publique sur le flux d'authentification de base, consultez la section Authentification de base HTTP.

OAuth 2.0 Authentification

AWS Glue prend également en charge la OAuth version 2.0 en tant que mécanisme d'authentification sécurisé pour établir des connexions à vos systèmes SAP. Cela permet une intégration fluide tout en garantissant la conformité aux normes d'authentification modernes et en renforçant la sécurité de l'accès aux données.

Type d'autorisation AUTHORIZATION_CODE

Le type de subvention détermine le AWS Glue mode de communication avec SAP OData pour demander l'accès à vos données. SAP ne OData prend en charge que le type de `AUTHORIZATION_CODE` subvention. Ce type de subvention est considéré comme « à trois étapes » OAuth car il repose sur la redirection des utilisateurs vers le serveur d'autorisation tiers pour authentifier l'utilisateur. Il est utilisé lors de la création de connexions via la AWS Glue console.

Les utilisateurs peuvent toujours choisir de créer leur propre application connectée dans SAP OData et de fournir leur propre identifiant client et leur propre secret client lors de la création de connexions via la AWS Glue console. Dans ce scénario, ils seront toujours redirigés vers SAP pour OData se connecter et autoriser AWS Glue l'accès à leurs ressources.

Ce type d'autorisation donne lieu à un jeton d'actualisation et à un jeton d'accès. Le jeton d'accès est de courte durée et peut être actualisé automatiquement sans interaction de l'utilisateur à l'aide du jeton d'actualisation.

Pour consulter OData la documentation SAP publique sur la création d'une application connectée pour le OAuth flux de code d'autorisation, voir Authentification Using OAuth 2.0.

AWS Secrets Manager pour stocker votre secret d'authentification

Vous devrez stocker les secrets de OData connexion SAP dans AWS Secrets Manager, configurer les autorisations nécessaires pour les récupérer, comme indiqué dans la politiques IAM section, et les utiliser lors de la création d'une connexion.

Utilisez la console AWS de gestion de AWS Secrets Manager pour créer un secret pour votre source SAP. Pour plus d'informations, consultez la section Créer un secret du Gestionnaire de AWS Secrets. Les informations AWS contenues dans Secrets Manager doivent inclure les éléments du code suivant.

Secret d'authentification personnalisé

Vous devrez saisir votre nom d'utilisateur du système SAP à la place de **<your SAP username>** et son mot de passe à la place de **<your SAP username password>**, ainsi que True ou False. Dans ce contexte, la définition de `basicAuthDisableSSO` sur `true` désactive l'authentification unique (SSO) pour les demandes d'authentification de base, en exigeant des informations d'identification utilisateur explicites pour chaque demande. Inversement, sa définition sur `false` autorise l'utilisation des sessions SSO existantes si elles sont disponibles.

```
{
   "basicAuthUsername": "<your SAP username>",
   "basicAuthPassword": "<your SAP username password>",
   "basicAuthDisableSSO": "<True/False>",
   "customAuthenticationType": "CustomBasicAuth"
}
```

OAuth 2.0 Secret

Si vous utilisez la OAuth version 2.0 comme mécanisme d'authentification, le secret du Gestionnaire de AWS Secrets doit avoir l'application ClientId cliente gérée par l'utilisateur au format suivant. Vous devrez saisir votre secret client SAP à la place de **<your client secret>**.

```
{"USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET": "<your client secret>"
}
```

Création de connexions

Pour configurer une OData connexion SAP :

1. Connectez-vous à la console AWS de gestion et AWS Glue ouvrez-la. Dans le AWS Glue Studio, créez une connexion en suivant les étapes ci-dessous :
   a. Cliquez sur Connexions de données sur le panneau de gauche.
   b. Cliquez sur Créer une connexion.
   c. Sélectionnez SAP OData dans Choisir une source de données
   d. Fournissez l'URL de l'hôte de l'application de l' OData instance SAP à laquelle vous souhaitez vous connecter. L'URL de l'hôte de l'application doit être accessible sur l'Internet public pour une connexion non-VPC.
   e. Indiquez le chemin du service d'application de l' OData instance SAP à laquelle vous souhaitez vous connecter. Il s'agit du même chemin d'accès que celui du service de catalogue. Par

exemple :`/sap/opu/odata/iwfnd/catalogservice;v=2`. AWS Glue n'accepte pas le chemin d'un objet spécifique.

f. Indiquez le numéro de client de l' OData instance SAP à laquelle vous souhaitez vous connecter. Les valeurs acceptables sont [001-999]. Exemple : 010

g. Indiquez le numéro de port de l' OData instance SAP à laquelle vous souhaitez vous connecter. Exemple : 443

h. Indiquez la langue d'ouverture de session de l' OData instance SAP à laquelle vous souhaitez vous connecter. Exemple : EN

i. Sélectionnez le rôle AWS IAM qui AWS Glue peut assumer et qui dispose des autorisations décrites dans la politiques IAM section.

j. Sélectionnez le type d'authentification que vous souhaitez utiliser pour cette connexion dans AWS Glue la liste déroulante : OAUTH2 ou PERSONNALISÉ

   i. PERSONNALISÉ : sélectionnez le secret que vous avez créé comme indiqué dans la section AWS Secrets Manager pour stocker votre secret d'authentification.

   ii. OAUTH 2.0 - entrez les entrées suivantes uniquement dans le cas de OAuth 2.0 :

      A. Sous Application client gérée par l'utilisateur ClientId, entrez votre identifiant client.

      B. `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET`(votre secret client) dans le Gestionnaire de AWS Secrets que vous avez créé dans la AWS Secrets Manager pour stocker votre secret d'authentification section.

      C. Sous URL du code d'autorisation, saisissez l'URL de votre code d'autorisation.

      D. Sous URL des jetons d'autorisation, saisissez l'URL de votre jeton d'autorisation.

      E. Sous OAuth Étendue, entrez vos OAuth portées en les séparant par un espace. Exemple : `/IWFND/SG_MED_CATALOG_0002 ZAPI_SALES_ORDER_SRV_0001`

k. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau. Pour en savoir plus, consultez Connectivité/connexion VPC.

2. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`. Pour plus de détails, consultez politiques IAM.

3. Choisissez Tester la connexion et testez votre connexion. Si le test de connexion réussit, cliquez sur Suivant, saisissez le nom de votre connexion et enregistrez-la. La fonctionnalité de test de connexion n'est pas disponible si vous avez sélectionné Options réseau (VPC).

Création d'un OData job SAP

Reportez-vous à la section Création de tâches ETL visuelles avec AWS Glue Studio

Sources ODP (Operational Data Provisioning)

L'ODP (Operational Data Provisioning) fournit une infrastructure technique que vous pouvez utiliser pour prendre en charge l'extraction et la réplication des données pour diverses applications cible et prend en charge les mécanismes Delta dans ces scénarios. Dans le cas d'une procédure Delta, les données d'une source (fournisseur ODP) sont automatiquement écrites dans une file d'attente Delta (Operational Delta Queue, ODQ) à l'aide d'un processus de mise à jour ou transmises à la file d'attente Delta à l'aide d'une interface d'extracteur. Un fournisseur ODP peut être un DataSource (extracteurs), des vues ABAP Core Data Services (vues ABAP CDS), SAP BW ou SAP BW/4HANA, un serveur SAP Landscape Transformation Replication (SLT) et des vues d'informations SAP HANA (vues de calcul). Les applications cible (appelées « abonnés » ODQ ou plus généralement « consommateurs ODP ») extraient les données de la file d'attente Delta et continuent à les traiter.

Chargement complet

Dans le contexte des entités SAP OData et ODP, un chargement complet fait référence au processus d'extraction de toutes les données disponibles d'une entité ODP en une seule opération. Cette opération extrait l'ensemble de données complet du système source, garantissant ainsi que le système cible dispose d'une up-to-date copie complète des données de l'entité. Les chargements complets sont généralement utilisés pour les sources qui ne prennent pas en charge les chargements incrémentiels ou lorsqu'une actualisation du système cible est requise.

Exemple

Vous pouvez définir explicitement l'`ENABLE_CDC`indicateur sur false lors de la création du DynamicFrame. Remarque : `ENABLE_CDC` est false par défaut, si vous ne souhaitez pas initialiser la file d'attente Delta, vous n'avez pas besoin d'envoyer cet indicateur ou de le définir sur true. Le fait de ne pas définir cet indicateur sur true entraînera une extraction de chargement complet.

```
sapodata_df = glueContext.create_dynamic_frame.from_options(
    connection_type="SAPOData",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "ENABLE_CDC": "false"
    }, transformation_ctx=key)
```

Chargement incrémentiel

Un chargement incrémentiel dans le contexte des entités ODP (Operational Data Provisioning) implique d'extraire uniquement les données nouvelles ou modifiées (deltas) du système source depuis la dernière extraction des données, en évitant ainsi de prétraiter les enregistrements déjà traités. Cette approche améliore considérablement l'efficacité, réduit les volumes de transfert de données, améliore les performances, garantit une synchronisation efficace entre les systèmes et minimise le temps de traitement, en particulier pour les grands jeux de données qui changent fréquemment.

Transferts incrémentiels basés sur des jetons Delta

Pour activer le transfert incrémentiel à l'aide de la capture des données de modification (CDC) pour les entités compatibles ODP qui le prennent en charge, procédez comme suit :

1. Créez la tâche de transfert incrémentiel en mode script.
2. Lorsque vous créez le DataFrame or Glue DynamicFrame, vous devez passer l'option`"ENABLE_CDC": "True"`. Cette option garantit que vous recevrez un jeton Delta de la part de SAP, qui pourra être utilisé pour extraire ultérieurement les données modifiées.

Le jeton Delta sera présent dans la dernière ligne du dataframe, dans la colonne DELTA_TOKEN. Ce jeton peut être utilisé comme option de connecteur lors des appels suivants pour récupérer de manière incrémentielle le prochain jeu de données.

Exemple

- Nous avons défini le `ENABLE_CDC` drapeau sur`true`, lors de la création du DynamicFrame. Remarque : `ENABLE_CDC` est `false` par défaut, si vous ne souhaitez pas initialiser la file d'attente Delta, vous n'avez pas besoin d'envoyer cet indicateur ou de le définir sur true. Le fait de ne pas définir cet indicateur sur true entraînera une extraction de chargement complet.

```
sapodata_df = glueContext.create_dynamic_frame.from_options(
    connection_type="SAPOData",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "ENABLE_CDC": "true"
    }, transformation_ctx=key)

# Extract the delta token from the last row of the DELTA_TOKEN column
```

```
delta_token_1 = your_logic_to_extract_delta_token(sapodata_df) # e.g.,
 D20241029164449_000370000
```

- Le jeton Delta extrait peut être transmis en option pour extraire de nouveaux événements.

```
sapodata_df_2 = glueContext.create_dynamic_frame.from_options(
    connection_type="SAPOData",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        // passing the delta token retrieved in the last run
        "DELTA_TOKEN": delta_token_1
    } , transformation_ctx=key)

# Extract the new delta token for the next run
delta_token_2 = your_logic_to_extract_delta_token(sapodata_df_2)
```

Notez que le dernier enregistrement, dans lequel le `DELTA_TOKEN` est présent, n'est pas un enregistrement transactionnel provenant de la source et n'est là que dans le but de transmettre la valeur du jeton Delta.

Outre `DELTA_TOKEN`, les champs suivants sont renvoyés dans chaque ligne du dataframe.

- GLUE_FETCH_SQ : il s'agit d'un champ de séquence, généré à partir de l'horodatage EPOC dans l'ordre dans lequel l'enregistrement a été reçu, et qui est unique pour chaque enregistrement. Il peut être utilisé si vous avez besoin de connaître ou d'établir l'ordre des modifications dans le système source. Ce champ ne sera présent que pour les entités compatibles ODP.
- DML_STATUS : cela affichera `UPDATED` pour tous les enregistrements récemment insérés et mis à jour à partir de la source, ainsi que `DELETED` pour les enregistrements supprimés de la source.

Pour plus de détails sur la façon de gérer l'état et de réutiliser le jeton Delta pour extraire les enregistrements modifiés à l'aide d'un exemple, reportez-vous à la section Utilisation du script de gestion OData d'état SAP.

Invalidation de jeton Delta

Un jeton Delta est associé à la collection de services et à un utilisateur. Si un nouveau pull with initial `"ENABLE_CDC" : "true"` est lancé pour la même collection de services et pour le même utilisateur, tous les jetons delta émis à la suite d'une initialisation précédente seront invalidés par

le service SAP OData . L'invocation du connecteur avec un jeton Delta expiré entraînera une exception :

```
Could not open data access via extraction API RODPS_REPL_ODP_OPEN
```

OData Services (sources autres que le PDO)

Chargement complet

Pour les systèmes non ODP (Operational Data Provisioning), un chargement complet consiste à extraire le jeu de données du système source et à le charger dans le système cible. Étant donné que les systèmes non ODP ne prennent pas en charge par nature les mécanismes avancés d'extraction de données tels que les deltas, le processus est simple, mais peut nécessiter beaucoup de ressources en fonction de la taille des données.

Chargement incrémentiel

Pour les systèmes ou les entités qui ne prennent pas en charge ODP (Operational Data Provisioning), le transfert de données incrémentiel peut être géré manuellement en mettant en œuvre un mécanisme basé sur l'horodatage pour suivre et extraire les modifications.

Transferts incrémentiels basés sur l'horodatage

Pour les entités non compatibles ODP (ou pour les entités compatibles ODP qui n'utilisent pas l'indicateur ENABLE_CDC), nous pouvons utiliser une option `filteringExpression` dans le connecteur pour indiquer l'intervalle `datetime` pendant lequel nous voulons extraire les données. Cette méthode repose sur un champ d'horodatage dans vos données qui représente la date à laquelle chaque enregistrement a été créé/modifié pour la dernière fois.

Exemple

Récupération des enregistrements modifiés après 2024-01-01T 00:00:00.000

```
sapodata_df = glueContext.create_dynamic_frame.from_options(
    connection_type="SAPOData",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "filteringExpression": "LastChangeDateTime >= 2024-01-01T00:00:00.000"
    }, transformation_ctx=key)
```

Remarque : Dans cet exemple, `LastChangeDateTime` est le champ qui représente la date de dernière modification de chaque enregistrement. Le nom du champ réel peut varier en fonction de votre OData entité SAP spécifique.

Pour obtenir un nouveau sous-ensemble de données lors des exécutions suivantes, vous devez mettre à jour la valeur `filteringExpression` avec un nouvel horodatage. Il s'agit généralement de la valeur d'horodatage maximale des données précédemment extraites.

Exemple

```
max_timestamp = get_max_timestamp(sapodata_df)  # Function to get the max timestamp
 from the previous run
next_filtering_expression = f"LastChangeDateTime > {max_timestamp}"

# Use this next_filtering_expression in your next run
```

Dans la section suivante, nous proposerons une approche automatisée pour gérer ces transferts incrémentiels basés sur l'horodatage, ce qui évite ainsi de devoir mettre à jour manuellement l'expression de filtrage entre les exécutions.

Écrire à SAP OData

Cette section décrit comment écrire des données dans votre OData service SAP à l'aide du AWS Glue connecteur pour SAP OData.

Conditions préalables

- Accès à un OData service SAP
- Un OData EntitySet objet SAP sur lequel vous souhaitez écrire. Vous aurez besoin du nom de l'objet.
- OData Informations d'identification SAP valides et connexion valide
- Autorisations appropriées telles que décrites dans Politiques IAM

Le OData connecteur SAP prend en charge deux opérations d'écriture :

- INSERT
- UPDATE

Lorsque vous utilisez l'opération d'écriture UPDATE, vous devez fournir l'élément ID_FIELD_NAMES afin de spécifier le champ d'ID externe pour les enregistrements.

Exemple :

```
sapodata_write = glueContext.write_dynamic_frame.from_options(
    frame=frameToWrite,
    connection_type="sapodata",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "WRITE_OPERATION": "INSERT"
    }
```

Utilisation du script de gestion OData d'état SAP

Pour utiliser le script de gestion OData d'état SAP dans le cadre de votre AWS Glue travail, procédez comme suit :

- Téléchargez le script de gestion d'état : `s3://aws-blogs-artifacts-public/artifacts/BDB-4789/sap_odata_state_management.zip` depuis le compartiment Amazon S3 public.
- Téléchargez le script dans un compartiment Amazon S3 auquel votre AWS Glue travail est autorisé à accéder.
- Référencez le script dans votre AWS Glue tâche : lors de la création ou de la mise à jour de votre AWS Glue tâche, transmettez l'`''--extra-py-files'` option faisant référence au chemin du script dans votre compartiment Amazon S3. Par exemple : `--extra-py-files s3://your-bucket/path/to/sap_odata_state_management.py`
- Importez et utilisez la bibliothèque de gestion des états dans vos scripts de AWS Glue travail.

Exemple de transfert incrémentiel basé sur des jetons Delta

Voici un exemple d'utilisation du script de gestion d'état pour les transferts incrémentiels basés sur des jetons Delta :

```
from sap_odata_state_management import StateManagerFactory, StateManagerType, StateType

# Initialize the state manager
state_manager = StateManagerFactory.create_manager(
    manager_type=StateManagerType.JOB_TAG,
    state_type=StateType.DELTA_TOKEN,
```

```
    options={
        "job_name": args['JOB_NAME'],
        "logger": logger
    }
)

# Get connector options (including delta token if available)
key = "SAPODataNode"
connector_options = state_manager.get_connector_options(key)

# Use the connector options in your Glue job
df = glueContext.create_dynamic_frame.from_options(
    connection_type="SAPOData",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "ENABLE_CDC": "true",
        **connector_options
    }
)

# Process your data here...

# Update the state after processing
state_manager.update_state(key, sapodata_df.toDF())
```

Exemple de transfert incrémentiel basé sur l'horodatage

Voici un exemple d'utilisation du script de gestion d'état pour les transferts incrémentiels basés sur des jetons Delta :

```
from sap_odata_state_management import StateManagerFactory, StateManagerType, StateType

# Initialize the state manager
state_manager = StateManagerFactory.create_manager(
    manager_type=StateManagerType.JOB_TAG,
    state_type=StateType.DELTA_TOKEN,
    options={
        "job_name": args['JOB_NAME'],
        "logger": logger
    }
)
```

```
# Get connector options (including delta token if available)
key = "SAPODataNode"
connector_options = state_manager.get_connector_options(key)

# Use the connector options in your Glue job
df = glueContext.create_dynamic_frame.from_options(
    connection_type="SAPOData",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "ENABLE_CDC": "true",
        **connector_options
    }
)

# Process your data here...

# Update the state after processing
state_manager.update_state(key, sapodata_df.toDF())
```

Dans les deux exemples, le script de gestion d'état gère les complexités liées au stockage de l'état (jeton Delta ou horodatage) entre les exécutions de tâches. Il récupère automatiquement le dernier état connu lors de l'obtention des options du connecteur et met à jour l'état après le traitement, ce qui garantit ainsi que chaque tâche exécutée ne traite que les données nouvelles ou modifiées.

Partitionnement pour les entités non ODP

Dans Apache Spark, le partitionnement fait référence à la manière dont les données sont divisées et distribuées entre les composants master d'un cluster pour un traitement parallèle. Chaque partition est un bloc logique de données qui peut être traité indépendamment par une tâche. Le partitionnement est un concept fondamental de Spark qui a un impact direct sur les performances, l'évolutivité et l'utilisation des ressources. AWS Glue les jobs utilisent le mécanisme de partitionnement de Spark pour diviser l'ensemble de données en petits morceaux (partitions) qui peuvent être traités en parallèle sur les nœuds de travail du cluster. Notez que le partitionnement n'est pas applicable aux entités ODP.

Pour plus de détails, consultez AWS Glue Spark et PySpark jobs.

Conditions préalables

Un objet SAP OData dont vous souhaitez lire un extrait. Vous aurez besoin du object/EntitySet nom. Par exemple : `/sap/opu/odata/sap/API_SALES_ORDER_SRV/A_SalesOrder` .

Exemple

```
sapodata_read = glueContext.create_dynamic_frame.from_options(
    connection_type="SAPOData",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "/sap/opu/odata/sap/API_SALES_ORDER_SRV/A_SalesOrder"
    }, transformation_ctx=key)
```

Requêtes de partitionnement

Partitionnement basé sur les champs

Vous pouvez indiquer les options Spark supplémentaires PARTITION_FIELD, LOWER_BOUND, UPPER_BOUND et NUM_PARTITIONS si vous souhaitez utiliser la simultanéité dans Spark. Avec ces paramètres, la requête d'origine serait divisée en NUM_PARTITIONS nombres de sous-requêtes pouvant être exécutées simultanément par les tâches Spark. Les nombres entiers, les dates et DateTime les champs prennent en charge le partitionnement basé sur les champs dans le connecteur SAP. OData

- PARTITION_FIELD : le nom du champ à utiliser pour partitionner la requête.
- LOWER_BOUND : une valeur limite inférieure inclusive du champ de partition choisi.

  Pour tout champ dont le type de données est DateTime, le format d'horodatage Spark utilisé dans les requêtes SQL Spark est accepté.

  Exemples de valeurs valides : "2000-01-01T00:00:00.000Z"
- UPPER_BOUND : une valeur limite supérieure exclusive du champ de partition choisi.
- NUM_PARTITIONS : nombre de partitions.
- PARTITION_BY : le type de partitionnement à effectuer, FIELD à transmettre en cas de partitionnement basé sur des champs.

Exemple

```
sapodata= glueContext.create_dynamic_frame.from_options(
    connection_type="sapodata",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "/sap/opu/odata/sap/SEPM_HCM_SCENARIO_SRV/EmployeeSet",
```

```
        "PARTITION_FIELD": "validStartDate"
        "LOWER_BOUND": "2000-01-01T00:00:00.000Z"
        "UPPER_BOUND": "2020-01-01T00:00:00.000Z"
        "NUM_PARTITIONS": "10",
        "PARTITION_BY": "FIELD"
    }, transformation_ctx=key)
```

Partitionnement basé sur des enregistrements

La requête d'origine serait divisée en `NUM_PARTITIONS` nombres de sous-requêtes qui peuvent être exécutées simultanément par les tâches Spark.

Le partitionnement basé sur les enregistrements n'est pris en charge que pour les entités non ODP, car la pagination dans les entités ODP est prise en charge par le jeton suivant. token/skip

- `PARTITION_BY` : le type de partitionnement à effectuer. `COUNT` doit être transmis en cas de partitionnement basé sur des enregistrements.

Exemple

```
sapodata= glueContext.create_dynamic_frame.from_options(
    connection_type="sapodata",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "/sap/opu/odata/sap/SEPM_HCM_SCENARIO_SRV/EmployeeSet",
        "NUM_PARTITIONS": "10",
        "PARTITION_BY": "COUNT"
    }, transformation_ctx=key)
```

Restrictions/alertes

- Les entités ODP ne sont pas compatibles avec le partitionnement basé sur les enregistrements car la pagination est gérée à l'aide d'un jeton de saut. token/delta Par conséquent, pour le partitionnement basé sur les enregistrements, la valeur par défaut de maxConcurrency est définie sur « null », quelle que soit la saisie de l'utilisateur.
- Lorsqu'une limite et une partition sont appliquées, la limite a priorité sur la partition.

Options de OData connexion SAP

Les options de connexion pour SAP sont les OData suivantes :

- ENTITY_NAME(Chaîne) - (Obligatoire) Utilisé pour la lecture. Le nom de votre objet dans SAP OData.

  Par exemple :/sap/opu/odata/sap/API_SALES_ORDER_SRV/A_ SalesOrder

- API_VERSION(Chaîne) - (Facultatif) Utilisé pour la lecture. Version de l' OData API SAP Rest que vous souhaitez utiliser. Exemple : 2.0.
- SELECTED_FIELDS(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.

  Exemples : SalesOrder

- FILTER_PREDICATE(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.

  Par exemple : `SalesOrder = "10"`

- QUERY(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.

  Par exemple : `SELECT * FROM /sap/opu/odata/sap/API_SALES_ORDER_SRV/A_SalesOrder`

- PARTITION_FIELD(Chaîne) - Utilisé pour la lecture. Champ à utiliser pour partitionner la requête.

  Par exemple : `ValidStartDate`

- LOWER_BOUND(Chaîne) - Utilisé pour la lecture. Une valeur limite inférieure inclusive du champ de partition choisi.

  Par exemple : `"2000-01-01T00:00:00.000Z"`

- UPPER_BOUND(Chaîne) - Utilisé pour la lecture. Une valeur limite supérieure exclusive du champ de partition choisi.

  Par exemple : `"2024-01-01T00:00:00.000Z"`

- NUM_PARTITIONS(Entier) - Par défaut : 1. Utilisé pour la lecture. Nombre de partitions pour la lecture.
- INSTANCE_URL(Chaîne) - L'URL de l'hôte de l'application de l'instance SAP.

  Par exemple : `https://example-externaldata.sierra.aws.dev`

- SERVICE_PATH(Chaîne) - Le chemin d'accès au service d'application de l'instance SAP.

  Par exemple : `/sap/opu/odata/iwfnd/catalogservice;v=2`

- `CLIENT_NUMBER`(Chaîne) - Le numéro de client de l'application de l'instance SAP.

  Par exemple : 100

- `PORT_NUMBER`(Chaîne) - Par défaut : le numéro de port de l'application de l'instance SAP.

  Par exemple : 443

- `LOGON_LANGUAGE`(Chaîne) - La langue de connexion à l'application de l'instance SAP.

  Par exemple : `EN`

- `ENABLE_CDC`(Chaîne) - Définit s'il faut exécuter une tâche avec CDC activé, c'est-à-dire avec le suivi des modifications.

  Par exemple : `True/False`

- `DELTA_TOKEN`(Chaîne) - Exécute une extraction de données incrémentielle en fonction du jeton Delta valide fourni.

  Par exemple : `D20241107043437_000463000`

- `PAGE_SIZE`(Entier) - Définit la taille de page pour interroger les enregistrements. La taille de page par défaut est de 50 000. Lorsqu'une taille de page est spécifiée, SAP renvoie uniquement le nombre défini d'enregistrements par appel d'API, plutôt que le jeu de données complet. Le connecteur fournira toujours le nombre total d'enregistrements et gérera la pagination en utilisant la taille de page que vous avez indiquée. Si vous avez besoin d'une taille de page plus importante, vous pouvez choisir n'importe quelle valeur jusqu'à 500 000, qui est le maximum autorisé. Toute taille de page spécifiée supérieure à 500 000 sera ignorée. Le système utilisera plutôt la taille de page maximale autorisée. Vous pouvez spécifier le format de page dans l' AWS Glue Studio interface utilisateur en ajoutant une option de connexion `PAGE_SIZE` avec la valeur souhaitée.

  Par exemple : `20000`

Détails de OData l'entité et du champ SAP

Entité	Type de données	Opérateurs pris en charge
Tables (entités dynamiques)	String	=, !=, >, >=, <, <=, BETWEEN, LIKE

Entité	Type de données	Opérateurs pris en charge
	Entier	=, !=, >, >=, <, <=, BETWEEN, LIKE
	Long	=, !=, >, >=, <, <=, BETWEEN, LIKE
	Double	=, !=, >, >=, <, <=, BETWEEN, LIKE
	Date	=, !=, <, <=, >, >=, BETWEEN, LIKE
	DateTime	=, !=, <, <=, >, >=, BETWEEN, LIKE
	Booléen	=, !=
	Struct	=, !=, >, >=, <, <=, BETWEEN, LIKE

Connexion à SendGrid

SendGrid est une plateforme de communication client pour les e-mails transactionnels et marketing.

- SendGrid Le connecteur aide à créer et à gérer des listes de contacts et à créer des campagnes de marketing par e-mail.
- SendGrid permet aux entreprises en ligne, aux organisations à but non lucratif et à d'autres entités en ligne de créer et d'envoyer des e-mails marketing à un large public et de surveiller l'engagement avec ces e-mails.

Rubriques

- AWS Glue support pour SendGrid
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration SendGrid
- Configuration des SendGrid connexions

- Lecture à partir d' SendGrid entités
- SendGrid options de connexion
- SendGrid limites

AWS Glue support pour SendGrid

AWS Glue prend en charge SendGrid les éléments suivants :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser des tâches AWS Glue ETL pour interroger des données à partir de SendGrid.

Prise en charge en tant que cible ?

Non.

Versions SendGrid d'API prises en charge

Les versions SendGrid d'API suivantes sont prises en charge :

- v3

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les autorisations AWS IAM requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
```

```
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Si vous ne voulez pas utiliser la méthode ci-dessus, vous pouvez également utiliser les politiques IAM gérées suivantes :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration SendGrid

Avant de pouvoir AWS Glue transférer des données depuis SendGrid, vous devez satisfaire aux exigences suivantes :

Configuration requise

Les exigences minimales sont les suivantes :

- Vous avez un SendGrid compte doté d'une clé API.
- Votre SendGrid compte dispose d'un accès à l'API avec une licence valide.

Si vous remplissez ces conditions, vous êtes prêt à vous connecter AWS Glue à votre SendGrid compte. Pour les connexions classiques, vous n'avez rien d'autre à faire dans SendGrid.

Configuration des SendGrid connexions

SendGrid prend en charge l'authentification personnalisée.

Pour consulter SendGrid la documentation publique sur la génération des clés d'API requises pour une authentification personnalisée, consultez Authentification.

Pour configurer une SendGrid connexion, procédez comme suit :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes :
   a. Pour l'application connectée gérée par le client, le secret doit contenir le secret du consommateur de l'application connectée avec *`api_key`* comme clé.
   b. Remarque : vous devez créer un secret pour vos connexions dans AWS Glue.

1. Dans AWS Glue Glue Studio, créez une connexion sous Data Connections en suivant les étapes ci-dessous :
   a. Lorsque vous sélectionnez un type de connexion, sélectionnez SendGrid.
   b. Indiquez `INSTANCE_URL` l' SendGrid instance à laquelle vous souhaitez vous connecter.
   c. Sélectionnez le rôle AWS IAM qui AWS Glue peut assumer les actions suivantes et qui dispose des autorisations nécessaires pour les actions suivantes :

      JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

d. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.

e. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.

2. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.

Lecture à partir d' SendGrid entités

Prérequis

Un SendGrid objet que vous aimeriez lire. Vous aurez besoin du nom de l'objet, comme `lists`, `singlesends` ou `segments`.

Entités prises en charge pour la source :

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Lists	Non	Oui	Non	Oui	Non
Single Sends	Oui	Oui	Non	Oui	Non
Marketing Campaign Stats-Aut omations	Oui	Oui	Non	Oui	Non
Marketing Campaign Stats-Single Sends	Oui	Oui	Non	Oui	Non
Segments	Oui	Non	Non	Oui	Non
Contacts	Oui	Non	Non	Oui	Non
Catégorie	Non	Non	Non	Oui	Non
Statistiques	Oui	Non	Non	Oui	Non

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Unsubscribe Groups	Oui	Non	Non	Oui	Non

Exemple :

```
sendgrid_read = glueContext.create_dynamic_frame.from_options(
    connection_type="sendgrid",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "lists",
        "API_VERSION": "v3",
        "INSTANCE_URL": "instanceUrl"
    }
```

SendGrid détails de l'entité et du champ :

Entités avec métadonnées statiques :

Entité	Champ	Type de données	Opérateurs pris en charge	
Lists	id	String	N/A	
	name	String	N/A	
	contact_count	Entier	N/A	
	_metadata	Struct	N/A	
Single Sends	id	String	N/A	
	name	String	EQUAL_TO	
	abtest	Struct	N/A	
	status	String	EQUAL_TO	

Entité	Champ	Type de données	Opérateurs pris en charge	
	categories	List	EQUAL_TO	
	send_at	String	N/A	
	is_abtest	Booléen	N/A	
	updated_at	String	N/A	
	created_at	String	N/A	
	canaux	List	N/A	
Marketing Campaign Stats-Automations	id	String	N/A	
	aggregation	String	N/A	
	step_id	String	N/A	
	stats	Struct	N/A	
	automation_ids	List	EQUAL_TO	
Marketing Campaign Stats-Singlesends	id	String	N/A	
	ab_variation	String	N/A	
	ab_phase	String	N/A	
	aggregation	String	N/A	
	stats	Struct	N/A	
	singlesend_ids	List	EQUAL_TO	
Segments	id	String	N/A	
	name	String	N/A	
	query_version	String	N/A	

Entité	Champ	Type de données	Opérateurs pris en charge	
	contacts_count	Entier	N/A	
	sample_updated_at	String	N/A	
	next_sample_update	String	N/A	
	created_at	String	N/A	
	updated_at	String	N/A	
	parent_list_id	String	N/A	
	status	Struct	N/A	
	parent_list_ids	String	EQUAL_TO	
	no_parent_list_id	Booléen	EQUAL_TO	
Contacts	id	String	N/A	
	first_name	String	N/A	
	last_name	String	N/A	
	unique_name	String	N/A	
	e-mail	String	N/A	
	alternate_emails	List	N/A	
	address_line_1	String	N/A	
	address_line_2	String	N/A	
	city	String	N/A	

Entité	Champ	Type de données	Opérateurs pris en charge	
	state_province_region	String	N/A	
	country	String	N/A	
	postal_code	String	N/A	
	phone_number	String	N/A	
	whatsapp	String	N/A	
	line	String	N/A	
	facebook	String	N/A	
	list_ids	List	N/A	
	custom_fields	Struct	N/A	
	created_at	String	N/A	
	updated_at	String	N/A	
	_metadata	Struct	N/A	
	event_timestamp	DateTime	BETWEEN	
Catégorie	categories	List	N/A	
Statistiques	date	String	N/A	
	stats	List	N/A	
	start_date	DateTime	EQUAL_TO, BETWEEN	
	aggregated_by	String	EQUAL_TO	

Entité	Champ	Type de données	Opérateurs pris en charge
Unsubscribe Groups	id	Entier	EQUAL_TO
	name	String	N/A
	description	String	N/A
	last_email_sent_at	Entier	N/A
	is_default	Booléen	N/A
	unsubscribes	Entier	N/A

> **Note**
>
> Les types de données Struct et List sont convertis en type de données String, et le type de DateTime données est converti en Timestamp dans la réponse des connecteurs.

Requêtes de partitionnement

SendGrid ne prend pas en charge le partitionnement basé sur des filtres ou le partitionnement basé sur des enregistrements.

SendGrid options de connexion

Les options de connexion suivantes sont disponibles pour SendGrid :

- `ENTITY_NAME`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Le nom de votre objet dans SendGrid.
- `API_VERSION`(Chaîne) - (Obligatoire) Utilisé pour la lecture. SendGrid Version de l'API Rest que vous souhaitez utiliser.
- `INSTANCE_URL`(Chaîne) - (Obligatoire) Utilisé pour la lecture. URL d' SendGrid instance valide.
- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.

- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.

SendGrid limites

Les restrictions ou remarques suivantes concernent SendGrid :

- L'extraction incrémentielle n'est prise en charge que par l'entité Stats sur le champ `start_date` et par l'entité Contact sur le champ `event_timestamp`.
- La pagination n'est prise en charge que par les entités Marketing Campaign Stats (Automations), Marketing Campaign Stats (Single Sends), Single Sends et Lists.
- Pour l'entité Stats, `start_date` est un paramètre de filtre obligatoire.
- Une clé d'API à accès restreint ne peut pas prendre en charge l'accès en lecture pour les entités Email API et Stats. Utilisez une clé d'API avec accès complet. Pour plus d'informations, consultez API Overview.

Connexion à ServiceNow

ServiceNow est une plateforme SaaS basée sur le cloud pour automatiser les flux de travail de gestion informatique. La ServiceNow plateforme s'intègre facilement à d'autres outils, permettant aux utilisateurs de gérer les projets, les équipes et les interactions avec les clients à l'aide de diverses applications et plugins. En tant qu' ServiceNow utilisateur, vous pouvez vous connecter AWS Glue à votre ServiceNow compte. Vous pouvez ensuite l'utiliser ServiceNow comme source de données dans vos tâches ETL. Exécutez ces tâches pour transférer des données entre ServiceNow des AWS services ou d'autres applications prises en charge.

Rubriques

- AWS Glue support pour ServiceNow
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration ServiceNow
- Configuration des ServiceNow connexions
- Lecture à partir d' ServiceNow entités
- ServiceNow options de connexion
- Limitations et remarques relatives au ServiceNow connecteur

AWS Glue support pour ServiceNow

AWS Glue prend en charge ServiceNow les éléments suivants :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser des tâches AWS Glue ETL pour interroger des données à partir de ServiceNow.

Prise en charge en tant que cible ?

Non.

Versions ServiceNow d'API prises en charge

Les versions ServiceNow d'API suivantes sont prises en charge :

- v2

Pour connaître la prise en charge des entités par version, consultez la section Entités prises en charge pour Source.

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les autorisations AWS IAM requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
```

```
      ],
      "Resource": "*"
    }
  ]
}
```

Si vous ne voulez pas utiliser la méthode ci-dessus, vous pouvez également utiliser les politiques IAM gérées suivantes :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration ServiceNow

Avant de pouvoir AWS Glue transférer des données depuis ServiceNow, vous devez satisfaire aux exigences suivantes :

Configuration requise

Les exigences minimales sont les suivantes :

- Vous avez un ServiceNow compte avec e-mail et mot de passe. Pour de plus amples informations, veuillez consulter Création d'un ServiceNow compte.
- Votre ServiceNow compte est activé pour accéder à l'API. Toute utilisation de l' ServiceNow API est disponible sans frais supplémentaires.

Si vous remplissez ces conditions, vous êtes prêt à vous connecter AWS Glue à votre ServiceNow compte.

Création d'un ServiceNow compte

Pour créer un ServiceNow compte :

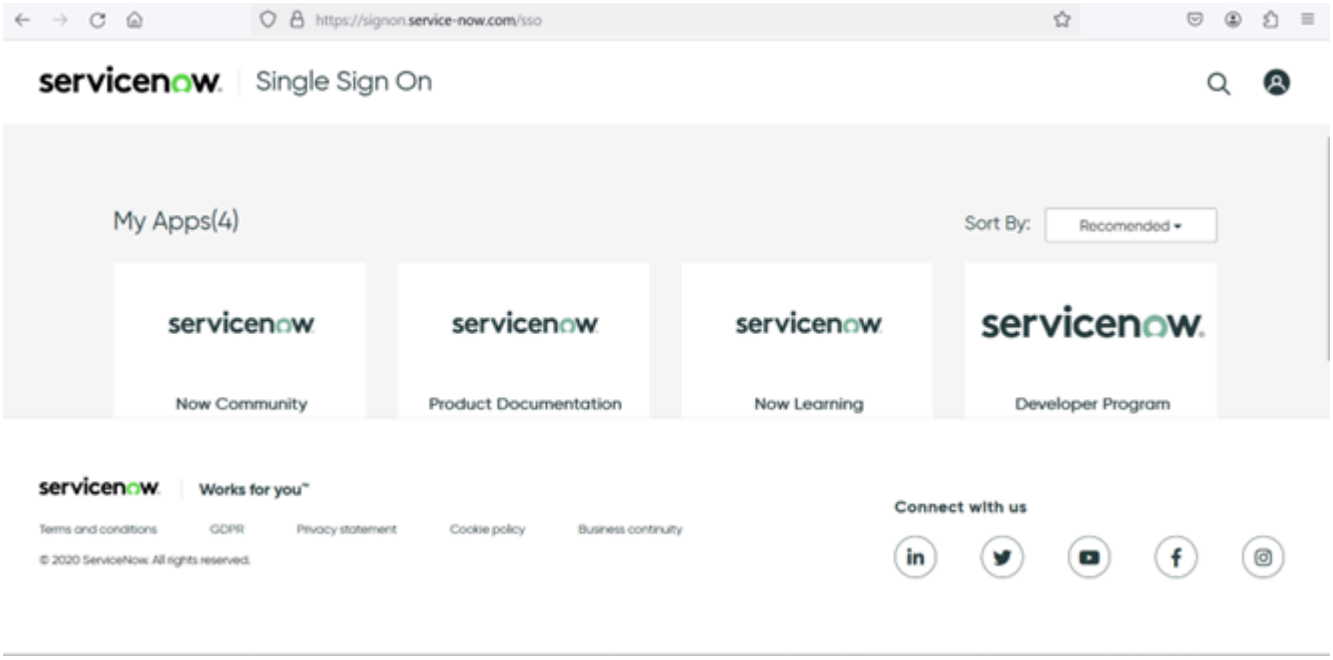
1. Accédez à la page d'inscription sur servicenow.com, saisissez vos détails, puis cliquez sur Continuer.
2. Lorsque vous recevez un code de vérification à votre adresse enregistrée, saisissez-le et choisissez Vérifier.
3. Configurez l'authentification multifactorielle ou ignorez cette opération.

Votre compte est créé et ServiceNow affiche votre profil.

Création d'une instance de ServiceNow développeur

Demandez une instance de développeur après vous être connecté à ServiceNow.

1. Sur la page de ServiceNow connexion, saisissez les informations d'identification de votre compte.
2. Choisissez le programme ServiceNow pour développeurs.



3. Choisissez Demander une instance en haut à droite.
4. Saisissez vos responsabilités professionnelles. Indiquez que vous acceptez les conditions d'utilisation, puis choisissez Terminer la configuration.
5. Une fois l'instance créée, notez l'URL et les informations d'identification de votre instance.

Récupération des informations d'identification BasicAuth

Pour récupérer les informations d'identification d'authentification de base d'un compte gratuit :

1. Sur la page de ServiceNow connexion, saisissez les informations d'identification de votre compte.
2. Sur la page d'accueil, choisissez la section Modifier le profil (coin supérieur droit), puis Gérer le mot de passe de l'instance.
3. Récupérez les informations d'identification de connexion telles que le nom d'utilisateur, le mot de passe et l'URL de l'instance.

> **Note**
>
> Si la MFA est activée pour le compte, ajoutez le jeton MFA à la fin du mot de passe de l'utilisateur dans l'authentification de base : <username>:<password><MFA Token>

Pour plus d'informations, consultez la section Création d'applications dans la ServiceNow documentation.

Création d'informations d'identification OAuth 2.0

Pour utiliser OAuth2 .0 dans le ServiceNow connecteur, vous devez créer un client entrant) afin de générer l'ID client et le secret du client :

1. Sur la page de ServiceNow connexion, saisissez les informations d'identification de votre compte.
2. Sur la page d'accueil, sélectionnez Démarrer la génération.
3. Sur la page App Engine Studio, recherchez Registre d'applications.
4. Choisissez Nouveau en haut à droite.
5. Choisissez l'option Créer un point de terminaison d' OAuth API pour les clients externes.
6. Apportez les modifications nécessaires à la OAuth configuration et choisissez Mettre à jour.

   Exemple d'URL de redirection : https://us-east-1.console.aws.amazon.com/gluestudio/ oauth
7. Sélectionnez l'application OAuth cliente nouvellement créée pour récupérer l'ID client et le secret du client.
8. Stockez l'ID client et le secret client en vue d'un traitement ultérieur.

Pour configurer OAuth dans un compte de développeur hors production :

1. Créez un profil d'authentification à l'aide de la rubrique Créer un profil d'authentification de la ServiceNow documentation.
2. Dans le profil d'authentification pour OAuth, sélectionnez Type as OAuth et sélectionnez le client entrant créé ci-dessus pour définir l'OAuth entité.
3. S'il existe plusieurs clients, vous devez créer plusieurs profils d'authentification pour définir l' OAuth entité requise dans le profil d'authentification.
4. Si elle n'est pas configurée, créez une stratégie d'accès à l'API REST pour donner accès à l'API TABLE. Consultez Create REST API access policy.

Configuration des ServiceNow connexions

Le type de subvention détermine le mode de AWS Glue communication avec ServiceNow lequel vous pouvez demander l'accès à vos données. Votre choix influe sur les exigences que vous devez respecter avant de créer la connexion. ServiceNow prend en charge uniquement le type de subvention AUTHORIZATION_CODE pour la version 2.0. OAuth

- Ce type de subvention est considéré comme « à trois étapes » OAuth car il repose sur la redirection des utilisateurs vers un serveur d'autorisation tiers pour authentifier l'utilisateur. Il est utilisé lors de la création de connexions via la AWS Glue console. La AWS Glue console redirige l'utilisateur vers l' ServiceNow endroit où il doit se connecter et autorise AWS Glue les autorisations demandées pour accéder à son ServiceNow instance.
- Les utilisateurs peuvent toujours choisir de créer leur propre application connectée ServiceNow et de fournir leur propre identifiant client et leur propre secret client lors de la création de connexions via la AWS Glue console. Dans ce scénario, ils seront toujours redirigés ServiceNow vers pour se connecter et autoriser AWS Glue l'accès à leurs ressources.
- Ce type d'autorisation donne lieu à un jeton d'actualisation et à un jeton d'accès. Le jeton d'accès est de courte durée et peut être actualisé automatiquement sans interaction de l'utilisateur à l'aide du jeton d'actualisation.
- Pour consulter ServiceNow la documentation publique sur la création d'une application connectée pour le OAuth flux de code d'autorisation, voir Configuration OAuth.

Pour configurer une ServiceNow connexion :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes :

a. Pour l'authentification de base, le secret doit contenir le secret du consommateur de l'application connectée avec `USERNAME` et `PASSWORD` comme clé.

b. Pour le type d'autorisation de code d'authentification, le secret doit contenir le secret du consommateur de l'application connectée avec `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` comme clé.

c. Remarque : Vous devez créer un secret par connexion dans AWS Glue.

2. Dans AWS Glue Glue Studio, créez une connexion sous Data Connections en suivant les étapes ci-dessous :

a. Lorsque vous sélectionnez un type de connexion, sélectionnez ServiceNow.

b. Indiquez l'INSTANCE_URL de l' ServiceNow instance à laquelle vous souhaitez vous connecter.

c. Sélectionnez le rôle AWS IAM qui AWS Glue peut assumer et dispose des autorisations nécessaires pour les actions suivantes :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

d. Sélectionnez le type d'authentification que vous souhaitez utiliser pour cette connexion dans AWS Glue.

i. Authentification de base : ce type d'authentification fonctionne bien pour les scénarios d'automatisation car il permet d'utiliser le nom d'utilisateur et le mot de passe dès le départ avec les autorisations d'un utilisateur particulier dans l' ServiceNow instance. AWS Glue est capable d'utiliser le nom d'utilisateur et le mot de passe pour s'authentifier. ServiceNow APIs

Saisissez les entrées suivantes uniquement en cas d'authentification de base : `Username` et `Password`.

ii. OAuth2: entrez les entrées suivantes uniquement en cas de OAuth2 :`ClientId`,`ClientSecret`,`Authorization URL`,`Authorization Token URL`.

e. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.

f. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.

3. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.

Lecture à partir d' ServiceNow entités

Prérequis

Un objet ServiceNow Tables que vous souhaitez lire. Vous aurez besoin du nom de l'objet, comme pa_bucket ou incident.

Exemple :

```
servicenow_read = glueContext.create_dynamic_frame.from_options(
    connection_type="servicenow",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "pa_buckets",
        "API_VERSION": "v2"
        "instanceUrl": "https://<instance-name>.service-now.com"
    }
)
```

ServiceNow détails de l'entité et du champ :

Pour les entités suivantes, ServiceNow fournit des points de terminaison permettant de récupérer les métadonnées de manière dynamique, afin que le support des opérateurs soit capturé au niveau du type de données pour chaque entité.

Entité	Type de données	Opérateurs pris en charge
Tables (entités dynamiques)	Entier	=, !=, <, <=, >, >=, BETWEEN

Entité	Type de données	Opérateurs pris en charge
	BigDecimal	=, !=, <, <=, >, >=, BETWEEN
	Float	=, !=, <, <=, >, >=, BETWEEN
	Long	=, !=, <, <=, >, >=, BETWEEN
	Date	=, !=, <, <=, >, >=, BETWEEN
	DateTime	=, !=, <, <=, >, >=, BETWEEN
	Booléen	=, !=
	String	=, !=, <, <=, >, >=, BETWEEN, LIKE
	Struct	N/A

> **Note**
>
> Le type de données Struct est converti en type de données String dans la réponse du connecteur.

> **Note**
>
> `DML_STATUS`est un attribut supplémentaire défini par l'utilisateur utilisé pour le suivi des CREATED/UPDATED enregistrements.

Requêtes de partitionnement

Partitionnement basé sur les champs :

Vous pouvez indiquer les options Spark supplémentaires `PARTITION_FIELD`, `LOWER_BOUND`, `UPPER_BOUND` et `NUM_PARTITIONS` si vous souhaitez utiliser la simultanéité dans Spark. Avec ces paramètres, la requête d'origine serait divisée en `NUM_PARTITIONS` nombres de sous-requêtes pouvant être exécutées simultanément par les tâches Spark.

Nom de l'entité	Champs de partitionnement	Type de données
Entité dynamique	sys_mod_count	Entier
	sys_created_on, sys_updated_on	DateTime

- `PARTITION_FIELD` : le nom du champ à utiliser pour partitionner la requête.
- `LOWER_BOUND` : une valeur limite inférieure inclusive du champ de partition choisi.

  Pour le champ Datetime, nous acceptons le format d'horodatage Spark utilisé dans SPark les requêtes SQL.

  Exemples de valeurs valides :

  ```
  "2024-01-30T06:47:51.000Z"
  ```
- `UPPER_BOUND` : une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS` : le nombre de partitions.

Le tableau suivant décrit les détails de prise en charge dans les champs de partitionnement d'entités :

Exemple :

```
servicenow_read = glueContext.create_dynamic_frame.from_options(
    connection_type="servicenow",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "pa_buckets",
        "API_VERSION": "v2",
        "instanceUrl": "https://<instance-name>.service-now.com"
        "PARTITION_FIELD": "sys_created_on"
        "LOWER_BOUND": "2024-01-30T06:47:51.000Z"
        "UPPER_BOUND": "2024-06-30T06:47:51.000Z"
        "NUM_PARTITIONS": "10"
    }
```

Partitionnement basé sur des enregistrements :

Vous pouvez indiquer l'option Spark supplémentaire NUM_PARTITIONS si vous souhaitez utiliser la simultanéité dans Spark. Avec ce paramètre, la requête d'origine est divisée en NUM_PARTITIONS nombres de sous-requêtes pouvant être exécutées simultanément par les tâches Spark.

Dans le partitionnement basé sur les enregistrements, le nombre total d'enregistrements présents est demandé par l' ServiceNow API, puis divisé par le nombre fourni. NUM_PARTITIONS Le nombre d'enregistrements qui en résulte est ensuite extrait simultanément par chaque sous-requête.

- NUM_PARTITIONS : le nombre de partitions.

Exemple :

```
servicenow_read = glueContext.create_dynamic_frame.from_options(
    connection_type="servicenow",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "pa_buckets",
        "API_VERSION": "v2",
        "instanceUrl": "https://<instance-name>.service-now.com"
        "NUM_PARTITIONS": "2"
    }
```

ServiceNow options de connexion

Les options de connexion suivantes sont disponibles pour ServiceNow :

- ENTITY_NAME(Chaîne) - (Obligatoire) Utilisé pour la lecture. Le nom de votre objet dans ServiceNow.
- API_VERSION(Chaîne) - (Obligatoire) Utilisé pour la lecture. ServiceNow Version de l'API Rest que vous souhaitez utiliser. Par exemple : v1, v2, v3, v4.
- SELECTED_FIELDS(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- FILTER_PREDICATE(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- QUERY(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.
- PARTITION_FIELD(Chaîne) - Utilisé pour la lecture. Champ à utiliser pour partitionner la requête.

- LOWER_BOUND(Chaîne) - Utilisé pour la lecture. Une valeur limite inférieure inclusive du champ de partition choisi. Par exemple : 2024-01-30T06:47:51.000Z.
- UPPER_BOUND(Chaîne) - Utilisé pour la lecture. Une valeur limite supérieure exclusive du champ de partition choisi. Par exemple : 2024-06-30T06:47:51.000Z.
- NUM_PARTITIONS(Entier) - Par défaut : 1. Utilisé pour la lecture. Nombre de partitions pour la lecture. Par exemple, 10.
- INSTANCE_URL(String) - (Obligatoire) Une URL d' ServiceNow instance valide au format https :// <instance-name>.service-now.com.
- PAGE_SIZE(Entier) - Définit la taille de page pour interroger les enregistrements. La taille de page par défaut est de 1 000. Lorsqu'une taille de page est spécifiée, ServiceNow renvoie uniquement le nombre défini d'enregistrements par appel d'API, plutôt que l'ensemble de données complet. Le connecteur fournira toujours le nombre total d'enregistrements et gérera la pagination en utilisant la taille de page que vous avez indiquée. Si vous avez besoin d'une taille de page plus importante, vous pouvez choisir n'importe quelle valeur jusqu'à 10 000, qui est le maximum autorisé. Toute taille de page spécifiée supérieure à 10 000 sera ignorée. Le système utilisera plutôt la taille de page maximale autorisée. Vous pouvez spécifier la taille de page dans l'interface utilisateur AWS Glue Studio en ajoutant une option de connexion PAGE_SIZE avec la valeur souhaitée. Par exemple, 5 000.

Limitations et remarques relatives au ServiceNow connecteur

Les limitations ou remarques suivantes concernent le ServiceNow connecteur :

- Selon la documentation SaaS, sys_created_on, sys_updated_on et sys_mod_count sont des champs générés par le système. Le connecteur s'appuie sur le SaaS APIs pour fournir ces champs dans le corps de réponse.
  - Si SaaS ne génère ces champs pour aucune entité, le partitionnement basé sur des filtres ne peut pas être pris en charge.
- Si le SaaS APIs ne s'affiche pas sys_created_on et que sys_updated_on les champs de la réponse DML_STATUS ne peuvent pas être calculés.
- Améliorer les performances et l'efficacité de la lecture
  - Le ServiceNow connecteur trie désormais automatiquement les enregistrements par ordre croissant en fonction du sys_id champ (doit être présent dans les métadonnées) lorsqu'aucune clause ORDER BY n'est spécifiée par l'utilisateur. Dans ce cas, les enregistrements seront paginés selon la nouvelle pagination optimisée basée sur le jeu de clés.

- Si la clause ORDER BY est spécifiée, la nouvelle optimisation ne sera pas utilisée et les enregistrements seront récupérés à l'aide de la méthode existante (pagination Order By définie par l'utilisateur et basée sur Offset-Limit).

Connexion à Slack dans AWS Glue Studio

Slack est une application de communication d'entreprise qui permet aux utilisateurs d'envoyer des messages et des pièces jointes via différents canaux publics et privés. Si vous utilisez Slack, vous pouvez vous connecter AWS Glue à votre compte Slack. Vous pouvez ensuite utiliser Slack comme source de données dans vos tâches ETL. Exécutez ces tâches pour transférer des données entre Slack et AWS des services ou d'autres applications compatibles.

Rubriques

- AWS Glue support pour Slack
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration de Slack
- Configuration de connexions Slack
- Lecture à partir d'entités Slack
- Options de connexion Slack
- Limitations
- Création d'un compte Slack et configuration de l'application client

AWS Glue support pour Slack

AWS Glue prend en charge Slack comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser les tâches AWS Glue ETL pour interroger les données de Slack.

Prise en charge en tant que cible ?

Non.

Versions de l'API Slack prises en charge

API Slack v2.

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les autorisations IAM requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Vous pouvez également utiliser les politiques IAM gérées suivantes pour autoriser l'accès :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM et Amazon EC2. Amazon CloudWatch Logs Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration de Slack

Avant de pouvoir transférer des données AWS Glue vers ou depuis Slack, vous devez satisfaire aux exigences suivantes :

Configuration requise

- Vous devez posséder un compte Slack. Pour de plus amples informations, veuillez consulter Création d'un compte Slack et configuration de l'application client.

Si vous remplissez ces conditions, vous êtes prêt à vous connecter AWS Glue à votre compte Slack.

Configuration de connexions Slack

Slack prend en charge le type de `AUTHORIZATION_CODE` subvention pour OAuth 2.

Ce type de subvention est considéré comme « à trois étapes » OAuth car il repose sur la redirection des utilisateurs vers le serveur d'autorisation tiers pour authentifier l'utilisateur. Il est utilisé lors de la création de connexions via la AWS Glue console. La AWS Glue console redirige l'utilisateur vers Slack où il doit se connecter et autoriser AWS Glue les autorisations demandées pour accéder à son instance Slack.

Les utilisateurs peuvent toujours choisir de créer leur propre application connectée dans Slack et de fournir leur propre identifiant client et leur propre secret client lors de la création de connexions via la AWS Glue console. Dans ce scénario, ils seront toujours redirigés vers Slack pour se connecter et autoriser l'accès AWS Glue à leurs ressources.

Ce type d'autorisation donne lieu à un jeton d'actualisation et à un jeton d'accès. Le jeton d'accès expire au bout d'une heure après sa création. Un nouveau jeton d'accès peut être récupéré à l'aide du jeton d'actualisation.

Pour plus d'informations sur la création d'une application connectée pour le OAuth flux de code d'autorisation, consultez l'API Slack.

Pour configurer une connexion Slack :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes. Vous devez créer un secret pour la connexion dans AWS Glue.

a. Pour l'application connectée gérée par le client : le secret doit contenir le secret du consommateur de l'application connectée avec `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` comme clé.

2. Dans AWS Glue Glue Studio, créez une connexion sous Data Connections en suivant les étapes ci-dessous :

   a. Lorsque vous sélectionnez un type de connexion, sélectionnez Slack.

   b. Indiquez l'environnement Slack.

   c. Sélectionnez le rôle IAM qui AWS Glue peut assumer et dispose des autorisations nécessaires pour les actions suivantes :

      JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

   d. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.

   e. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.

3. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.

Lecture à partir d'entités Slack

Conditions préalables

- Un objet Slack à partir duquel vous souhaitez lire.

Entités prises en charge

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
conversations	Oui	Oui	Non	Oui	Oui

Exemple

```
slack_read = glueContext.create_dynamic_frame.from_options(
    connection_type="slack",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "conversations/C058W38R5J8"
    }
)
```

Détails des entités et des champs Slack

Entité	Champ	Type de données	Opérateurs pris en charge
conversations	attachments	List	NA
conversations	bot_id	String	NA
conversations	Blocs de	List	NA
conversations	client_msg_id	String	NA
conversations	is_starred	Booléen	NA
conversations	last_read	String	NA
conversations	latest_reply	String	NA

Entité	Champ	Type de données	Opérateurs pris en charge
conversations	reactions	List	NA
conversations	replies	List	NA
conversations	reply_count	Entier	NA
conversations	reply_users	List	NA
conversations	reply_users_count	Entier	NA
conversations	subscribed	Booléen	NA
conversations	subtype	String	NA
conversations	text	String	NA
conversations	team	String	NA
conversations	thread_ts	String	NA
conversations	ts	String	EQUAL_TO, BETWEEN, LESS_THAN , LESS_THAN _OR_EQUAL_TO, GREATER_T HAN, GREATER_T HAN_OR_EQUAL_TO
conversations	type	String	NA
conversations	user	String	NA
conversations	inviter	String	NA
conversations	racine	Struct	NA
conversations	is_locked	Booléen	NA

Entité	Champ	Type de données	Opérateurs pris en charge
conversations	files	List	NA
conversations	salle	Struct	NA
conversations	charger	Booléen	NA
conversations	display_as_bot	Booléen	NA
conversations	channel	String	NA
conversations	no_notifications	Booléen	NA
conversations	permalink	String	NA
conversations	pinned_to	List	NA
conversations	pinned_info	Struct	NA
conversations	edited	Struct	NA
conversations	app_id	String	NA
conversations	bot_profile	Struct	NA
conversations	metadata	Struct	NA

Requêtes de partitionnement

Les options Spark supplémentaires `PARTITION_FIELD`, `LOWER_BOUND`, `UPPER_BOUND` et `NUM_PARTITIONS` peuvent être indiquées si vous souhaitez utiliser la simultanéité dans Spark. Avec ces paramètres, la requête d'origine serait divisée en `NUM_PARTITIONS` nombres de sous-requêtes pouvant être exécutées simultanément par les tâches Spark.

- `PARTITION_FIELD` : le nom du champ à utiliser pour partitionner la requête.
- `LOWER_BOUND` : une valeur limite inférieure inclusive du champ de partition choisi.

Pour le champ Date, nous acceptons le format de date Spark utilisé dans les requêtes SQL Spark. Exemple de valeur valide : `"2024-07-01T00:00:00.000Z"`.

- `UPPER_BOUND` : une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS` : nombre de partitions.

Les détails relatifs à la prise en charge des champs de partitionnement par entité sont présentés dans le tableau ci-dessous.

Nom de l'entité	Champ de partitionnement	Type de données
conversations	ts	String

Exemple

```
slack_read = glueContext.create_dynamic_frame.from_options(
    connection_type="slack",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "conversations/C058W38R5J8",
        "PARTITION_FIELD": "ts"
        "LOWER_BOUND": "2022-12-01T00:00:00.000Z"
        "UPPER_BOUND": "2024-09-23T15:00:00.000Z"
        "NUM_PARTITIONS": "2"
    }
)
```

Options de connexion Slack

Les options de connexion pour Slack sont les suivantes :

- `ENTITY_NAME`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Nom de l'entité prise en charge. Exemple: `conversations/C058W38R5J8`.
- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Champs que vous souhaitez sélectionner pour l'objet.
- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.

- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.
- `NUM_PARTITIONS`(Entier) - Par défaut : 1. Utilisé pour la lecture. Nombre de partitions pour la lecture.

Limitations

Les restrictions du connecteur Slack sont les suivantes :

- Le partitionnement basé sur les enregistrements n'est pas pris en charge, car le connecteur ne fournit aucun moyen de récupérer le nombre total d'enregistrements (messages) disponibles dans une conversation donnée.

Création d'un compte Slack et configuration de l'application client

Création d'un compte Slack

1. Ouvrez la page d'accueil de Slack pour créer un compte.
2. Choisissez S'INSCRIRE AVEC UNE ADRESSE E-MAIL. Saisissez votre adresse e-mail, puis choisissez Continuer.
3. Saisissez le code à six caractères envoyé à votre adresse e-mail. Vous serez alors redirigé afin de créer un espace de travail ou rejoindre un espace de travail existant.
4. Choisissez Créer un espace de travail pour créer un espace de travail. Vous serez alors redirigé pour répondre à quelques questions dans le cadre du processus de configuration.
   - Nom de l'entreprise
   - Votre nom
   - Pour ajouter des collègues par e-mail
   - Sur quoi travaille votre équipe ? (Ce sera le nom du canal)
5. Remplissez les champs de saisie correspondant à ces questions et poursuivez. Votre compte est maintenant prêt à être utilisé.

Création d'une application pour développeurs Slack

1. Connectez-vous à votre compte Slack et accédez à votre espace de travail Slack.

2. Dans le menu de l'espace de travail, sélectionnez Outils et paramètres, puis sélectionnez Gérer les applications.
3. Dans le menu de la liste des applications Slack, sélectionnez Créer.
4. Sur la page Vos applications, sélectionnez Créer une application.
5. Sur la page Créer une application, sélectionnez À partir de zéro.
6. Dans la boîte de dialogue Nommer l'application et choisir l'espace de travail qui s'ouvre, ajoutez un nom d'application et choisissez un espace de travail pour y déployer votre application. Sélectionnez ensuite Créer une application.
7. Notez votre ID et votre secret client affichés dans les informations d'identification de l'application.
8. Dans la barre latérale OAuth & Permissions, accédez à Étendue et choisissez Ajouter une OAuth portée. Vous pouvez ajouter la redirection URLs vers votre application pour la configuration afin de générer automatiquement le bouton « Ajouter à Slack » ou de distribuer votre application. Faites défiler la page jusqu'à la URLs section Redirection, choisissez Ajouter une nouvelle URL de redirection et enregistrez.
9. Accédez ensuite à la section OAuth Tokens pour votre espace de travail, puis choisissez Installer dans l'espace de travail.
10. Dans la boîte de dialogue qui s'ouvre pour vous informer que l'application que vous avez créée demande l'autorisation d'accéder à l'espace de travail Slack auquel vous souhaitez la connecter, sélectionnez Autoriser.
11. Une fois l'opération terminée, la console affichera un écran OAuth Tokens pour votre espace de travail.
12. À partir de l'écran OAuth Jetons pour votre espace de travail, copiez et enregistrez le OAuth jeton que vous utiliserez pour vous connecter à AWS Glue
13. Ensuite, vous allez récupérer votre identifiant d'équipe Slack. Dans le menu de l'espace de travail Slack, sélectionnez Outils et paramètres, puis sélectionnez Gérer les applications. Vous trouverez votre identifiant d'équipe dans l'URL de la page qui s'ouvre.
14. Pour distribuer publiquement votre application, vous pouvez l'activer en accédant au bouton Gérer la distribution dans la barre latérale. Faites défiler la page jusqu'à la section Partager votre application avec d'autres espaces de travail et choisissez Supprimer les informations codées en dur. Donnez votre consentement et choisissez Distribution publique active.
15. Votre application est désormais distribuée au public. Pour accéder à l'entité APIs, l'application doit être ajoutée à chaque canal de l'espace de travail à partir duquel l'utilisateur souhaite accéder.

16. Connectez-vous à votre compte Slack et ouvrez l'espace de travail dont vous souhaitez accéder au canal.
17. Dans l'espace de travail, ouvrez le canal auquel l'application souhaite accéder et choisissez le titre du canal. Sélectionnez l'onglet Intégrations dans la fenêtre contextuelle et ajoutez l'application. De cette façon, l'application est intégrée au canal pour avoir accès à son API.

    L'ID client OAuth 2.0 doit comporter une ou plusieurs redirections autorisées URLs. URLs Les redirections ont le format suivant :

    > **Note**
    >
    > Les redirections Appflow URLs sont sujettes à modification. Les redirections de publication URLs pour AWS Glue la plateforme sont disponibles. L'ID client et le secret client proviennent des paramètres de votre identifiant client OAuth 2.0.

L'URL de redirection peut être l'une des suivantes	
URL de redirection pour l'environnement Gamma	
https://us-west-2.console.aws.amazon.com/appflow/oauth	https://us-east-1.awsc-integ.aws.amazon.com/appflow/oauth
https://us-east-2.console.aws.amazon.com/appflow/oauth	
https://us-west-1.console.aws.amazon.com/appflow/oauth	
https://ap-south-1.console.aws.amazon.com/appflow/oauth	
https://ap-southeast-1.console.aws.amazon.com/appflow/oauth	

L'URL de redirection peut être l'une des suivantes	
URL de redirection pour l'environnement Gamma	
https://ap-southeast-2.console.aws.amazon.com/appflow/oauth	
https://ap-northeast-1.console.aws.amazon.com/appflow/oauth	
https://ap-northeast-2.console.aws.amazon.com/appflow/oauth	
https://ca-central-1.console.aws.amazon.com/appflow/oauth	
https://eu-central-1.console.aws.amazon.com/appflow/oauth	
https://eu.west-1.console.aws.amazon.com/appflow/oauth	
https://eu-west-2.console.aws.amazon.com/appflow/oauth	
https://eu-west-3.console.aws.amazon.com/appflow/oauth	
https://sa-east-1.console.aws.amazon.com/appflow/oauth	
https://us-west-2.awsc-integ.aws.amazon.com/appflow/oauth	
https://af-south-1.console.aws.amazon.com/appflow/oauth	

Connexion à Smartsheet

Smartsheet est un produit SaaS de gestion du travail et de collaboration. Smartsheet permet essentiellement aux utilisateurs d'utiliser des objets de type feuille de calcul pour créer, stocker et utiliser des données métier.

Rubriques

- AWS Glue support pour Smartsheet
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration de Smartsheet
- Configuration de connexions Smartsheet
- Lecture à partir d'entités Smartsheet
- Options de connexion Smartsheet
- Création d'un compte Smartsheet
- Limitations

AWS Glue support pour Smartsheet

AWS Glue prend en charge Smartsheet comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser des tâches AWS Glue ETL pour interroger les données de Smartsheet.

Prise en charge en tant que cible ?

Non.

Versions de l'API Smartsheet prises en charge

v2.0

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les AWS autorisations requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Si vous ne voulez pas utiliser la méthode ci-dessus, vous pouvez également utiliser les politiques IAM gérées suivantes :

- AWSGlueServiceRole — Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess — Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration de Smartsheet

Avant de pouvoir AWS Glue effectuer un transfert depuis Smartsheet, vous devez satisfaire aux exigences suivantes :

Configuration requise

- Vous disposez d'un compte Smartsheet avec adresse e-mail et mot de passe. Pour plus d'informations sur la création d'un compte, consultez Creating a Smartsheet account.
- Votre compte Smartsheet dispose d'un accès à l'API avec une licence valide.
- Votre compte Smartsheet propose un forfait Tarif Pro pour l'entité `Sheets` et un forfait Tarif entreprise avec le module complémentaire Rapports d'événements pour les entités `Events`.

Si vous répondez à ces exigences, vous êtes prêt à vous connecter AWS Glue à votre compte Smartsheet. Pour les connexions classiques, vous n'avez rien d'autre à faire dans Smartsheet.

Configuration de connexions Smartsheet

Smartsheet prend en charge le type de `AUTHORIZATION_CODE` subvention pour OAuth2.

Ce type d'autorisation est considéré comme `OAuth` tridimensionnel, car il repose sur la redirection des utilisateurs vers le serveur d'autorisation tiers pour authentifier l'utilisateur. Les utilisateurs peuvent choisir de créer leur propre application connectée dans Smartsheet et de fournir leur propre identifiant client et leur propre secret client lors de la création de connexions via la AWS Glue console. Dans ce scénario, ils seront toujours redirigés vers Smartsheet pour se connecter et autoriser l'accès AWS Glue à leurs ressources.

Ce type d'autorisation donne lieu à un jeton d'actualisation et à un jeton d'accès. Le jeton d'accès est de courte durée et peut être actualisé automatiquement sans interaction de l'utilisateur à l'aide du jeton d'actualisation.

Pour consulter la documentation publique de Smartsheet sur la création d'une application connectée pour le flux AUTHORIZATION_CODE OAuth , consultez Smartsheet. APIs

Pour configurer une connexion Smartsheet :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes :

   Pour l'application connectée gérée par le client : le secret doit contenir le secret du consommateur de l'application connectée avec `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` comme clé.

> **Note**
>
> Il est indispensable de créer un secret par connexion dans AWS Glue.

2. Dans le AWS Glue Studio, créez une connexion sous Connexions de données en suivant les étapes ci-dessous :
   a. Lorsque vous sélectionnez un type de connexion, sélectionnez Smartsheet.
   b. Indiquez l'`instanceUrl` de Smartsheet à laquelle vous souhaitez vous connecter.
   c. Sélectionnez le rôle IAM pour lequel AWS Glue vous pouvez assumer et qui dispose des autorisations nécessaires pour effectuer les actions suivantes :

      JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

   d. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.
   e. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.
3. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.

Lecture à partir d'entités Smartsheet

Conditions préalables

Un objet `Smartsheet` à partir duquel vous souhaitez lire. Reportez-vous au tableau des entités prises en charge ci-dessous pour vérifier les entités disponibles.

Entités prises en charge

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
List Sheet	Oui	Oui	Non	Oui	Non
Row Metadata	Oui	Oui	Non	Oui	Non
Sheet Metadata	Non	Non	Non	Oui	Non
Sheet Data	Oui	Oui	Oui	Oui	Non
Événement	Oui	Oui	Non	Oui	Non

Exemple

```
Smartsheet_read = glueContext.create_dynamic_frame.from_options(
    connection_type="smartsheet",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "list-sheets",
        "API_VERSION": "2.0",
        "INSTANCE_URL": "https://api.smartsheet.com"
    })
```

Détails des entités et des champs Smartsheet

Entité	Champ	Type de données	Opérateurs pris en charge
List Sheets	id	Long	NA

Entité	Champ	Type de données	Opérateurs pris en charge
	accessLevel	String	NA
	createdAt	DateTime	NA
	modifiedAt	DateTime	NA
	name	String	NA
	permalink	String	NA
	modifiedSince	DateTime	>=
	version	Entier	NA
	source	Struct	NA
Row Metadata	id	Long	NA
	sheetId	Long	NA
	accessLevel	String	NA
	attachments	List	NA
	columns	List	NA
	conditionalFormat	String	NA
	createdAt	DateTime	NA
	createdBy	Struct	NA
	discussions	List	NA
	proofs	Struct	NA
	expanded	Booléen	NA
	filteredOut	Booléen	NA

Entité	Champ	Type de données	Opérateurs pris en charge
	format	String	NA
	inCriticalPath	Booléen	NA
	locked	Booléen	NA
	lockedForUser	Booléen	NA
	modifiedAt	DateTime	NA
	modifiedBy	Struct	NA
	permalink	String	NA
	rowNumber	Entier	NA
	version	Entier	NA
	totalRowCount	Entier	NA
	rowsModifiedSince	DateTime	>
	filterId	Long	“="
	siblingId	Long	NA
	parentId	Long	NA
Sheet metadata	id	Long	NA
	fromId	Long	NA
	ownerId	Long	NA
	accessLevel	String	NA
	attachments	List	NA
	columns	List	NA

Entité	Champ	Type de données	Opérateurs pris en charge
	createdAt	DateTime	NA
	crossSheetReferences	List	NA
	dependenciesEnabled	Booléen	NA
	discussions	List	NA
	effectiveAttachmentOptions	List	NA
	favorite	Booléen	NA
	ganttEnabled	Booléen	NA
	hasSummaryFields	Booléen	NA
	modifiedAt	DateTime	NA
	name	String	NA
	owner	String	NA
	permalink	String	NA
	projectSettings	Struct	NA
	readOnly	Booléen	NA
	resourceManagementEnabled	Booléen	NA
	showParentRowsForFilters	Booléen	NA
	source	Struct	NA

Entité	Champ	Type de données	Opérateurs pris en charge
	récapitulatif	Struct	NA
	totalRowCount	Entier	NA
	userPermissions	Struct	NA
	userSettings	Struct	NA
	version	Entier	NA
	espace de travail	Struct	NA
	filtres	List	NA
	ganttConfig	Struct	NA
	resourceManagementType	String	NA
	cellImageUploadActivé	Booléen	NA
	isMultiPicklistActivé	Booléen	NA
Événements	eventId	String	NA
	objectType	String	NA
	action	String	NA
	objectId	Long	NA
	eventTimestamp	DateTime	NA
	userId	Long	NA
	requestUserId	Long	NA
	accessTokenName	String	NA

Entité	Champ	Type de données	Opérateurs pris en charge
	source	String	NA
	additionalDetails	Struct	NA
	since	DateTime	>=

Entités avec métadonnées dynamiques :

Pour l'entité suivante, Smartsheet fournit un point de terminaison permettant de récupérer les métadonnées de manière dynamique, afin que la prise en charge des opérateurs soit capturée au niveau du type de données.

Entité	Type de données	Opérateurs pris en charge
Sheet Data	String	NA
Sheet Data	Long	"="
Sheet Data	Entier	NA
Sheet Data	DateTime	>

Exemple

```
Smartsheet_read = glueContext.create_dynamic_frame.from_options(
    connection_type="smartsheet",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "list-sheets",
        "API_VERSION": "2.0",
        "INSTANCE_URL": "https://api.smartsheet.com"
    }
```

Options de connexion Smartsheet

Les options de connexion pour Smartsheet sont les suivantes :

- `ENTITY_NAME`(Chaîne) - (Obligatoire) Utilisé pour la lecture/l'écriture. Le nom de votre objet dans Smartsheet.
- `API_VERSION`(Chaîne) - (Obligatoire) Utilisé pour la lecture/l'écriture. Version d'API Rest Smartsheet que vous souhaitez utiliser. Par exemple : v2.0.
- `INSTANCE_URL`(Chaîne) – (Obligatoire) Utilisé pour la lecture. URL d'instance Smartsheet.
- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.

Création d'un compte Smartsheet

1. Ouvrez un compte Smartsheet en accédant à la page d'inscription à Smartsheet.
2. Choisissez Créer un compte pour créer un compte ou connectez-vous à l'aide de votre compte Google, Microsoft ou Apple enregistré.
3.
4.
5. Ouvrez l'e-mail de confirmation envoyé par Smartsheet et cliquez sur le lien de confirmation pour vérifier votre compte.

   Vous serez abonné au forfait d'essai par défaut.
6. Dans le coin inférieur gauche, cliquez sur l'icône du compte, puis sur Ajouter des licences/Mettre à niveau pour améliorer votre plan tarifaire.

   > **Note**
   >
   > Cette opération est nécessaire pour accéder à Event Reporting, qui est un module complémentaire du forfait Entreprise.
7. Sous le forfait Entreprise, choisissez Contactez-nous pour demander une mise à niveau de compte auprès de l'équipe d'assistance.
8. Dans le formulaire de demande d'assistance, fournissez les informations requises et vos exigences pour mettre le forfait à niveau.

Cette opération permet de finaliser la mise à niveau vers le forfait Entreprise.

Création d'informations d'identification **`OAuth2.0`**

1. Après avoir mis à niveau le plan tarifaire de votre compte pour accéder aux outils pour développeurs, accédez à Développeurs Smartsheet.

   Vous recevrez un e-mail d'activation.
2. Ouvrez l'e-mail d'activation envoyé par Smartsheet et cliquez sur le lien d'activation pour activer les outils de développement sur votre compte.

   L'outil Développeurs vous permet de créer l'application.
3. Ouvrez la page d'accueil de votre compte Smartsheet et sélectionnez Compte pour vérifier l'accès.
4. Choisissez Outils pour développeurs dans la liste des services, puis saisissez les détails du profil du développeur.
5. Choisissez Créer une nouvelle application.
6. Saisissez les informations suivantes dans le formulaire d'inscription à l'application :

   - Nom : nom de l'application.
   - Description : description de l'application.
   - URL : URL qui vous permet de lancer votre application ou URL de la page de destination.
   - Contact/support : coordonnées de l'équipe d'assistance.
   - URL de redirection : URL (également appelée URL de rappel) de votre application qui recevra les informations d'identification OAuth 2.0.
7. Choisissez Enregistrer.

   Smartsheet attribue un identifiant client et un secret client à votre application. Enregistrez ces valeurs pour les étapes suivantes. Vous pourrez également les consulter à nouveau ultérieurement dans la section Outils pour développeurs.

Limitations

Smartsheet ne prend pas en charge le partitionnement basé sur les champs ou les enregistrements.

Connexion à Snapchat Ads dans AWS Glue Studio

Snapchat est une application et un service de messagerie instantanée multimédia développés par Snap Inc., à l'origine Snapchat Inc. L'une des principales caractéristiques de Snapchat est que les photos et les messages ne sont généralement disponibles que pendant une courte période avant de devenir inaccessibles à leurs destinataires. Le marketing Snapchat désigne des publications pour lesquelles les entreprises peuvent payer pour qu'elles soient diffusées aux utilisateurs de Snapchat.

Rubriques

- AWS Glue support pour Snapchat Ads
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration de Snapchat Ads
- Configuration des connexions Snapchat Ads
- Lecture depuis des entités Snapchat Ads
- Options de connexion Snapchat Ads
- Création d'un compte Snapchat Ad et configuration de l'application client
- Création d'une application dans votre compte Snapchat Ads

AWS Glue support pour Snapchat Ads

AWS Glue prend en charge les publicités Snapchat comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser les tâches AWS Glue ETL pour interroger les données de Snapchat Ads.

Prise en charge en tant que cible ?

Non.

Versions d'API Snapchat Ads prises en charge

v1.

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les AWS autorisations requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Vous pouvez également utiliser les politiques IAM gérées suivantes pour autoriser l'accès :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM et Amazon EC2. Amazon CloudWatch Logs Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration de Snapchat Ads

Avant de pouvoir AWS Glue effectuer un transfert depuis Snapchat Ads, vous devez remplir les conditions suivantes :

Configuration requise

- Vous disposez d'un compte Snapchat Ads. Pour plus d'informations sur la création d'un compte, consultez Création d'un compte Snapchat Ad et configuration de l'application client.
- Vous avez créé une OAuth2 application dans votre compte Snapchat Ads. Cette intégration fournit les informations d'identification qu'utilise AWS Glue pour accéder à vos données en toute sécurité au moment de passer des appels authentifiés vers votre compte. Pour de plus amples informations, veuillez consulter Création d'une application dans votre compte Snapchat Ads.

Si vous remplissez ces conditions, vous êtes prêt à vous connecter AWS Glue à votre compte Snapchat Ads. Dans Snapchat Ads, une application connectée est un framework qui autorise des applications externes, par exemple AWS Glue, à accéder à vos données Snapchat Ads.

Configuration des connexions Snapchat Ads

Snapchat Ads ne prend en charge que le type d'autorisation `AUTHORIZATION_CODE`.

Ce type de subvention est considéré comme « à trois étapes » OAuth car il repose sur la redirection des utilisateurs vers le serveur d'autorisation tiers pour authentifier l'utilisateur. Il est utilisé lors de la création de connexions via la AWS Glue console. L'utilisateur qui crée une connexion peut par défaut s'appuyer sur une application connectée qu'il AWS Glue possède (application cliente AWS Glue gérée) sur laquelle il n'a pas besoin de fournir d'informations OAuth connexes, à l'exception de l'URL de son instance Snapchat Ads. La AWS Glue console redirigera l'utilisateur vers Snapchat Ads où il devra se connecter et autoriser AWS Glue les autorisations demandées pour accéder à son instance Snapchat Ads.

Les utilisateurs peuvent toujours choisir de créer leur propre application connectée dans Snapchat Ads et de fournir leur propre identifiant client et leur secret client lors de la création de connexions via la AWS Glue console. Dans ce scénario, ils seront toujours redirigés vers Snapchat Ads pour se connecter et autoriser AWS Glue l'accès à leurs ressources.

Ce type d'autorisation donne lieu à un jeton d'actualisation et à un jeton d'accès. Le jeton d'accès expire au bout d'une heure après sa création. Un nouveau jeton d'accès peut être récupéré à l'aide du jeton d'actualisation.

Pour plus d'informations sur la création d'une application connectée pour le OAuth flux de code d'autorisation, consultez la section API des annonces.

Pour configurer une connexion Snapchat Ads :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes. Il est nécessaire de créer un secret pour chaque connexion dans AWS Glue.
   a. Pour l'application connectée gérée par le client : le secret doit contenir le secret du consommateur de l'application connectée avec `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` comme clé.
2. Dans AWS Glue Glue Studio, créez une connexion sous Data Connections en suivant les étapes ci-dessous :
   a. Lorsque vous sélectionnez un type de connexion, sélectionnez Snapchat Ads.
   b. Indiquez l'environnement Snapchat Ads.
   c. Sélectionnez le rôle IAM qui AWS Glue peut assumer les actions suivantes et qui dispose des autorisations nécessaires pour les actions suivantes :

      JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

   d. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.
   e. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.
3. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.

Lecture depuis des entités Snapchat Ads

Conditions préalables

- Un objet Snapchat Ads à partir duquel vous souhaitez lire. Reportez-vous au tableau des entités prises en charge ci-dessous pour vérifier les entités disponibles.

Entités prises en charge

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Organisation	Non	Non	Non	Oui	Non
Ad Account	Non	Non	Non	Oui	Non
Creative	Non	Non	Non	Oui	Non
Multimédia	Non	Non	Non	Oui	Non
Campaign	Oui	Non	Non	Oui	Non
Ad Under Ad Account	Oui	Non	Non	Oui	Non
Ad Under Campaign	Non	Non	Non	Oui	Non
Ad Squad	Oui	Non	Non	Oui	Non
Segment	Non	Non	Non	Oui	Non

Exemple

```
snapchatads_read = glueContext.create_dynamic_frame.from_options(
    connection_type="snapchatAds",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "organization",
```

```
        "API_VERSION": "v1"
    }
)
```

Détails des entités et des champs Snapchat Ads

Snapchat Ads charge dynamiquement les champs disponibles sous l'entité sélectionnée. Les opérateurs de filtrage suivants sont pris en charge en fonction du type de données du champ.

Type de données de champs	Opérateurs de filtre pris en charge
Booléen	=

Requêtes de partitionnement

- Partitionnement basé sur les champs : non pris en charge.
- Partitionnement basé sur les enregistrements : non pris en charge.

Options de connexion Snapchat Ads

Les options de connexion pour Snapchat Ads sont les suivantes :

- `ENTITY_NAME`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Le nom de l'entité Snapchat Ads. Exemple: `campaign` .
- `API_VERSION`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Version d'API Snapchat Ads que vous souhaitez utiliser. La valeur sera v1, car Snapchat Ads ne prend actuellement en charge que la version 1.
- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Liste de colonnes séparées par des virgules que vous souhaitez sélectionner pour l'entité sélectionnée.
- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.

Création d'un compte Snapchat Ad et configuration de l'application client

Rubriques

- Inscription à Snapchat Ads
- Étapes pour créer un compte Snapchat Ad

Inscription à Snapchat Ads

Pour s'inscrire à Snapchat Ads :

1. Accédez à Snapchat Ads Manager. Choisissez S'inscrire à côté de Nouveau sur Snapchat ?.
2. Sur l'écran Créer un compte, suivez les instructions pour saisir le nom de votre entreprise, votre e-mail, votre mot de passe, etc. Choisissez Suivant.
3. Sur l'écran Créer votre profil, entrez des valeurs pour le nom d'utilisateur, le site Web (facultatif), puis choisissez Créer un compte. Cela vous donnera la possibilité d'ajouter une photo de profil et une biographie sur l'écran Modifier votre profil. Choisissez Confirmer.
4. Sur l'écran Informations de l'entreprise, remplissez les champs obligatoires tels que le pays, la devise, le numéro de téléphone, le code GSTIN, etc., et terminez le processus de création du compte en choisissant Suivant.

Étapes pour créer un compte Snapchat Ad

Pour créer un compte Snapchat Ad :

1. Connectez-vous à Ads Manager. Cliquez ensuite sur le menu de navigation dans le coin supérieur et sélectionnez Comptes publicitaires.
2. Choisissez + Nouveau compte publicitaire. Saisissez les informations de votre annonceur :

    - Indiquez si vous êtes une agence qui achète des publicités pour le compte d'un annonceur. Si vous sélectionnez « Oui », votre annonce peut être rejetée si elle utilise des paramètres de ciblage tels que le ciblage par âge, sexe ou code postal. Le ciblage relatif à l'âge minimum peut être appliqué jusqu'à 21 ans.
    - Indiquez si votre compte publicitaire diffusera ou non des annonces de logement, de crédit ou d'emploi. Si vous sélectionnez « Oui », votre annonce peut être rejetée si elle utilise des paramètres de ciblage tels que le ciblage par âge, sexe ou code postal. Le ciblage relatif à l'âge minimum peut être appliqué jusqu'à 21 ans.
    - Indiquez si vous souhaitez utiliser le compte publicitaire pour des annonces politiques. Si vous diffusez une annonce politique, indiquez l'organisation politique ou le groupe d'intérêt qui finance la publicité. Si vous ne saisissez pas correctement l'organisation politique, vos

annonces risquent d'être rejetées. Vous devrez également remplir le « formulaire d'évaluation des annonces politiques » obligatoire avant de soumettre des annonces.

3. Choisissez Détails du compte et renseignez les informations de votre compte publicitaire :

Champ	Description
Nom	Le nom de votre compte publicitaire
Type de compte	Sélectionnez votre type de compte, s'il n'est pas renseigné automatiquement.
Type de facturation	Si vous souhaitez avoir accès en permanence à votre source de financement, choisissez « Renouvelable ». Cette option vous permet de réapprovisionner votre marge de crédit et est souvent choisie pour diffuser des annonces dans Ads Manager. Si vous préférez lancer une campagne ponctuelle avec un plafond de dépenses, vous pouvez choisir Ordre d'insertion. Si vous créez un compte avec un ordre d'insertion, mais que vous souhaitez fixer un plafond de dépenses ultérieurement, vous pouvez laisser ce dernier à 0 USD. De cette façon, le compte aura une limite de dépenses de 0 USD, mais vous pourrez en appliquer une plus tard.
Organisation publicitaire	L'organisation qui achète des publicités.
Centre de facturation	Sélectionnez le centre de facturation dans lequel vous souhaitez recevoir des factures. Vous pouvez utiliser celui créé automatiquement lorsque vous avez revendiqué votre entreprise pour la première fois ou ajouter

Champ	Description
	un autre centre de facturation en suivant la procédure du Centre d'aide aux entreprises.
Currency	Choisissez votre devise.
Plafond de dépenses	Si vous avez choisi « Ordre d'insertion » comme type de facturation, saisissez le budget d'un compte publicitaire.
Fuseau horaire	Sélectionnez votre fuseau horaire.

4. Choisissez Créer un compte. Votre compte publicitaire sera créé et vous le trouverez dans la section Comptes publicitaires d'Ads Manager. Pour commencer à lancer des annonces, vous devez saisir un mode de paiement. Vous pouvez également ajouter des membres à votre compte publicitaire.
5. Choisissez d'utiliser un paiement existant ou d'en créer un. Choisissez ensuite Enregistrer le mode de paiement.
6. Sélectionnez les membres que vous avez invités à rejoindre votre entreprise pour les ajouter au compte publicitaire. Pour plus d'informations sur les rôles et les autorisations qui peuvent être attribués, consultez la section Rôles et autorisations de l'organisation. Les membres ajoutés pourront ensuite se connecter à Ads Manager et accéder à ce compte publicitaire. Lorsque vous avez terminé, enregistrez vos membres.

Pour plus d'informations sur les comptes publicitaires, consultez https://businesshelp.snapchat.com/s/article/roles-permissions?language=en_US https://businesshelp.snapchat.com/s/ article/roles-permissions ? language=fr_US

Création d'une application dans votre compte Snapchat Ads

Pour activer l'accès à l'API marketing de Snapchat, assurez-vous d'avoir configuré un compte professionnel. Ensuite, procédez de la façon suivante.

1. Connectez-vous à Ads Manager. Choisissez ensuite le menu dans le coin supérieur gauche et sélectionnez Tableau de bord de l'entreprise, puis sélectionnez Informations sur l'entreprise.
2. Choisissez + OAuth App.

3. Entrez le nom de votre application et ajoutez l'URL suivante en tant qu'URI de redirection Snap `https://<aws-region>.console.aws.amazon.com/gluestudio/oauth`. Par exemple, si vous utilisez la région us-west-1, l'URL serait `https://us-west-1.console.aws.amazon.com/gluestudio/oauth) and choose` **`Create OAuth App`**. Choisissez Create OAuth App.
4. Les informations d'identification de votre application (ID et secret client) seront affichées. Enregistrez-les, car vous en aurez besoin pour créer une connexion.

Connexion à Snowflake dans AWS Glue Studio

> **Note**
>
> Vous pouvez utiliser AWS Glue pour Spark pour lire et écrire dans des tables dans Snowflake dans AWS Glue 4.0 et versions ultérieures. Pour configurer une connexion Snowflake avec des tâches AWS Glue par programmation, consultez Connexions Redshift.

AWS Glue fournit une prise en charge intégrée pour Snowflake. AWS Glue Studio fournit une interface visuelle pour se connecter à Snowflake, créer des tâches d'intégration de données et les exécuter sur l'exécution Spark AWS Glue Studio sans serveur.

AWS Glue Studio crée une connexion unifiée pour Snowflake. Pour de plus amples informations, veuillez consulter Considérations.

Rubriques

- Création d'une connexion Snowflake
- Création d'un nœud source Snowflake
- Création d'un nœud cible Snowflake
- Options avancées

Création d'une connexion Snowflake

> **Note**
>
> Les connexions unifiées (connexion v2) normalisent toutes les connexions pour utiliser les clés `USERNAME` et `PASSWORD` pour les informations d'identification d'authentification de base.

Vous pouvez toujours créer une connexion v1 via l'API avec des secrets contenant `sfUser`, `sfPassword`.

Lorsque vous ajoutez une source de données - nœud SnowflakeAWS Glue Studio, vous pouvez choisir une connexion AWS Glue Snowflake existante ou créer une nouvelle connexion. Vous devez choisir un type de connexion `SNOWFLAKE` et non un type de connexion `JDBC` configuré pour se connecter à Snowflake. Suivez la procédure suivante pour créer une connexion AWS Glue Snowflake :

Pour créer une connexion Snowflake

1. Dans Snowflake, générez un utilisateur *`snowflakeUser`* et un mot de passe. *`snowflakePassword`*
2. Déterminez avec quel entrepôt Snowflake cet utilisateur va interagir,. *`snowflakeWarehouse`* Vous pouvez le `DEFAULT_WAREHOUSE` définir comme forme *`snowflakeUser`* dans Snowflake ou vous en souvenir pour l'étape suivante.
3. Dans AWS Secrets Manager, créez un secret à l'aide de vos informations d'identification Snowflake. Pour créer un secret dans Secrets Manager, suivez le didacticiel disponible dans la section Créer un AWS Secrets Manager secret dans la AWS Secrets Manager documentation. Après avoir créé le secret, conservez le nom du secret *`secretName`* pour l'étape suivante.

   - Lorsque vous sélectionnez des paires clé/valeur, créez une paire pour *`snowflakeUser`* avec la clé. `sfUser`
   - Lorsque vous sélectionnez des paires clé/valeur, créez une paire pour *`snowflakePassword`* avec la clé. `sfPassword`
   - Lorsque vous sélectionnez des paires clé/valeur, créez une paire pour *`snowflakeWarehouse`* avec la clé. `sfWarehouse` Cela n'est pas nécessaire si une paire par défaut est définie dans Snowflake.
4. Dans le catalogue de AWS Glue données, créez une connexion en suivant les étapes décrites dans Ajouter une AWS Glue connexion. Après avoir créé la connexion, conservez le nom de la connexion pour l'étape suivante. *`connectionName`*

   - Lorsque vous sélectionnez un type de connexion, sélectionnez Snowflake.
   - Lorsque vous sélectionnez URL Snowflake, indiquez le nome d'hôte de votre instance Snowflake. L'URL utilisera un nom d'hôte sous la forme *`account_identifier`*`.snowflakecomputing.com`.

- Lorsque vous sélectionnez un AWS secret, fournissez*secretName*.

Création d'un nœud source Snowflake

Autorisations nécessaires

AWS Glue Studio les tâches utilisant des sources de données Snowflake nécessitent des autorisations supplémentaires. Pour plus d'informations sur la façon d'ajouter des autorisations aux tâches ETL, consultez Review IAM permissions needed for ETL jobs.

`SNOWFLAKE` AWS Glue les connexions utilisent un AWS Secrets Manager secret pour fournir des informations d'identification. Vos rôles de prévisualisation des tâches et des données AWS Glue Studio doivent être autorisés à lire ce secret.

Ajout d'une source de données Snowflake

Prérequis :

- Un AWS Secrets Manager secret pour vos identifiants Snowflake
- Une connexion au catalogue de AWS Glue données de type Snowflake

Pour ajouter un nœud Source de données – Snowflake :

1. Choisissez la connexion pour votre source de données Snowflake. Cela suppose que la connexion existe déjà et que vous pouvez la sélectionner parmi les connexions existantes. Si vous devez créer une connexion, choisissez Créer une connexion Snowflake. Pour en savoir plus, consultez Overview of using connectors and connections .

   Une fois que vous avez choisi une connexion, vous pouvez afficher ses propriétés en cliquant sur Afficher les propriétés. Les informations relatives à la connexion sont visibles, notamment l'URL, les groupes de sécurité, le sous-réseau, la zone de disponibilité, la description et les horodatages de création (UTC) et de dernière mise à jour (UTC).
2. Choisissez une option de source Snowflake :

   - Choisir une seule table : il s'agit de la table qui contient les données auxquelles vous souhaitez accéder à partir d'une seule table Snowflake.
   - Saisir une requête personnalisée : vous permet d'accéder à un jeu de données à partir de plusieurs tables Snowflake en fonction de votre requête personnalisée.

3. Si vous avez choisi une seule table, saisissez le nom d'un schéma Snowflake.

   Vous pouvez également choisir Saisir une requête personnalisée. Choisissez cette option pour accéder à un jeu de données personnalisé à partir de plusieurs tables Snowflake. Lorsque vous choisissez cette option, saisissez la requête Snowflake.

4. Dans les options Performance et sécurité (facultatif),

   - Activer le pushdown de requêtes : choisissez cette option si vous souhaitez décharger le travail vers l'instance Snowflake.

5. Dans Propriétés Snowflake personnalisées (facultatif), saisissez les paramètres et les valeurs nécessaires.

Création d'un nœud cible Snowflake

Autorisations nécessaires

AWS Glue Studio les tâches utilisant des sources de données Snowflake nécessitent des autorisations supplémentaires. Pour plus d'informations sur la façon d'ajouter des autorisations aux tâches ETL, consultez Review IAM permissions needed for ETL jobs.

`SNOWFLAKE` AWS Glue les connexions utilisent un AWS Secrets Manager secret pour fournir des informations d'identification. Vos rôles de prévisualisation des tâches et des données AWS Glue Studio doivent être autorisés à lire ce secret.

Ajout d'une cible de données Snowflake

Pour créer un nœud cible Snowflake :

1. Choisissez une table Snowflake existante comme cible ou saisissez un nouveau nom de table.
2. Lorsque vous utilisez le nœud cible Cible de données – Snowflake, vous pouvez choisir l'une des options suivantes :

   - APPREND : si une table existe déjà, insérez toutes les nouvelles données dans la table sous forme d'insertion. Si la table n'existe pas, créez-la, puis insérez toutes les nouvelles données.
   - MERGE : AWS Glue met à jour ou ajoute des données à votre table cible en fonction des conditions que vous spécifiez.

     Choisissez parmi les options :

- Choisir des clés et des actions simples : choisissez les colonnes à utiliser comme clés de correspondance entre les données source et votre jeu de données cible.

  Spécifiez les options suivantes lorsqu'elles correspondent :

  - Mettez à jour l'enregistrement dans votre jeu de données cible avec les données de la source.
  - Supprimez l'enregistrement dans votre jeu de données cible.

  Spécifiez les options suivantes lorsqu'elles ne correspondent pas :

  - Insérez les données source en tant que nouvelle ligne dans votre jeu de données cible.
  - Ne rien faire.
- Saisir une instruction MERGE personnalisée : vous pouvez ensuite choisir Valider l'instruction de fusion pour vérifier si l'instruction est valide ou non.

- TRUNCATE : si une table existe déjà, tronquez les données de la table en effaçant d'abord le contenu de la table cible. Si la troncature est réussie, insérez toutes les données. Si la table n'existe pas, créez-la, puis insérez toutes les nouvelles données. Si la troncature échoue, l'opération échouera.
- DROP : si une table existe déjà, supprimez les métadonnées et les données de la table. Si la suppression est réussie, insérez toutes les données. Si la table n'existe pas, créez-la, puis insérez toutes les nouvelles données. Si la suppression échoue, l'opération échouera.

Options avancées

Consultez la section Connexions Snowflake dans le guide du AWS Glue développeur.

Connexion à Stripe dans AWS Glue Studio

Stripe est une plateforme de traitement des paiements en ligne et de traitement des cartes de crédit destinée aux entreprises. La plateforme Stripe permet aux entreprises d'accepter des paiements en ligne, de créer des abonnements (facturation récurrente) pour leur commerce électronique et de configurer un compte bancaire pour recevoir les paiements. Stripe prend également en charge les paiements multipartites, dans lesquels il permet aux entreprises de configurer leur site de vente et de collecter les paiements, puis d'effectuer des paiements aux vendeurs ou aux fournisseurs de services via un compte « connecté ».

Rubriques

- AWS Glue support pour Stripe
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration de Stripe
- Configuration de connexions Stripe
- Lecture à partir d'entités Stripe
- Options de connexion Stripe
- Limitations
- Création d'un compte Stripe et configuration de l'application client

AWS Glue support pour Stripe

AWS Glue prend en charge Stripe comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser les tâches AWS Glue ETL pour interroger les données de Stripe.

Prise en charge en tant que cible ?

Non.

Versions de l'API Slack prises en charge

v1.

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les autorisations IAM requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
```

```
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Vous pouvez également utiliser les politiques IAM gérées suivantes pour autoriser l'accès :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM et Amazon EC2. Amazon CloudWatch Logs Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration de Stripe

Avant de pouvoir AWS Glue transférer des données depuis Stripe, vous devez satisfaire aux exigences suivantes :

Configuration requise

- Vous disposez d'un compte Stripe avec e-mail et mot de passe. Pour de plus amples informations, veuillez consulter Création d'un compte Stripe et configuration de l'application client.
- Votre compte Stripe est activé pour accéder à l'API. Toute utilisation de l'API Stripe est disponible sans frais supplémentaires.

Si vous répondez à ces exigences, vous êtes prêt à vous connecter AWS Glue à votre compte Stripe.

Configuration de connexions Stripe

Stripe prend en charge l'authentification personnalisée. Pour plus d'informations sur la génération des clés d'API requises pour l'autorisation personnalisée, consultez la documentation de l'API REST STRIPE.

Pour configurer une connexion Stripe :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes. Il est nécessaire de créer un secret pour chaque connexion dans AWS Glue.
   a. Pour l'application connectée gérée par le client : le secret doit contenir le secret du consommateur de l'application connectée avec `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` comme clé.
2. Dans AWS Glue Glue Studio, créez une connexion sous Data Connections en suivant les étapes ci-dessous :
   a. Lorsque vous sélectionnez un type de connexion, sélectionnez Stripe.
   b. Sélectionnez le rôle IAM qui AWS Glue peut assumer et dispose des autorisations nécessaires pour les actions suivantes :

      JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

   c. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.

d. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.

3. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.

Lecture à partir d'entités Stripe

Conditions préalables

- Un objet Stripe à partir duquel vous souhaitez lire.

Entités prises en charge

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Balance	Non	Non	Non	Oui	Non
Balance Transactions	Oui	Oui	Non	Oui	Oui
Frais	Oui	Oui	Non	Oui	Oui
Disputes	Oui	Oui	Non	Oui	Oui
File Links	Oui	Oui	Non	Oui	Oui
PaymentIn tents	Oui	Oui	Non	Oui	Oui
SetupIntents	Oui	Oui	Non	Oui	Oui
Payouts	Oui	Oui	Non	Oui	Oui
Rembourse ments	Oui	Oui	Non	Oui	Oui
Produits	Oui	Oui	Non	Oui	Oui
Prices	Oui	Oui	Non	Oui	Oui

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Coupons	Oui	Oui	Non	Oui	Oui
Promotion Codes	Oui	Oui	Non	Oui	Oui
Tax Codes	Non	Oui	Non	Oui	Non
Tax Rates	Oui	Oui	Non	Oui	Oui
Shipping Rates	Oui	Oui	Non	Oui	Oui
Séances	Oui	Oui	Non	Oui	Oui
Credit Notes	Oui	Oui	Non	Oui	Oui
Client	Oui	Oui	Non	Oui	Oui
Factures	Oui	Oui	Non	Oui	Oui
Invoice Items	Oui	Oui	Non	Oui	Non
Plans	Oui	Oui	Non	Oui	Oui
Quotes	Oui	Oui	Non	Oui	Non
Abonnements	Oui	Oui	Non	Oui	
Subscription Items	Non	Oui	Non	Oui	Non
Subscription Schedules	Oui	Oui	Non	Oui	Oui
Comptes	Non	Oui	Non	Oui	Oui

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Application Fees	Oui	Oui	Non	Oui	Oui
Country Specs	Non	Oui	Non	Oui	Non
Transferts	Oui	Oui	Non	Oui	Oui
Early Fraud Warnings	Oui	Oui	Non	Oui	Oui
Types de rapport	Non	Non	Non	Oui	Non

Exemple

```
stripe_read = glueContext.create_dynamic_frame.from_options(
    connection_type="stripe",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "coupons",
        "API_VERSION": "v1"
    }
)
```

Détails des entités et des champs Stripe

Entité	Champ	Type de données	Opérateurs pris en charge
Balance			
	available	List	
	connect_reserved	List	

Entité	Champ	Type de données	Opérateurs pris en charge
	en suspens	List	
	livemode	Booléen	
	objet	String	
	instant_available	List	
	issuing	Struct	
Balance Transactions			
	id	String	
	objet	String	
	montant	Entier	
	available_on	DateTime	=, >=, <=,<,>
	créé	DateTime	=, >=, <=,<,>
	currency	String	
	description	String	
	exchange_rate	BigDecimal	
	fee	Entier	
	fee_details	List	
	net	Entier	
	reporting_category	String	
	source	String	=
	status	String	

Entité	Champ	Type de données	Opérateurs pris en charge
	type	String	=
	cross_border_classification	String	
Frais			
	id	String	
	objet	String	
	montant	Entier	=, <, >
	amount_captured	Entier	
	amount_refunded	Entier	
	vigie	String	
	application_fee	String	
	application_fee_amount	Entier	
	balance_transaction	String	
	billing_details	Struct	
	calculated_statement_descriptor	String	
	captured	Booléen	
	créé	DateTime	=, >=, <=,<,>
	currency	String	
	customer	String	=

Entité	Champ	Type de données	Opérateurs pris en charge
	description	String	
	destination	String	
	dispute	String	
	disputed	Booléen	=
	failure_balance_transaction	String	
	failure_code	String	
	failure_message	String	
	fraud_details	Struct	
	facture	String	
	livemode	Booléen	
	metadata	Struct	
	on_behalf_of	String	
	order	String	
	outcome	Struct	
	paid	Booléen	
	payment_intent	String	=
	payment_method	String	
	payment_method_details	Struct	
	receipt_email	String	

Entité	Champ	Type de données	Opérateurs pris en charge
	receipt_number	String	
	receipt_url	String	
	refunded	Booléen	=
	refunds	Struct	
	review	String	
	shipping	Struct	
	source	Struct	
	source_transfer	String	
	statement_descriptor	String	
	statement_descriptor_suffix	String	
	status	String	
	transfert	String	
	transfer_data	Struct	
	transfer_group	String	=
Disputes			
	id	String	
	objet	String	
	montant	Entier	=, <, >
	balance_transaction	String	

Entité	Champ	Type de données	Opérateurs pris en charge
	balance_transactions	List	
	charge	String	=
	créé	DateTime	=, >=, <=,<,>
	currency	String	
	evidence	Struct	
	evidence_details	Struct	
	is_charge_refundable	Booléen	
	livemode	Booléen	
	metadata	Struct	
	payment_intent	String	=
	raison	String	=
	status	String	
	payment_method_details	Struct	
File Links			
	id	String	
	objet	String	
	créé	DateTime	=, >=, <=,<,>
	expired	Booléen	=
	expires_at	DateTime	

Entité	Champ	Type de données	Opérateurs pris en charge
	dans le fichier	String	=
	livemode	Booléen	
	metadata	Struct	
	url	String	
PaymentIntents			
	id	String	
	objet	String	
	montant	Entier	
	amount_capturable	Entier	
	amount_details	Struct	
	amount_received	Entier	
	vigie	String	
	application_fee_amount	Entier	
	automatic_payment_methods	Struct	
	canceled_at	DateTime	
	cancellation_reason	String	
	capture_method	String	
	client_secret	String	
	confirmation_method	String	

Entité	Champ	Type de données	Opérateurs pris en charge
	créé	DateTime	=, >=, <=,<,>
	currency	String	
	customer	String	=
	description	String	
	facture	String	
	last_payment_error	Struct	
	latest_charge	String	
	livemode	Booléen	
	metadata	Struct	
	next_action	Struct	
	on_behalf_of	String	
	payment_method	String	
	payment_method_options	Struct	
	payment_method_types	List	
	payment_method_configuration_details	Struct	
	traitement	Struct	
	receipt_email	String	
	review	String	

Entité	Champ	Type de données	Opérateurs pris en charge
	setup_future_usage	String	
	shipping	Struct	
	source	String	
	statement_descriptor	String	
	statement_descriptor_suffix	String	
	status	String	
	transfer_data	Struct	
	transfer_group	String	
SetupIntents			
	id	String	
	objet	String	
	vigie	String	
	cancellation_reason	String	
	client_secret	String	
	créé	DateTime	=, >=, <=,<,>
	customer	String	=
	description	String	
	flow_directions	List	
	last_setup_error	Struct	

Entité	Champ	Type de données	Opérateurs pris en charge
	latest_attempt	String	
	livemode	Booléen	
	mandate	String	
	metadata	Struct	
	next_action	Struct	
	on_behalf_of	String	
	payment_method	String	
	payment_method_options	Struct	
	payment_method_types	List	
	single_use_mandate	String	
	status	String	
	utilisation	String	
	automatic_payment_methods	Struct	
Payouts			
	id	String	
	objet	String	
	montant	Entier	=, <, >
	arrival_date	DateTime	=, >=, <=,<,>

Entité	Champ	Type de données	Opérateurs pris en charge
	automatic	Booléen	
	balance_transaction	String	
	créé	DateTime	=, >=, <=,<,>
	currency	String	
	description	String	=
	destination	String	
	failure_balance_transaction	String	
	failure_code	String	
	failure_message	String	
	livemode	Booléen	
	metadata	Struct	
	méthode	String	
	original_payout	String	
	reversed_by	String	
	reconciliation_status	String	
	source_type	String	
	statement_descriptor	String	
	status	String	
	type	String	

Entité	Champ	Type de données	Opérateurs pris en charge
	application_fee	String	
	application_fee_amount	Entier	
Remboursements			
	id	String	
	objet	String	
	montant	Entier	
	balance_transaction	String	
	charge	String	=
	créé	DateTime	=, >=, <=,<,>
	currency	String	
	metadata	Struct	
	destination_details	Struct	
	payment_intent	String	=
	raison	String	
	receipt_number	String	
	source_transfer_reversal	String	
	status	String	
	transfer_reversal	String	
Produits			

Entité	Champ	Type de données	Opérateurs pris en charge
	id	String	
	objet	String	
	actif	Booléen	=
	attributs	List	
	créé	DateTime	=, >=, <=,<,>
	default_price	String	
	description	String	
	images	List	
	livemode	Booléen	
	metadata	Struct	
	name	String	
	package_dimensions	Struct	
	shippable	Booléen	
	statement_descriptor	String	
	tax_code	String	
	type	String	=
	unit_label	String	
	updated	DateTime	
	url	String	
	fonctionnalités	List	

Entité	Champ	Type de données	Opérateurs pris en charge
Prices			
	id	String	
	objet	String	
	actif	Booléen	=
	billing_scheme	String	
	créé	DateTime	=, >=, <=,<,>
	currency	String	=
	custom_unit_amount	Struct	
	livemode	Booléen	
	lookup_key	String	
	metadata	Struct	
	nickname	String	
	product	String	=
	recurring	Struct	
	tax_behavior	String	
	tiers_mode	String	
	transform_quantity	Struct	
	type	String	=
	unit_amount	Entier	
	unit_montant_decimal	String	

Entité	Champ	Type de données	Opérateurs pris en charge
Coupons			
	Id	String	
	objet	String	
	amount_off	Entier	
	créé	DateTime	=, >=, <=,<,>
	currency	String	=
	duration	String	=
	duration_in_months	Entier	=,<,>
	livemode	Booléen	
	max_redemptions	Entier	=, <, >
	metadata	Struct	
	name	String	
	percent_off	Double	=
	redeem_by	DateTime	=, >=, <=, <, >
	times_redeemed	Entier	
	valid	Booléen	
Promotion Codes			
	Id	String	
	objet	String	
	actif	Booléen	=

Entité	Champ	Type de données	Opérateurs pris en charge
	code	String	=
	coupon	Struct	
	créé	DateTime	=,>=,<=,<,>
	customer	String	
	expires_at	DateTime	
	livemode	Booléen	
	max_redemptions	Entier	
	metadata	Struct	
	restrictions	Struct	
	times_redeemed	Entier	
Tax Codes			
	Id	String	
	objet	String	
	description	String	
	name	String	
Tax Rates			
	Id	String	
	objet	String	
	actif	Booléen	=
	country	String	

Entité	Champ	Type de données	Opérateurs pris en charge
	créé	DateTime	=, >=, <=, <, >
	description	String	
	display_name	String	
	inclusive	Booléen	=
	jurisdiction	String	
	jurisdiction_level	String	
	livemode	Booléen	
	metadata	Struct	
	percentage	Double	
	effective_percentage	Double	
	state	String	
	tax_type	String	
Shipping Rates			
	Id	String	
	objet	String	
	actif	Booléen	=
	créé	DateTime	=, >=, <=, <, >
	delivery_estimate	Struct	
	display_name	String	
	fixed_amount	Struct	

Entité	Champ	Type de données	Opérateurs pris en charge
	livemode	Booléen	
	metadata	Struct	
	tax_behavior	String	
	tax_code	String	
	type	String	
Séances			
	id	String	
	objet	String	
	after_expiration	Struct	
	allow_promotion_codes	Booléen	
	amount_subtotal	Entier	
	amount_total	Entier	
	automatic_tax	Struct	
	billing_address_collection	String	
	cancel_url	String	
	client_reference_id	String	
	consent	Struct	
	consent_collection	Struct	
	créé	DateTime	=, >=, <=, <, >

Entité	Champ	Type de données	Opérateurs pris en charge
	currency	String	
	custom_text	Struct	
	customer	String	
	customer_creation	String	
	customer_details	Struct	
	customer_email	String	
	expires_at	DateTime	
	facture	String	
	invoice_creation	Struct	
	livemode	Booléen	
	locale	String	
	metadata	Struct	
	mode	String	
	payment_intent	String	=
	payment_link	String	
	payment_method_collection	String	
	payment_method_options	Struct	
	payment_method_types	List	

Entité	Champ	Type de données	Opérateurs pris en charge
	payment_status	String	
	phone_number_collection	Struct	
	recovered_from	String	
	setup_intent	String	
	shipping_address_collection	Struct	
	shipping_cost	Struct	
	shipping_details	Struct	
	shipping_options	List	
	status	String	
	submit_type	String	
	abonnement	String	
	success_url	String	
	tax_id_collection	Struct	
	total_details	Struct	
	url	String	
	ui_mode	String	
Credit Notes			
	id	String	
	objet	String	

Entité	Champ	Type de données	Opérateurs pris en charge
	montant	Entier	
	créé	DateTime	=, >=, <=, <, >
	currency	String	
	customer	String	=
	customer_balance_transaction	String	
	discount_amount	Entier	
	discount_amounts	List	
	facture	String	=
	lines	Struct	
	livemode	Booléen	
	memo	String	
	metadata	Struct	
	number	String	
	out_of_band_amount	Entier	
	pdf	String	
	raison	String	
	refund	String	
	status	String	
	subtotal	Entier	

Entité	Champ	Type de données	Opérateurs pris en charge
	subtotal_excluding_tax	Entier	
	tax_amounts	List	
	total	Entier	
	total_excluding_tax	Entier	
	type	String	
	voided_at	DateTime	
	amount_shipping	Entier	
	effective_at	DateTime	
	shipping_cost	Struct	
Client			
	id	String	
	objet	String	
	adresse	Struct	
	balance	Entier	
	créé	DateTime	
	currency	String	=, >=, <=, <, >
	default_source	String	
	delinquent	Booléen	=
	description	String	

Entité	Champ	Type de données	Opérateurs pris en charge
	discount	Struct	
	e-mail	String	=
	invoice_prefix	String	
	invoice_settings	Struct	
	livemode	Booléen	
	metadata	Struct	
	name	String	
	next_invoice_sequence	Entier	
	phone	String	
	preferred_locales	List	
	shipping	Struct	
	tax_exempt	String	
	test_clock	String	
Factures			
	id	String	
	objet	String	
	account_country	String	
	account_name	String	
	account_tax_ids	List	

Entité	Champ	Type de données	Opérateurs pris en charge
	amount_due	Entier	
	amount_paid	Entier	
	amount_remaining	Entier	
	vigie	String	
	application_fee_amount	Entier	
	attempt_count	Entier	
	attempted	Booléen	=
	auto_advance	Booléen	=
	automatic_tax	Struct	
	billing_reason	String	
	charge	String	
	collection_method	String	=
	créé	DateTime	=, >=, <=, <, >
	currency	String	
	custom_fields	List	
	customer	String	=
	customer_address	Struct	
	customer_email	String	
	customer_name	String	

Entité	Champ	Type de données	Opérateurs pris en charge
	customer_phone	String	
	customer_shipping	Struct	
	customer_tax_exempt	String	
	customer_tax_ids	List	
	default_payment_method	String	
	default_source	String	
	default_tax_rates	List	
	description	String	
	discount	Struct	
	discounts	List	
	due_date	DateTime	=, >=, <=, <, >
	ending_balance	Entier	
	footer	String	
	from_invoice	Struct	
	hosted_invoice_url	String	
	invoice_pdf	String	
	last_finalization_error	Struct	
	latest_revision	String	
	lines	Struct	

Entité	Champ	Type de données	Opérateurs pris en charge
	livemode	Booléen	
	metadata	Struct	
	next_payment_attempt	DateTime	
	number	String	
	on_behalf_of	String	
	paid	Booléen	=
	paid_out_of_band	Booléen	
	payment_intent	String	
	payment_settings	Struct	
	period_end	DateTime	=, >=, <=, <, >
	period_start	DateTime	=, >=, <=, <, >
	post_payment_credit_notes_amount	Entier	
	pre_payment_credit_notes_amount	Entier	
	quote	String	
	receipt_number	String	
	rendering	Struct	
	rendering_options	Struct	
	starting_balance	Entier	

Entité	Champ	Type de données	Opérateurs pris en charge
	statement_descriptor	String	
	status	String	=
	status_transitions	Struct	
	abonnement	String	
	subscription_details	Struct	
	subtotal	Entier	=, <, >
	subtotal_excluding_tax	Entier	
	tax	Entier	
	test_clock	String	
	total	Entier	=, <, >
	total_discount_amounts	List	
	total_excluding_tax	Entier	
	total_tax_amounts	List	
	transfer_data	Struct	
	webhooks_delivered_at	DateTime	
	automatically_finalizes_at	DateTime	
	effective_at	DateTime	

Entité	Champ	Type de données	Opérateurs pris en charge
	émetteur	Struct	
Invoice Items			
	id	String	
	objet	String	
	montant	Entier	=, <, >
	currency	String	
	customer	String	=
	date	DateTime	
	description	String	
	discountable	Booléen	
	discounts	List	
	facture	String	=
	livemode	Booléen	
	metadata	Struct	
	point	Struct	
	plan	Struct	
	prix	Struct	
	proration	Booléen	=
	quantity	Entier	
	abonnement	String	

Entité	Champ	Type de données	Opérateurs pris en charge
	subscription_item	String	
	tax_rates	List	
	test_clock	String	
	unit_amount	Entier	
	unit_montant_decimal	String	
Plans			
	id	String	
	objet	String	
	actif	Booléen	=
	aggregate_usage	String	
	montant	Entier	
	amount_decimal	String	
	billing_scheme	String	
	créé	DateTime	=, >=, <=, <, >
	currency	String	=
	interval	String	=
	interval_count	Entier	
	livemode	Booléen	
	metadata	Struct	
	nickname	String	

Entité	Champ	Type de données	Opérateurs pris en charge
	product	String	=
	tiers_mode	String	
	transform_usage	Struct	
	trial_period_days	Entier	=, <, >
	usage_type	String	
	meter	String	
Quotes			
	id	String	
	objet	String	
	amount_subtotal	Entier	
	amount_total	Entier	
	vigie	String	
	application_fee_amount	Entier	
	application_fee_percent	Double	
	automatic_tax	Struct	
	collection_method	String	
	computed	Struct	
	créé	DateTime	
	currency	String	

Entité	Champ	Type de données	Opérateurs pris en charge
	customer	String	=
	default_tax_rates	List	
	description	String	
	discounts	List	
	expires_at	DateTime	
	footer	String	
	from_quote	Struct	
	header	String	
	facture	String	
	invoice_settings	Struct	
	livemode	Booléen	
	metadata	Struct	
	number	String	
	on_behalf_of	String	
	status	String	=
	status_transitions	Struct	
	abonnement	String	
	subscription_data	Struct	
	subscription_schedule	String	
	test_clock	String	

Entité	Champ	Type de données	Opérateurs pris en charge
	total_details	Struct	
	transfer_data	Struct	
Abonnements			
	id	String	
	objet	String	
	vigie	String	
	application_fee_percent	Double	
	automatic_tax	Struct	
	billing_cycle_anchor	DateTime	
	billing_thresholds	Struct	
	cancel_at	DateTime	
	cancel_at_period_end	Booléen	
	canceled_at	DateTime	
	collection_method	String	=
	créé	DateTime	=, >=, <=,<,>
	currency	String	
	current_period_end	DateTime	=, >=, <=
	current_period_start	DateTime	=, >=, <=
	customer	String	=

Entité	Champ	Type de données	Opérateurs pris en charge
	days_until_due	Entier	
	default_payment_method	String	
	default_source	String	
	default_tax_rates	List	
	description	String	
	discount	Struct	
	ended_at	DateTime	
	items	Struct	
	latest_invoice	String	
	livemode	Booléen	
	metadata	Struct	
	next_pending_invoice_item_invoice	DateTime	
	pause_collection	Struct	
	payment_settings	Struct	
	pending_invoice_item_interval	Struct	
	pending_setup_intent	String	
	pending_update	Struct	
	plan	Struct	

Entité	Champ	Type de données	Opérateurs pris en charge
	quantity	Entier	
	schedule	String	
	start_date	DateTime	
	status	String	=
	test_clock	String	
	transfer_data	Struct	
	trial_end	DateTime	
	trial_start	DateTime	
Subscription Items			
	Id	String	
	objet	String	
	billing_thresholds	Struct	
	créé	DateTime	=, >=, <=, <, >
	metadata	Struct	
	plan	Struct	
	prix	Struct	
	abonnement	String	
	tax_rates	List	
	discounts	List	

Entité	Champ	Type de données	Opérateurs pris en charge
Subscription Schedules			
	objet	String	
	vigie	String	
	canceled_at	DateTime	
	completed_at	DateTime	
	créé	DateTime	
	current_phase	Struct	
	customer	String	=
	default_settings	Struct	
	end_behavior	String	
	livemode	Booléen	
	metadata	Struct	
	phases	List	
	released_at	DateTime	
	released_subscription	String	
	renewal_interval	String	
	status	String	
	abonnement	String	
	test_clock	String	

Entité	Champ	Type de données	Opérateurs pris en charge
Comptes			
	details_submitted	Booléen	
	tos_acceptance	Struct	
	type	String	
	metadata	Struct	
	id	String	
	objet	String	
	default_currency	String	
	caractéristiques	Struct	
	charges_enabled	Booléen	
	paramètres	Struct	
	requirements	Struct	
	payouts_enabled	Booléen	
	future_requirements	Struct	
	external_accounts	Struct	
	contrôleur	Struct	
	country	String	
	e-mail	String	
	créé	DateTime	=, >=, <=, <, >
	business_profile	Struct	

Entité	Champ	Type de données	Opérateurs pris en charge
	business_type	String	
	company	Struct	
Application Fees			
	id	String	
	objet	String	
	compte	String	
	montant	Entier	=, <, >
	amount_refunded	Entier	=, <, >
	vigie	String	
	balance_transaction	String	
	charge	String	=
	créé	DateTime	
	currency	String	
	livemode	Booléen	
	originating_transaction	String	
	refunded	Booléen	=
	refunds	Struct	
	fee_source	Struct	
Country Specs			

Entité	Champ	Type de données	Opérateurs pris en charge
	id	String	
	objet	String	
	default_currency	String	
	supported_bank_account_currencies	Struct	
	supported_payment_currencies	List	
	supported_payment_methods	List	
	supported_transfer_countries	List	
	verification_fields	Struct	
Transferts			
	id	String	
	objet	String	
	montant	Entier	=, <, >
	amount_reversed	Entier	
	balance_transaction	String	
	créé	DateTime	=, >=, <=, <, >
	currency	String	=
	description	String	

Entité	Champ	Type de données	Opérateurs pris en charge
	destination	String	=
	destination_payment	String	
	livemode	Booléen	
	metadata	Struct	
	reversals	Struct	
	reversed	Booléen	
	source_transaction	String	
	source_type	String	
	transfer_group	String	=
Early Fraud Warnings			
	id	String	
	objet	String	
	actionable	Booléen	
	charge	String	=
	créé	DateTime	=, >=, <=, <, >
	fraud_type	String	
	livemode	Booléen	
	payment_intent	String	=
Types de rapport			
	id	String	

Entité	Champ	Type de données	Opérateurs pris en charge
	objet	String	
	data_available_end	DateTime	
	data_available_start	DateTime	
	default_columns	List	
	livemode	Booléen	
	name	String	
	updated	DateTime	
	version	Entier	

Requêtes de partitionnement

Les options Spark supplémentaires `PARTITION_FIELD`, `LOWER_BOUND`, `UPPER_BOUND` et `NUM_PARTITIONS` peuvent être indiquées si vous souhaitez utiliser la simultanéité dans Spark. Avec ces paramètres, la requête d'origine serait divisée en `NUM_PARTITIONS` nombres de sous-requêtes pouvant être exécutées simultanément par les tâches Spark.

- `PARTITION_FIELD` : le nom du champ à utiliser pour partitionner la requête.
- `LOWER_BOUND` : une valeur limite inférieure inclusive du champ de partition choisi.

  Pour le champ Date, nous acceptons le format de date Spark utilisé dans les requêtes SQL Spark. Exemple de valeur valide : `"2024-07-01T00:00:00.000Z"`.
- `UPPER_BOUND` : une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS` : nombre de partitions.

Les détails relatifs à la prise en charge des champs de partitionnement par entité sont présentés dans le tableau ci-dessous.

Nom de l'entité	Champ de partitionnement	Type de données
Balance Transactions	créé	DateTime
Frais	créé	DateTime
Disputes	créé	DateTime
File Links	créé	DateTime
PaymentIntents	créé	DateTime
SetupIntents	créé	DateTime
Payouts	créé	DateTime
Remboursements	créé	DateTime
Produits	créé	DateTime
Prices	créé	DateTime
Coupons	créé	DateTime
Promotion Codes	créé	DateTime
Tax Rates	créé	DateTime
Shipping Rates	créé	DateTime
Séances	créé	DateTime
Credit Notes	créé	DateTime
Client	créé	DateTime
Factures	créé	DateTime
Plans	créé	DateTime
Abonnements	créé	DateTime

Nom de l'entité	Champ de partitionnement	Type de données
Subscription Schedules	créé	DateTime
Comptes	créé	DateTime
Application Fees	créé	DateTime
Transferts	créé	DateTime
Early Fraud Warnings	créé	DateTime

Exemple

```
stripe_read = glueContext.create_dynamic_frame.from_options(
    connection_type="stripe",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "coupons",
        "API_VERSION": "v1",
        "PARTITION_FIELD": "created"
        "LOWER_BOUND": "2024-05-01T20:55:02.000Z"
        "UPPER_BOUND": "2024-07-11T20:55:02.000Z"
        "NUM_PARTITIONS": "10"
    }
)
```

Options de connexion Stripe

Les options de connexion pour Stripe sont les suivantes :

- `ENTITY_NAME`(Chaîne) - (Obligatoire) Utilisé pour la lecture/l'écriture. Le nom de votre objet dans Stripe.
- `API_VERSION`(Chaîne) - (Obligatoire) Utilisé pour la lecture/l'écriture. Version d'API Rest Stripe que vous souhaitez utiliser. Exemple : v1.
- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.

- QUERY(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.
- PARTITION_FIELD(Chaîne) - Utilisé pour la lecture. Champ à utiliser pour partitionner la requête.
- LOWER_BOUND(Chaîne) - Utilisé pour la lecture. Une valeur limite inférieure inclusive du champ de partition choisi.
- UPPER_BOUND(Chaîne) - Utilisé pour la lecture. Une valeur limite supérieure exclusive du champ de partition choisi.
- NUM_PARTITIONS(Entier) - Par défaut : 1. Utilisé pour la lecture. Nombre de partitions pour la lecture.

Limitations

Les restrictions du connecteur Stripe sont les suivantes :

- Seul le partitionnement basé sur les champs est pris en charge par le connecteur.
- Le partitionnement basé sur les enregistrements n'est pas pris en charge par le connecteur, aucune disposition ne permet de récupérer le nombre total d'enregistrements.
- Le type de données de clé primaire étant String, le partitionnement basé sur l'identifiant n'est donc pas pris en charge par le connecteur.

Création d'un compte Stripe et configuration de l'application client

Création d'un compte Stripe

1. Choisissez sur le lien https://dashboard.stripe.com/enregistrement.
2. Saisissez votre e-mail, votre nom complet, votre mot de passe et choisissez Créer un compte.
3. Après vous être connecté avec le compte, vérifiez le compte en choisissant Ouvrir Gmail.
4. Vérifiez le compte en cliquant sur le lien de vérification reçu par e-mail.
5. Après avoir cliqué sur Vérifier l'adresse e-mail, vous serez redirigé vers une autre page.
6. Après avoir cliqué sur Activer les paiements pour activer le compte, il sera redirigé vers la page Activer les paiements (https://dashboard.stripe.com/bienvenue). Assurez-vous de renseigner toutes vos informations valides, puis de choisir le bouton Continuer.

Création d'une application pour développeurs Slack

1. Connectez-vous à Stripe.
2. Choisissez Développeurs comme indiqué tout en haut de l'image ci-dessous.
3. Choisissez Clés d'API sous Développeurs.
4. Choisissez Révéler la clé de test pour obtenir les clés d'API.

Connexion à Teradata Vantage dans AWS Glue Studio

AWS Glue fournit une prise en charge intégrée pour Teradata Vantage. AWS Glue Studio fournit une interface visuelle pour se connecter à Teradata, créer des tâches d'intégration de données et les exécuter sur l'exécution Spark AWS Glue Studio sans serveur.

AWS Glue Studio crée une connexion unifiée pour Teradata Vantage. Pour de plus amples informations, veuillez consulter Considérations.

Rubriques

- Création d'une connexion Teradata Vantage
- Création d'un nœud source Teradata
- Création d'un nœud cible Teradata
- Options avancées

Création d'une connexion Teradata Vantage

Pour vous connecter à Teradata Vantage depuis AWS Glue, vous devez créer et stocker vos informations d'identification Teradata dans un AWS Secrets Manager secret, puis associer ce secret à une AWS Glue connexion Teradata.

Prérequis :

- Si vous accédez à votre environnement Teradata via Amazon VPC, configurez Amazon VPC pour permettre à AWS Glue votre tâche de communiquer avec l'environnement Teradata. Nous vous déconseillons d'accéder à l'environnement Teradata via l'Internet public.

  Dans Amazon VPC, identifiez ou créez un VPC, un sous-réseau et un groupe de sécurité qui AWS Glue seront utilisés lors de l'exécution de la tâche. En outre, vous devez vous assurer qu'Amazon VPC est configuré pour autoriser le trafic réseau entre votre instance Teradata et cet emplacement.

Votre tâche devra établir une connexion TCP avec votre port client Teradata. Pour de plus amples informations sur les ports Teradata, consultez la documentation Teradata.

Selon la configuration de votre réseau, la connectivité VPC sécurisée peut nécessiter des modifications dans Amazon VPC et dans d'autres services réseau. Pour plus d'informations sur AWS la connectivité, consultez les options de AWS connectivité dans la documentation Teradata.

Pour configurer une connexion AWS Glue Teradata :

1. Dans votre configuration Teradata, identifiez ou créez un utilisateur et un mot de passe AWS Glue vous permettra de vous connecter à, *teradataUser* et*teradataPassword*. Pour plus d'informations, consultez la présentation de Vantage Security dans la documentation Teradata.
2. Dans AWS Secrets Manager, créez un secret à l'aide de vos informations d'identification Teradata. Pour créer un secret dans Secrets Manager, suivez le didacticiel disponible dans la section Créer un AWS Secrets Manager secret dans la AWS Secrets Manager documentation. Après avoir créé le secret, conservez le nom du secret *secretName* pour l'étape suivante.

   - Lorsque vous sélectionnez des paires clé/valeur, créez une paire pour la clé `user` contenant la valeur. *teradataUsername*
   - Lorsque vous sélectionnez des paires clé/valeur, créez une paire pour la clé `password` contenant la valeur. *teradataPassword*
3. Dans la AWS Glue console, créez une connexion en suivant les étapes décrites dansthe section called “Ajouter une AWS Glue connexion”. Après avoir créé la connexion, conservez le nom de la connexion pour l'étape suivante. *connectionName*

   - Lorsque vous sélectionnez un type de connexion, sélectionnez Teradata.
   - Lorsque vous fournissez l'URL JDBC, indiquez l'URL de votre instance. Vous pouvez également coder en dur certains paramètres de connexion séparés par des virgules dans votre URL JDBC. L'URL doit être conforme au format suivant : `jdbc:teradata://`*teradataHostname*`/`*ParameterName*`=`*ParameterValue*`,`*ParameterName*

     Les paramètres d'URL pris en charge sont les suivants :

     - `DATABASE` – nom de la base de données sur l'hôte à laquelle accéder par défaut.
     - `DBS_PORT` – le port de base de données utilisé lors de l'exécution sur un port non standard.
   - Lorsque vous sélectionnez un type d'identifiant, sélectionnez AWS Secrets Manager, puis définissez AWS Secret sur*secretName*.

4. Dans les situations suivantes, vous pouvez avoir besoin d'une configuration supplémentaire :

   - Pour les instances Teradata hébergées sur AWS un Amazon VPC
     - Vous devrez fournir les informations de connexion Amazon VPC à la AWS Glue connexion qui définit vos informations d'identification de sécurité Teradata. Lorsque vous créez ou mettez à jour votre connexion, définissez le VPC, le sous-réseau et les groupes de sécurité dans les options réseau.

Création d'un nœud source Teradata

Prérequis

- Une connexion AWS Glue Teradata Vantage, configurée avec un AWS Secrets Manager secret, comme décrit dans la section précédente,. the section called “Création d'une connexion Teradata Vantage”
- Les autorisations appropriées sur votre tâche pour lire le secret utilisé par la connexion.
- Une table Teradata que vous souhaitez lire ou interroger*`targetQuery`*. *`tableName`*

Ajout d'une source de données Teradata

Pour ajouter un nœud Source de données – Teradata :

1. Choisissez la connexion pour votre source de données Teradata. Puisque vous l'avez créé, il devrait être disponible dans le menu déroulant. Si vous devez créer une connexion, choisissez Créer une nouvelle connexion. Pour de plus amples informations, veuillez consulter la section précédente the section called “Création d'une connexion Teradata Vantage”.

   Une fois que vous avez choisi une connexion, vous pouvez afficher ses propriétés en cliquant sur Afficher les propriétés.
2. Choisissez une option Source Teradata :

   - Choisissez une seule table : accéder à toutes les données à partir d'une seule table.
   - Saisir une requête personnalisée : accéder à un jeu de données à partir de plusieurs tables en fonction de votre requête personnalisée.
3. Si vous avez choisi une seule table, entrez*`tableName`*.

   Si vous avez choisi Saisir une requête personnalisée, saisissez une requête SQL SELECT.

4. Dans les Propriétés Teradata personnalisées, saisissez les paramètres et les valeurs nécessaires.

Création d'un nœud cible Teradata

Prérequis

- Une connexion AWS Glue Teradata Vantage, configurée avec un AWS Secrets Manager secret, comme décrit dans la section précédente,. the section called "Création d'une connexion Teradata Vantage"
- Les autorisations appropriées sur votre tâche pour lire le secret utilisé par la connexion.
- Une table Teradata dans laquelle vous souhaitez écrire,*tableName*.

Ajout d'une cible de données Teradata

Pour ajouter un nœud Cible de données – Teradata :

1. Choisissez la connexion pour votre source de données Teradata. Puisque vous l'avez créé, il devrait être disponible dans le menu déroulant. Si vous devez créer une connexion, choisissez Créer une connexion Teradata. Pour en savoir plus, consultez Overview of using connectors and connections .

   Une fois que vous avez choisi une connexion, vous pouvez afficher ses propriétés en cliquant sur Afficher les propriétés.
2. Configurez le nom de la table en fournissant*tableName*.
3. Dans les Propriétés Teradata personnalisées, saisissez les paramètres et les valeurs nécessaires.

Options avancées

Vous pouvez fournir des options avancées lors de la création d'un nœud Teradata. Ces options sont les mêmes que celles disponibles lors de la programmation AWS Glue de scripts Spark.

Consultez the section called "Connexions Teradata Vantage".

Connexion à Twilio

Twilio fournit des outils de communication programmables pour passer et recevoir des appels téléphoniques, envoyer et recevoir des messages texte, et exécuter d'autres fonctions de communication à l'aide de son service APIs Web. Twilio APIs alimente sa plateforme de communication. Derrière cela se APIs trouve une couche logicielle qui connecte et optimise les réseaux de communication du monde entier pour permettre à vos utilisateurs d'appeler et d'envoyer des messages à n'importe qui, dans le monde entier. En tant qu'utilisateur de Twilio, vous pouvez vous connecter AWS Glue à votre compte Twilio. Vous pouvez ensuite utiliser Twilio comme source de données dans vos tâches ETL. Exécutez ces jobs pour transférer des données entre Twilio et AWS des services ou d'autres applications prises en charge.

Rubriques

- AWS Glue support pour Twilio
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration de Twilio
- Configuration de connexions Twilio
- Lecture à partir d'entités Twilio
- Options de connexion Twilio
- Restrictions et remarques relatives au connecteur Twilio

AWS Glue support pour Twilio

AWS Glue soutient Twilio comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser les tâches AWS Glue ETL pour interroger les données de Twilio.

Prise en charge en tant que cible ?

Non.

Versions de l'API Twilio prises en charge

Les versions d'API Twilio suivantes sont prises en charge :

- v1
- 2010-04-01

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les autorisations AWS IAM requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Si vous ne voulez pas utiliser la méthode ci-dessus, vous pouvez également utiliser les politiques IAM gérées suivantes :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la

convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration de Twilio

Avant de pouvoir AWS Glue transférer des données depuis Twilio, vous devez répondre aux exigences suivantes :

Configuration requise

Les exigences minimales sont les suivantes :

- Vous disposez d'un compte Twilio avec e-mail et mot de passe.
- Votre compte Twilio est activé pour accéder à l'API.

Si vous remplissez ces conditions, vous êtes prêt à vous connecter AWS Glue à votre compte Twilio. Pour les connexions classiques, vous n'avez rien d'autre à faire dans Twilio.

Configuration de connexions Twilio

Twilio prend en charge le nom d'utilisateur et le mot de passe pour l'authentification de base. L'authentification de base est une méthode d'authentification simple dans laquelle les clients fournissent des informations d'identification directement pour accéder aux ressources protégées. AWS Glue est capable d'utiliser le nom d'utilisateur (SID du compte) et le mot de passe (jeton d'authentification) pour authentifier Twilio. APIs

Pour consulter la documentation publique de Twilio pour le flux d'authentification de base, consultez Basic Authentication | Twilio.

Pour configurer une connexion Twilio :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes :
   - Pour l'authentification de base : le secret doit contenir le secret du consommateur de l'application connectée avec le SID du compte (nom d'utilisateur) et le jeton d'authentification (mot de passe).

> **Note**
>
> Vous devez créer un secret pour vos connexions dans AWS Glue.

2. Dans AWS Glue Glue Studio, créez une connexion sous Data Connections en suivant les étapes ci-dessous :
   a. Lorsque vous sélectionnez un type de connexion, sélectionnez Twilio.
   b. Indiquez l'`Edge_Location` de l'instance Twilio à laquelle vous souhaitez vous connecter.
   c. Sélectionnez le rôle AWS IAM qui AWS Glue peut assumer et dispose des autorisations nécessaires pour les actions suivantes :

      JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

   d. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.
   e. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.
3. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.

Lecture à partir d'entités Twilio

Prérequis

Un objet Twilio à partir duquel vous souhaitez lire. Vous aurez besoin du nom de l'objet, comme `SMS-Message` ou `SMS-CountryPricing`.

Entités prises en charge pour la source :

Entité	Interface	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partitionnement pris en charge
SMS-Message	REST	Oui	Oui	Non	Oui	Oui
SMS-CountryPricing	REST	Non	Non	Non	Oui	Non
Voice-Call	REST	Oui	Oui	Non	Oui	Non
Voice-Application	REST	Oui	Oui	Non	Oui	Non
OutgoingCallerIdentification vocale	REST	Oui	Oui	Non	Oui	Non
Voice-Queue	REST	Oui	Oui	Non	Oui	Non
Conversations-Conversation	REST	Oui	Oui	Non	Oui	Non
Conversations-User	REST	Non	Oui	Non	Oui	Non
Conversations-Role	REST	Non	Oui	Non	Oui	Non

Entité	Interface	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Conversations-Configuration	REST	Non	Non	Non	Oui	Non
Conversations-AddressConfiguration	REST	Oui	Oui	Non	Oui	Non
Conversations-WebhookConfiguration	REST	Non	Non	Non	Oui	Non
Conversations-ParticipantConversation	REST	Non	Non	Non	Oui	Non
Conversations-Credential	REST	Non	Oui	Non	Oui	Non
Conversations-ConversationService	REST	Non	Oui	Non	Oui	Non

Exemple :

```
twilio_read = glueContext.create_dynamic_frame.from_options(
    connection_type="twilio",
```

```
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "sms-message",
        "API_VERSION": "2010-04-01",
        "Edge_Location": "sydney.us1"
    }
```

Détails des entités et des champs Twilio :

Entité	Champ	Type de données	Opérateurs pris en charge
SMS-Message	account_sid	String	N/A
	api_version	String	N/A
	body	String	N/A
	date_created	Datetime	N/A
	date_sent	Datetime	>=, <=, =
	date_updated	Datetime	N/A
	direction	String	N/A
	error_code	Entier	N/A
	error_message	String	N/A
	from	Entier	=
	messaging_service_sid	String	N/A
	num_media	String	N/A
	num_segments	String	N/A
	prix	String	N/A
	price_unit	Struct	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	sid	Entier	N/A
	status	String	N/A
	subresource_uris	Map	N/A
	to	Entier	=
	uri	Datetime	N/A
SMS- CountryPricing	country	String	N/A
	iso_country	String	N/A
	url	String	N/A
	outbound_sms_prices	List	N/A
	inbound_sms_prices	List	N/A
	price_unit	String	N/A
Voice-Call	account_sid	String	N/A
	annotation	String	N/A
	answered_by	String	N/A
	api_version	String	N/A
	caller_name	String	N/A
	date_created	Datetime	N/A
	date_updated	Datetime	N/A
	direction	String	N/A
	duration	String	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	end_time	Datetime	>=, <=, =
	forwarded_from	String	N/A
	from	String	=
	from_formatted	String	N/A
	group_sid	String	N/A
	parent_call_sid	String	N/A
	phone_number_sid	String	N/A
	prix	String	N/A
	price_unit	String	N/A
	sid	String	N/A
	start_time	Datetime	>=, <=, =
	status	String	=
	subresource_uris	String	N/A
	to	String	=
	to_formatted	String	N/A
	trunk_sid	String	N/A
	uri	String	N/A
	queue_time	String	N/A
Voice-Application	account_sid	String	N/A
	api_version	String	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	date_created	Datetime	N/A
	date_updated	Datetime	N/A
	friendly_name	String	=
	message_status_callback	String	N/A
	sid	String	N/A
	sms_fallback_method	String	N/A
	sms_fallback_url	String	N/A
	sms_method	String	N/A
	sms_status_callback	String	N/A
	sms_url	String	N/A
	status_callback	String	N/A
	status_callback_method	String	N/A
	uri	String	N/A
	voice_caller_id_lookup	Booléen	N/A
	voice_fallback_method	String	N/A
	voice_fallback_url	String	N/A
	voice_method	String	N/A
	voice_url	String	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
public_application_connect_enabled	Booléen	N/A	
OutgoingCallerIdentification vocale	sid	String	N/A
	date_created	Datetime	N/A
	date_updated	Datetime	N/A
	account_sid	String	N/A
	friendly_name	String	=
	phone_number	String	=
	uri	String	N/A
Voice-Queue	date_created	Datetime	N/A
	date_updated	Datetime	N/A
	current_size	Entier	N/A
	friendly_name	String	N/A
	uri	String	N/A
	account_sid	String	N/A
	average_wait_time	Entier	N/A
	sid	String	N/A
	max_size	Entier	N/A
Conversations-Conversation	account_sid	String	N/A
	chat_service_sid	String	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	messaging_service_sid	String	N/A
	sid	String	N/A
	friendly_name	String	N/A
	unique_name	String	N/A
	attributs	String	N/A
	state	String	=
	date_created	Datetime	N/A
	date_updated	Datetime	N/A
	timers	Struct	N/A
	url	String	N/A
	links	Struct	N/A
	bindings	Struct	N/A
	start_date	Datetime	=
	end_date	Datetime	=
	Minuteries. DateInactive	String	N/A
	Minuteries. DateClosed	String	N/A
Conversations-User	sid	String	N/A
	account_sid	String	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	chat_service_sid	String	N/A
	role_sid	String	N/A
	une	String	N/A
	friendly_name	String	N/A
	attributs	String	N/A
	is_online	Booléen	N/A
	is_notifiable	Booléen	N/A
	date_created	Datetime	N/A
	date_updated	Datetime	N/A
	url	String	N/A
	links	Struct	N/A
Conversations-Role	sid	String	N/A
	account_sid	String	N/A
	chat_service_sid	String	N/A
	friendly_name	String	N/A
	type	String	N/A
	des autorisations	String	N/A
	date_created	Datetime	N/A
	date_updated	Datetime	N/A
	url	String	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
Conversations-Configuration	account_sid	Long	N/A
	default_chat_service_sid	String	N/A
	default_messaging_service_sid	String	N/A
	default_inactive_timer	String	N/A
	default_closed_timer	String	N/A
	url	String	N/A
	links	Map	N/A
Conversations-AddressConfiguration	sid	String	N/A
	account_sid	String	N/A
	type	String	N/A
	adresse	String	N/A
	friendly_name	String	N/A
	auto_creation	Struct	N/A
	date_created	Datetime	N/A
	date_updated	Datetime	N/A
	url	String	N/A
	address_country	String	N/A
	AutoCreation.Activé	Booléen	N/A
	AutoCreation.Type	String	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	AutoCreation.ConversationServiceSid	String	N/A
	AutoCreation.WebhookUrl	String	N/A
	AutoCreation.WebhookMethod	String	N/A
	AutoCreation.WebhookFilters	List	N/A
	AutoCreation.StudioFlowSid	String	N/A
	AutoCreation.StudioRetryCount	Entier	N/A
Conversations-WebhookConfiguration	account_sid	String	N/A
	méthode	String	N/A
	filtres	List	N/A
	pre_webhook_url	String	N/A
	post_webhook_url	String	N/A
	cible	String	N/A
	url	String	N/A
Conversations-ParticipantConversation	account_sid	String	N/A
	chat_service_sid	String	N/A
	participant_sid	String	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	participant_user_sid	String	N/A
	participant_identity	String	N/A
	participant_messaging_binding	Struct	N/A
	Conversation_sid	String	N/A
	conversation_unique_name	String	N/A
	conversation_friendly_name	String	N/A
	conversation_attributes	String	N/A
	conversation_date_created	Datetime	N/A
	conversation_date_updated	Datetime	N/A
	conversation_created_by	String	N/A
	conversation_state	String	N/A
	conversation_timers	Struct	N/A
	links	Map	N/A
	adresse	String	=
	une	String	=

Entité	Champ	Type de données	Opérateurs pris en charge
Conversation-Credentials	sid	String	N/A
	account_sid	String	N/A
	friendly_name	String	N/A
	type	String	N/A
	sandbox	String	N/A
	date_created	Datetime	N/A
	dated_updated	Datetime	N/A
	url	String	N/A
	certificat	String	N/A
	private_key	String	N/A
	api_key	String	N/A
	secret	String	N/A
Conversations-ConversationService	sid	String	N/A
	account_sid	String	N/A
	friendly_name	String	N/A
	date_created	Datetime	N/A
	date_updated	Datetime	N/A
	url	String	N/A
	links	Map	N/A

Requêtes de partitionnement

Champs prenant en charge le partitionnement :

Dans Twilio, les champs de type de DateTime données prennent en charge le partitionnement basé sur les champs.

Vous pouvez indiquer les options Spark supplémentaires `PARTITION_FIELD`, `LOWER_BOUND`, `UPPER_BOUND` et `NUM_PARTITIONS` si vous souhaitez utiliser la simultanéité dans Spark. Avec ces paramètres, la requête d'origine serait divisée en `NUM_PARTITIONS` nombres de sous-requêtes pouvant être exécutées simultanément par les tâches Spark.

- `PARTITION_FIELD` : le nom du champ à utiliser pour partitionner la requête.
- `LOWER_BOUND` : une valeur limite inférieure inclusive du champ de partition choisi.

  Pour le champ Datetime, nous acceptons le format d'horodatage Spark utilisé dans les requêtes SQL Spark.

  Exemples de valeurs valides :

  ```
  "2024-05-01T20:55:02.000Z"
  ```

- `UPPER_BOUND` : une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS` : le nombre de partitions.

Exemple :

```
twilio_read = glueContext.create_dynamic_frame.from_options(
    connection_type="twilio",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "sms-message",
        "API_VERSION": "2010-04-01",
        "PARTITION_FIELD": "date_sent"
        "LOWER_BOUND": "2024-05-01T20:55:02.000Z"
        "UPPER_BOUND": "2024-06-01T20:55:02.000Z"
        "NUM_PARTITIONS": "10"
    }
```

Options de connexion Twilio

Les options de connexion pour Twilio sont les suivantes :

- `ENTITY_NAME`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Le nom de votre objet dans Twilio.
- `EDGE_LOCATION`(Chaîne) - (Obligatoire) Un emplacement périphérique Twilio valide.
- `API_VERSION`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Version d'API Rest Twilio que vous souhaitez utiliser. Twilio prend en charge deux versions d'API : « v1 » et « 2010-04-01 ».
- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.
- `PARTITION_FIELD`(Chaîne) - Utilisé pour la lecture. Champ à utiliser pour partitionner la requête.
- `LOWER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite inférieure inclusive du champ de partition choisi.
- `UPPER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS`(Entier) - Par défaut : 1. Utilisé pour la lecture. Nombre de partitions pour la lecture.
- `INSTANCE_URL`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Une URL d'instance Twilio valide.

Restrictions et remarques relatives au connecteur Twilio

Voici les restrictions ou les remarques relatives au connecteur Twilio :

- Le partitionnement basé sur les enregistrements n'est pas pris en charge par le connecteur, car aucune disposition ne permet de récupérer le nombre total d'enregistrements de Twilio.
- Les champs `date_sent`, `start_time` et `end_time` sont du type Datetime, mais lors du filtrage, ils ne prennent en charge que les valeurs date (les composants temporels ne sont pas pris en compte).
- Le filtrage des champs « from » ou « to » ne fonctionne que si les valeurs n'incluent aucun préfixe (par exemple, un protocole ou une étiquette). Si un préfixe est présent, le filtrage du champ correspondant ne fonctionne pas. Par exemple, si vous transmettez « to » : « whatsapp :

+14xxxxxxxxxx » comme filtre, Twilio ne renverra aucune réponse. Vous devez le transmettre en tant que « to » : « +14xxxxxxxx », puis il renverra les enregistrements s'ils existent.

- Le filtre de champ « identity » est obligatoire lorsque vous interrogez l'entité `conversation-participant-conversation`.

Connexion à Vertica dans AWS Glue Studio

AWS Glue fournit une prise en charge intégrée pour Vertica. AWS Glue Studio fournit une interface visuelle pour se connecter à Vertica, créer des tâches d'intégration de données et les exécuter sur l'exécution Spark AWS Glue Studio sans serveur.

AWS Glue Studio crée une connexion unifiée pour Vertica. Pour de plus amples informations, veuillez consulter Considérations.

Rubriques

- Création d'une connexion Vertica
- Création d'un nœud source Vertica
- Création d'un nœud cible Vertica
- Options avancées

Création d'une connexion Vertica

Prérequis :

- Un compartiment ou un dossier Amazon S3 à utiliser pour le stockage temporaire lors de la lecture et de l'écriture dans la base de données, désigné par*`tempS3Path`*.

> **Note**
>
> Lorsque vous utilisez Vertica dans les aperçus des données de AWS Glue travail, il est possible que les fichiers temporaires ne soient pas automatiquement supprimés. *`tempS3Path`* Pour garantir la suppression des fichiers temporaires, mettez directement fin à la session de prévisualisation des données en choisissant Mettre fin à la session dans le volet Prévisualisation des données.
> Si vous ne pouvez pas garantir la fin directe de la session de prévisualisation des données, pensez à configurer le cycle de vie d'Amazon S3 pour supprimer les anciennes données.

> Nous recommandons de supprimer les données de plus de 49 heures, sur la base de la durée d'exécution maximale des tâches plus une marge. Pour de plus amples informations sur la configuration du cycle de vie Amazon S3, consultez Gestion du cycle de vie de votre stockage dans la documentation Amazon S3.

- Une politique IAM avec les autorisations appropriées pour votre chemin Amazon S3 que vous pouvez associer à votre rôle AWS Glue professionnel.
- Si votre instance Vertica se trouve dans un Amazon VPC, configurez Amazon VPC pour permettre à AWS Glue votre tâche de communiquer avec l'instance Vertica sans que le trafic ne transite par l'Internet public.

  Dans Amazon VPC, identifiez ou créez un VPC, un sous-réseau et un groupe de sécurité qui AWS Glue seront utilisés lors de l'exécution de la tâche. En outre, vous devez vous assurer qu'Amazon VPC est configuré pour autoriser le trafic réseau entre votre instance Vertica et cet emplacement. Votre tâche devra établir une connexion TCP avec votre port client Vertica (par défaut 5433). Selon la configuration de votre réseau, cela peut nécessiter des modifications des règles du groupe de sécurité, du réseau ACLs, des passerelles NAT et des connexions d'appairage.

Pour configurer une connexion à Vertica :

1. Dans AWS Secrets Manager, créez un secret à l'aide de vos informations d'identification Vertica, *`verticaUsername`* et*`verticaPassword`*. Pour créer un secret dans Secrets Manager, suivez le didacticiel disponible dans la section Créer un AWS Secrets Manager secret dans la AWS Secrets Manager documentation. Après avoir créé le secret, conservez le nom du secret *`secretName`* pour l'étape suivante.

   - Lorsque vous sélectionnez des paires clé/valeur, créez une paire pour la clé `user` contenant la valeur. *`verticaUsername`*
   - Lorsque vous sélectionnez des paires clé/valeur, créez une paire pour la clé `password` contenant la valeur. *`verticaPassword`*
2. Dans la AWS Glue console, créez une connexion en suivant les étapes décrites dansthe section called “Ajouter une AWS Glue connexion”. Après avoir créé la connexion, conservez le nom de la connexion pour l'étape suivante. *`connectionName`*

   - Lorsque vous sélectionnez un type de connexion, sélectionnez Vertica.
   - Lorsque vous sélectionnez l'hôte Vertica, indiquez le nome d'hôte de votre installation Vertica.

- Lorsque vous sélectionnez le port Vertica, le port via lequel votre installation Vertica est disponible.
- Lorsque vous sélectionnez un AWS secret, fournissez*secretName*.

3. Dans les situations suivantes, vous pouvez avoir besoin d'une configuration supplémentaire :

   - Pour les instances Vertica hébergées AWS dans un Amazon VPC
     - Fournissez les informations de connexion Amazon VPC à la AWS Glue connexion qui définit vos identifiants de sécurité Vertica. Lorsque vous créez ou mettez à jour votre connexion, définissez le VPC, le sous-réseau et les groupes de sécurité dans les options réseau.

Vous devez effectuer les étapes suivantes avant d'exécuter votre AWS Glue tâche :

- Accordez le rôle IAM associé à vos autorisations de AWS Glue travail à*tempS3Path*.
- Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire*secretName*.

Création d'un nœud source Vertica

Prérequis

- Une connexion au catalogue de AWS Glue données de type Vertica *connectionName* et un emplacement Amazon S3 temporaire*tempS3Path*, comme décrit dans la section précédente,the section called “Création d'une connexion Vertica”.
- Une table Vertica que vous souhaitez lire ou interroger*targetQuery*. *tableName*

Ajout d'une source de données Vertica

Pour ajouter un nœud Source de données – Vertica :

1. Choisissez la connexion pour votre source de données Vertica. Puisque vous l'avez créé, il devrait être disponible dans le menu déroulant. Si vous devez créer une connexion, choisissez Créer une connexion Vertica. Pour de plus amples informations, veuillez consulter la section précédente the section called “Création d'une connexion Vertica”.

   Une fois que vous avez choisi une connexion, vous pouvez afficher ses propriétés en cliquant sur Afficher les propriétés.

2. Choisissez la Base de données contenant votre table.

3. Choisissez la zone de transit dans Amazon S3, entrez un URI S3A pour*tempS3Path*.
4. Choisissez la Source Vertica.
   - Choisissez une seule table : accéder à toutes les données à partir d'une seule table.
   - Saisir une requête personnalisée : accéder à un jeu de données à partir de plusieurs tables en fonction de votre requête personnalisée.
5. Si vous avez choisi une seule table, entrez *tableName* et sélectionnez éventuellement un schéma.

   Si vous avez choisi Saisir une requête personnalisée, saisissez une requête SQL SELECT et sélectionnez éventuellement un Schéma.
6. Dans les Propriétés Vertica personnalisées, saisissez les paramètres et les valeurs nécessaires.

Création d'un nœud cible Vertica

Prérequis

- Une connexion au catalogue de AWS Glue données de type Vertica *connectionName* et un emplacement Amazon S3 temporaire*tempS3Path*, comme décrit dans la section précédente,the section called “Création d'une connexion Vertica”.

Ajout d'une cible de données Vertica

Pour ajouter un nœud Cible de données – Vertica :

1. Choisissez la connexion pour votre source de données Vertica. Puisque vous l'avez créé, il devrait être disponible dans le menu déroulant. Si vous devez créer une connexion, choisissez Créer une connexion Vertica. Pour de plus amples informations, veuillez consulter la section précédente the section called “Création d'une connexion Vertica”.

   Une fois que vous avez choisi une connexion, vous pouvez afficher ses propriétés en cliquant sur Afficher les propriétés.
2. Choisissez la Base de données contenant votre table.
3. Choisissez la zone de transit dans Amazon S3, entrez un URI S3A pour*tempS3Path*.
4. Entrez *tableName* et sélectionnez éventuellement un schéma.
5. Dans les Propriétés Vertica personnalisées, saisissez les paramètres et les valeurs nécessaires.

Options avancées

Vous pouvez fournir des options avancées lors de la création d'un nœud Vertica. Ces options sont les mêmes que celles disponibles lors de la programmation AWS Glue de scripts Spark.

Consultez the section called "Connexions Vertica".

Connexion à WooCommerce

WooCommerce est une solution logicielle flexible open source conçue pour les sites Web WordPress basés sur le Web. Elle est couramment utilisée pour créer des boutiques de commerce électronique en ligne. Avec cette solution logicielle, n'importe qui peut transformer son site Web habituel en un magasin en ligne entièrement fonctionnel.

Rubriques

- AWS Glue support pour WooCommerce
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration WooCommerce
- Configuration des WooCommerce connexions
- Lecture à partir d' WooCommerce entités
- WooCommerce options de connexion

AWS Glue support pour WooCommerce

AWS Glue prend en charge WooCommerce les éléments suivants :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser des tâches AWS Glue ETL pour interroger des données à partir de WooCommerce.

Prise en charge en tant que cible ?

Non.

Versions WooCommerce d'API prises en charge

Les versions WooCommerce d'API suivantes sont prises en charge :

- v3

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les autorisations AWS IAM requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Si vous ne voulez pas utiliser la méthode ci-dessus, vous pouvez également utiliser les politiques IAM gérées suivantes :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs

bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration WooCommerce

Avant de pouvoir AWS Glue transférer des données depuis WooCommerce, vous devez satisfaire aux exigences suivantes :

Configuration requise

Les exigences minimales sont les suivantes :

- Vous avez un WooCommerce compte avec a `consumerKey` et `consumerSecret` a.
- Votre WooCommerce compte dispose d'un accès à l'API avec une licence valide.

Si vous remplissez ces conditions, vous êtes prêt à vous connecter AWS Glue à votre WooCommerce compte. Pour les connexions classiques, vous n'avez rien d'autre à faire dans WooCommerce.

Configuration des WooCommerce connexions

WooCommerce prend en charge l'authentification personnalisée. Pour consulter WooCommerce la documentation publique sur la génération des clés d'API requises pour une autorisation personnalisée, consultez Authentication — Documentation de l' WooCommerce API REST.

Pour configurer une WooCommerce connexion :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes :
   - Pour une application connectée gérée par le client, le secret doit contenir le secret du consommateur de l'application connectée avec `consumerKey` et `consumerSecret` comme clés. Remarque : Vous devez créer un secret par connexion dans AWS Glue.

1. Dans AWS Glue Glue Studio, créez une connexion sous Data Connections en suivant les étapes ci-dessous :
   a. Lorsque vous sélectionnez un type de connexion, sélectionnez WooCommerce.
   b. Indiquez `INSTANCE_URL` l' WooCommerce instance à laquelle vous souhaitez vous connecter.
   c. Sélectionnez le rôle AWS IAM qui AWS Glue peut assumer et dispose des autorisations nécessaires pour les actions suivantes :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

d. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.

e. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.

2. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.

Lecture à partir d' WooCommerce entités

Prérequis

Un WooCommerce objet que vous aimeriez lire. Vous aurez besoin du nom de l'objet tel que coupon, order, product, etc.

Entités prises en charge pour la source :

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Coupon	Oui	Oui	Oui	Oui	Oui

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Coupon Total	Non	Non	Non	Oui	Non
Customers Total	Non	Non	Non	Oui	Non
Ordre	Oui	Oui	Oui	Oui	Oui
Orders Total	Non	Non	Non	Oui	Non
Payment Gateway	Non	Non	Non	Oui	Non
Produit (langue française non garantie)	Oui	Oui	Oui	Oui	Oui
Product attribute	Oui	Oui	Oui	Oui	Oui
Product category	Oui	Oui	Oui	Oui	Oui
Product review	Oui	Oui	Oui	Oui	Oui
Product shipping class	Oui	Oui	Oui	Oui	Oui
Product tag	Oui	Oui	Oui	Oui	Oui
Product variation	Oui	Oui	Oui	Oui	Oui

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Products Total	Non	Non	Non	Oui	Non
Report (List)	Non	Non	Non	Oui	Non
Reviews Total	Non	Non	Non	Oui	Non
Sales Report	Oui	Non	Non	Oui	Non
Shipping Method	Non	Non	Non	Oui	Non
Shipping Zone	Non	Non	Non	Oui	Non
Shipping Zone Location	Non	Non	Non	Oui	Non
Shipping Zone Method	Non	Non	Non	Oui	Non
Tax Rate	Oui	Oui	Oui	Oui	Oui
Tax Class	Non	Non	Non	Oui	Non
Top Sellers Report	Oui	Non	Non	Oui	Non

Exemple :

```
woocommerce_read = glueContext.create_dynamic_frame.from_options(
    connection_type="glue.spark.woocommerce",
    connection_options={
        "connectionName": "connectionName",
```

```
        "ENTITY_NAME": "coupon",
        "API_VERSION": "v3",
        "INSTANCE_URL": "instanceUrl"
    }
```

WooCommerce détails de l'entité et du champ :

Entité	Champ	Type de données	Opérateurs pris en charge
coupon	id	Entier	N/A
	code	String	EQUAL_TO
	montant	String	N/A
	status	String	N/A
	date_created	DateTime	N/A
	date_created_gmt	DateTime	N/A
	date_modified	DateTime	N/A
	date_modified_gmt	DateTime	N/A
	discount_type	String	N/A
	description	String	N/A
	date_expires	String	N/A
	date_expires_gmt	String	N/A
	usage_count	Entier	N/A
	individual_use	Booléen	N/A
	product_ids	List	N/A
	excluded_product_ids	List	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	usage_limit	Entier	N/A
	usage_limit_per_user	Entier	N/A
	limit_usage_to_x_items	Entier	N/A
	free_shipping	Booléen	N/A
	product_categories	List	N/A
	excluded_product_categories	List	N/A
	exclude_sale_items	Booléen	N/A
	minimum_amount	String	N/A
	maximum_amount	String	N/A
	email_restrictions	List	N/A
	used_by	List	N/A
	meta_data	List	N/A
	contexte	String	EQUAL_TO
	search	String	EQUAL_TO
	after	DateTime	EQUAL_TO
	before	DateTime	EQUAL_TO
	order	String	EQUAL_TO
	orderby	String	EQUAL_TO
	modified_after	DateTime	EQUAL_TO

Entité	Champ	Type de données	Opérateurs pris en charge
	modified_before	DateTime	EQUAL_TO
	dates_are_gmt	Booléen	EQUAL_TO
coupon-total	slug	String	N/A
	name	String	N/A
	total	Entier	N/A
customer-total	slug	String	N/A
	name	String	N/A
	total	Entier	N/A
order	id	Entier	N/A
	parent_id	Entier	N/A
	number	String	N/A
	order_key	String	N/A
	created_via	String	N/A
	status	String	N/A
	currency	String	N/A
	version	String	N/A
	date_created	DateTime	N/A
	date_modified	DateTime	N/A
	discount_total	String	N/A
	discount_tax	String	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	shipping_total	String	N/A
	shipping_tax	String	N/A
	cart_tax	String	N/A
	total	String	N/A
	total_tax	String	N/A
	prices_include_tax	Booléen	N/A
	customer_id	Entier	N/A
	customer_ip_address	String	N/A
	customer_user_agent	String	N/A
	customer_note	String	N/A
	facturation	Struct	N/A
	shipping	Struct	N/A
	payment_method	String	N/A
	payment_method_title	String	N/A
	transaction_id	String	N/A
	date_paid	DateTime	N/A
	date_completed	DateTime	N/A
	cart_hash	String	N/A
	meta_data	List	N/A
	line_items	List	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	tax_lines	List	N/A
	shipping_lines	List	N/A
	fee_lines	List	N/A
	coupon_lines	List	N/A
	refunds	List	N/A
	payment_url	String	N/A
	is_editable	Booléen	N/A
	needs_payment	Booléen	N/A
	needs_processing	Booléen	N/A
	date_created_gmt	DateTime	N/A
	date_modified_gmt	DateTime	N/A
	date_completed_gmt	DateTime	N/A
	date_paid_gmt	DateTime	N/A
	currency_symbol	String	N/A
	set_paid	Booléen	N/A
	contexte	String	EQUAL_TO
	search	String	EQUAL_TO
	after	DateTime	EQUAL_TO
	before	DateTime	EQUAL_TO
	order	String	EQUAL_TO

Entité	Champ	Type de données	Opérateurs pris en charge
	orderby	String	EQUAL_TO
	customer	Entier	EQUAL_TO
	product	Entier	EQUAL_TO
	dp	Entier	EQUAL_TO
	modified_before	DateTime	EQUAL_TO
	modified_after	DateTime	EQUAL_TO
	dates_are_gmt	Booléen	EQUAL_TO
order-total	slug	String	N/A
	name	String	N/A
	total	Entier	N/A
payment-gateway	title	String	N/A
	description	String	N/A
	order	String	N/A
	enabled	Booléen	N/A
	method_title	String	N/A
	method_description	String	N/A
	method_supports	List	N/A
	paramètres	String	N/A
	needs_setup	Booléen	N/A
	post_install_scripts	List	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	settings_url	String	N/A
	connection_url	String	N/A
	setup_help_text	String	N/A
	required_settings_keys	List	N/A
product	id	Entier	N/A
	name	String	N/A
	type	String	EQUAL_TO
	permalink	String	N/A
	date_created	DateTime	N/A
	date_created_gmt	DateTime	N/A
	date_modified	DateTime	N/A
	date_modified_gmt	DateTime	N/A
	catalog_visibility	String	N/A
	description	String	N/A
	short_description	String	N/A
	prix	String	N/A
	regular_price	String	N/A
	sale_price	String	N/A
	date_on_sale_from	DateTime	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	date_on_sale_from_gmt	DateTime	N/A
	date_on_sale_to	DateTime	N/A
	date_on_sale_to_gmt	DateTime	N/A
	price_html	String	N/A
	purchasable	Booléen	N/A
	total_sales	Entier	N/A
	virtual	Booléen	N/A
	downloadable	Booléen	N/A
	downloads	List	N/A
	download_limit	Entier	N/A
	download_expiry	Entier	N/A
	external_url	String	N/A
	button_text	String	N/A
	tax_status	String	N/A
	manage_stock	Booléen	N/A
	stock_quantity	Entier	N/A
	backorders	String	N/A
	backorders_allowed	Booléen	N/A
	backordered	Booléen	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	sold_individually	Booléen	N/A
	weight	String	N/A
	dimensions	Struct	N/A
	shipping_required	Booléen	N/A
	shipping_taxable	Booléen	N/A
	shipping_class_id	Entier	N/A
	reviews_allowed	Booléen	N/A
	average_rating	String	N/A
	rating_count	Entier	N/A
	related_ids	List	N/A
	upsell_ids	List	N/A
	cross_sell_ids	List	N/A
	parent_id	Entier	N/A
	purchase_note	String	N/A
	categories	List	N/A
	tags	List	N/A
	images	List	N/A
	attributs	List	N/A
	default_attributes	List	N/A
	variations	List	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	grouped_products	List	N/A
	menu_order	Entier	N/A
	meta_data	List	N/A
	low_stock_amount	Entier	N/A
	jetpack_publicize_connections	List	N/A
	jetpack-related-posts	List	N/A
	jetpack_likes_enabled	Booléen	N/A
	jetpack_sharing_enabled	Booléen	N/A
	contexte	String	EQUAL_TO
	search	String	EQUAL_TO
	after	DateTime	EQUAL_TO
	before	DateTime	EQUAL_TO
	order	String	EQUAL_TO
	orderby	String	EQUAL_TO
	slug	String	EQUAL_TO
	status	String	EQUAL_TO
	sku	String	EQUAL_TO
	featured	Booléen	EQUAL_TO
	balise	String	EQUAL_TO

Entité	Champ	Type de données	Opérateurs pris en charge
	shipping_class	String	EQUAL_TO
	tax_class	String	EQUAL_TO
	on_sale	Booléen	EQUAL_TO
	stock_status	String	EQUAL_TO
	has_options	Booléen	N/A
	modified_after	DateTime	EQUAL_TO
	modified_before	DateTime	EQUAL_TO
	dates_are_gmt	Booléen	EQUAL_TO
	category	String	EQUAL_TO
	attribute	String	EQUAL_TO
	min_price	String	EQUAL_TO
	max_price	String	EQUAL_TO
product-attribute	id	Entier	N/A
	name	String	N/A
	slug	String	N/A
	type	String	N/A
	order_by	String	N/A
	has_archives	Booléen	N/A
	contexte	String	EQUAL_TO
product-attribute-term	id	Entier	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	name	String	N/A
	slug	String	N/A
	description	String	N/A
	menu_order	Entier	N/A
	count	Entier	N/A
	contexte	String	EQUAL_TO
	search	String	EQUAL_TO
	order	String	EQUAL_TO
	orderby	String	EQUAL_TO
	hide_empty	Booléen	EQUAL_TO
	parent	Entier	EQUAL_TO
	product	Entier	EQUAL_TO
product-category	id	Entier	N/A
	name	String	N/A
	slug	String	EQUAL_TO
	description	String	N/A
	display	String	N/A
	image	Struct	N/A
	menu_order	Entier	N/A
	count	Entier	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	contexte	String	EQUAL_TO
	search	String	EQUAL_TO
	order	String	EQUAL_TO
	orderby	String	EQUAL_TO
	hide_empty	Booléen	EQUAL_TO
	parent	Entier	EQUAL_TO
	product	Entier	EQUAL_TO
product-review	id	Entier	N/A
	date_created	DateTime	N/A
	date_created_gmt	DateTime	N/A
	product_id	Entier	N/A
	product_name	String	N/A
	product_permalink	String	N/A
	review	String	N/A
	rating	Entier	N/A
	verified	Booléen	N/A
	reviewer	String	N/A
	reviewer_email	String	N/A
	reviewer_avatar_urls	Struct	N/A
	contexte	String	EQUAL_TO

Entité	Champ	Type de données	Opérateurs pris en charge
	search	String	EQUAL_TO
	after	DateTime	EQUAL_TO
	before	DateTime	EQUAL_TO
	order	String	EQUAL_TO
	orderby	String	EQUAL_TO
	status	String	EQUAL_TO
product-shipping-class	id	Entier	N/A
	name	String	N/A
	slug	String	EQUAL_TO
	description	String	N/A
	count	Entier	N/A
	contexte	String	EQUAL_TO
	search	String	EQUAL_TO
	order	String	EQUAL_TO
	orderby	String	EQUAL_TO
	hide_empty	String	EQUAL_TO
	product	Entier	EQUAL_TO
product-tag	id	Entier	N/A
	name	String	N/A
	slug	String	EQUAL_TO

Entité	Champ	Type de données	Opérateurs pris en charge
	description	String	N/A
	count	Entier	N/A
	contexte	String	EQUAL_TO
	search	String	EQUAL_TO
	order	String	EQUAL_TO
	orderby	String	EQUAL_TO
	hide_empty	Booléen	EQUAL_TO
	product	Entier	EQUAL_TO
product-total	slug	String	N/A
	name	String	N/A
	total	Entier	N/A
product-variation	id	Entier	N/A
	date_created	DateTime	N/A
	date_created_gmt	DateTime	N/A
	date_modified	DateTime	N/A
	date_modified_gmt	DateTime	N/A
	description	String	N/A
	permalink	String	N/A
	prix	String	N/A
	regular_price	String	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	sale_price	String	N/A
	date_on_sale_from	DateTime	N/A
	date_on_sale_from_gmt	DateTime	N/A
	date_on_sale_to	DateTime	N/A
	date_on_sale_to_gmt	DateTime	N/A
	purchasable	Booléen	N/A
	virtual	Booléen	N/A
	downloadable	Booléen	N/A
	downloads	List	N/A
	download_limit	Entier	N/A
	download_expiry	Entier	N/A
	tax_status	String	N/A
	manage_stock	Booléen	N/A
	stock_quantity	Entier	N/A
	backorders	String	N/A
	backorders_allowed	Booléen	N/A
	backordered	Booléen	N/A
	low_stock_amount	Entier	N/A
	weight	String	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	dimensions	Struct	N/A
	shipping_class	String	N/A
	shipping_class_id	Entier	N/A
	image	Struct	N/A
	attributs	List	N/A
	menu_order	Entier	N/A
	meta_data	List	N/A
	contexte	String	EQUAL_TO
	search	String	EQUAL_TO
	after	DateTime	EQUAL_TO
	before	DateTime	EQUAL_TO
	order	String	EQUAL_TO
	orderby	String	EQUAL_TO
	slug	String	EQUAL_TO
	status	String	EQUAL_TO
	sku	String	EQUAL_TO
	tax_class	String	EQUAL_TO
	on_sale	Booléen	EQUAL_TO
	min_price	String	EQUAL_TO
	max_price	String	EQUAL_TO

Entité	Champ	Type de données	Opérateurs pris en charge
	stock_status	String	EQUAL_TO
report	slug	String	N/A
	description	String	N/A
review-total	slug	String	N/A
	name	String	N/A
	total	Entier	N/A
sales-report	total_sales	String	N/A
	net_sales	String	N/A
	average_sales	String	N/A
	total_orders	Entier	N/A
	total_items	Entier	N/A
	total_tax	String	N/A
	total_shipping	String	N/A
	total_refunds	Entier	N/A
	total_discount	String	N/A
	totals_grouped_by	String	N/A
	totals	Struct	N/A
	total_customers	Entier	N/A
	contexte	String	EQUAL_TO
	point	String	EQUAL_TO

Entité	Champ	Type de données	Opérateurs pris en charge
	date_min	Date	EQUAL_TO
	date_max	Date	EQUAL_TO
shipping-method	id	String	N/A
	title	String	N/A
	description	String	N/A
shipping-zone	id	Entier	EQUAL_TO
	name	String	N/A
	order	Entier	N/A
shipping-zone-location	code	String	N/A
	type	String	N/A
shipping-zone-method	instance_id	Entier	N/A
	id	Entier	EQUAL_TO
	title	String	N/A
	order	Entier	N/A
	enabled	Booléen	N/A
	method_id	String	N/A
	method_title	String	N/A
	method_description	String	N/A
	paramètres	Struct	N/A
tax-class	slug	String	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	name	String	N/A
tax-rate	id	Entier	N/A
	country	String	N/A
	state	String	N/A
	postcode	String	N/A
	city	String	N/A
	postcodes	List	N/A
	cities	List	N/A
	rate	String	N/A
	name	String	N/A
	priority	Entier	N/A
	compound	Booléen	N/A
	shipping	Booléen	N/A
	contexte	String	EQUAL_TO
	order	String	EQUAL_TO
	orderby	String	EQUAL_TO
	class	String	EQUAL_TO
top-seller-report	name	String	N/A
	product_id	Entier	N/A
	quantity	Entier	N/A

Entité	Champ	Type de données	Opérateurs pris en charge
	contexte	String	EQUAL_TO
	point	String	EQUAL_TO
	date_min	Date	EQUAL_TO
	date_max	Date	EQUAL_TO

> **Note**
>
> Les types de données Struct et List sont convertis en type de données String, et le type de DateTime données est converti en Timestamp dans la réponse des connecteurs.

Requêtes de partitionnement

Partitionnement basé sur des enregistrements :

Vous pouvez indiquer l'option Spark supplémentaire `NUM_PARTITIONS` si vous souhaitez utiliser la simultanéité dans Spark. Avec ces paramètres, la requête d'origine serait divisée en `NUM_PARTITIONS` nombres de sous-requêtes pouvant être exécutées simultanément par les tâches Spark.

Dans le partitionnement basé sur les enregistrements, le nombre total d'enregistrements présents est demandé à partir de l' WooCommerce API et divisé par le nombre fourni. `NUM_PARTITIONS` Le nombre d'enregistrements qui en résulte est ensuite extrait simultanément par chaque sous-requête.

- `NUM_PARTITIONS` : le nombre de partitions.

Les entités suivantes prennent en charge le partitionnement basé sur les enregistrements :

- coupon
- order
- product
- product-attribute

- product-attribute-term
- product-category
- product-review
- product-shipping-class
- product-tag
- product-variation
- tax-rate

Exemple :

```
woocommerce_read = glueContext.create_dynamic_frame.from_options(
    connection_type="glue.spark.woocommerce",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "coupon",
        "API_VERSION": "v3",
        "INSTANCE_URL": "instanceUrl"
        "NUM_PARTITIONS": "10"
    }
```

Partitionnement basé sur des enregistrements :

La requête d'origine est divisée en NUM_PARTITIONS nombres de sous-requêtes qui peuvent être exécutées simultanément par les tâches Spark :

- NUM_PARTITIONS : le nombre de partitions.

Exemple :

```
WooCommerce_read = glueContext.create_dynamic_frame.from_options(
    connection_type="WooCommerce",
    connection_options={
        "connectionName": "connectionName",
        "REALMID": "1234567890123456789",
        "ENTITY_NAME": "Bill",
        "API_VERSION": "v3",
        "NUM_PARTITIONS": "10"
    }
```

WooCommerce options de connexion

Les options de connexion suivantes sont disponibles pour WooCommerce :

- `ENTITY_NAME`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Le nom de votre objet dans WooCommerce.
- `API_VERSION`(Chaîne) - (Obligatoire) Utilisé pour la lecture. WooCommerce Version de l'API Rest que vous souhaitez utiliser.
- `REALM_ID`(Chaîne) - Un identifiant identifiant une entreprise WooCommerce en ligne individuelle à laquelle vous envoyez des demandes.
- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.
- `INSTANCE_URL`(String) - (Obligatoire) Une URL d' WooCommerce instance valide au format : https ://<instance>.wpcomstaging.com
- `NUM_PARTITIONS`(Entier) - Par défaut : 1. Utilisé pour la lecture. Nombre de partitions pour la lecture.

Connexion à Zendesk

Zendesk est une solution de gestion de centre d'assistance cloud qui propose des outils personnalisables pour créer un portail de service client, une base de connaissances et des communautés en ligne.

Rubriques

- AWS Glue support pour Zendesk
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration de Zendesk
- Configuration de connexions Zendesk
- Lecture à partir d'entités Zendesk
- Options de connexion Zendesk

- Limitations

AWS Glue support pour Zendesk

AWS Glue prend en charge Zendesk comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser les tâches AWS Glue ETL pour interroger les données de Zendesk.

Prise en charge en tant que cible ?

Non.

Versions d'API Zendesk prises en charge

Les versions d'API Zendesk suivantes sont prises en charge :

- v2

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les autorisations AWS IAM requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
```

```
}
```

Si vous ne voulez pas utiliser la méthode ci-dessus, vous pouvez également utiliser les politiques IAM gérées suivantes :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration de Zendesk

Avant de pouvoir AWS Glue transférer des données depuis Zendesk, vous devez répondre aux exigences suivantes :

Configuration requise

Les exigences minimales sont les suivantes :

- Vous disposez d'un compte Zendesk. Pour de plus amples informations, veuillez consulter Création d'un compte Zendesk.
- Votre compte Zendesk est activé pour accéder à l'API.
- Votre compte Zendesk vous permet d'installer des applications connectées.

Si vous répondez à ces exigences, vous êtes prêt à vous connecter AWS Glue à votre compte Zendesk.

Création d'un compte Zendesk

Pour créer un compte Zendesk :

1. Accéder à l' https://www.zendesk.com/in/enregistrement/
2. Saisissez les informations telles que votre adresse e-mail professionnelle, votre prénom, votre nom de famille, votre numéro de téléphone, le titre du poste, le nom de l’entreprise, le nombre d’employés de l’entreprise, le mot de passe et la langue de votre choix. Choisissez ensuite Inscription complète à l’essai.
3. Une fois votre compte créé, utilisez le lien de vérification que vous avez reçu pour vérifier votre adresse e-mail.
4. Une fois l’adresse e-mail professionnelle vérifiée, vous êtes redirigé vers votre compte Zendesk. Choisissez l’option Acheter Zendesk pour votre plan préféré. Remarque : Pour le connecteur Zendesk, il est recommandé d’acheter le plan Suite Enterprise.

Création d'une application cliente et d'informations d'identification OAuth 2.0

Pour créer une application cliente et des informations d'identification OAuth 2.0 :

1. Connectez-vous à votre compte Zendesk où vous souhaitez que l'application OAuth 2.0 soit créée https://www.zendesk.com/in/ login/
2. Cliquez sur l’icône d’engrenage. Cliquez sur le lien Accéder au centre d’administration pour ouvrir la page du centre d’administration.
3. Choisissez Applications et intégrations dans la barre latérale gauche, puis sélectionnez APIs**>** API Zendesk.
4. Sur la page de l'API Zendesk, choisissez l'onglet OAuth Clients.
5. Choisissez Ajouter un client OAuth sur le côté droit.
6. Renseignez les champs suivants pour créer un client :
   a. Nom du client : saisissez un nom pour votre application. Il s’agit du nom que les utilisateurs verront lorsqu’on leur demandera d’autoriser l’accès à votre application et lorsqu’ils consulteront la liste des applications tierces ayant accès à leur application Zendesk.
   b. Description (facultatif). Brève description de votre application que les utilisateurs verront lorsqu’on leur demandera d’autoriser l’accès à celle-ci.
   c. Entreprise (facultatif). Le nom de l’entreprise que les utilisateurs verront lorsqu’ils seront invités à autoriser l’accès à votre application. Ces informations peuvent les aider à comprendre à qui ils accordent l’accès.
   d. Logo (facultatif). Il s’agit du logo que les utilisateurs verront lorsqu’on leur demandera d’autoriser l’accès à votre application. L’image peut être au format JPG, GIF ou PNG. Pour

obtenir de meilleurs résultats, importez une image carrée. Elle sera redimensionnée pour la page d’autorisation.

e. Identifiant unique : le champ est rempli automatiquement avec une version reformatée du nom que vous avez saisi pour votre application. Vous pouvez modifier les noms si vous le souhaitez.

f. Redirection URLs : entrez l'URL ou l'URL URLs que Zendesk doit utiliser pour envoyer la décision de l'utilisateur d'autoriser l'accès à votre application.

   Par exemple : https://us-east-1.console.aws.amazon.com/gluestudio/ oauth

7. Cliquez sur Sauvegarder
8. Une fois la page actualisée, un nouveau champ Secret prérempli apparaît en bas de la page. Il s'agit de la valeur « client_secret » spécifiée dans la OAuth2 spécification. Copiez la valeur Secret dans votre presse-papiers et enregistrez-la dans un endroit sûr. Remarque : Les caractères peuvent dépasser la largeur de la zone de texte. Assurez-vous donc de tout sélectionner avant de copier.
9. Cliquez sur Sauvegarder

Configuration de connexions Zendesk

Le connecteur Zendesk prend en charge le type d’autorisation Authorization Code.

- Ce type de subvention est considéré comme « à trois étapes » OAuth car il repose sur la redirection des utilisateurs vers un serveur d'autorisation tiers pour authentifier l'utilisateur. Il est utilisé lors de la création de connexions via la AWS Glue console. L'utilisateur qui crée une connexion peut par défaut s'appuyer sur une application connectée qu'il AWS Glue possède (application client AWS Glue gérée) pour laquelle il n'a pas besoin de fournir d'informations OAuth connexes, à l'exception de l'URL de son instance Zendesk. La AWS Glue console redirige l'utilisateur vers Zendesk où il doit se connecter et autoriser AWS Glue les autorisations demandées pour accéder à son instance Zendesk.
- Vous pouvez toujours choisir de créer votre propre application connectée dans Zendesk et de fournir votre propre identifiant client et votre secret client lors de la création de connexions via la AWS Glue console. Dans ce scénario, vous serez toujours redirigé vers Zendesk pour vous connecter et autoriser l'accès AWS Glue à vos ressources.
- Ce type d’autorisation donne lieu à un jeton d’accès. Le jeton d’accès n’expire jamais.

Pour consulter la documentation publique de Zendesk sur la création d'une application connectée pour le OAuth flux de code d'autorisation, consultez la section OAuth Tokens pour les types de subvention.

Pour configurer une connexion Zendesk :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes :
   a. Pour le type de AuthorizationCode subvention : pour une application connectée gérée par le client, le secret doit contenir le secret du consommateur de l'application connectée avec `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` comme clé.
   b. Remarque : vous devez créer un secret par connexion dans AWS Glue.
2. Dans AWS Glue Glue Studio, créez une connexion sous Data Connections en suivant les étapes ci-dessous :
   a. Lorsque vous sélectionnez un type de connexion, sélectionnez Zendesk.
   b. Indiquez l'INSTANCE_URL de l'instance Zendesk à laquelle vous souhaitez vous connecter.
   c. Indiquez l'environnement Zendesk.
   d. Sélectionnez le rôle AWS IAM qui AWS Glue peut assumer les actions suivantes et qui dispose des autorisations nécessaires pour les actions suivantes :

      JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
  ]
}
```

e. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.

f. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.

3. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.

Lecture à partir d'entités Zendesk

Prérequis

Un objet Zendesk à partir duquel vous souhaitez lire. Vous aurez besoin du nom de l'objet tel que ticket, user ou article, comme indiqué dans le tableau suivant.

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Ticket	Y	Y	Y	Y	N
Utilisateur	Y	Y	Y	Y	N
Organisation	Y	Y	Y	Y	N
Article	Y	Y	N	Y	N
Ticket Event	Y	Y	N	Y	N
Ticket Metric Event	Y	Y	N	Y	N
Ticket Comment	Y	Y	Y	Y	N
Ticket Field	Y	Y	N	Y	N
Ticket Metric	Y	Y	N	Y	N
Ticket Activity	Y	Y	N	Y	N
Ticket Skip	N	Y	N	Y	N

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Groupe	Y	Y	Y	Y	N
Group Membership	N	Y	Y	Y	N
Satisfaction Rating	Y	Y	N	Y	N
Afficher	Y	Y	Y	Y	N
Déclencheur	Y	Y	Y	Y	N
Trigger Category	N	Y	Y	Y	N
Macro	Y	Y	Y	Y	N
Automatis ation	N	Y	Y	Y	N

Exemple :

```
Zendesk_read = glueContext.create_dynamic_frame.from_options(
    connection_type="Zendesk",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "Account",
        "API_VERSION": "v2"
    }
```

Entités Zendesk et détails des champs :

Entité	Champ	Type de données	Opérateurs pris en charge	Commentaires
articles	url	String		
	id	Long		
	author_id	Long		
	body	String		
	comments_ disabled	Booléen		
	draft	Booléen		
	edited_at	DateTime		
	html_url	String		
	label_names	List		
	locale	String	EQUAL_TO	
	outdated	Booléen		
	outdated_locales	List		
	permissio n_group_id	Long		
	position	Entier		
	promoted	Booléen		
	section_id	Long		
	source_locale	String		
	name	String		

Entité	Champ	Type de données	Opérateurs pris en charge	Commentaires
	title	String		
	user_segment_id	Long		
	content_tags_id	List		
	vote_count	Entier		
	vote_sum	Entier		
	created_at	DateTime		
	updated_at	DateTime	EQUAL_TO	
	label_name	String	EQUAL_TO	
groupe	url	String		
	id	Long		
	is_public	Booléen		
	name	String		
	description	String		
	default	Booléen		
	deleted	Booléen		
	created_at	DateTime		
	updated_at	DateTime		
	exclude_deleted	Booléen	EQUAL_TO	
Automatisation	url	String		
	id	Long		

Entité	Champ	Type de données	Opérateurs pris en charge	Commentaires
	title	String		
	actif	Booléen		
	created_at	DateTime		
	updated_at	DateTime		
	default	Booléen		
	actions	List		
	positions	Entier		
	conditions	Struct		
	raw_title	String		
group-membership	url	String		
	id	Long		
	user_id	Long		
	group_id	Long		
	default	Booléen		
	created_at	DateTime		
	updated_at	DateTime		
macro	url	String		
	id	Long		
	title	String		
	actif	Booléen	EQUAL_TO	

Entité	Champ	Type de données	Opérateurs pris en charge	Commentaires
	created_at	DateTime		
	updated_at	DateTime		
	default	Booléen		
	actions	List		
	position	Entier		
	description	String		
	raw_title	String		
	restriction	Struct		
	accès	String	EQUAL_TO	
	category	Entier	EQUAL_TO	
	group_id	Long	EQUAL_TO	
	only_viewable	Booléen	EQUAL_TO	
organisations	url	String		
	id	Long		
	external_id	String		
	name	String		
	domain_names	List		
	détails	String		
	notes	String		
	group_id	Long		

Entité	Champ	Type de données	Opérateurs pris en charge	Commentaires
	shared_tickets	Booléen		
	shared_comments	Booléen		
	tags	List		
	organization_fields	Struct		
	created_at	DateTime		
	updated_at	DateTime	EQUAL_TO	
	DML_STATUS	String		Champ défini par l'utilisateur utilisé pour suivre l'état créé, mis à jour et supprimé de l'enregistrement.
satisfaction-rating	url	String		
	id	Long		
	assignee_id	Long		
	comment	String		
	group_id	Long		
	raison	String		
	reason_code	Entier		
	reason_id	Long		
	requester_id	Long		

Entité	Champ	Type de données	Opérateurs pris en charge	Commentaires
	score	String	EQUAL_TO	
	ticket_id	Entier		
	created_at	DateTime		
	updated_at	DateTime	EQUAL_TO	
	start_time	DateTime	EQUAL_TO	
	end_time	DateTime	EQUAL_TO	
	DML_STATUS	String		Champ défini par l'utilisateur utilisé pour suivre l'état créé, mis à jour et supprimé de l'enregistrement.
ticket-activity	acteur	Struct		
	actor_id	Long		
	created_at	DateTime		
	id	Long		
	objet	Struct		
	cible	Struct		
	title	String		
	updated_at	DateTime		
	url	String		
	user	Struct		

Entité	Champ	Type de données	Opérateurs pris en charge	Commentaires
	user_id	Long		
	verb	String		
	since	DateTime	EQUAL_TO	
ticket-comment	id	Long		
	type	String		
	author_id	Long		
	body	String		
	html_body	String		
	plain_body	String		
	public	Booléen		
	attachments	List		
	audit_id	Long		
	via	Struct		
	created_at	DateTime		
	metadata	Struct		
	ticket_id	Entier	EQUAL_TO	
	include_inline_images	Booléen	EQUAL_TO	
ticket-events	id	Long		
	ticket_id	Long		

Entité	Champ	Type de données	Opérateurs pris en charge	Commentaires
	timestamp	Long		
	created_at	DateTime		
	updater_id	Long		
	child_events	List		
	via	String		
	system	Struct		
	event_type	String		
	comment_p resent	Booléen		
	comment_public	Booléen		
	via_reference_id	Long		
	created_at	DateTime	EQUAL_TO	
	DML_STATUS	String		Champ défini par l'utilisateur utilisé pour suivre l'état créé, mis à jour et supprimé de l'enregistrement.
ticket-field	url	String		
	id	Long		
	type	String		
	title	String		

Entité	Champ	Type de données	Opérateurs pris en charge	Commentaires
	raw_title	String		
	description	String		
	raw_description	String		
	position	Entier		
	actif	Booléen		
	obligatoire	Booléen		
	collapsed_for_agents	Booléen		
	regexp_for_validation	String		
	title_in_portal	String		
	raw_title_in_portal	String		
	visible_in_portal	Booléen		
	editable_on_portal	Booléen		
	required_in_portal	Booléen		
	balise	String		
	created_at	DateTime		
	updated_at	DateTime		
	removable	Booléen		

Entité	Champ	Type de données	Opérateurs pris en charge	Commentaires
	agent_description	String		
	custom_field_options	List		
	custom_statuses	List		
	relationship_filter	Struct		
	relationship_target_type	String		
	sub_type_id	Entier		
	system_field_options	List		
	locale	String	EQUAL_TO	
ticket-metric-events	id	Long		
	time	DateTime	EQUAL_TO	
	ticket_id	Entier		
	métrique	String		
	instance_id	Entier		
	type	String		

Entité	Champ	Type de données	Opérateurs pris en charge	Commentaires
	DML_STATUS	String	EQUAL_TO	Champ défini par l'utilisateur utilisé pour suivre l'état créé, mis à jour et supprimé de l'enregistrement.
ticket-metric	url	String		
	id	Long		
	ticket_id	Entier		
	created_at	DateTime		
	updated_at	DateTime		
	group_stations	Entier		
	assignee_ stations	Entier		
	reopens	Entier		
	replies	Entier		
	assignee_ updated_at	DateTime		
	requester _updated_at	DateTime		
	initially_assigned _at	DateTime		
	assigned_at	DateTime		

Entité	Champ	Type de données	Opérateurs pris en charge	Commentaires
	solved_at	DateTime		
	last_comment_added_at	DateTime		
	reply_time_in_minutes	Struct		
	first_resolution_time_in_minutes	Struct		
	full_resolution_time_in_minutes	Struct		
	agent_wait_time_in_minutes	Struct		
	requester_wait_time_in_minutes	Struct		
	on_hold_time_in_seconds	Struct		
	reply_time_in_seconds	Struct		
	custom_status_updated_at	DateTime		
ticket-skip	created_at	DateTime		
	id	Long		
	raison	String		

Entité	Champ	Type de données	Opérateurs pris en charge	Commentaires
	ticket	Struct		
	ticket_id	Entier		
	updated_at	DateTime		
	user_id	Long		
billets	url	String		
	id	Long		
	external_id	String	EQUAL_TO	
	type	String		
	subject	String		
	raw_subject	String		
	description	String		
	priority	String		
	status	String		
	destinataire	String		
	demandeur	Struct		
	requester_id	Long		
	submitter_id	Long		
	assignee_id	Long		
	organization_id	Long		
	group_id	Long		

Entité	Champ	Type de données	Opérateurs pris en charge	Commentaires
	collaborator_ids	List		
	emails_cc_ids	List		
	follower_ids	List		
	forum_topic_id	Ling		
	problem_id	Long		
	has_incidents	Booléen		
	due_at	DateTime		
	tags	List		
	via	Struct		
	custom_fields	List		
	satisfaction_rating	Struct		
	sharing_agreement_ids	List		
	followup_ids	List		
	via_followup_source_id	Long		
	ticket_form_id	Long		
	brand_id	Long		
	allow_channelback	Booléen		

Entité	Champ	Type de données	Opérateurs pris en charge	Commentaires
	allow_attachments	Booléen		
	is_public	Booléen		
	from_messaging_channel	Booléen		
	created_at	DateTime		
	updated_at	DateTime	EQUAL_TO	
	assignee_email	String		
	attribute_value_ids	List		
	collaborators	List		
	comment	Struct		
	custom_status_id	Long		
	email_ccs	Struct		
	followers	Struct		
	macro_id	Long		
	macros_ids	List		
	metadata	Struct		
	safe_update	Booléen		
	updated_stamp	DateTime		
	via_id	Long		

Entité	Champ	Type de données	Opérateurs pris en charge	Commentaires
	voice_comment	Struct		
	DML_STATUS	String		Champ défini par l'utilisateur utilisé pour suivre l'état créé, mis à jour et supprimé de l'enregistrement.
trigger-category	url	String		
	id	String		
	name	String		
	updated_at	DateTime		
	created_at	DateTime		
	position	Entier		
déclencheur	url	String		
	id	Long		
	title	String		
	actif	Booléen	EQUAL_TO	
	updated_at	DateTime		
	created_at	DateTime		
	default	Booléen		
	actions	List		
	conditions	Struct		

Entité	Champ	Type de données	Opérateurs pris en charge	Commentaires
	description	String		
	position	Entier		
	raw_title	String		
	category_id	String	EQUAL_TO	
users	url	String		
	id	Long		
	external_id	String	EQUAL_TO	
	e-mail	String		
	actif	Booléen		
	alias	String		
	chat_only	Booléen		
	custom_roll_id	Long		
	roll_type	Entier		
	détails	String		
	last_login_at	DateTime		
	locale	String		
	locale_id	Entier		
	moderator	Booléen		
	notes	String		
	name	String		

Entité	Champ	Type de données	Opérateurs pris en charge	Commentaires
	only_private_comments	Booléen		
	organization_id	Long		
	default_group_id	Long		
	phone	String		
	photo	Struct		
	remote_photo_url	String		
	restricted_agent	Booléen		
	rôle	String	EQUAL_TO	
	shared	Booléen		
	shared_agent	Booléen		
	balise	List		
	Signature	String		
	suspended	Booléen		
	ticket_restriction	String		
	time_zone	String		
	iana_time_zone			
	two_factor_auth_enabled			
	user_fields			

Entité	Champ	Type de données	Opérateurs pris en charge	Commentaires
	verified	Booléen		
	report_csv	Booléen		
	created_at	DateTime		
	updated_at	DateTime	EQUAL_TO	
	permission_set	Long	EQUAL_TO	
	shared_phone_number	Booléen		
	DML_STATUS	String		Champ défini par l'utilisateur utilisé pour suivre l'état créé, mis à jour et supprimé de l'enregistrement.
afficher	url	String		
	id	Long		
	title	String		
	actif	Booléen	EQUAL_TO	
	updated_at	DateTime		
	created_at	DateTime		
	default	Booléen		
	position	Entier		
	description	String		

Entité	Champ	Type de données	Opérateurs pris en charge	Commentaires
	execution	Struct		
	restriction	Struct		
	raw_title	String		
	conditions	Struct		
	accès	String	EQUAL_TO	
	group_id	Long	EQUAL_TO	

> **Note**
>
> Les types de données Struct et List sont convertis en type de données String dans la réponse du connecteur.

Requêtes de partitionnement

Les partitions ne sont pas prises en charge dans Zendesk.

Options de connexion Zendesk

Les options de connexion pour Zendesk sont les suivantes :

- `ENTITY_NAME`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Le nom de votre objet dans Zendesk.
- `API_VERSION`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Version d'API Rest Zendesk que vous souhaitez utiliser. Par exemple : v2.
- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet. Par exemple : id, name, url, created_at
- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark. Par exemple : group_id = 100
- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète. Par exemple : "SELECT id,url FROM users WHERE role=\"end-user\""

- PARTITION_FIELD(Chaîne) - Utilisé pour la lecture. Champ à utiliser pour partitionner la requête. Le champ par défaut est update_at pour les entités prenant en charge l'API d'exportation incrémentielle (created_at pour ticket-events et time pour ticket-metric-events).
- LOWER_BOUND(Chaîne) - Utilisé pour la lecture. Une valeur limite inférieure inclusive du champ de partition choisi.
- UPPER_BOUND(Chaîne) - Utilisé pour la lecture. Une valeur limite supérieure exclusive du champ de partition choisi. Facultatif : cette option sera gérée par le connecteur si elle n'est pas fournie dans l'option de tâche. Valeur par défaut : "2024-05-01T20:55:02.000Z
- NUM_PARTITIONS(Entier) - Par défaut : 1. Utilisé pour la lecture. Nombre de partitions pour la lecture. Facultatif : cette option sera gérée par le connecteur si elle n'est pas fournie dans l'option de tâche. Valeur par défaut : 1.
- IMPORT_DELETED_RECORDS(Chaîne) - Par défaut : FALSE. Utilisé pour la lecture. Pour obtenir les enregistrements supprimés lors de la requête.
- ACCESS_TOKEN : jeton d'accès à utiliser dans la demande.
- INSTANCE_URL : URL de l'instance sur laquelle l'utilisateur souhaite exécuter les opérations. Par exemple : https://{subdomain}/.zendesk.com

Limitations

Les restrictions du connecteur Zendesk sont les suivantes :

- La pagination basée sur le décalage limite le nombre de pages pouvant être extraites à 100, mais cela n'est pas recommandé, car le nombre total d'enregistrements pouvant être extraits est de 10 000. Cependant, la pagination basée sur un curseur mise en œuvre pour le connecteur Zendesk permet de contourner cette restriction. Seul l'opérateur de filtre EQUAL_TO est pris en charge par l'API Zendesk.

  En raison de cette restriction, le partitionnement n'est pas pris en charge pour le connecteur Zendesk.
- Pour l'entité « Ticket Event », la limite de débit est de 10 demandes par minute. Lors de l'exécution d'une tâche AWS Glue ETL, vous pouvez recevoir un message d'erreur 429 (trop de demandes).

Connexion à Zoho CRM

Zoho CRM fait office de référentiel unique pour regrouper les activités de vente, de marketing et de support client, et rationaliser les processus, les politiques et les personnes dans une seule plateforme. Zoho CRM peut être facilement personnalisé pour répondre aux besoins spécifiques des entreprises de tout type et de toute taille.

La plateforme de développement de Zoho CRM offre la bonne combinaison d'outils low-code et pro-code pour automatiser le travail, intégrer les données businesses/enterprises au sein de l'entreprise et créer des solutions personnalisées pour le Web et le mobile.

Rubriques

- AWS Glue support pour Zoho CRM
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration de Zoho CRM
- Configuration de connexions Zoho CRM
- Lecture depuis les entités Zoho CRM
- Options de connexion Zoho CRM
- Restrictions et remarques relatives au connecteur Zoho CRM

AWS Glue support pour Zoho CRM

AWS Glue prend en charge Zoho CRM comme suit :

Prise en charge en tant que source ?

Oui : synchrone et asynchrone. Vous pouvez utiliser les tâches AWS Glue ETL pour interroger les données de Zoho CRM.

Prise en charge en tant que cible ?

Non.

Versions d'API Zoho CRM prises en charge

Les versions d'API Zoho CRM suivantes sont prises en charge :

- v7

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les autorisations AWS IAM requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Si vous ne voulez pas utiliser la méthode ci-dessus, vous pouvez également utiliser les politiques IAM gérées suivantes :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration de Zoho CRM

Avant de pouvoir AWS Glue transférer des données depuis Zoho CRM, vous devez répondre aux exigences suivantes :

Configuration requise

Les exigences minimales sont les suivantes :

- Vous possédez un compte Zoho CRM.
- Votre compte Zoho CRM est activé pour accéder à l'API.
- Vous avez un client d'API enregistré dans la console API pour obtenir des OAuth informations d'identification.

Configuration de connexions Zoho CRM

Le type de subvention détermine le mode AWS Glue de communication avec Zoho CRM pour demander l'accès à vos données. Votre choix influe sur les exigences que vous devez respecter avant de créer la connexion. Zoho CRM prend uniquement en charge le type de subvention AUTHORIZATION_CODE pour la version 2.0. OAuth

- Ce type de subvention est considéré comme « à trois étapes » OAuth car il repose sur la redirection des utilisateurs vers un serveur d'autorisation tiers pour authentifier l'utilisateur. Il est utilisé lors de la création de connexions via la AWS Glue console. La AWS Glue console redirige l'utilisateur vers Zoho CRM où il doit se connecter et accorder à Glue les autorisations demandées pour accéder à son instance Zoho CRM.
- Les utilisateurs peuvent toujours choisir de créer leur propre application connectée dans Zoho CRM et de fournir leur propre identifiant client, URL d'authentification, URL de jeton et URL d'instance lors de la création de connexions via la AWS Glue console. Dans ce scénario, ils seront toujours redirigés vers Zoho CRM pour se connecter et autoriser l'accès AWS Glue à leurs ressources.
- Ce type d'autorisation donne lieu à un jeton d'actualisation et à un jeton d'accès. Le jeton d'accès restera valide pendant une heure et pourra être actualisé automatiquement sans interaction de l'utilisateur à l'aide du jeton d'actualisation.
- Pour consulter la documentation publique de Zoho CRM sur la création d'une application connectée pour le OAuth flux de code d'autorisation, consultez Authentification.

Pour configurer une connexion Zoho CRM :

1. Dans AWS Glue Glue Studio, créez une connexion sous Data Connections en suivant les étapes ci-dessous :
   a. Lorsque vous sélectionnez un type de connexion, sélectionnez Zoho CRM.
   b. Indiquez l'`INSTANCE_URL` de l'instance Zoho CRM à laquelle vous souhaitez vous connecter.
   c. Indiquez l'ID client de l'application du client utilisateur.
   d. Sélectionnez l'URL d'authentification appropriée dans le menu déroulant.
   e. Sélectionnez l'URL du jeton appropriée dans le menu déroulant.
   f. Sélectionnez le rôle AWS IAM qui AWS Glue peut assumer et dispose des autorisations nécessaires pour les actions suivantes :

      JSON

      ```
      {
        "Version":"2012-10-17",
        "Statement": [
          {
            "Effect": "Allow",
            "Action": [
              "secretsmanager:DescribeSecret",
              "secretsmanager:GetSecretValue",
              "secretsmanager:PutSecretValue",
              "ec2:CreateNetworkInterface",
              "ec2:DescribeNetworkInterfaces",
              "ec2:DeleteNetworkInterface"
            ],
            "Resource": "*"
          }
        ]
      }
      ```

   g. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.
   h. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.
2. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.
3. Dans la configuration de votre AWS Glue tâche, fournissez `connectionName` une connexion réseau supplémentaire.

Lecture depuis les entités Zoho CRM

Prérequis

Objets Zoho CRM à partir desquels vous souhaitez lire. Vous aurez besoin du nom de l'objet.

Entités prises en charge pour la source de synchronisation :

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Produit (langue française non garantie)	Oui	Oui	Oui	Oui	Oui
Quote	Oui	Oui	Oui	Oui	Oui
Bon de commande	Oui	Oui	Oui	Oui	Oui
Solution	Oui	Oui	Oui	Oui	Oui
Appelez	Oui	Oui	Oui	Oui	Oui
Sous-tâche	Oui	Oui	Oui	Oui	Oui
Événement	Oui	Oui	Oui	Oui	Oui
Invoice	Oui	Oui	Oui	Oui	Oui
Compte	Oui	Oui	Oui	Oui	Oui
Contact	Oui	Oui	Oui	Oui	Oui
Vendor	Oui	Oui	Oui	Oui	Oui
Campaign	Oui	Oui	Oui	Oui	Oui
Deal	Oui	Oui	Oui	Oui	Oui

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Lead	Oui	Oui	Oui	Oui	Oui
Custom Module	Oui	Oui	Oui	Oui	Oui
Sales Order	Oui	Oui	Oui	Oui	Oui
Price Books	Oui	Oui	Oui	Oui	Oui
Cas	Oui	Oui	Oui	Oui	Oui

Exemple :

```
zoho_read = glueContext.create_dynamic_frame.from_options(
    connection_type="ZOHO",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "API_VERSION": "v7",
        "INSTANCE_URL": "https://www.zohoapis.in/"
    }
```

Entités prises en charge pour la source asynchrone :

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Produit (langue française non garantie)	Oui	Non	Non	Oui	Non
Quote	Oui	Non	Non	Oui	Non

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Bon de commande	Oui	Non	Non	Oui	Non
Solution	Oui	Non	Non	Oui	Non
Appelez	Oui	Non	Non	Oui	Non
Sous-tâche	Oui	Non	Non	Oui	Non
Événement	Oui	Non	Non	Oui	Non
Invoice	Oui	Non	Non	Oui	Non
Compte	Oui	Non	Non	Oui	Non
Contact	Oui	Non	Non	Oui	Non
Vendor	Oui	Non	Non	Oui	Non
Campaign	Oui	Non	Non	Oui	Non
Deal	Oui	Non	Non	Oui	Non
Lead	Oui	Non	Non	Oui	Non
Custom Module	Oui	Non	Non	Oui	Non
Sales Order	Oui	Non	Non	Oui	Non
Price Books	Oui	Non	Non	Oui	Non
Cas	Oui	Non	Non	Oui	Non

Exemple :

```
zoho_read = glueContext.create_dynamic_frame.from_options(
```

```
    connection_type="ZOHO",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "API_VERSION": "v7",
        "INSTANCE_URL": "https://www.zohoapis.in/",
        "TRANSFER_MODE": "ASYNC"
    }
```

Détails du champ Zoho CRM :

Zoho CRM fournit des points de terminaison permettant de récupérer les métadonnées de manière dynamique pour les entités prises en charge. Par conséquent, la prise en charge des opérateurs est capturée au niveau du type de données.

Entité	Type de données	Opérateurs pris en charge
Entités Zoho (toutes les entités)	Entier	!=, =, <, <=, >, >=, BETWEEN
	String	Like, =, !=
	BigInteger	!=, =, <, <=, >, >=, BETWEEN
	Booléen	=
	Double	!=, =, <, <=, >, >=, BETWEEN
	BigDecimal	!=, =, <, <=, >, >=, BETWEEN
	Date	!=, =, <, <=, >, >=, BETWEEN
	DateTime	!=, =, <, <=, >, >=, BETWEEN
	Struct	N/A
	List	N/A

Requêtes de partitionnement

Le partitionnement n'est pas pris en charge en mode asynchrone.

Partitionnement basé sur des filtres (mode de synchronisation) :

Vous pouvez indiquer les options Spark supplémentaires `PARTITION_FIELD`, `LOWER_BOUND`, `UPPER_BOUND` et `NUM_PARTITIONS` si vous souhaitez utiliser la simultanéité dans Spark. Avec ces paramètres, la requête d'origine serait divisée en `NUM_PARTITIONS` nombres de sous-requêtes pouvant être exécutées simultanément par les tâches Spark.

- `PARTITION_FIELD` : le nom du champ à utiliser pour partitionner la requête.
- `LOWER_BOUND` : une valeur limite inférieure inclusive du champ de partition choisi.

  Pour le champ Datetime, nous acceptons le format d'horodatage Spark utilisé dans les requêtes SQL Spark.

  Exemples de valeurs valides :

  ```
  "2024-09-30T01:01:01.000Z"
  ```

- `UPPER_BOUND` : une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS` : le nombre de partitions.

Exemple :

```
zoho_read = glueContext.create_dynamic_frame.from_options(
    connection_type="zohocrm",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "entityName",
        "API_VERSION": "v7",
        "PARTITION_FIELD": "Created_Time"
        "LOWER_BOUND": "2022-01-01T01:01:01.000Z"
        "UPPER_BOUND": "2024-01-01T01:01:01.000Z"
        "NUM_PARTITIONS": "10"
    }
```

Options de connexion Zoho CRM

Les options de connexion pour Zoho CRM sont les suivantes :

- `ENTITY_NAME`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Le nom de votre objet dans Zoho CRM.

- `API_VERSION`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Version d'API Rest Zoho CRM que vous souhaitez utiliser.
- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Colonnes que vous souhaitez sélectionner pour l'objet.
- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.
- `QUERY`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.
- `PARTITION_FIELD`(Chaîne) - Utilisé pour la lecture. Champ à utiliser pour partitionner la requête.
- `LOWER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite inférieure inclusive du champ de partition choisi.
- `UPPER_BOUND`(Chaîne) - Utilisé pour la lecture. Une valeur limite supérieure exclusive du champ de partition choisi.
- `NUM_PARTITIONS`(Entier) - Par défaut : 1. Utilisé pour la lecture. Nombre de partitions pour la lecture.
- `INSTANCE_URL`(Chaîne) - (Obligatoire) Utilisé pour la lecture. URL d'instance Zoho CRM valide.
- `TRANSFER_MODE`(Chaîne) - Utilisé pour indiquer si la requête doit être exécutée en mode asynchrone.

Restrictions et remarques relatives au connecteur Zoho CRM

Voici les restrictions ou les remarques relatives au connecteur Zoho CRM :

- Avec la version 7 de l'API, vous pouvez récupérer un maximum de 100 000 enregistrements. Consultez la documentation Zoho.
- Pour l'entité Event, l'étiquette « Meeting » s'affiche comme indiqué dans la documentation Zoho.
- Pour la fonctionnalité Tout sélectionner :
  - Vous pouvez récupérer un maximum de 50 champs depuis SaaS pour les appels GET et POST.
  - Si vous souhaitez avoir des données pour un champ spécifique qui n'appartient pas aux 50 premiers champs, vous devez indiquer manuellement la liste des champs sélectionnés.
  - Si plus de 50 champs sont sélectionnés, tous les champs au-delà des 50 champs seront rognés et contiendront des données nulles dans Amazon S3.
  - Dans le cas d'une expression de filtre, si la liste de 50 champs fournie par l'utilisateur n'inclut pas les champs « id » et « Created_Time », une exception personnalisée sera déclenchée pour inviter l'utilisateur à les inclure.

- Les opérateurs de filtre peuvent être différents field-to-field même s'ils ont le même type de données. Par conséquent, vous devez spécifier manuellement un opérateur différent pour chaque champ qui déclenche une erreur dans la plateforme SaaS.
- Pour la fonctionnalité Trier par :
  - Les données ne peuvent être triées que selon un seul champ sans expression de filtre, tandis que les données peuvent être triées selon plusieurs champs lorsqu'une expression de filtre est appliquée.
  - Si aucun ordre de tri n'est spécifié pour le champ sélectionné, les données seront extraites par ordre croissant par défaut.
- Les régions prises en charge par le connecteur Zoho CRM sont les États-Unis, l'Europe, l'Inde, l'Australie et le Japon.
- Restrictions de la fonctionnalité de lecture asynchrone :
  - Limit, Order by et le partitionnement ne sont pas pris en charge en mode asynchrone.
  - En mode asynchrone, nous pouvons transférer jusqu'à 500 pages de données avec 200 000 enregistrements par page.
  - Pendant un intervalle d'une minute, seules 10 demandes sont autorisées à être téléchargées. Lorsque vous dépassez la limite de téléchargement, le système renvoie une erreur HTTP 429 et suspend toutes les demandes de téléchargement pendant une minute avant que le traitement ne puisse reprendre.
  - Une fois la tâche groupée terminée, vous ne pouvez accéder au fichier téléchargeable que pendant une journée. Ensuite, vous ne pouvez plus accéder au fichier via les points de terminaison.
  - Un maximum de 200 champs de sélection peuvent être fournis via un point de terminaison. Si vous spécifiez plus de 200 champs de sélection dans un point de terminaison, le système exportera automatiquement tous les champs disponibles pour ce module.
  - Les champs externes créés dans un module ne sont pas pris en charge dans Bulk Read APIs.
  - Le tri et les clauses `Group_by` ne sont pas pris en charge via ce point de terminaison d'API.
  - Les valeurs des champs contenant des données d'intégrité sensibles ne seront récupérées que lorsque l'option Restreindre l'accès aux données via l'API dans les paramètres de conformité est désactivée. Si l'option est activée, la valeur sera vide dans le résultat.
  - Limites de filtrage/critères
    - Le nombre maximum de critères qu'une requête peut utiliser est de 25.
    - Le filtrage/les critères sur les champs de texte à lignes multiples ne sont pas pris en charge.

Connexion à Zoom Meetings

Zoom Meetings est une plateforme de visioconférence cloud qui peut être utilisée pour les réunions par visioconférence, les audioconférences, les webinaires, les enregistrements de réunions et le chat en direct.

Rubriques

- AWS Glue support pour Zoom Meetings
- Politiques contenant les opérations API pour la création et l'utilisation des connexions
- Configuration de Zoom Meetings
- Configuration de l'application client Zoom Meetings
- Configuration des connexions Zoom Meetings
- Lecture depuis les entités de Zoom Meetings
- Options de connexion Zoom Meetings
- Restrictions de Zoom Meetings

AWS Glue support pour Zoom Meetings

AWS Glue prend en charge Zoom Meetings comme suit :

Prise en charge en tant que source ?

Oui. Vous pouvez utiliser les tâches AWS Glue ETL pour interroger les données de Zoom Meetings.

Prise en charge en tant que cible ?

Non.

Versions d'API Zoom Meetings prises en charge

Les versions d'API Zoom Meetings suivantes sont prises en charge :

- v2

Politiques contenant les opérations API pour la création et l'utilisation des connexions

L'exemple de politique suivant décrit les autorisations AWS IAM requises pour créer et utiliser des connexions. Si vous créez un rôle, créez une politique contenant les éléments suivants :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListConnectionTypes",
        "glue:DescribeConnectionType",
        "glue:RefreshOAuth2Tokens",
        "glue:ListEntities",
        "glue:DescribeEntity"
      ],
      "Resource": "*"
    }
  ]
}
```

Si vous ne voulez pas utiliser la méthode ci-dessus, vous pouvez également utiliser les politiques IAM gérées suivantes :

- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiée dans cette politique, AWS Glue les processus disposent des autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise la console AWS de gestion. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console.

Configuration de Zoom Meetings

Avant de pouvoir AWS Glue transférer des données depuis Zoom Meetings, vous devez satisfaire aux exigences suivantes :

Configuration requise

Les exigences minimales sont les suivantes :

- Vous disposez d'un compte Zoom Meetings.
- Votre compte Zoom est activé pour accéder à l'API.
- Vous avez créé une OAuth2 application dans votre compte Zoom Meetings. Cette intégration fournit les informations d'identification qui AWS Glue permettent d'accéder à vos données en toute sécurité lorsqu'elle passe des appels authentifiés vers votre compte. Pour de plus amples informations, veuillez consulter the section called "Configuration de l'application client Zoom Meetings".

Si vous répondez à ces exigences, vous êtes prêt à vous connecter AWS Glue à votre compte Zoom Meetings. Pour les connexions classiques, vous n'avez rien d'autre à faire dans Zoom Meetings.

Configuration de l'application client Zoom Meetings

1. Connectez-vous à la marketplace de l'application Zoom.
2. Choisissez Développer > Créer l'application.
3. Choisissez General App pour une application basée sur la OAuth version 2.0.
4. Sur la page Informations de base, ajoutez ou mettez à jour des informations sur l'application, telles que le nom de l'application, son mode de gestion, ses informations d'identification et autres OAuth informations.
5. Dans la section Sélectionner le mode de gestion de l'application, confirmez la manière dont vous souhaitez que votre application soit gérée :
   a. Gestion par l'administrateur : les administrateurs de compte ajoutent et gèrent l'application.
   b. Gestion par l'utilisateur : des utilisateurs individuels ajoutent et gèrent l'application. L'application n'a accès qu'aux données autorisées de l'utilisateur.
6. Identifiants de l'application : le flux de création génère automatiquement les informations d'identification (ID client et secret client) pour votre application.
7. Dans la section OAuth Informations, configurez OAuth votre application.
   a. OAuth URL de redirection (obligatoire) : entrez l'URL de redirection ou le point de terminaison à configurer OAuth entre votre application et Zoom.
   b. Utiliser l'URL en mode strict (facultatif)
   c. Vérification du sous-domaine (facultatif)

d. OAuth listes d'autorisation (obligatoire) : ajoutez toute adresse unique URLs que Zoom doit autoriser comme redirections valides pour vos OAuth flux.

8. Sur la page Portées, sélectionnez les méthodes de l'API Zoom que votre application est autorisée à appeler. Les portées définissent les informations et les fonctionnalités mises à la disposition de votre utilisateur. Sélectionnez les portées granulaires suivantes :

   - user:read:list_users:admin
   - zoom_rooms:read:list_rooms:admin
   - group:read:list_members:admin
   - group:read:administrator:admin
   - group:read:list_groups:admin
   - report:read:admin
   - role:read:list_roles, role:read:list_roles:admin

   Une fois les portées ajoutées, choisissez Continuer et l'application sera alors prête à être utilisée.

Pour plus d'informations sur la configuration OAuth 2.0, voir Intégrations (OAuth applications).

Configuration des connexions Zoom Meetings

Zoom Meetings prend en charge le type de subvention AUTHORIZATION_CODE pour. OAuth2
Le type de subvention détermine le mode AWS Glue de communication avec Zoom Meetings pour demander l'accès à vos données.

- Ce type de subvention est considéré comme « à trois étapes » OAuth car il repose sur la redirection des utilisateurs vers un serveur d'autorisation tiers pour authentifier l'utilisateur. Il est utilisé lors de la création de connexions via la AWS Glue console. L'utilisateur qui crée une connexion doit fournir des informations OAuth connexes telles que l'ID client et le secret du client pour son application client Zoom Meetings. La AWS Glue console redirige l'utilisateur vers Zoom où il doit se connecter et autoriser AWS Glue les autorisations demandées pour accéder à son instance Zoom Meetings.
- Les utilisateurs peuvent toujours choisir de créer leur propre application connectée dans Zoom Meetings et de fournir leur propre identifiant client et leur secret client lors de la création de connexions via la AWS Glue console. Dans ce scénario, ils seront toujours redirigés vers Zoom Meetings pour se connecter et autoriser AWS Glue l'accès à leurs ressources.

- Ce type d'autorisation donne lieu à un jeton d'actualisation et à un jeton d'accès. Le jeton d'accès est de courte durée et peut être actualisé automatiquement sans interaction de l'utilisateur à l'aide du jeton d'actualisation.
- Pour consulter la documentation publique de Zoom Meetings sur la création d'une application connectée pour le OAuth flux de code d'autorisation, voir Utilisation de la OAuth version 2.0.

Pour configurer une connexion Zoom Meetings :

1. Dans AWS Secrets Manager, créez un secret avec les informations suivantes :
   a. Pour l'application connectée gérée par le client, le secret doit contenir le secret du consommateur de l'application connectée avec `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` comme clé.
   b. Remarque : vous devez créer un secret pour vos connexions dans AWS Glue.

1. Dans AWS Glue Glue Studio, créez une connexion sous Data Connections en suivant les étapes ci-dessous :
   a. Lorsque vous sélectionnez un type de connexion, sélectionnez Zoom Meetings.
   b. Indiquez l'environnement Zoom Meetings auquel vous souhaitez vous connecter.
   c. Sélectionnez le rôle AWS IAM qui AWS Glue peut assumer et dispose des autorisations nécessaires pour les actions suivantes :

      JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetSecretValue",
        "secretsmanager:PutSecretValue",
        "ec2:CreateNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterface"
      ],
      "Resource": "*"
    }
```

```
  ]
}
```

d. Sélectionnez celui `secretName` que vous souhaitez utiliser pour cette connexion AWS Glue afin d'y placer les jetons.

e. Sélectionnez les options réseau si vous souhaitez utiliser votre réseau.

2. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire`secretName`.

Lecture depuis les entités de Zoom Meetings

Prérequis

Un objet Zoom Meetings à partir duquel vous souhaitez lire. Vous aurez besoin du nom de l'objet, comme `Group` ou `Zoom Rooms`.

Entités prises en charge pour la source :

Entité	Peut être filtré	Limit prise en charge	Order by prise en charge	Select prise en charge*	Partition nement pris en charge
Zoom Rooms	Non	Oui	Non	Oui	Non
Groupe	Non	Non	Non	Oui	Non
Group Member	Oui	Oui	Non	Oui	Non
Group Admin	Non	Oui	Non	Oui	Non
Report (daily)	Oui	Non	Non	Oui	Non
Roles	Non	Non	Non	Oui	Non
Utilisateurs	Oui	Oui	Non	Oui	Non

Exemple :

```
zoom_read = glueContext.create_dynamic_frame.from_options(
```

```
    connection_type="zoom",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "organization",
        "API_VERSION": "v2"
    }
)
```

Détails des entités et des champs Zoom Meetings :

Zoom Meetings charge dynamiquement les champs disponibles sous l'entité sélectionnée. Les opérateurs de filtrage suivants sont pris en charge en fonction du type de données du champ.

Entité	Champ	Type de données	Opérateurs pris en charge	
Zoom Room	status	String	=	
	type	String	=	
	unassigned_rooms	Booléen	=	
	location_id	String	=	
	room_id	String	N/A	
	activation_code	String	N/A	
	id	String	N/A	
	name	String	N/A	
	tag_ids	String	N/A	
	query_name	String	N/A	
Daily Report	month	Date	=	
	date	Date	N/A	
	meeting_minutes	Entier	N/A	

Entité	Champ	Type de données	Opérateurs pris en charge	
	meetings	Entier	N/A	
	new_users	Entier	N/A	
	participants	Entier	N/A	
	group_id	String	N/A	
Utilisateur	created_at	DateTime	N/A	
	dept	String	N/A	
	e-mail	String	N/A	
	employee_ unique_id	String	N/A	
	first_name	String	N/A	
	group_ids	List	N/A	
	host_key	String	N/A	
	id	String	N/A	
	im_group_ids	String	N/A	
	last_client_versio n	String	N/A	
	last_login_time	DateTime	N/A	
	last_name	String	N/A	
	plan_united_type	String	N/A	
	custom_at tributes	List	N/A	

Entité	Champ	Type de données	Opérateurs pris en charge	
	pmi	BigInteger	N/A	
	role_id	String	=	
	status	String	=	
	timezone	String	N/A	
	type	Entier	N/A	
	verified	Entier	N/A	
	user_created_at	DateTime	N/A	
	display_name	String	N/A	
	phone_number	String	N/A	
	langage	String	N/A	
	license	String	=	
Groupe	id	String	N/A	
	name	String	N/A	
	total_members	Entier	N/A	
Group Member	e-mail	String	N/A	
	first_name	String	N/A	
	id	String	N/A	
	last_name	String	N/A	
	type	Entier	N/A	
	primary_group	Booléen	N/A	

Entité	Champ	Type de données	Opérateurs pris en charge	
	member_id	String	N/A	
Group Admin	id	String	N/A	
	e-mail	String	N/A	
	name	String	N/A	
rôle	description	String	N/A	
	id	String	N/A	
	name	String	N/A	
	total_members	Entier	N/A	
	type	String	=	

Requêtes de partitionnement

Zoom Meetings ne prend pas en charge le partitionnement basé sur des filtres ni le partitionnement basé sur des enregistrements.

Options de connexion Zoom Meetings

Les options de connexion pour Zoom Meetings sont les suivantes :

- `ENTITY_NAME`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Le nom de l'entité Zoom Meetings. Par exemple, `group`.
- `API_VERSION`(Chaîne) - (Obligatoire) Utilisé pour la lecture. Version d'API Rest de Zoom Meetings que vous souhaitez utiliser. La valeur sera `v2`, car Zoom Meetings ne prend actuellement en charge que la version 2.
- `SELECTED_FIELDS`(Liste<Chaîne>) - Par défaut : vide (SELECT *). Utilisé pour la lecture. Liste de colonnes séparées par des virgules que vous souhaitez sélectionner pour l'entité sélectionnée.
- `FILTER_PREDICATE`(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Doit être au format SQL Spark.

- QUERY(Chaîne) - Par défaut : vide. Utilisé pour la lecture. Requête Spark SQL complète.

Restrictions de Zoom Meetings

Les restrictions ou remarques suivantes concernent Zoom Meetings :

- Zoom Meetings ne prend pas en charge la commande Order by.
- Zoom Meetings ne prend pas en charge le partitionnement basé sur des filtres, car aucun champ ne peut répondre aux critères requis.
- Zoom Meetings ne prend pas en charge le partitionnement basé sur les enregistrements, car la limite de pagination et la pagination basée sur le décalage ne sont pas prises en charge.

Ajout d'une connexion JDBC à l'aide de vos propres pilotes JDBC

Vous pouvez utiliser votre propre pilote JDBC lorsque vous utilisez une connexion JDBC. Lorsque le pilote par défaut utilisé par le AWS Glue robot d'exploration ne parvient pas à se connecter à une base de données, vous pouvez utiliser votre propre pilote JDBC. Par exemple, si vous souhaitez utiliser SHA-256 avec votre base de données Postgres et que les anciens pilotes Postgres ne le prennent pas en charge, vous pouvez utiliser votre propre pilote JDBC.

Sources de données prises en charge

Sources de données prises en charge	Sources de données non prises en charge
MySQL	Snowflake
Postgres	
Oracle	
Redshift	
SQL Server	
Aurora*	

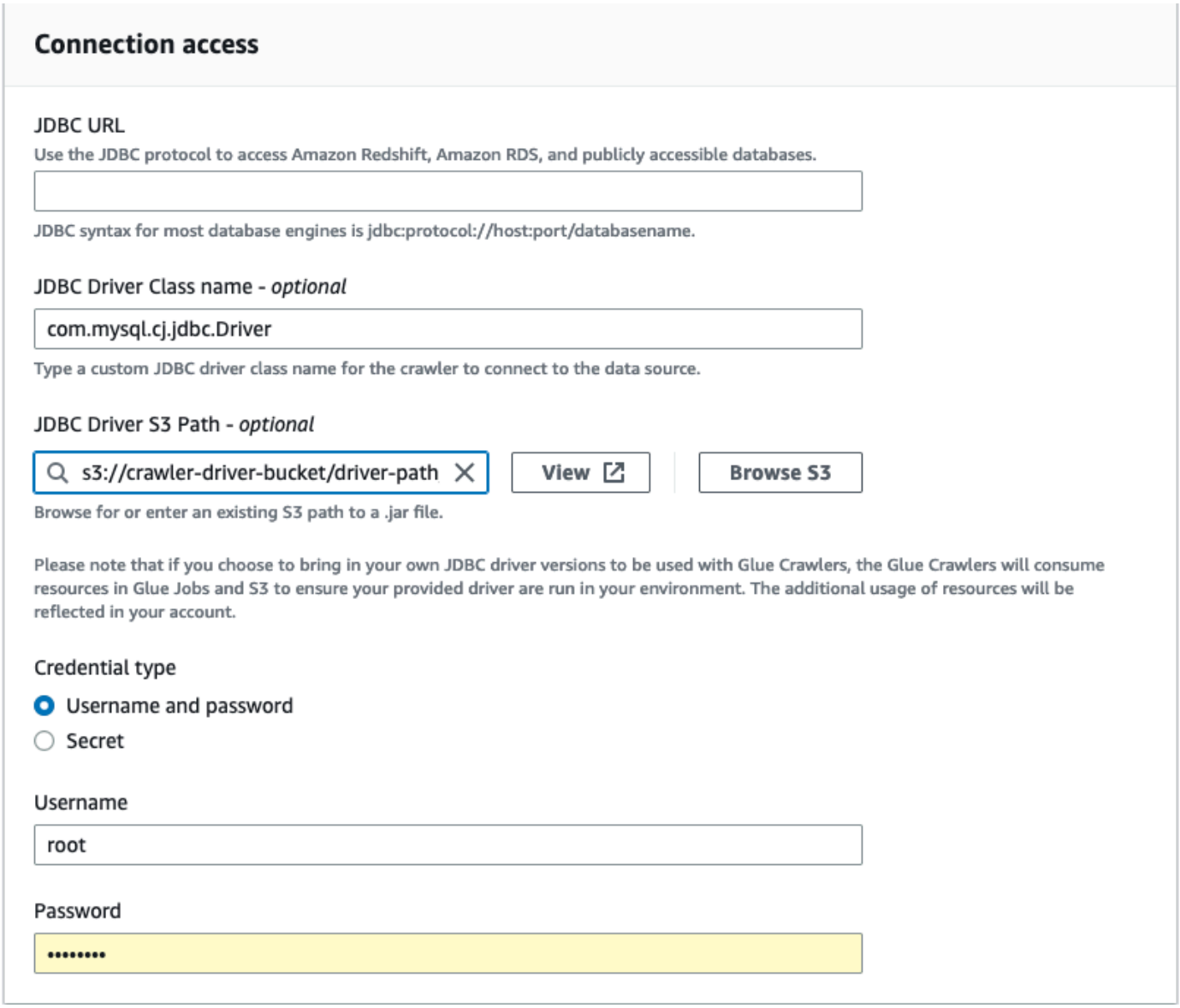
* Pris en charge si le pilote JDBC natif est utilisé. Toutes les fonctionnalités du pilote ne peuvent pas être exploitées.

Ajout d'un pilote JDBC à une connexion JDBC

> **Note**
>
> Si vous choisissez d'utiliser vos propres versions de pilote JDBC, les robots d' AWS Glue exploration consommeront des ressources dans les AWS Glue tâches et les compartiments Amazon S3 pour s'assurer que le pilote que vous avez fourni est exécuté dans votre environnement. L'utilisation supplémentaire des ressources sera reflétée sur votre compte. Le coût des AWS Glue robots d'exploration et des tâches figure dans la AWS Glue catégorie indiquée dans la facturation. De plus, le fait de fournir votre propre pilote JDBC ne signifie pas que le Crawler est capable de tirer parti de toutes les fonctionnalités du pilote.

Pour ajouter votre propre pilote JDBC à une connexion JDBC :

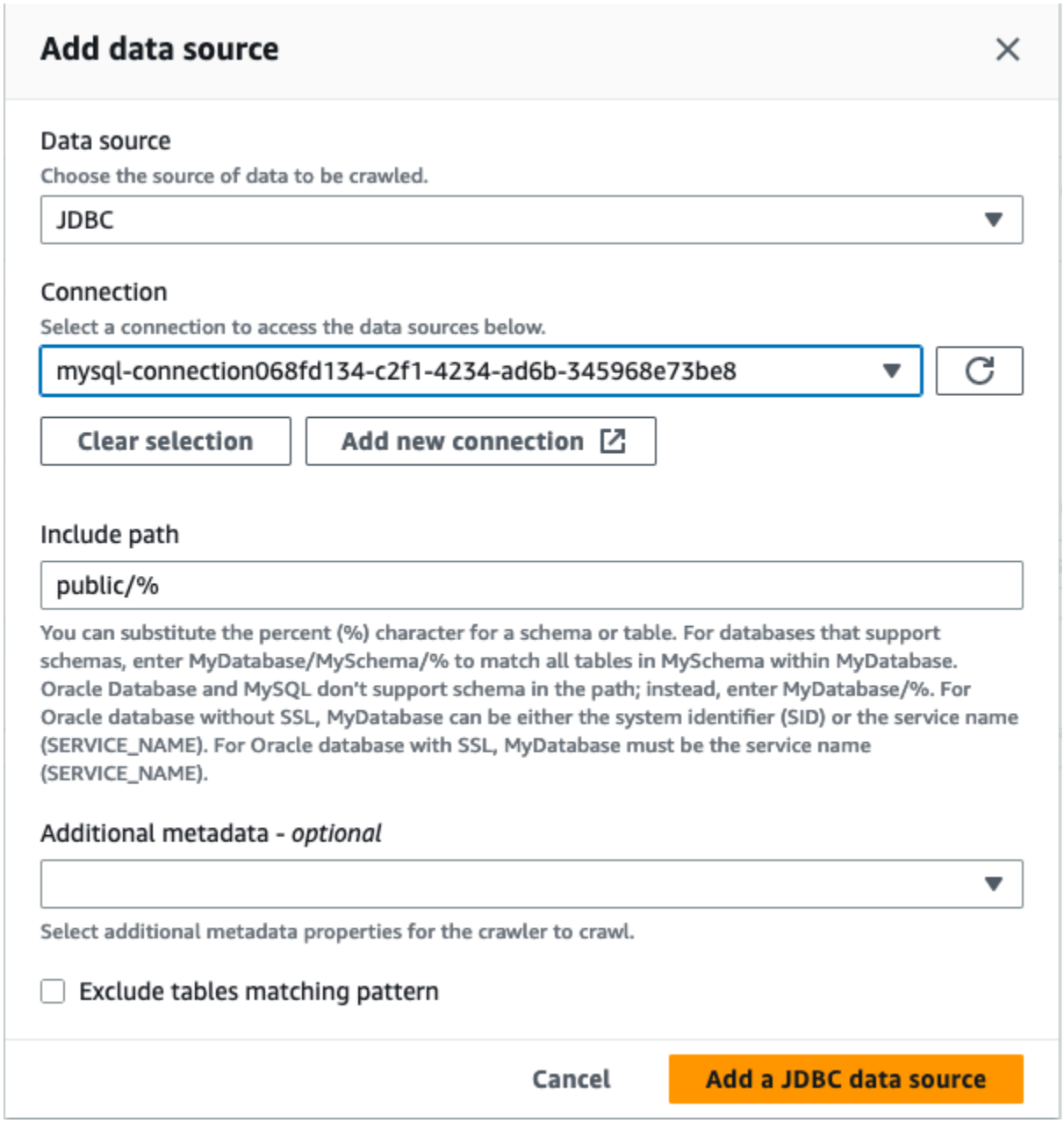
1. Ajoutez le fichier du pilote JDBC à un emplacement Amazon S3. Vous pouvez créer un and/or dossier de bucket ou utiliser un and/or dossier de bucket existant.
2. Dans la AWS Glue console, choisissez Connexions dans le menu de gauche sous Catalogue de données, puis créez une nouvelle connexion.
3. Complétez les champs pour Propriétés de connexion et choisissez JDBC pour Type de connexion.
4. Dans Accès à la connexion, entrez l'URL JDBC et le nom de la classe de pilote JDBC (facultatif). Le nom de classe du pilote doit être celui d'une source de données prise en charge par les robots d' AWS Glue exploration.



5. Choisissez le chemin Amazon S3 où se trouve le pilote JDBC dans chemin Amazon S3 du pilote JDBC : champ (facultatif).
6. Complétez les champs du type d'informations d'identification si vous saisissez un nom d'utilisateur et un mot de passe ou un code secret. Lorsque vous avez terminé, choisissez Créer une connexion.

> **Note**
>
> Le test de connexion n'est pas pris en charge pour le moment. Lorsque vous effectuez une indexation de site web de la source de données à l'aide d'un pilote JDBC que vous avez fourni, le Crawler ignore cette étape.

7. Ajoutez la connexion nouvellement créée à un Crawler. Dans la AWS Glue console, choisissez Crawlers dans le menu de gauche sous Data Catalog, puis créez un nouveau robot d'exploration.
8. Dans l'assistant Ajouter un Crawler, à l'étape 2, choisissez Ajouter une source de données.



9. Choisissez JDBC comme source de données et choisissez la connexion créée lors des étapes précédentes. Complet
10. Pour utiliser votre propre pilote JDBC avec un AWS Glue robot d'exploration, ajoutez les autorisations suivantes au rôle utilisé par le robot d'exploration :

- Accordez des autorisations pour les actions de tâches suivantes : `CreateJob`, `DeleteJob`, `GetJob`, `GetJobRun`, `StartJobRun`.
- Accordez des autorisations pour les actions IAM : `iam:PassRole`
- Accordez des autorisations pour les actions Amazon S3 : `s3:DeleteObjects`, `s3:GetObject`, `s3:ListBucket`, `s3:PutObject`.
- Accordez l'accès principal au service bucket/folder dans la politique IAM.

Exemple de politique IAM :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "VisualEditor0",
      "Effect": "Allow",
      "Action": [
        "s3:PutObject",
        "s3:GetObject",
        "s3:ListBucket",
        "s3:DeleteObject"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/driver-parent-folder/driver.jar",
        "arn:aws:s3:::amzn-s3-demo-bucket"
      ]
    }
  ]
}
```

Le AWS Glue robot crée deux dossiers : _glue_job_crawler et _crawler.

Si le fichier JAR du pilote se trouve dans le dossier `s3://amzn-s3-demo-bucket/driver.jar"`, ajoutez les ressources suivantes :

```
"Resource": [
```

```
                "arn:aws:s3:::amzn-s3-demo-bucket/_glue_job_crawler/*",
    "arn:aws:s3:::amzn-s3-demo-bucket/_crawler/*"
            ]
```

Si le fichier JAR du pilote se trouve dans le dossier `s3://amzn-s3-demo-bucket/tmp/driver/subfolder/driver.jar"`, ajoutez les ressources suivantes :

```
"Resource": [
               "arn:aws:s3:::amzn-s3-demo-bucket/tmp/_glue_job_crawler/*",
    "arn:aws:s3:::amzn-s3-demo-bucket/tmp/_crawler/*"
            ]
```

11. Si vous utilisez un VPC, vous devez autoriser l'accès au point de AWS Glue terminaison en créant le point de terminaison de l'interface et en l'ajoutant à votre table de routage. Pour plus d'informations, consultez Création d'un point de terminaison VPC d'interface pour AWS Glue
12. Si vous utilisez le chiffrement dans votre catalogue de données, créez le point de terminaison de l' AWS KMS interface et ajoutez-le à votre table de routage. Pour de plus amples informations, consultez Creating a VPC endpoint for AWS KMS.

Utilisation de connecteurs personnalisés et de connexions avec AWS Glue Studio

AWS Glue fournit une prise en charge intégrée des magasins de données les plus couramment utilisés (tels que Amazon Redshift, Amazon Aurora, Microsoft SQL Server, MySQL, MongoDB et PostgreSQL) à l'aide de connexions JDBC. AWS Glue vous permet également d'utiliser des pilotes JDBC personnalisés dans vos tâches Extract-transform-load (ETL). Pour les magasins de données qui ne sont pas pris en charge en mode natif, comme les applications SaaS, vous pouvez utiliser des connecteurs.

Un connecteur est un package de code facultatif qui facilite l'accès aux magasins de données dans AWS Glue Studio Glue Studio. Vous pouvez vous abonner à plusieurs connecteurs proposés dans AWS Marketplace.

Lorsque vous créez des tâches ETL, vous pouvez utiliser un magasin de données pris en charge de manière native AWS Marketplace, un connecteur ou vos propres connecteurs personnalisés. Si vous utilisez un connecteur, vous devez d'abord créer une connexion pour le connecteur. Une connexion

contient les propriétés requises pour se connecter à un magasin de données particulier. Vous utilisez la connexion avec vos sources de données et cibles de données dans la tâche ETL. Les connecteurs et les connexions fonctionnent ensemble pour faciliter l'accès aux magasins de données.

Les connexions suivantes sont disponibles lors de la création de connexions pour les connecteurs :

- Amazon Aurora: un moteur de base de données relationnelle évolutif et très performant doté de fonctionnalités intégrées de sécurité, de sauvegarde et de restauration, ainsi que d'accélération en mémoire.
- Amazon DocumentDB : un service de base de données de documents évolutif, hautement disponible et entièrement géré qui prend en charge MongoDB et SQL. APIs
- Amazon Redshift: un service de base de données de documents évolutif, hautement disponible et entièrement géré qui prend en charge MongoDB et SQL. APIs
- Azure SQL : un service de base de données relationnelle basé sur le cloud de Microsoft Azure qui fournit des fonctionnalités de stockage et de gestion de données évolutives, fiables et sécurisées.
- Cosmos DB : un service de base de données basé sur le cloud distribué dans le monde entier de Microsoft Azure qui fournit des fonctionnalités de stockage et de requête de données évolutives et très performantes.
- Google BigQuery : un entrepôt de données cloud sans serveur permettant d'exécuter des requêtes SQL rapides sur de grands ensembles de données.
- JDBC : système de gestion de base de données relationnelle (RDBMS) qui utilise une API Java pour se connecter et interagir avec les connexions de données.
- Kafka : une plateforme de traitement de flux open source utilisée pour le streaming de données et la messagerie en temps réel.
- MariaDB : une fourche développée par la communauté de MySQL qui offre des performances, une évolutivité et des fonctionnalités améliorées.
- MongoDB : une base de données multiplateforme orientée documents qui offre une évolutivité, une flexibilité et des performances élevées.
- MongoDB Atlas : une offre de base de données en tant que service (DBaaS) basée sur le cloud de MongoDB qui simplifie la gestion et le dimensionnement des déploiements MongoDB.
- Microsoft SQL Server : système de gestion de base de données relationnelle (RDBMS) de Microsoft qui fournit des fonctionnalités robustes de stockage, d'analyse et de création de rapports.
- Mixpanel : plateforme analytique qui aide les entreprises à analyser la manière dont les utilisateurs interagissent avec leurs sites Web, leurs applications mobiles et leurs autres produits numériques.

- MySQL : un système de gestion de base de données relationnelle (RDBMS) open source largement utilisé dans les applications web et connu pour sa fiabilité et sa capacité de mise à l'échelle.
- Réseau : une source de données réseau représente une ressource ou un service accessible par le réseau auquel une plateforme d'intégration de données peut accéder.
- OpenSearch— une source de OpenSearch données est une application qui OpenSearch peut se connecter à des données et les ingérer depuis.
- Oracle : un système de gestion de base de données relationnelle (RDBMS) d'Oracle Corporation qui fournit des fonctionnalités robustes de stockage, d'analyse et de création de rapports.
- PostgreSQL : un système de gestion de base de données relationnelle (RDBMS) open source qui fournit des fonctionnalités robustes de stockage, d'analyse et de création de rapports.
- Salesforce : Salesforce fournit un logiciel de gestion de la relation client (CRM) qui vous aide dans les domaines des ventes, du service client, du commerce électronique, etc. Si vous êtes un utilisateur de Salesforce, vous pouvez vous connecter AWS Glue à votre compte Salesforce. Vous pouvez ensuite utiliser Salesforce comme source de données ou comme destination dans vos tâches ETL. Exécutez ces tâches pour transférer des données entre Salesforce et les AWS services ou d'autres applications prises en charge.
- SAP HANA : une base de données en mémoire et plateforme d'analyse qui permet un traitement rapide des données, des analyses avancées et une intégration des données en temps réel.
- Snowflake : un entrepôt des données basé sur le cloud qui fournit des services d'analyse et de stockage de données évolutifs et très performants.
- Teradata : un système de gestion de base de données relationnelle (RDBMS) qui fournit des fonctionnalités très performantes de stockage, d'analyse et de création de rapports.
- Vertica : un entrepôt des données analytiques orienté colonnes conçu pour l'analytique du big data qui offre des performances de requêtes rapides, des analyses avancées et capacité de mise à l’échelle.

Création de connecteurs personnalisés

Vous pouvez également créer votre propre connecteur, puis charger le code du connecteur dans AWS Glue Studio.

Les connecteurs personnalisés sont intégrés à AWS Glue Studio Glue Studio via l'API d'exécution Spark AWS Glue. L'exécution Spark AWS Glue vous permet de brancher n'importe quel connecteur

compatible avec l'interface Spark, Athena ou JDBC. Cela vous permet de transférer n'importe quelle option de connexion disponible avec le connecteur personnalisé.

Vous pouvez encapsuler toutes vos propriétés de connexion avec AWS Glue Connections (Connexions Glue) et fournir le nom de la connexion à votre tâche ETL. L'intégration aux connexions Data Catalog vous permet d'utiliser les mêmes propriétés de connexion sur plusieurs appels dans une seule application Spark ou différentes applications.

Vous pouvez spécifier des options supplémentaires pour la connexion. Le script de tâche généré par AWS Glue Studio Glue Studio contient une entrée `Datasource` qui utilise la connexion pour brancher votre connecteur avec les options de connexion spécifiées. Par exemple :

```
Datasource = glueContext.create_dynamic_frame.from_options(connection_type =
"custom.jdbc", connection_options = {"dbTable":"Account","connectionName":"my-custom-
jdbc-
connection"}, transformation_ctx = "DataSource0")
```

Pour ajouter un connecteur personnalisé à AWS Glue Studio Glue Studio

1. Créez le code de votre connecteur personnalisé. Pour de plus amples informations, veuillez consulter Développement de connecteurs personnalisés.
2. Ajoutez la prise en charge des fonctions AWS Glue à votre connecteur. Voici quelques exemples de ces fonctions et de leur utilisation dans le script de tâche généré par AWS Glue Studio :

   - Mappage des types de données : votre connecteur peut convertir les colonnes tout en les lisant à partir du magasin de données sous-jacent. Par exemple, un `dataTypeMapping` de `{"INTEGER":"STRING"}` convertit toutes les colonnes de type `Integer` en colonnes de type `String` lors de l'analyse des enregistrements et de la création du fichier `DynamicFrame`. Cela aide les utilisateurs à convertir les colonnes en types de leur choix.

     ```
     DataSource0 = glueContext.create_dynamic_frame.from_options(connection_type
     = "custom.jdbc", connection_options = {"dataTypeMapping":{"INTEGER":"STRING"}",
     connectionName":"test-connection-jdbc"}, transformation_ctx = "DataSource0")
     ```

   - Partitionnement pour les lectures parallèles : AWS Glue permet des lectures de données parallèles à partir du magasin de données en partitionnant les données sur une colonne. Vous devez spécifier la colonne de partition, la limite de partition inférieure, la limite de partition supérieure et le nombre de partitions. Cette fonction vous permet d'utiliser à la fois le parallélisme des données et plusieurs exécuteurs Spark alloués à l'application Spark.

```
DataSource0 = glueContext.create_dynamic_frame.from_options(connection_type
= "custom.jdbc", connection_options = {"upperBound":"200","numPartitions":"4",
"partitionColumn":"id","lowerBound":"0","connectionName":"test-connection-jdbc"},
transformation_ctx = "DataSource0")
```

- AWS Secrets Manager À utiliser pour stocker les informations d'identification : la connexion au catalogue de données peut également contenir un `secretId` code secret stocké dans AWS Secrets Manager. Le AWS secret peut stocker en toute sécurité les informations d'authentification et d'identification et les fournir AWS Glue lors de l'exécution. Vous pouvez également spécifier le `secretId` à partir du script Spark comme suit :

```
DataSource = glueContext.create_dynamic_frame.from_options(connection_type
= "custom.jdbc", connection_options = {"connectionName":"test-connection-jdbc",
 "secretId"-> "my-secret-id"}, transformation_ctx = "DataSource0")
```

- Filtrage des données source avec des prédicats de ligne et des projections de colonne : l'exécution Spark AWS Glue permet également aux utilisateurs d'envoyer des requêtes SQL pour filtrer les données à la source avec des prédicats de ligne et des projections de colonne. Cela permet à votre tâche ETL de charger plus rapidement les données filtrées à partir des magasins de données qui prennent en charge les transferts. Voici un exemple de requête SQL transmise à une source de données JDBC : `SELECT id, name, department FROM department WHERE id < 200.`

```
DataSource = glueContext.create_dynamic_frame.from_options(connection_type =
"custom.jdbc", connection_options = {"query":"SELECT id, name, department FROM
 department
WHERE id < 200","connectionName":"test-connection-jdbc"}, transformation_ctx =
"DataSource0")
```

- Marque-pages de tâche : AWS Glue prend en charge le chargement progressif de données à partir de sources JDBC. AWS Glue assure le suivi du dernier registre traité à partir du magasin de données et traite les nouveaux registres de données dans les exécutions de tâches ETL suivantes. Les marque-pages de tâche utilisent la clé primaire comme colonne par défaut pour la clé de marque-page, à condition que cette colonne augmente ou diminue de manière séquentielle. Pour plus d'informations sur les marque-pages de tâche, consultez Marque-pages de tâche dans le Guide du développeur AWS Glue .

```
DataSource0 = glueContext.create_dynamic_frame.from_options(connection_type =
```

```
"custom.jdbc", connection_options = {"jobBookmarkKeys":["empno"],
 "jobBookmarkKeysSortOrder"
:"asc", "connectionName":"test-connection-jdbc"}, transformation_ctx =
 "DataSource0")
```

3. Empaquetez le connecteur personnalisé en tant que fichier JAR et téléchargez le fichier sur Amazon S3.
4. Testez votre connecteur personnalisé. Pour plus d'informations, consultez les instructions sur GitHub at Glue Custom Connectors : Local Validation Tests Guide.
5. Dans la console AWS Glue Studio Glue Studio, sélectionnez Connectors (Connecteurs) dans le panneau de navigation de la console.
6. Sur la page Connectors (Connecteurs), sélectionnez Create custom connector (Créer un connecteur personnalisé).
7. Sur la page Create custom connector (Créer un connecteur personnalisé), saisissez les informations suivantes :
   - Chemin d'accès à l'emplacement du fichier JAR de code personnalisé dans Amazon S3.
   - Nom du connecteur qui sera utilisé par AWS Glue Studio Glue Studio.
   - Votre type de connecteur, JDBC, Spark ou Athena.
   - Nom du point d'entrée dans votre code personnalisé qu'AWS Glue Studio Glue Studio appelle pour utiliser le connecteur.
     - Pour les connecteurs JDBC, ce champ doit être le nom de classe de votre pilote JDBC.
     - Pour les connecteurs Spark, ce champ doit être le nom complet de la classe de source de données, ou son alias, que vous utilisez lors du chargement de la source de données Spark avec l'opérateur `format`.
   - (JDBC uniquement) URL de base utilisée par la connexion JDBC pour le magasin de données.
   - (Facultatif) Description du connecteur personnalisé.
8. Sélectionnez Create connector (Créer un connecteur).
9. Depuis la page Connectors (Connecteurs), créez une connexion qui utilise ce connecteur, comme décrit dans Création de connexions pour les connecteurs.

Ajouter des connecteurs àAWS Glue Studio

Un connecteur est un morceau de code qui facilite la communication entre votre magasin de données et AWS Glue. Vous pouvez soit vous abonner à un connecteur proposé dans AWS Marketplace, soit créer votre propre connecteur personnalisé.

Abonnement à des connecteurs AWS Marketplace

AWS Glue Studiopermet d'ajouter facilement des connecteurs à partir de AWS Marketplace.

Pour ajouter un connecteur de AWS Marketplace à AWS Glue Studio

1. Dans la console AWS Glue Studio Glue Studio, sélectionnez Connectors (Connecteurs) dans le panneau de navigation de la console.
2. Sur la page Connectors (Connecteurs), sélectionnez Go to AWS Marketplace (Accéder à MKT).
3. Dans AWS Marketplace, dans Produits en vedette, choisissez le connecteur que vous souhaitez utiliser. Vous pouvez choisir l'un des connecteurs disponibles ou utiliser la recherche. Vous pouvez effectuer une recherche sur le nom ou le type de connecteur, et utiliser des options pour affiner les résultats de cette recherche.

   Si vous souhaitez utiliser l'un des connecteurs proposés, sélectionnez View product (Afficher le produit). Si vous avez utilisé la recherche pour localiser un connecteur, choisissez le nom de celui-ci.
4. Sur la page produit du connecteur, utilisez les onglets pour afficher des informations sur le connecteur. Si vous décidez d'acheter ce connecteur, sélectionnez Continue to Subscribe (Continuer pour s'abonner).
5. Fournissez les informations de paiement, puis sélectionnez Continue to Configure (Continuer pour configurer).
6. Sur la page Configure this software (Configurer ce logiciel), choisissez la méthode de déploiement et la version du connecteur à utiliser. Puis sélectionnez Continue to Launch (Continuer pour lancer).
7. Sur la page Launch this software (Lancer ce logiciel), vous pouvez consulter les instructions d'utilisation fournies par le fournisseur du connecteur. Lorsque vous êtes prêt à continuer, sélectionnez Activate connection in (Activer la connexion dans)AWS Glue Studio.

   Après un bref instant, la console affiche la page Create marketplace connection (Créer une connexion de site de vente) dans AWS Glue Studio Glue Studio.

8. Créez une connexion qui utilise ce connecteur, comme décrit dans Création de connexions pour les connecteurs.

   Vous pouvez également sélectionner Activate connector only (Activer le connecteur uniquement) pour ignorer la création d'une connexion à ce stade. Vous devez créer une connexion à une date ultérieure avant de pouvoir utiliser le connecteur.

Création de connexions pour les connecteurs

Une AWS Glue connexion est un objet du catalogue de données qui stocke les informations de connexion pour un magasin de données spécifique. Les connexions stockent les informations de connexion, les chaînes d'URI, les informations du cloud privé virtuel (VPC), etc. La création de connexions dans le catalogue de données vous évite d'avoir à spécifier tous les détails de connexion à chaque fois que vous créez une tâche.

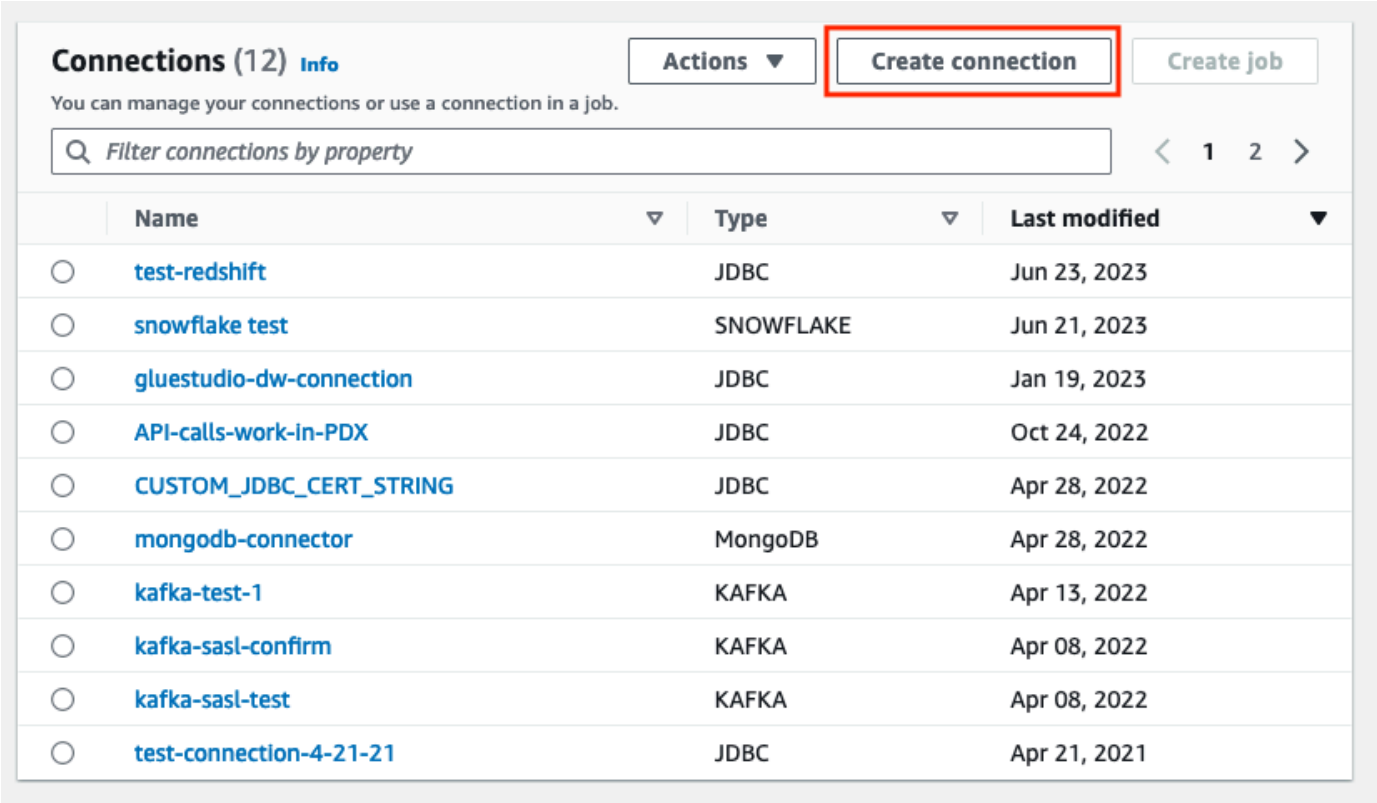
Pour créer une connexion pour un connecteur

1. Dans la AWS Glue Studio console, choisissez Connectors dans le volet de navigation de la console. Dans la section Connexions, choisissez Créer une connexion.
2. Choisissez la source de données pour laquelle vous souhaitez créer une connexion dans la première étape de l'assistant Créer une connexion de données. Il existe plusieurs façons de visualiser les sources de données disponibles, notamment :

   - Filtrez les sources de données disponibles en choisissant un onglet. Par défaut, Tous les connecteurs est sélectionné.
   - Basculez vers Liste pour afficher les sources de données sous forme de liste ou revenez à la Grille pour afficher les connecteurs disponibles dans la disposition en grille.
   - Utilisez la barre de recherche pour affiner la liste des sources de données. Lorsque vous tapez, les résultats de recherche s'affichent et les sources non correspondantes sont supprimées de la vue.

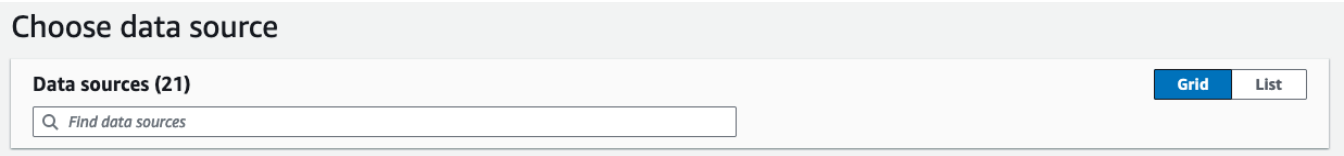
   Une fois que vous avez choisi la source de données, choisissez Suivant.
3. Configurez la connexion dans la deuxième étape de l'assistant.

   Saisissez les informations de connexion. Selon le type de connecteur que vous avez sélectionné, vous êtes invité à saisir des informations supplémentaires :



4. Choisissez la source de données pour laquelle vous souhaitez créer une connexion dans la première étape de l'assistant Créer une connexion de données. Il existe plusieurs méthodes pour afficher les sources de données disponibles. Par défaut, vous verrez toutes les sources de données disponibles sous forme de grille. Vous pouvez également :

   - Basculez vers Liste pour afficher les sources de données sous forme de liste ou revenez à la Grille pour afficher les connecteurs disponibles dans la disposition en grille.
   - Utilisez la barre de recherche pour affiner la liste des sources de données. Lorsque vous tapez, les résultats de recherche s'affichent et les sources non correspondantes sont supprimées de la vue.

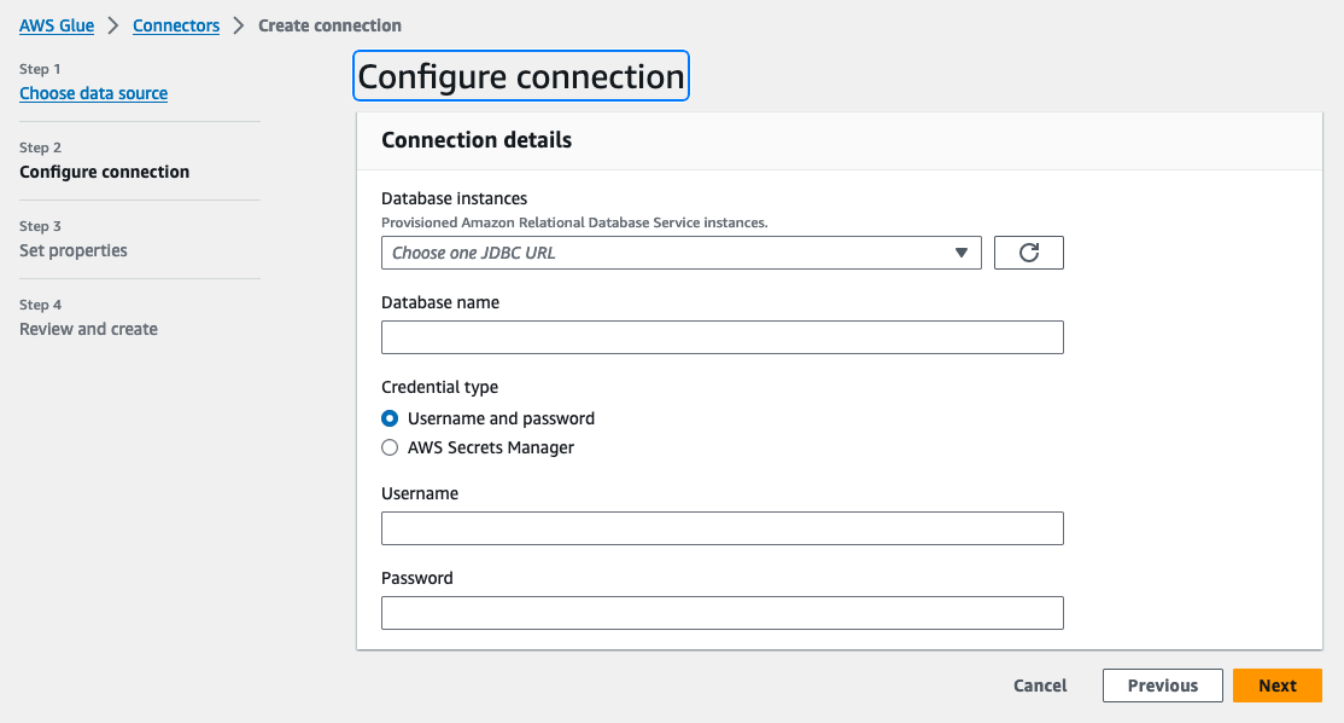


   Une fois que vous avez choisi la source de données, choisissez Suivant.

5. Configurez la connexion dans la deuxième étape de l'assistant.

Saisissez les informations de connexion. Selon le type de connecteur que vous avez sélectionné, vous devrez peut-être saisir des informations de connexion supplémentaires. Cela peut inclure :

- Détails de connexion : ces champs changeront en fonction de la source de données à laquelle vous vous connectez. Par exemple, si vous vous connectez aux bases de données Amazon DocumentDB, vous devez saisir l'URL Amazon DocumentDB. Si vous vous connectez à Amazon Aurora, vous devez choisir l'instance de base de données et entrer le nom de la base de données. Voici les détails de connexion requis pour Amazon Aurora :



- Type d'informations d'identification : choisissez le nom d'utilisateur et mot de passe ou AWS Secrets Manager. Saisissez les informations d'authentification demandées.
- Pour les connecteurs qui utilisent JDBC, entrez les informations requises pour créer l'URL JDBC pour le magasin de données.
- Si vous utilisez un Virtual Private Cloud (VPC), saisissez les informations réseau de votre VPC.

6. Définissez les propriétés de connexion dans la troisième étape de l'assistant. Vous pouvez ajouter une description et des balises en tant que partie facultative de cette étape. Le nom est obligatoire et est prérempli avec une valeur par défaut. Choisissez Suivant.
7. Vérifiez la source, les détails et les propriétés de la connexion. Si vous devez apporter des modifications, choisissez Modifier pour l'étape de l'assistant. Lorsque vous avez terminé, choisissez Créer une connexion.

Choisissez Créer une connexion.

Vous revenez à la page Connectors (Connecteurs) et la bannière d'information indique la connexion qui a été créée. Vous pouvez maintenant utiliser la connexion dans vos tâches AWS Glue Studio .

Création d'une connexion Kafka

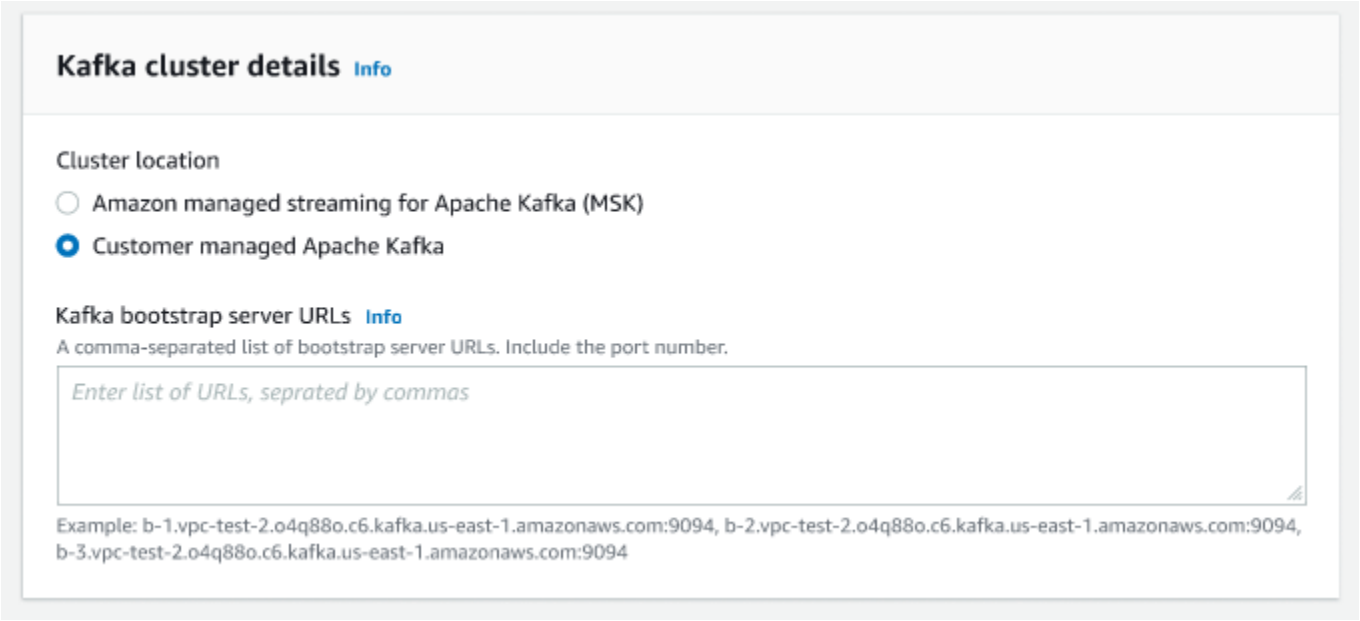
Lorsque vous créez une connexion Kafka, la sélection de Kafka dans le menu déroulant fait apparaitre des paramètres supplémentaires à configurer :

- Détails du cluster Kafka
- Authentification
- Chiffrement
- Options réseau

Configurer les détails du cluster Kafka

1. Choisissez l'emplacement du cluster. Vous pouvez choisir parmi un cluster Streaming géré par Amazon pour Apache Kafka (MSK) (Amazon Managed Streaming for Apache Kafka (MSK)) ou un cluster Customer managed Apache Kafka (Apache Kafka géré par le client). Pour plus d'informations sur Amazon Managed Streaming pour Apache Kafka, consultez Amazon Managed Streaming for Apache Kafka (MSK).

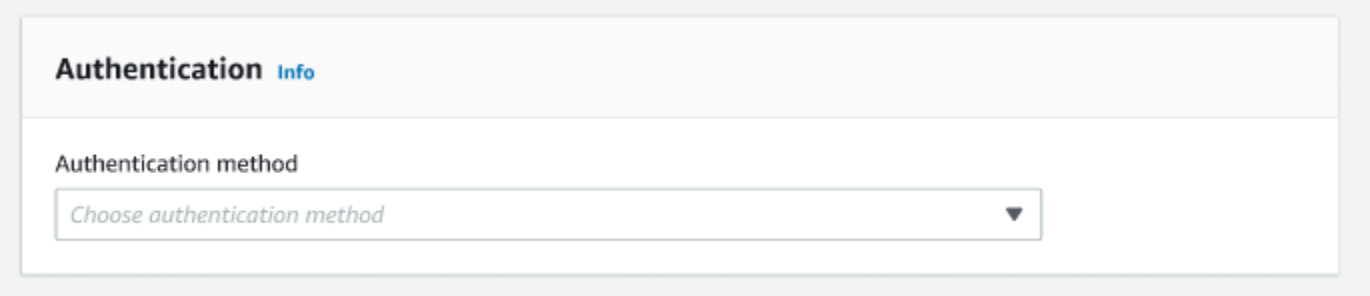
   > **Note**
   >
   > Amazon Managed Streaming pour Apache Kafka prend uniquement en charge les méthodes d'authentification TLS et SASL/SCRAM-SHA-512.



2. Entrez le URLs pour vos serveurs bootstrap Kafka. Vous pouvez en saisir plusieurs en séparant chaque serveur par une virgule. Incluez le numéro de port à la fin de l'URL en ajoutant `:<port number>`.

   Par exemple : `b-1.vpc-test-2.034a88o.kafka-us-east-1.amazonaws.com:9094`

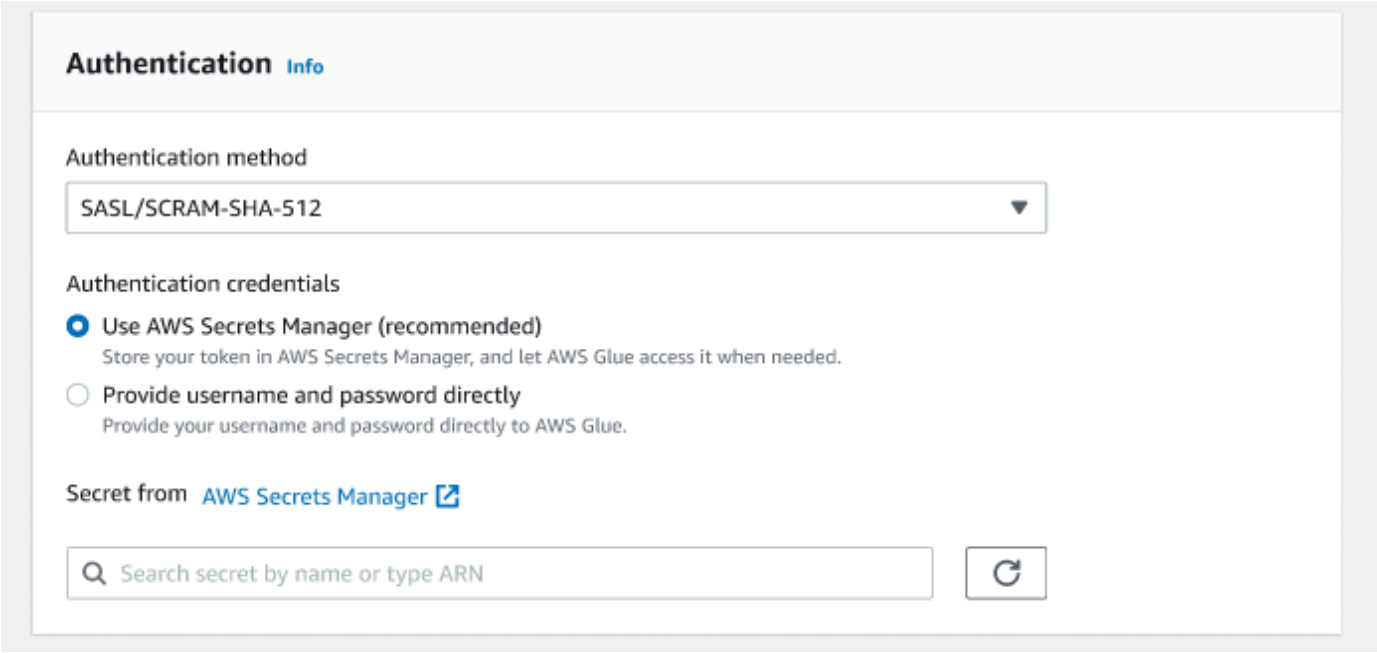
Sélection d'une méthode d'authentification



AWS Glue prend en charge le framework SASL (Simple Authentication and Security Layer) pour l'authentification. Le framework SASL prend en charge divers mécanismes d'authentification et AWS Glue propose les protocoles SCRAM (nom d'utilisateur et mot de passe), GSSAPI (protocole Kerberos) et PLAIN (nom d'utilisateur et mot de passe).

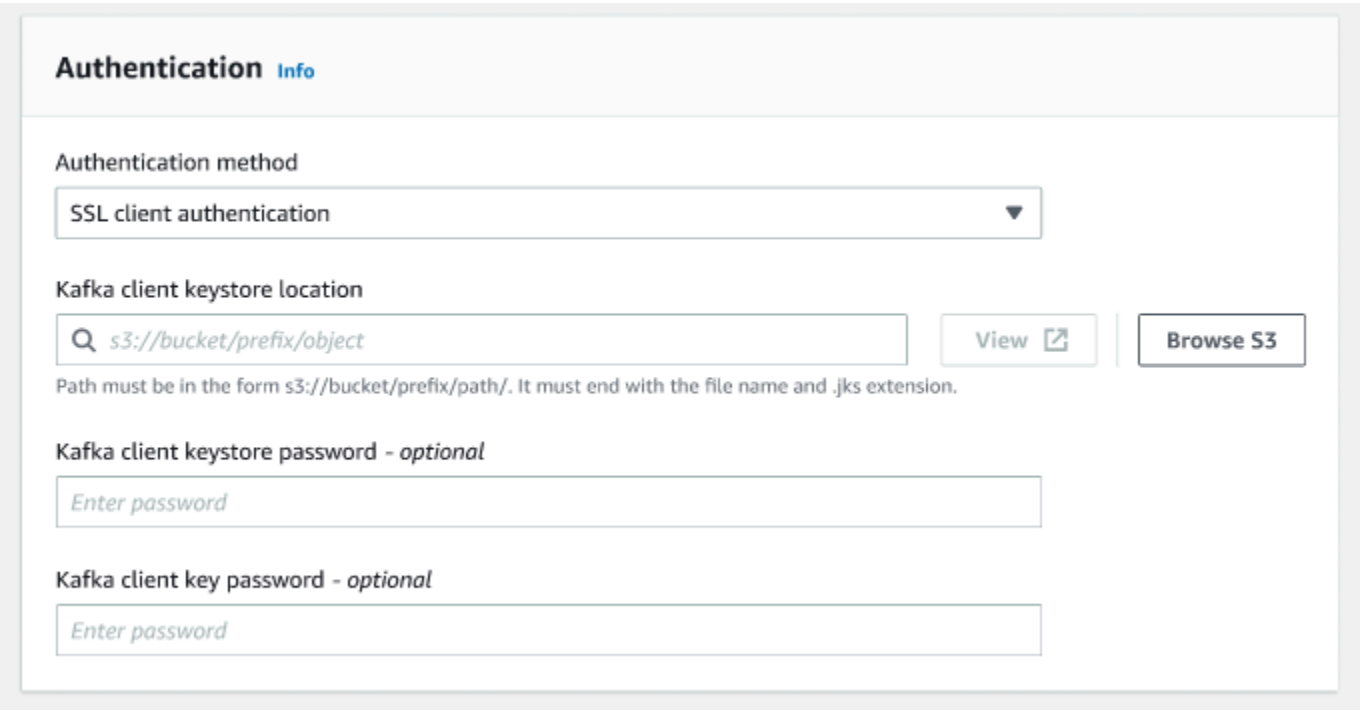
Lorsque vous choisissez une méthode d'authentification dans le menu déroulant, les méthodes d'authentification client suivantes peuvent être sélectionnées :

- Aucune - Aucune authentification. Cette option est utile si vous créez une connexion pour des raisons de tests.
- SASL/SCRAM-SHA-512 - Choisissez cette méthode d'authentification pour spécifier les informations d'identification d'authentification. Deux options s'offrent à vous :
  - Utiliser AWS Secrets Manager (recommandé) : si vous sélectionnez cette option, vous pouvez enregistrer vos informations d'identification dans AWS Secrets Manager et autoriser l' AWS Glue accès aux informations en cas de besoin. Spécifiez le secret qui stocke les informations d'identification d'authentification SSL ou SASL.



  - Fournissez directement un nom d'utilisateur et un mot de passe.
- SASL/GSSAPI (Kerberos) - si vous sélectionnez cette option, vous pouvez sélectionner l'emplacement du fichier keytab, du fichier krb5.conf et entrer le nom principal Kerberos et le nom du service Kerberos. Les emplacements du fichier keytab et du fichier krb5.conf doivent se trouver dans un emplacement Amazon S3. Puisque MSK ne prend pas encore en charge SASL/GSSAPI, cette option n'est disponible que pour les clusters Apache Kafka gérés par le client. Pour en savoir plus, consultez MIT Kerberos Documentation: Keytab (Documentation du MIT Kerberos : Keytab).
- SASL/PLAIN - Choisissez cette méthode d'authentification pour spécifier les informations d'authentification. Deux options s'offrent à vous :
  - Utiliser AWS Secrets Manager (recommandé) : si vous sélectionnez cette option, vous pouvez enregistrer vos informations d'identification dans AWS Secrets Manager et autoriser l' AWS Glue accès aux informations en cas de besoin. Spécifiez le secret qui stocke les informations d'identification d'authentification SSL ou SASL.

  - Fournissez directement un nom d'utilisateur et un mot de passe.
- Authentification client SSL : si vous sélectionnez cette option, vous pouvez sélectionner l'emplacement du centre de stockage des clés client Kafka en naviguant sur Amazon S3. Vous pouvez également entrer le mot de passe du centre de stockage des clés client Kafka et le mot de passe de la clé client Kafka.



Configuration des paramètres de chiffrement

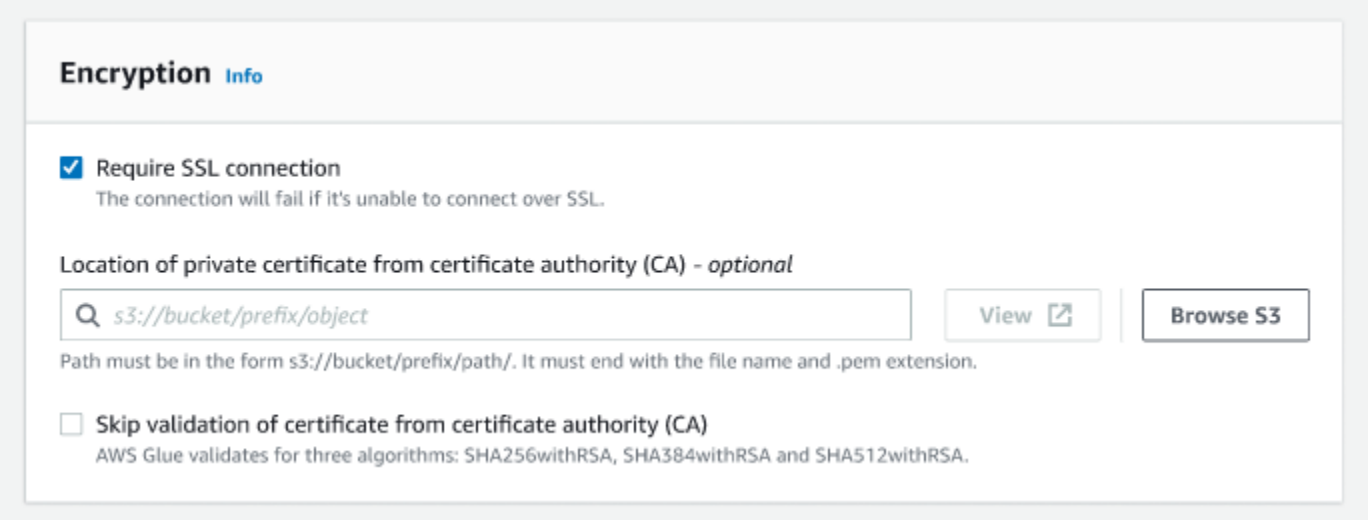
1. Si la connexion Kafka nécessite une connexion SSL, cochez la case correspondant à Require SSL connection (Connexion SSL obligatoire). Notez que la connexion échouera si elle ne parvient pas à se connecter via SSL. Le protocole SSL pour le chiffrement peut être utilisé avec n'importe quelle méthode d'authentification (SASL/SCRAM-SHA-512, SASL/GSSAPI, SASL/PLAINou authentification client SSL) et est facultatif.

   Si la méthode d'authentification est définie sur Authentification client SSL, cette option sera sélectionnée automatiquement et sera désactivée pour empêcher toute modification.
2. (Facultatif). Choisissez l'emplacement du certificat privé auprès de l'autorité de certification (CA). Notez que l'emplacement de la certification doit se trouver dans un emplacement S3. Choisissez Browse (Parcourir) pour choisir le fichier dans un compartiment S3 connecté. Le chemin doit être

de la forme `s3://bucket/prefix/filename.pem`. Il doit se terminer par le nom du fichier et l'extension .pem.

3. Vous pouvez choisir d'ignorer la validation d'un certificat auprès d'une autorité de certification (CA). Choisissez la case à cocher Skip validation of certificate from certificate authority (CA) (Sauter l'étape de validation du certificat de l'autorité de certification (CA)). Si cette case n'est pas cochée, AWS Glue valide les certificats pour trois algorithmes :

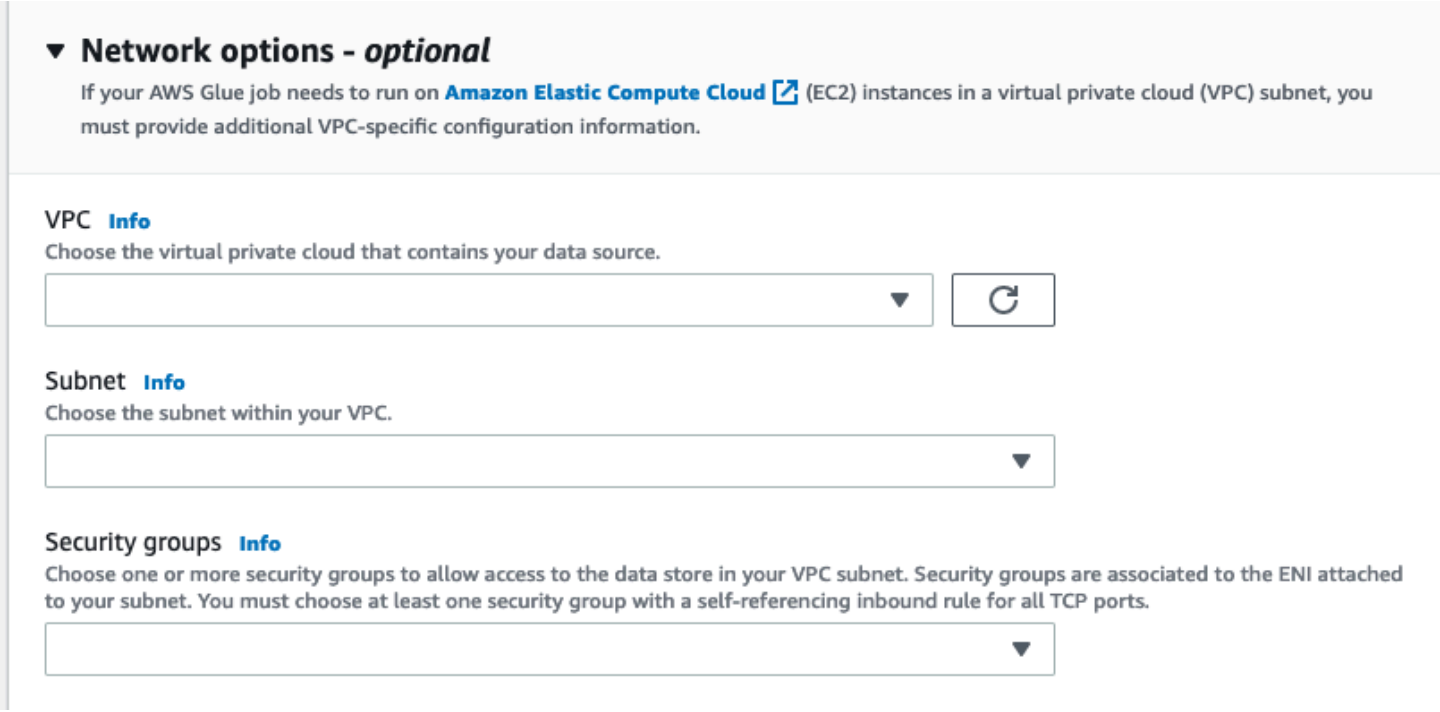
   - SHA256withRSA
   - SHA384withRSA
   - SHA512withRSA



(Facultatif) Options réseau

Les étapes suivantes sont facultatives pour configurer des groupes de VPC, de sous-réseau et de sécurité. Si votre AWS Glue tâche doit être exécutée sur des instances Amazon EC2 dans un sous-réseau de cloud privé virtuel (VPC), vous devez fournir des informations de configuration supplémentaires spécifiques au VPC.

1. Choisissez le nom du cloud privé virtuel (VPC) qui contient votre sources de données.
2. Choisissez le sous-réseau au sein de votre VPC.
3. Choisissez un ou plusieurs groupes de sécurité qui autorisent l'accès au stockage des données dans votre sous-réseau VPC. Les groupes de sécurité sont associés à l'ENI attaché à votre sous-réseau. Vous devez choisir au moins un groupe de sécurité avec une règle entrante avec référencement automatique pour tous les ports TCP.



Création de tâches avec des connecteurs personnalisés

Vous pouvez utiliser des connecteurs et des connexions pour les nœuds de source de données et les nœuds de cible de données dans AWS Glue Studio Glue Studio.

Rubriques

- Créer des tâches qui utilisent un connecteur pour la source de données
- Configurer les propriétés source pour les nœuds qui utilisent des connecteurs
- Configurer les propriétés cibles pour les nœuds qui utilisent des connecteurs

Créer des tâches qui utilisent un connecteur pour la source de données

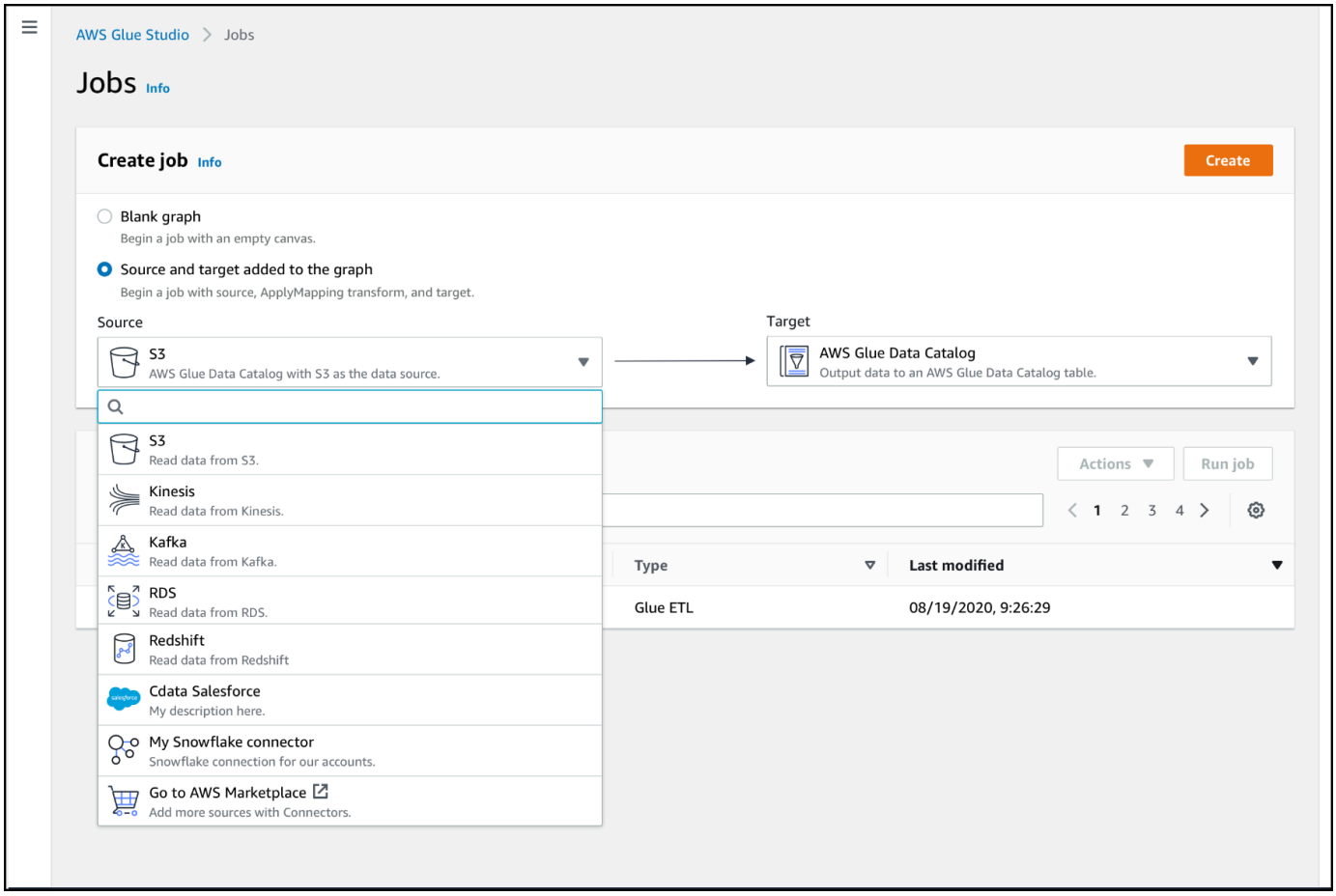
Lorsque vous créez une tâche, vous pouvez choisir un connecteur pour la source de données et les cibles de données.

Pour créer une tâche qui utilise des connecteurs pour la source de données ou la cible de données

1. Connectez-vous à la AWS Glue Studio console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/gluestudio/.

2. Depuis la page Connectors (Connecteurs), dans la liste de ressources Your connections (Vos connexions), choisissez la connexion que vous souhaitez utiliser dans votre tâche, puis sélectionnez Create job (Créer une tâche).

   Sinon, sur la page Jobs (Tâches) d'AWS Glue Studio Glue Studio, sous Create job (Créer une tâche), sélectionnez Source and target added to the graph (Source et cible ajoutées au graphique). Dans la liste déroulante Source, choisissez le connecteur personnalisé que vous souhaitez utiliser dans votre tâche. Vous pouvez également sélectionner un connecteur pour Target (Cible).



3. Sélectionnez ensuite Create (Créer) pour ouvrir l'éditeur de tâches visuelles.
4. Configurez le nœud de source de données, comme décrit dans Configurer les propriétés source pour les nœuds qui utilisent des connecteurs.
5. Continuez à créer votre tâche ETL en ajoutant des transformations, des magasins de données supplémentaires et des cibles de données, comme décrit dans Démarrage de tâches ETL visuelles dans AWS Glue Studio.

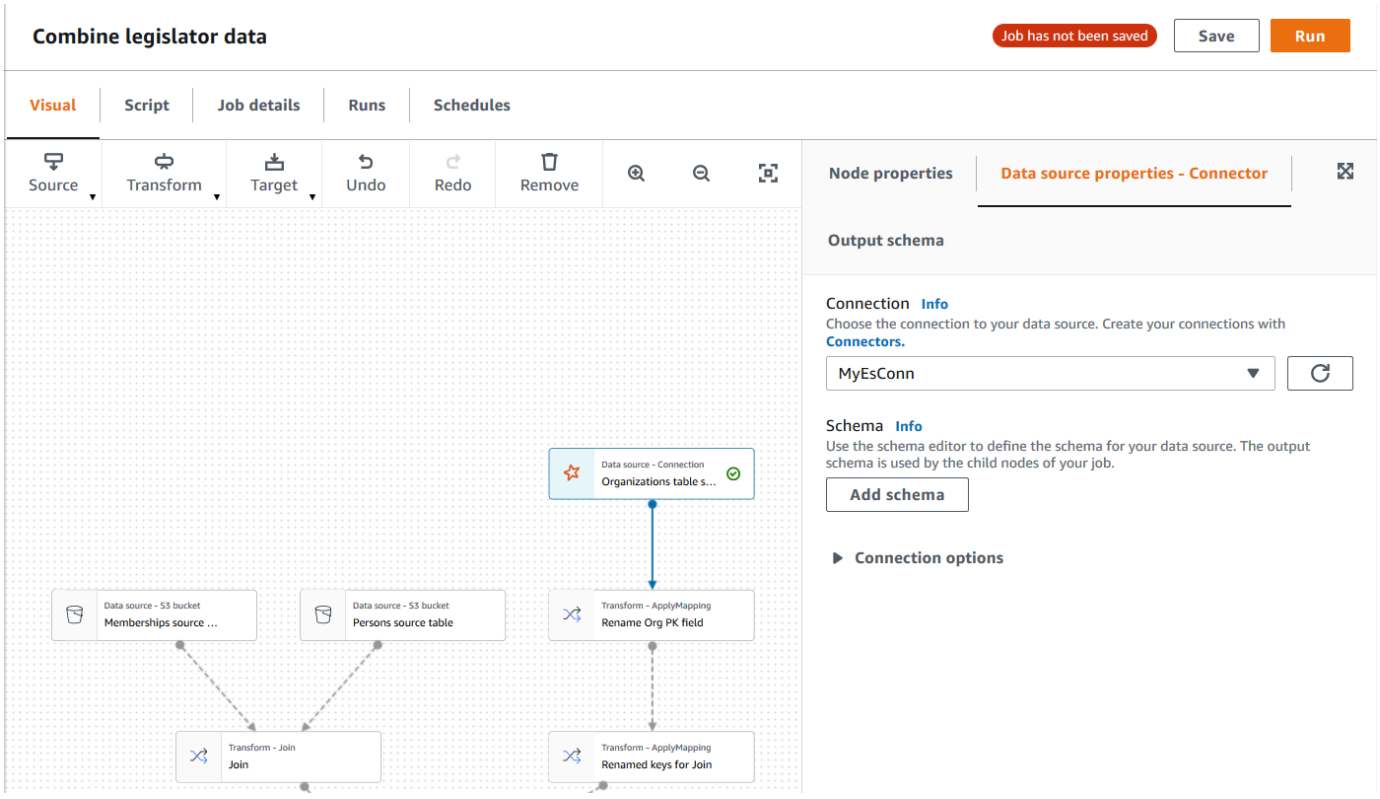
6. Personnalisez l'environnement d'exécution de la tâche en configurant les propriétés de la tâche comme décrit dans Modifier les propriétés de tâche.
7. Enregistrez, puis exécutez la tâche.

Configurer les propriétés source pour les nœuds qui utilisent des connecteurs

Après avoir créé une tâche qui utilise un connecteur pour la source de données, l'éditeur de tâche visuelle affiche un graphique de tâche avec un nœud de source de données configuré pour le connecteur. Vous devez configurer les propriétés de la source de données pour ce nœud.

Pour configurer les propriétés d'un nœud de source de données qui utilise un connecteur

1. Choisissez le nœud de la source de données du connecteur dans le graphique de la tâche ou ajoutez un nouveau nœud et choisissez le connecteur pour le type de nœud. Ensuite, sur la droite, dans le panneau des détails du nœud, sélectionnez l'onglet Data source properties (Propriétés de la source de données), s'il n'est pas déjà sélectionné.



2. Sous l'onglet Data source properties (Propriétés de la source de données), choisissez la connexion que vous souhaitez utiliser pour cette tâche.

   Saisissez les informations supplémentaires requises pour chaque type de connexion :

JDBC

- Type d'entrée de source de données : fournissez un nom de table ou une requête SQL comme source de données. Selon votre choix, vous devrez ensuite fournir les informations complémentaires suivantes :
  - Nom de la table : nom de la table dans la source de données. Si la source de données n'utilise pas le terme table, indiquez le nom d'une structure de données appropriée, comme indiqué par les informations d'utilisation du connecteur personnalisé (disponibles dans AWS Marketplace).
  - Prédicat de filtre : clause de condition à utiliser lors de la lecture de la source de données, similaire à une clause `WHERE`, qui est utilisée pour récupérer un sous-ensemble des données.
  - Code de requête : saisissez une requête SQL à utiliser pour récupérer un ensemble de données spécifique à partir de la source de données. Voici un exemple d'une requête SQL de base :

    ```
    SELECT column_list FROM
                              table_name WHERE where_clause
    ```

- Schéma : étant donné qu'AWS Glue Studio Glue Studio utilise les informations stockées dans la connexion pour accéder à la source de données au lieu de récupérer les informations de métadonnées à partir d'une table Data Catalog, vous devez fournir les métadonnées de schéma pour la source de données. Sélectionnez Add schema (Ajouter un schéma) pour ouvrir l'éditeur de schéma.

  Pour obtenir des instructions sur l'utilisation de l'éditeur de schéma, veuillez consulter Modifier le schéma d'un nœud de transformation personnalisé.

- Colonne de partition : (facultatif) vous pouvez choisir de partitionner les lectures de données en fournissant des valeurs pour Partition column (Colonne de partition), Lower bound (Limite inférieure), Upper bound (Limite supérieure) et Number of partitions (Nombre de partitions).

  Les valeurs `lowerBound` et `upperBound` sont utilisées pour décider de la progression de la partition, pas pour filtrer les lignes de la table. Toutes les lignes de la table sont partitionnées et renvoyées.

> **Note**
>
> Le partitionnement de colonne ajoute une condition de partitionnement supplémentaire à la requête utilisée pour lire les données. Lorsque vous utilisez une requête au lieu d'un nom de table, vous devez valider que la requête fonctionne avec la condition de partitionnement spécifiée. Par exemple :
>
> - Si le format de votre requête est `"SELECT col1 FROM table1"`, testez la requête en ajoutant une clause `WHERE` à la fin de la requête qui utilise la colonne de partition.
> - Si le format de votre requête est `"SELECT col1 FROM table1 WHERE col2=val"`, testez la requête en étendant la clause `WHERE` avec `AND` et une expression qui utilise la colonne de partition.

- Conversion de type de données : si la source de données utilise des types de données qui ne sont pas disponibles dans JDBC, utilisez cette section pour spécifier comment un type de données de la source de données doit être converti en types de données JDBC. Vous pouvez spécifier jusqu'à 50 conversions de types de données différents. Toutes les colonnes de la source de données qui utilisent le même type de données sont converties de la même manière.

  Par exemple, si vous avez trois colonnes dans la source de données qui utilisent le type de données `Float` et que vous indiquez que le type de données `Float` doit être converti en type de données JDBC `String`, les trois colonnes qui utilisent le type de données `Float` sont converties en types de données `String`.

- Job bookmark keys (Clés de marque-pages de tâche) : les marque-pages de tâche permettent à AWS Glue de conserver des informations d'état et d'empêcher le retraitement des anciennes données. Spécifiez une ou plusieurs colonnes comme clés de marque-page. AWS Glue Studio utilise les clés de marque-page pour suivre les données qui ont déjà été traitées au cours d'une exécution précédente de la tâche ETL. Toutes les colonnes que vous utilisez pour les clés de marque-page personnalisées doivent être croissantes ou décroissantes de manière monotonique et stricte, mais les espaces sont autorisés.

  Si vous saisissez plusieurs clés de marque-page, elles sont combinées pour former une seule clé composée. Une clé de marque-page de tâche composée ne doit pas contenir de colonnes en double. Si vous ne spécifiez pas de clés de marque-page, AWS Glue Studio utilise par défaut la clé primaire comme clé de marque-page, à condition que la clé primaire

augmente ou diminue de manière séquentielle (sans espace). Si la table n'a pas de clé primaire, mais que la propriété de marque-page de tâche est activée, vous devez fournir des clés de marque-page de tâche personnalisées. Sinon, la recherche des clés primaires à utiliser par défaut échouera et l'exécution du travail échouera.

- Ordre de tri des clés de marque-page de la tâche : indiquez si les valeurs clés augmentent ou diminuent de façon séquentielle.

Spark

- Schéma : étant donné qu'AWS Glue Studio Glue Studio utilise les informations stockées dans la connexion pour accéder à la source de données au lieu de récupérer les informations de métadonnées à partir d'une table Data Catalog, vous devez fournir les métadonnées de schéma pour la source de données. Sélectionnez Add schema (Ajouter un schéma) pour ouvrir l'éditeur de schéma.

  Pour obtenir des instructions sur l'utilisation de l'éditeur de schéma, veuillez consulter Modifier le schéma d'un nœud de transformation personnalisé.

- Options de connexion : saisissez des paires clé-valeur supplémentaires si nécessaire pour fournir des informations ou des options de connexion supplémentaires. Par exemple, vous pouvez saisir un nom de base de données, un nom de table, un nom d'utilisateur et un mot de passe.

  Par exemple, pour OpenSearch, vous entrez les paires clé-valeur suivantes, comme décrit dans : the section called "Tutoriel : Utilisation du AWS Glue connecteur pour Elasticsearch"

  - `es.net.http.auth.user` : *`username`*
  - `es.net.http.auth.pass` : *`password`*
  - `es.nodes` : `https://`*`<Elasticsearch endpoint>`*
  - `es.port` : `443`
  - path: *`<Elasticsearch resource>`*
  - `es.nodes.wan.only` : `true`

Pour un exemple des options de connexion minimales à utiliser, consultez l'exemple de script de test MinimalSparkConnectorTest.scala on GitHub, qui indique les options de connexion que vous devez normalement fournir dans une connexion.

Athena

- Nom de la table : nom de la table dans la source de données. Si vous utilisez un connecteur pour lire les CloudWatch journaux d'Athena, vous devez entrer le nom de la table. `all_log_streams`
- Nom du schéma Athena : choisissez le schéma dans votre source de données Athena qui correspond à la base de données qui contient la table. Si vous utilisez un connecteur pour lire les CloudWatch journaux d'Athena, vous devez entrer un nom de schéma similaire à. `/aws/glue/name`
- Schéma : étant donné qu'AWS Glue Studio Glue Studio utilise les informations stockées dans la connexion pour accéder à la source de données au lieu de récupérer les informations de métadonnées à partir d'une table Data Catalog, vous devez fournir les métadonnées de schéma pour la source de données. Sélectionnez Add schema (Ajouter un schéma) pour ouvrir l'éditeur de schéma.

  Pour obtenir des instructions sur l'utilisation de l'éditeur de schéma, veuillez consulter Modifier le schéma d'un nœud de transformation personnalisé.
- Options de connexion supplémentaires : saisissez des paires clé-valeur supplémentaires si nécessaire pour fournir des informations ou des options de connexion supplémentaires.

Pour un exemple, consultez le `README.md` fichier à l'adresse https://github.com/aws-samples/aws-glue-samples/tree/master/GlueCustomConnectors/development/Athena. Dans les étapes de ce document, l'exemple de code montre les options de connexion minimales requises, qui sont `tableName`, `schemaName` et `className`. L'exemple de code spécifie ces options dans le cadre de la variable `optionsMap`, mais vous pouvez les spécifier pour votre connexion, puis utiliser la connexion.

3. (Facultatif) Après avoir fourni les informations requises, vous pouvez afficher le schéma de données résultant pour votre source de données en choisissant l'onglet Output schema (Schéma de sortie) dans le panneau des détails du nœud. Le schéma affiché sur cet onglet est utilisé par tous les nœuds enfants que vous ajoutez au graphique de tâche.
4. (Facultatif) Après avoir configuré les propriétés du nœud et les propriétés de la source de données, vous pouvez prévisualiser le jeu de données à partir de votre source de données en sélectionnant l'onglet Data preview (Prévisualisation des données) dans le volet de détails du nœud. La première fois que vous choisissez cet onglet pour un nœud de votre tâche, vous êtes invité à fournir un rôle IAM pour accéder aux données. Il y a un coût associé à l'utilisation de cette fonction, et la facturation commence dès que vous fournissez le rôle IAM.

Configurer les propriétés cibles pour les nœuds qui utilisent des connecteurs

Si vous utilisez un connecteur pour le type de cible de données, vous devez configurer les propriétés du nœud de cible de données.

Pour configurer les propriétés d'un nœud de cible de données qui utilise un connecteur

1. Choisissez le nœud cible des données du connecteur dans le graphique de la tâche. Ensuite, sur la droite, dans le panneau des détails du nœud, sélectionnez l'onglet Data target properties (Propriétés de la cible de données), s'il n'est pas déjà sélectionné.
2. Dans Data target properties (Propriétés des cibles de données), choisissez la connexion à utiliser pour écrire sur la cible.

   Saisissez les informations supplémentaires requises pour chaque type de connexion :

   JDBC

   - Connexion : choisissez la connexion à utiliser avec votre connecteur. Pour plus d'informations sur la création d'une connexion, consultez Création de connexions pour les connecteurs.
   - Nom de la table : nom de la table dans la cible de données. Si la cible de données n'utilise pas le terme table, indiquez le nom d'une structure de données appropriée, comme indiqué par les informations d'utilisation du connecteur personnalisé (disponibles dans AWS Marketplace).
   - Taille de lot (facultatif) : saisissez le nombre de lignes ou d'enregistrements à insérer dans la table cible en une seule opération. La valeur par défaut est de 1 000 lignes.

   Spark

   - Connexion : choisissez la connexion à utiliser avec votre connecteur. Si vous n'avez pas créé de connexion auparavant, sélectionnez Create connection (Créer une connexion) pour en créer une. Pour plus d'informations sur la création d'une connexion, consultez Création de connexions pour les connecteurs.
   - Options de connexion : saisissez des paires clé-valeur supplémentaires si nécessaire pour fournir des informations ou des options de connexion supplémentaires. Vous pouvez saisir un nom de base de données, un nom de table, un nom d'utilisateur et un mot de passe.

Par exemple, pour OpenSearch, vous entrez les paires clé-valeur suivantes, comme décrit dans : the section called "Tutoriel : Utilisation du AWS Glue connecteur pour Elasticsearch"

- `es.net.http.auth.user` : *`username`*
- `es.net.http.auth.pass` : *`password`*
- `es.nodes` : `https://`*`<Elasticsearch endpoint>`*
- `es.port` : `443`
- `path`: *`<Elasticsearch resource>`*
- `es.nodes.wan.only` : `true`

Pour un exemple des options de connexion minimales à utiliser, consultez l'exemple de script de test MinimalSparkConnectorTest.scala on GitHub, qui indique les options de connexion que vous devez normalement fournir dans une connexion.

3. Après avoir fourni les informations requises, vous pouvez afficher le schéma de données résultant pour votre source de données en choisissant l'onglet Output schema (Schéma de sortie) dans le panneau des détails du nœud.

Gestion des connecteurs et des connexions

Vous utilisez la page Connexions dans AWS Glue pour gérer vos connecteurs et connexions.

Rubriques

- Affichage des détails du connecteur et de la connexion
- Modification des connecteurs et des connexions
- Suppression des connecteurs et des connexions
- Annuler un abonnement pour un connecteur

Affichage des détails du connecteur et de la connexion

Vous pouvez afficher des informations récapitulatives sur vos connecteurs et connexions dans les tableaux de ressources Your connectors (Vos connecteurs) et Your connections (Vos connexions) sur la page Connectors (Connecteurs). Pour afficher les informations détaillées, procédez comme suit.

Pour afficher les détails des connecteurs ou des connexions

1. Dans la console AWS Glue Studio Glue Studio, sélectionnez Connectors (Connecteurs) dans le panneau de navigation de la console.
2. Choisissez le connecteur ou la connexion dont vous souhaitez afficher les informations détaillées.
3. Sélectionnez Actions, puis View details (Afficher les détails) pour ouvrir la page de détail de ce connecteur ou de cette connexion.
4. Sur la page détaillée, vous pouvez choisir de Edit (Modifier) ou de Delete (Supprimer) le connecteur ou la connexion.

   - Pour les connecteurs, vous pouvez sélectionner Create connecton (Créer une connexion) pour créer une connexion qui utilise le connecteur.
   - Pour les connexions, vous pouvez sélectionner Create job (Créer une tâche) pour créer une tâche qui utilise la connexion.

Modification des connecteurs et des connexions

Vous utilisez la page Connectors (Connecteurs) pour modifier les informations stockées dans vos connecteurs et connexions.

Pour modifier un connecteur ou une connexion

1. Dans la console AWS Glue Studio Glue Studio, sélectionnez Connectors (Connecteurs) dans le panneau de navigation de la console.
2. Sélectionnez le connecteur ou la connexion que vous souhaitez modifier.
3. Sélectionnez Actions, puis Edit (Modifier).

   Vous pouvez également sélectionner View details (Afficher les détails) et sur la page des détails du connecteur ou de la connexion, vous pouvez sélectionner Edit (Modifier).
4. Sur la page Edit connector (Modifier le connecteur) ou Edit connection (Modifier la connexion), mettez à jour les informations, puis sélectionnez Save (Enregistrer).

Suppression des connecteurs et des connexions

Vous pouvez utiliser la page Connectors (Connecteurs) pour supprimer les connecteurs et les connexions. Si vous supprimez un connecteur, toutes les connexions créées pour ce connecteur doivent également être supprimées.

Pour retirer des connecteurs de AWS Glue Studio

1. Dans la console AWS Glue Studio Glue Studio, sélectionnez Connectors (Connecteurs) dans le panneau de navigation de la console.
2. Sélectionnez le connecteur ou la connexion que vous souhaitez supprimer.
3. Sélectionnez Actions, puis sélectionnez Delete (Supprimer).

   Vous pouvez également sélectionner View details (Afficher les détails), puis sur la page détaillée du connecteur ou de la connexion, vous pouvez sélectionnez Delete (Supprimer).
4. Vérifiez que vous souhaitez supprimer le connecteur ou la connexion en saisissant **`Delete`**, puis sélectionnez Delete (Supprimer).

   Lors de la suppression d'un connecteur, toutes les connexions créées pour celui-ci sont également supprimées.

Toutes les tâches qui utilisent une connexion supprimée ne fonctionneront plus. Vous pouvez modifier les tâches pour utiliser un autre magasin de données ou les supprimer. Pour plus d'informations sur la suppression d'une tâche, veuillez consulter Supprimer une tâche.

Si vous supprimez un connecteur, cela n'annule pas l'abonnement au connecteur dans AWS Marketplace. Pour supprimer un abonnement à un connecteur supprimé, suivez les instructions de la rubrique Annuler un abonnement pour un connecteur.

Annuler un abonnement pour un connecteur

Après avoir supprimé les connexions et le connecteurAWS Glue Studio, vous pouvez annuler votre abonnement AWS Marketplace si vous n'en avez plus besoin.

> **Note**
>
> Si vous annulez votre abonnement à un connecteur, cela ne supprime pas le connecteur ou la connexion de votre compte. Toutes les tâches qui utilisent le connecteur et les connexions associées ne pourront plus utiliser le connecteur et échoueront.

Avant de vous désabonner ou de vous réabonner à un connecteur de AWS Marketplace, vous devez supprimer les connexions existantes et les connecteurs associés à ce AWS Marketplace produit.

Pour vous désabonner d'un connecteur dans AWS Marketplace

1. Connectez-vous à la AWS Marketplace console sur https://console.aws.amazon.com/marketket.
2. Sélectionnez Manage subscriptions (Gérer les abonnements).
3. Depuis la page Manage subscriptions (Gérer les abonnements), sélectionnez Manage (Gérer) en regard de l'abonnement au connecteur que vous souhaitez annuler.
4. Sélectionnez Actions, puis Cancel subscription (Annuler l'abonnement).
5. Cochez la case pour confirmer que les instances en cours d'exécution sont facturées sur votre compte, puis sélectionnez Yes, cancel subscription (Oui, annuler l'abonnement).

Développement de connecteurs personnalisés

Vous pouvez écrire le code qui lit ou écrit des données dans votre magasin de données et formate les données à utiliser avec les tâches AWS Glue Studio Glue Studio. Vous pouvez créer des connecteurs pour les magasins de données Spark, Athena et JDBC. L'exemple de code publié sur GitHub fournit un aperçu des interfaces de base que vous devez implémenter.

Vous aurez besoin d'un environnement de développement local pour créer votre code de connecteur. Vous pouvez utiliser n'importe quel IDE ou même simplement un éditeur de ligne de commande pour écrire votre connecteur. Voici quelques exemples d'environnements de développement :

- Un environnement Scala local avec une bibliothèque AWS Glue ETL Maven locale, comme décrit dans Développement local avec Scala dans le Guide du développeur AWS Glue .
- IntelliJ IDE, en téléchargeant l'IDE depuis. https://www.jetbrains.com/idea/

Rubriques

- Développement de connecteurs Spark
- Développement de connecteurs Athena
- Développement de connecteurs JDBC
- Exemples d'utilisation de connecteurs personnalisés avec AWS Glue Studio Glue Studio

- Développement de AWS Glue connecteurs pour AWS Marketplace

Développement de connecteurs Spark

Vous pouvez créer un connecteur Spark avec DataSource l'API Spark V2 (Spark 2.4) pour lire les données.

Pour créer un connecteur Spark personnalisé

Suivez les étapes de la bibliothèque AWS Glue GitHub d'exemples pour développer des connecteurs Spark, qui se trouve à l'adresse https://github.com/aws-samples/aws-glue-samples/tree/master/GlueCustomConnectors/development/Spark/README.md.

Développement de connecteurs Athena

Vous pouvez créer un connecteur Athena à utiliser par AWS Glue et AWS Glue Studio Glue Studio pour interroger une source de données personnalisée.

Pour créer un connecteur Athena personnalisé

Suivez les étapes de la bibliothèque AWS Glue GitHub d'exemples pour développer les connecteurs Athena, qui se trouve à https://github.com/aws-samples/ aws-glue-samples l'adresse/. tree/master/GlueCustomConnectors/development/Athena

Développement de connecteurs JDBC

Vous pouvez créer un connecteur qui utilise JDBC pour accéder à vos magasins de données.

Pour créer un connecteur JDBC personnalisé

1. Installez les bibliothèques d'exécution Spark AWS Glue dans votre environnement de développement local. Reportez-vous aux instructions de la bibliothèque AWS Glue GitHub d'exemples à l'adresse https://github.com/aws-samples/aws-glue-samples/tree/master/GlueCustomConnectors/development/GlueSparkRuntime/README.md.
2. Implémentez le pilote JDBC chargé de récupérer les données de la source de données. Reportez-vous à la documentation Java pour Java SE 8.

   Créez un point d'entrée dans votre code qu'AWS Glue Studio Glue Studio utilise pour localiser votre connecteur. Le champ Class name (Nom de classe) doit être le chemin complet de votre pilote JDBC.

3. Utilisation de l'API `GlueContext` pour lire les données avec le connecteur. Les utilisateurs peuvent ajouter plus d'options d'entrée dans la console AWS Glue Studio Glue Studio pour configurer la connexion à la source de données, si nécessaire. Pour un exemple de code qui montre comment lire et écrire dans une base de données JDBC avec un connecteur JDBC personnalisé, voir Valeurs personnalisées et ConnectionType. AWS Marketplace

Exemples d'utilisation de connecteurs personnalisés avec AWS Glue Studio Glue Studio

Vous pouvez consulter les blogs suivants pour des exemples d'utilisation de connecteurs personnalisés :

- Développer, tester et déployer des connecteurs personnalisés pour vos magasins de données avec AWS Glue
- Apache Hudi : Écrire dans des tables Apache Hudi à l'aide du AWS Glue connecteur personnalisé
- Google BigQuery : migration de données de Google vers Amazon S3 BigQuery à l'aide de connecteurs AWS Glue personnalisés
- Snowflake (JDBC) : Exécution de transformations de données à l'aide de Snowflake et AWS Glue
- SingleStore: Création d'un ETL rapide à l'aide SingleStore de et AWS Glue
- Salesforce : ingérez des données Salesforce dans Amazon S3 à l'aide du connecteur personnalisé CData JDBC avec - AWS Glue
- MongoDB : Création de AWS Glue tâches ETL Spark à l'aide d'Amazon DocumentDB (avec compatibilité MongoDB) et MongoDB
- Amazon Relational Database Service (Amazon RDS) : création de tâches ETL AWS Glue Spark en apportant vos propres pilotes JDBC pour Amazon RDS
- MySQL (JDBC) : https://github.com/aws-samples/aws-glue-samples//.scala blob/master/GlueCustomConnectors/development/Spark/SparkConnectorMySQL

Développement de AWS Glue connecteurs pour AWS Marketplace

En tant que AWS partenaire, vous pouvez créer des connecteurs personnalisés et les télécharger AWS Marketplace pour les vendre aux AWS Glue clients.

Le processus de développement du code du connecteur est le même que pour les connecteurs personnalisés, mais le processus de téléchargement et de vérification du code du connecteur

est plus détaillé. Reportez-vous aux instructions de la section Création de connecteurs pour AWS Marketplace sur le GitHub site Web.

Restrictions d'utilisation des connecteurs et des connexions dans AWS Glue Studio Glue Studio

Lorsque vous utilisez des connecteurs personnalisés ou des connecteurs provenant de AWS Marketplace, prenez note des restrictions suivantes :

- L'API testConnection n'est pas prise en charge avec les connexions créées pour les connecteurs personnalisés.
- Le chiffrement du mot de passe de connexion Data Catalog n'est pas pris en charge avec les connecteurs personnalisés.
- Vous ne pouvez pas utiliser de marque-pages de tâche si vous spécifiez un prédicat de filtre pour un nœud de source de données qui utilise un connecteur JDBC.
- La création d'une connexion Marketplace n'est pas prise en charge en dehors de l'interface AWS Glue Studio utilisateur.

Test d'une connexion AWS Glue

Comme meilleure pratique, avant d'utiliser une connexion AWS Glue dans une tâche ETL, utilisez la console AWS Glue pour tester la connexion. AWS Glue utilise les paramètres de votre connexion pour confirmer qu'il peut accéder à votre magasin de données et signale toute erreur. Pour plus d'informations sur les connexions AWS Glue, consultez Connexion aux données.

Pour tester une connexion AWS Glue

1. Connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/.
2. Dans le volet de navigation, sous Catalogue de données, choisissez Connexions. Vous pouvez également sélectionner Connexions aux données au-dessus de Catalogue de données dans le volet de navigation.
3. Dans Connexions, cochez la case en regard de la connexion souhaitée, puis choisissez Actions. Dans le menu déroulant, choisissez Tester la connexion.

4. Dans la boîte de dialogue Tester la connexion, sélectionnez un rôle ou choisissez Create IAM role pour accéder à la console Gestion des identités et des accès AWS (IAM) afin de créer un nouveau rôle. Le rôle doit disposer d'autorisations dans le magasin de données.
5. Choisissez Confirmer.

   Le test commence et peut prendre plusieurs minutes. Si le test échoue, choisissez Dépanner pour voir les étapes à suivre pour résoudre le problème.
6. Choisissez Logs pour afficher les connexions CloudWatch. Vous devez disposer des autorisations IAM requises pour afficher les journaux. Pour plus d'informations, consultez la section Politiques AWS gérées (prédéfinies) pour les CloudWatch journaux dans le guide de l'utilisateur Amazon CloudWatch Logs.

Configuration des AWS appels pour qu'ils passent par votre VPC

Le paramètre spécial de tâche vous `disable-proxy-v2` permet d'acheminer vos appels vers des services tels qu'Amazon S3 et AWS Glue via votre VPC. CloudWatch Par défaut, AWS Glue utilise un proxy local pour envoyer du trafic via le AWS Glue VPC afin de télécharger des scripts et des bibliothèques depuis Amazon S3, d'envoyer des demandes de publication de journaux et de métriques, et d'envoyer des demandes d'accès à AWS Glue des catalogues de données. CloudWatch Ce proxy permet à la tâche de fonctionner normalement même si votre VPC ne configure pas une route appropriée vers d'autres AWS services, tels qu'Amazon S3 CloudWatch, et. AWS Glue AWS Glue propose désormais un paramètre vous permettant de désactiver ce comportement. Pour plus d'informations, consultez la section Paramètres du Job utilisés par AWS Glue. AWS Glue continuera à utiliser un proxy local pour publier CloudWatch les journaux de vos AWS Glue tâches.

> **Note**
>
> - Cette fonctionnalité est prise en charge pour les AWS Glue tâches avec AWS Glue la version 2.0 et supérieure. Lorsque vous utilisez cette fonction, vous devez vous assurer que votre VPC a configuré un acheminement vers Amazon S3 via un point de terminaison d'un VPC de service ou via un NAT.
> - Le paramètre de tâche obsolète `disable-proxy` achemine uniquement vos appels vers Amazon S3 pour le téléchargement de scripts et de bibliothèques via votre VPC. Il est recommandé d'utiliser le nouveau paramètre `disable-proxy-v2` à la place.

Exemple d'utilisation

Créez un AWS Glue emploi avec `disable-proxy-v2` :

```
aws glue create-job \
    --name no-proxy-job \
    --role GlueDefaultRole \
    --command "Name=glueetl,ScriptLocation=s3://my-bucket/glue-script.py" \
    --connections Connections="traffic-monitored-connection" \
    --default-arguments '{"--disable-proxy-v2" : "true"}'
```

Connexion à un magasin de données JDBC dans un VPC

Généralement, vous créez des ressources dans Amazon Virtual Private Cloud (Amazon VPC) afin qu'elles ne soient pas accessibles via l'Internet public. Par défaut, AWS Glue impossible d'accéder aux ressources d'un VPC. Pour permettre l'accès AWS Glue aux ressources de votre VPC, vous devez fournir des informations de configuration supplémentaires spécifiques au VPC, notamment le sous-réseau et le groupe de sécurité du VPC. IDs IDs AWS Glue utilise ces informations pour configurer des interfaces réseau élastiques qui permettent à votre fonction de se connecter en toute sécurité à d'autres ressources de votre VPC privé.

Lorsque vous utilisez un point de terminaison d'un VPC, ajoutez-le à votre table de routage. Pour de plus amples informations, consultez Creating an interface VPC endpoint for AWS Glue et Conditions préalables.

Lorsque vous utilisez le chiffrement dans le catalogue de données, créez le point de terminaison de l'interface KMS et ajoutez-le à votre table de routage. Pour de plus amples informations, consultez Creating a VPC endpoint for AWS KMS.

Accès à des données de VPC par l'intermédiaire d'interfaces réseau Elastic

Lorsque vous AWS Glue vous connectez à un magasin de données JDBC dans un VPC, il AWS Glue crée une interface réseau élastique (avec le préfixe`Glue_`) dans votre compte pour accéder à vos données VPC. Vous ne pouvez pas supprimer cette interface réseau tant qu'elle est connectée à AWS Glue. Dans le cadre de la création de l'interface elastic network, AWS Glue associez un ou plusieurs groupes de sécurité à celle-ci. Pour AWS Glue permettre de créer l'interface réseau, les groupes de sécurité associés à la ressource doivent autoriser l'accès entrant avec une règle source. Cette règle contient un groupe de sécurité associé à la ressource. Cela permet à l'interface réseau Elastic d'accéder à votre magasin de données avec le même groupe de sécurité.

AWS Glue Pour autoriser la communication avec ses composants, spécifiez un groupe de sécurité doté d'une règle d'autoréférencement entrant pour tous les ports TCP. En créant une règle avec référence circulaire, vous pouvez restreindre la source au même groupe de sécurité dans le VPC et ne pas l'ouvrir à tous les réseaux. Le groupe de sécurité par défaut pour votre VPC peut déjà avoir une règle entrante avec référence circulaire pour `ALL Traffic`.

Vous pouvez créer des règles dans la console Amazon VPC. Pour mettre à jour les paramètres de règle via l' AWS Management Console, accédez à la console VPC (https://console.aws.amazon.com/vpc/) et sélectionnez le groupe de sécurité approprié. Spécifiez la règle entrante pour `ALL TCP` afin de disposer du même nom de groupe de sécurité en tant que source. Pour plus d'informations sur les règles des groupes de sécurité, consultez Groupes de sécurité pour votre VPC.

Chaque interface elastic network se voit attribuer une adresse IP privée issue de la plage d'adresses IP des sous-réseaux que vous spécifiez. Aucune adresse IP publique n'est attribuée à l'interface réseau. AWS Glue nécessite un accès à Internet (par exemple, pour accéder à AWS des services qui ne disposent pas de points de terminaison VPC). Vous pouvez configurer une instance NAT (Network Address Translation, traduction d'adresses réseau) dans votre VPC, ou utiliser la passerelle NAT Amazon VPC. Pour plus d'informations, veuillez consulter NAT Gateways (Passerelles NAT) dans le Guide de l'utilisateur Amazon VPC. Vous ne pouvez pas utiliser directement une passerelle Internet attachée à votre VPC en tant que routage dans votre table de routage de sous-réseau, car cela nécessite que l'interface réseau ait des adresses IP publiques.

Les attributs de réseau VPC `enableDnsHostnames` et `enableDnsSupport` doivent être définis sur true. Pour plus d'informations, consultez Utilisation de DNS avec votre VPC.

> **Important**
>
> Ne placez pas votre banque de données dans un sous-réseau public ou privé qui n'a pas accès à Internet. En revanche, attachez-le uniquement à des sous-réseaux privés disposant d'un accès Internet via une instance NAT ou une passerelle NAT Amazon VPC.

Propriétés d'interface réseau Elastic

Pour créer l'interface réseau Elastic, vous devez indiquer les propriétés suivantes :

VPC

Nom du VPC qui contient votre magasin de données.

Sous-réseau

Sous-réseau du VPC qui contient votre magasin de données.

Groupes de sécurité

Les groupes de sécurité qui sont associés à votre magasin de données. AWS Glue associe ces groupes de sécurité à l'interface réseau Elastic qui est attachée à votre sous-réseau VPC. Pour autoriser des composants AWS Glue à communiquer, ainsi que pour empêcher l'accès depuis d'autres réseaux, au moins un groupe de sécurité choisi doit spécifier une règle entrante avec référence circulaire pour tous les ports TCP.

Pour en savoir plus sur la gestion d'un VPC avec Amazon Redshift, veuillez consulter Gestion des clusters dans un Amazon Virtual Private Cloud (VPC).

Pour plus d'informations sur la gestion d'un VPC avec Amazon Relational Database Service (Amazon RDS), veuillez consulter Utilisation d'une instance de base de données Amazon RDS dans un VPC.

Utilisation d'une connexion MongoDB ou MongoDB Atlas

Après avoir créé une connexion pour MongoDB ou MongoDB Atlas, vous pouvez utiliser la connexion dans votre tâche ETL. Vous créez une table dans le AWS Glue Data Catalog et spécifiez la connexion MongoDB ou MongoDB Atlas pour l'`connection`attribut de la table.

AWS Glue stocke votre connexion `url` et vos informations d'identification dans la connexion MongoDB. Les formats d'URI de connexion sont les suivants :

- Pour MongoDB : mongodb://host:port/database. L'hôte peut être un nom d'hôte, une adresse IP ou un socket de domaine UNIX. Si la chaîne de connexion ne spécifie aucun port, elle utilise le port MongoDB par défaut, 27017.
- Pour MongoDB Atlas : mongodb+srv://server.example.com/database. L'hôte peut être un nom d'hôte qui suit et correspond à un enregistrement DNS SRV. Le format SRV ne nécessite pas de port et utilisera le port MongoDB par défaut, 27017.

En outre, vous pouvez spécifier les options dans votre script de tâche. Pour de plus amples informations, veuillez consulter the section called "Connexion MongoDB".

Analyse d'un magasin de données Amazon S3 à l'aide d'un point de terminaison d'un VPC

Pour des raisons de sécurité, d'audit ou de contrôle, vous pouvez souhaiter que votre magasin de données Amazon S3 ou vos tables du catalogue de données basées sur Amazon S3 soient uniquement accessibles via un environnement Amazon Virtual Private Cloud (Amazon VPC). Cette rubrique décrit la manière de créer et de tester une connexion au magasin de données Amazon S3 ou des tables du catalogue de données basées sur Amazon S3 dans un point de terminaison d'un VPC à l'aide du type de connexion `Network`.

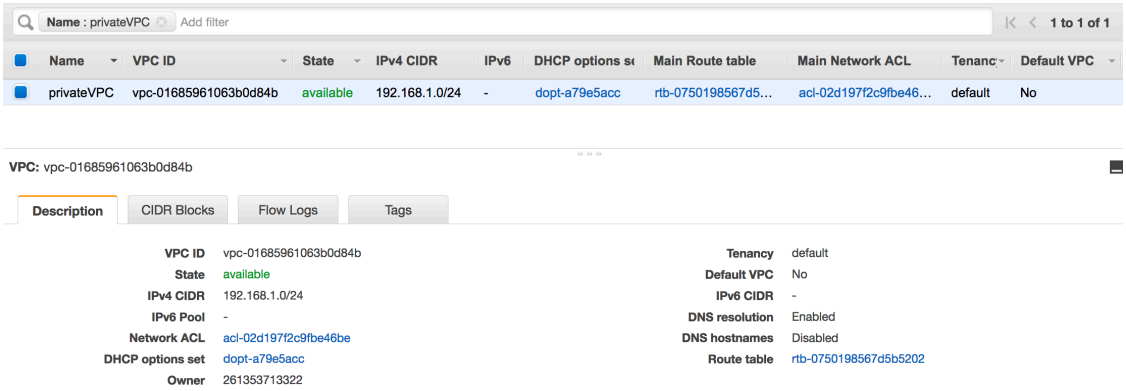
Effectuez les tâches suivantes pour exécuter un crawler sur le magasin de données :

- the section called “Conditions préalables”
- the section called “Création de la connexion à Amazon S3”
- the section called “Test de la connexion à Amazon S3”
- the section called “Création d'un crawler pour un magasin de données Amazon S3”
- the section called “Exécution d'un crawler”

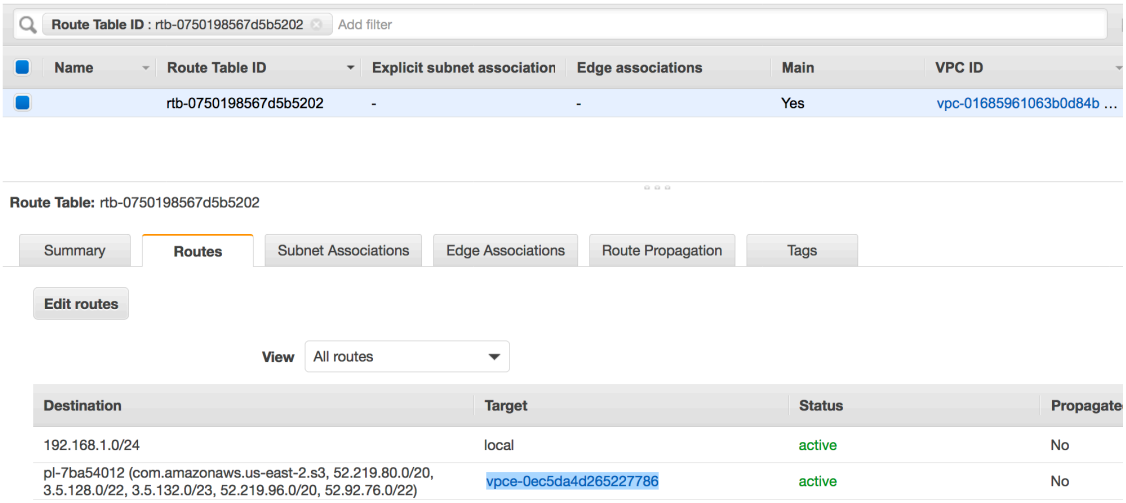
Conditions préalables

Vérifier que vous avez satisfait à ces conditions préalables pour la configuration de votre magasin de données Amazon S3 ou vos tables du catalogue de données basées sur Amazon S3 pour y accéder via un environnement Amazon Virtual Private Cloud (Amazon VPC).

- Un VPC configuré. Par exemple : vpc-01685961063b0d84b. Pour plus d'informations, consultez Mise en route avec Amazon VPC dans le Guide de l'utilisateur Amazon VPC.
- Un point de terminaison Amazon S3 attaché au VPC. Par exemple : vpc-01685961063b0d84b. Pour de plus amples informations, veuillez consulter Points de terminaison pour Amazon S3 dans le Guide de l'utilisateur Amazon VPC.



- Entrée d'acheminement pointant vers le point de terminaison d'un VPC. Par exemple vpce-0ec5da4d265227786 dans la table de routage utilisée par le point de terminaison d'un VPC (vpce-0ec5da4d265227786).



- Une ACL réseau attachée au VPC autorise le trafic.
- Un groupe de sécurité attaché au VPC autorise le trafic.

Création de la connexion à Amazon S3

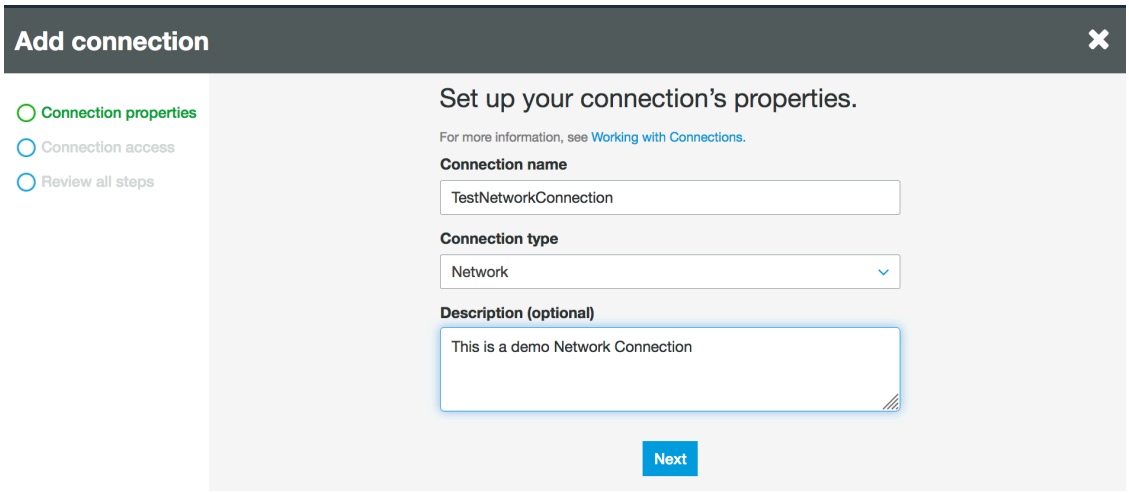
Généralement, vous créez des ressources dans Amazon Virtual Private Cloud (Amazon VPC) afin qu'elles ne soient pas accessibles via l'Internet public. Par défaut, AWS Glue impossible d'accéder aux ressources d'un VPC. Pour permettre l'accès AWS Glue aux ressources de votre VPC, vous devez fournir des informations de configuration supplémentaires spécifiques au VPC, notamment le sous-réseau et le groupe de sécurité du VPC. IDs IDs Pour créer une connexion `Network`, vous devez spécifier les informations suivantes :

- ID d'un VPC

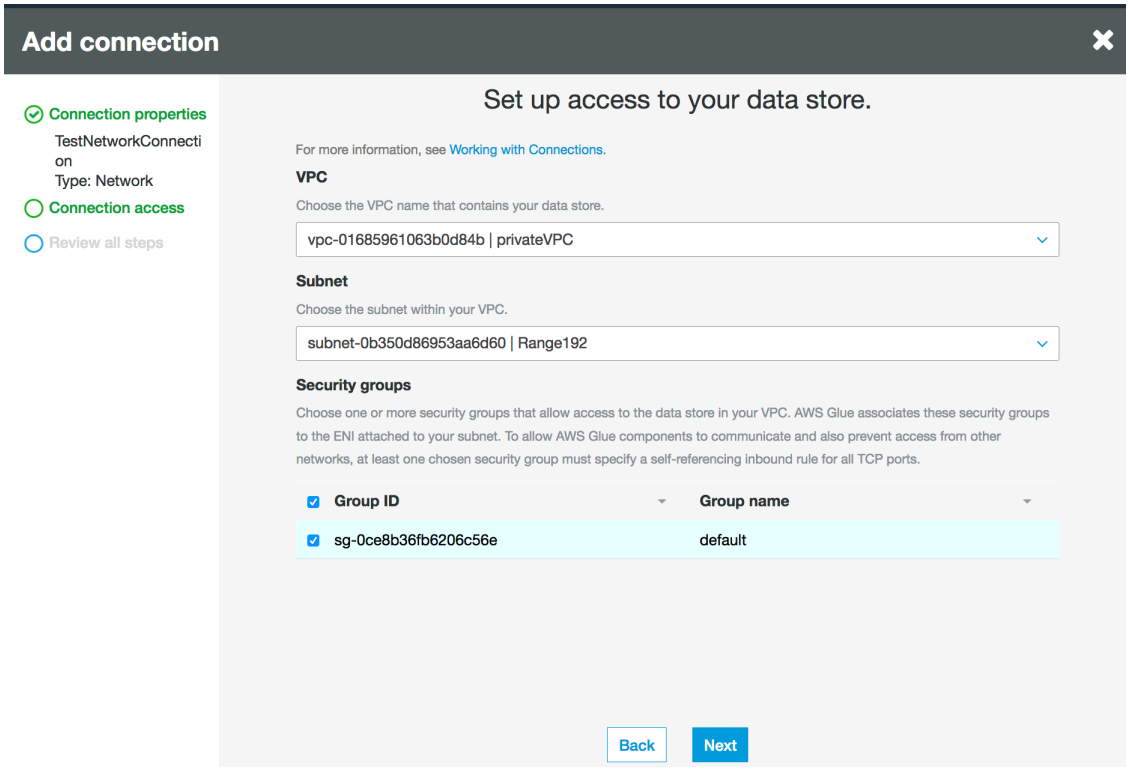
- Un sous-réseau au sein du VPC
- Un groupe de sécurité

Pour configurer une connexion `Network` :

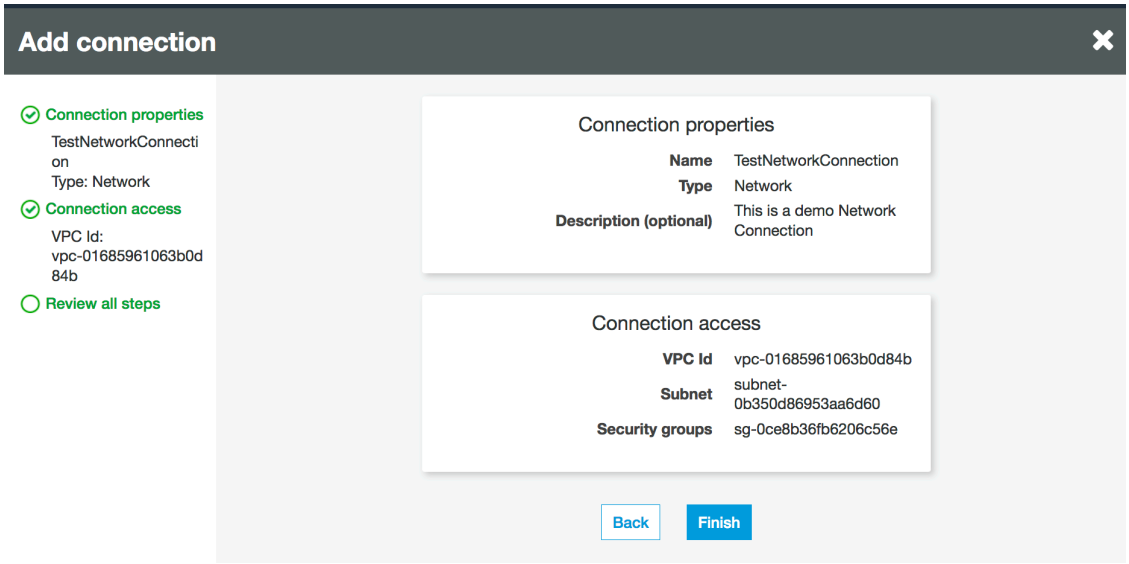
1. Choisissez Add connection (Ajouter une connexion) dans le panneau de navigation de la console AWS Glue .
2. Saisissez le nom de la connexion, puis choisissez Network (Réseau) comme type de connexion. Choisissez Suivant.



3. Configurez les informations du VPC, du sous-réseau et des groupes de sécurité.
   - VPC : choisissez le nom du VPC qui contient votre magasin de données.
   - Subnet (Sous-réseau) : choisissez le sous-réseau au sein de votre VPC.
   - Security groups (Groupes de sécurité) : choisissez un ou plusieurs groupes de sécurité qui autorisent l'accès au stockage des données dans votre VPC.



4. Choisissez Suivant.
5. Vérifiez les informations de connexion et choisissez Finish (Terminer).



Test de la connexion à Amazon S3

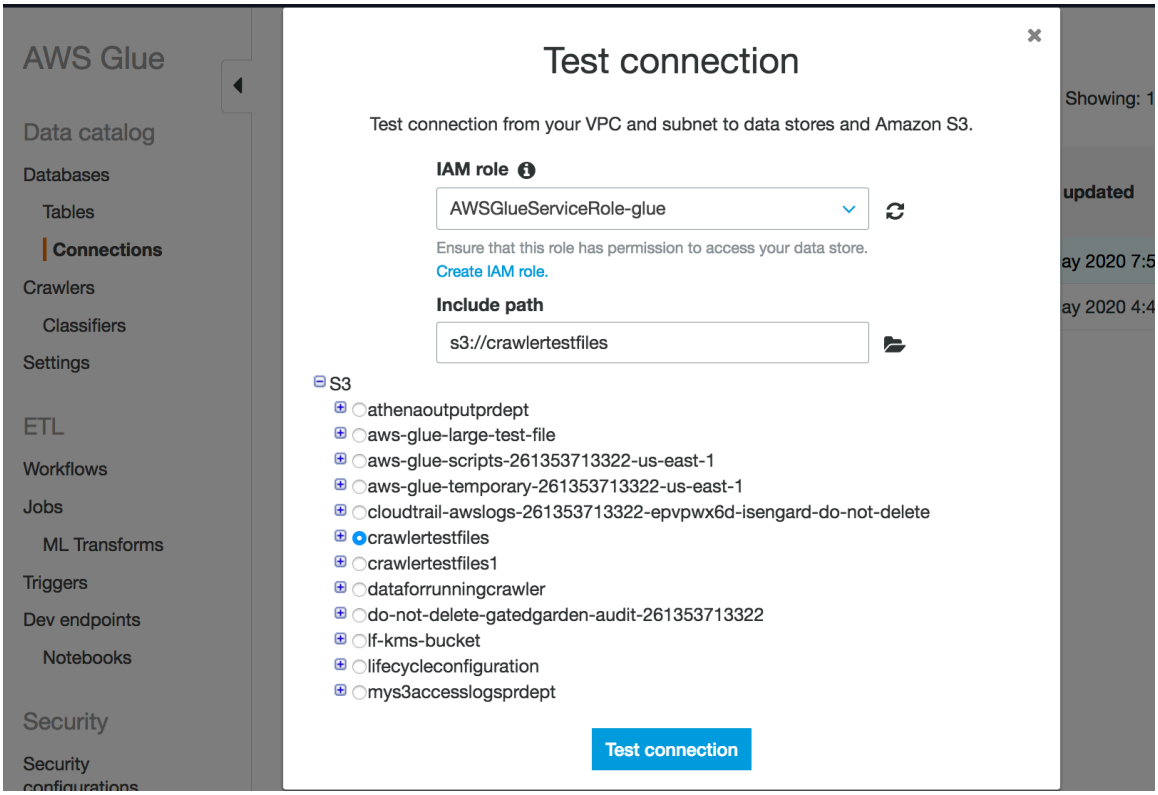
Une fois que vous avez créé votre connexion `Network`, vous pouvez tester la connectivité à votre magasin de données Amazon S3 dans un point de terminaison d'un VPC.

Les erreurs suivantes peuvent se produire lors du test d'une connexion :

- INTERNET CONNECTION ERROR (Erreur de connexion Internet) : indique un problème de connexion Internet
- INVALID BUCKET ERROR (Erreur de compartiment non valide) : indique un problème avec le compartiment Amazon S3
- S3 CONNECTION ERROR (Erreur de connexion S3) : indique un échec de connexion à Amazon S3
- INVALID CONNECTION TYPE (Type de connexion non valide) : indique que le type de connexion n'a pas la valeur attendue, `NETWORK`
- INVALID CONNECTION TEST TYPE (Type de test de connexion non valide) : indique un problème avec le type de test de connexion réseau
- INVALID TARGET (Cible non valide) : indique que le compartiment Amazon S3 n'a pas été spécifié correctement

Pour tester une connexion `Network` :

1. Sélectionnez la connexion Network (Réseau) dans la console AWS Glue .
2. Choisissez Test connection (Tester la connexion).
3. Choisissez le rôle IAM que vous avez créé à l'étape précédente et spécifiez un compartiment Amazon S3.
4. Choisissez Test connection (Tester la connexion) pour démarrer le test. Cela peut prendre quelques instants pour afficher le résultat.



Si vous recevez une erreur, effectuez les vérifications suivantes :

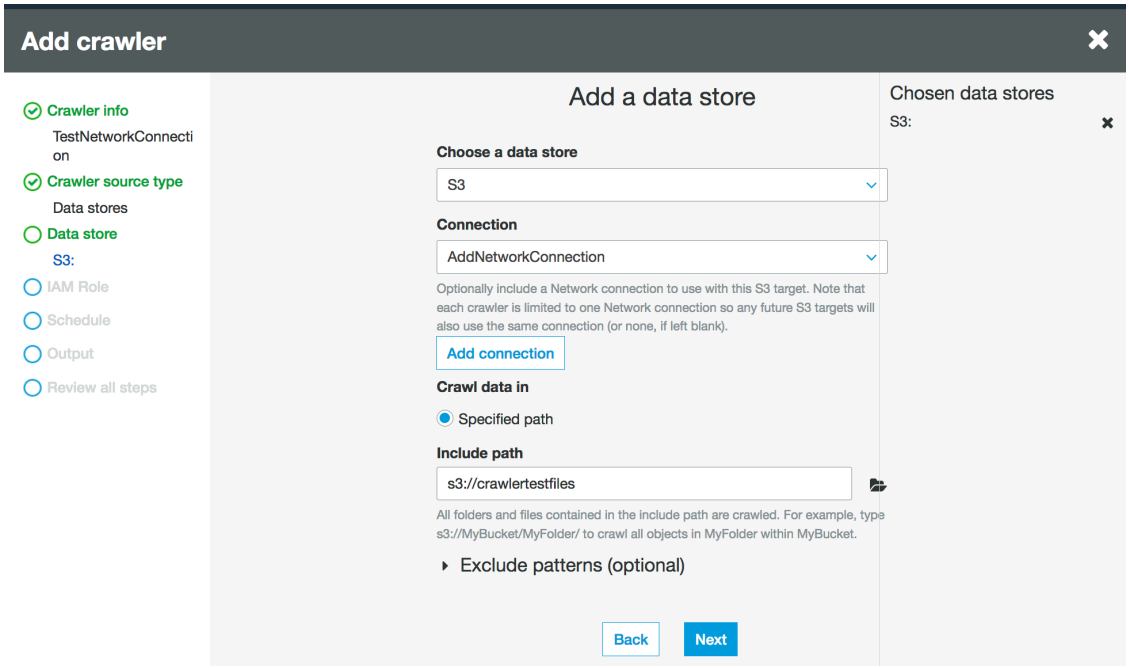
- Les privilèges corrects sont fournis au rôle sélectionné.
- Le compartiment Amazon S3 approprié est fourni.
- Les groupes de sécurité et la liste ACL réseau autorisent le trafic entrant et sortant requis.
- Le VPC que vous avez spécifié est connecté à un point de terminaison d'un VPC Amazon S3.

Après avoir testé la connexion, vous pouvez créer un crawler.

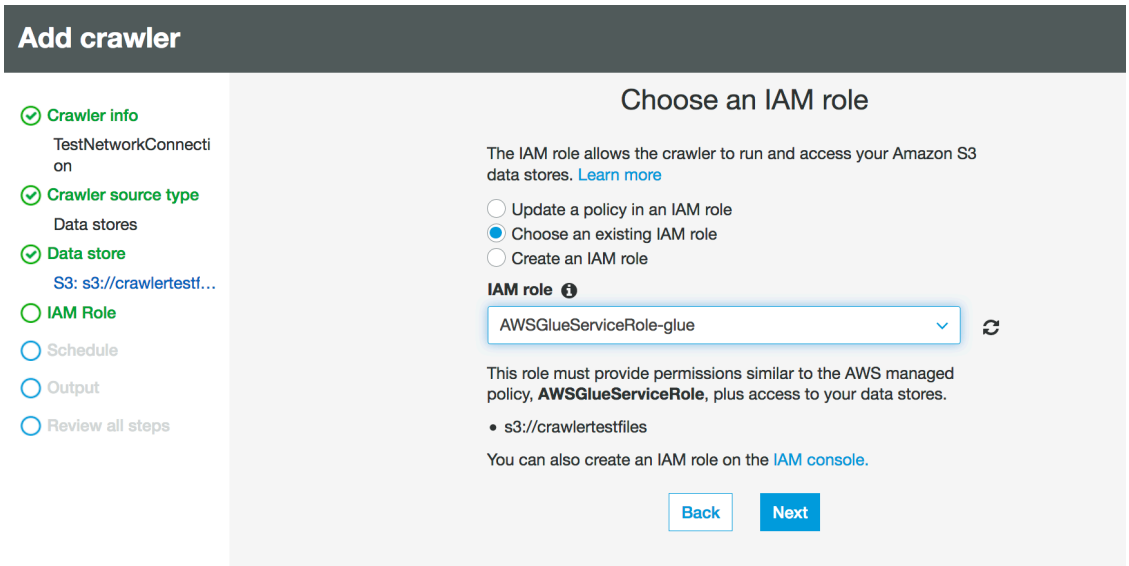
Création d'un crawler pour un magasin de données Amazon S3

Vous pouvez désormais créer un crawler qui spécifie la connexion `Network` que vous avez créée. Pour plus d'informations sur la création d'un crawler, veuillez consulter Configuration d'un robot.

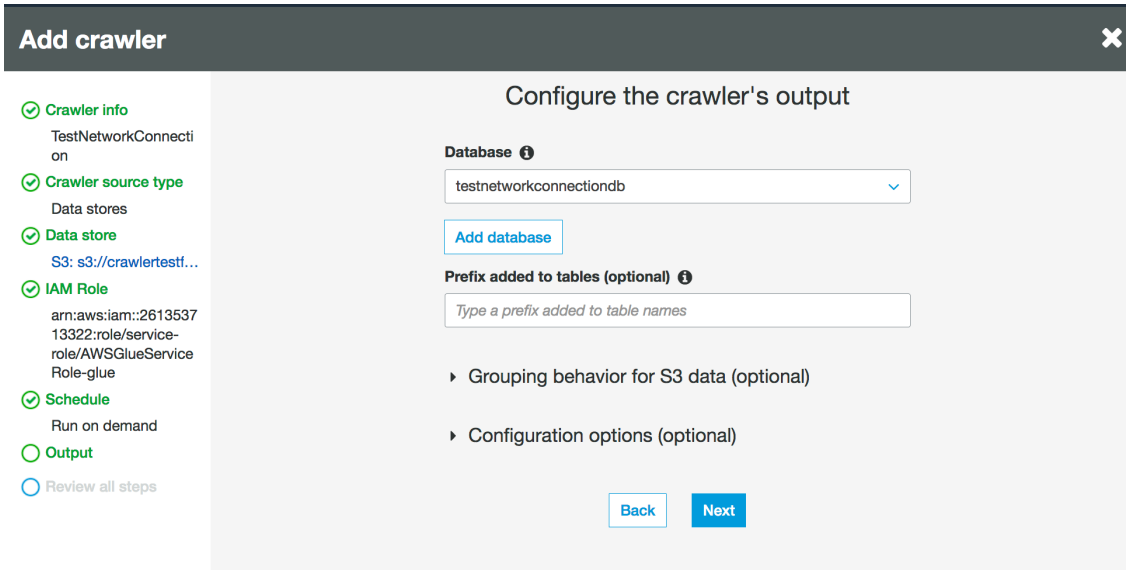
1. Commencez par choisir Crawlers dans le volet de navigation de la AWS Glue console.
2. Choisissez Add crawler (Ajouter un crawler).
3. Indiquez le nom du crawler, puis choisissez Next (Suivant).
4. Lorsque vous êtes invité à indiquer la source de données, sélectionnez S3, puis spécifiez le préfixe du compartiment Amazon S3 et la connexion que vous avez créée plus tôt.



5. Si nécessaire, ajoutez un autre magasin de données sur la même connexion réseau.
6. Choisissez le rôle IAM. Le rôle IAM doit autoriser l'accès au service AWS Glue et au compartiment Amazon S3. Pour de plus amples informations, veuillez consulter the section called "Configuration d'un robot".



7. Définissez le planificateur pour le crawler.
8. Choisissez une base de données existante dans le catalogue de données ou créez-en une.

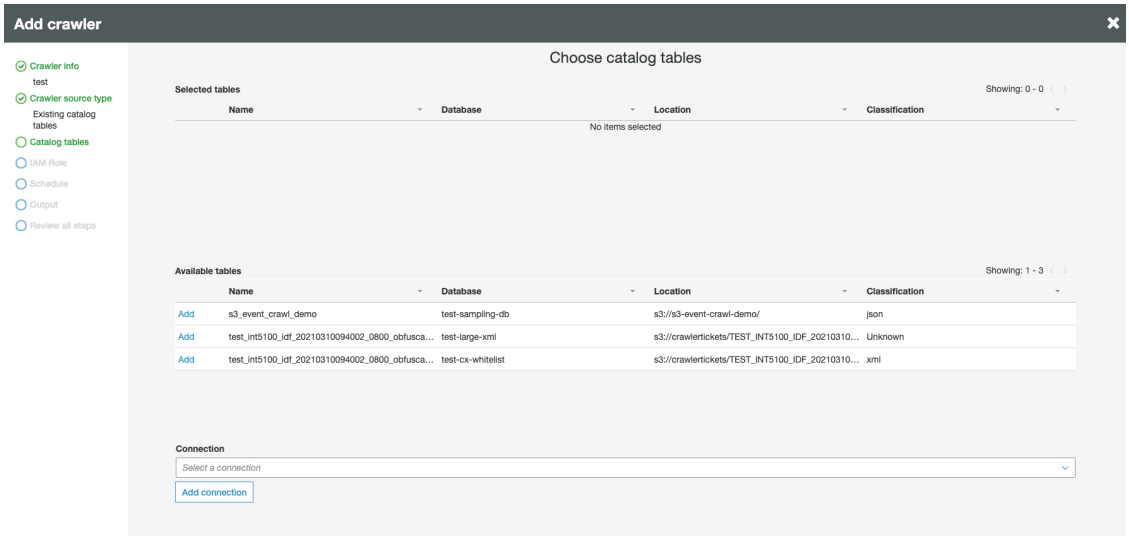


9. Terminez la configuration restante.

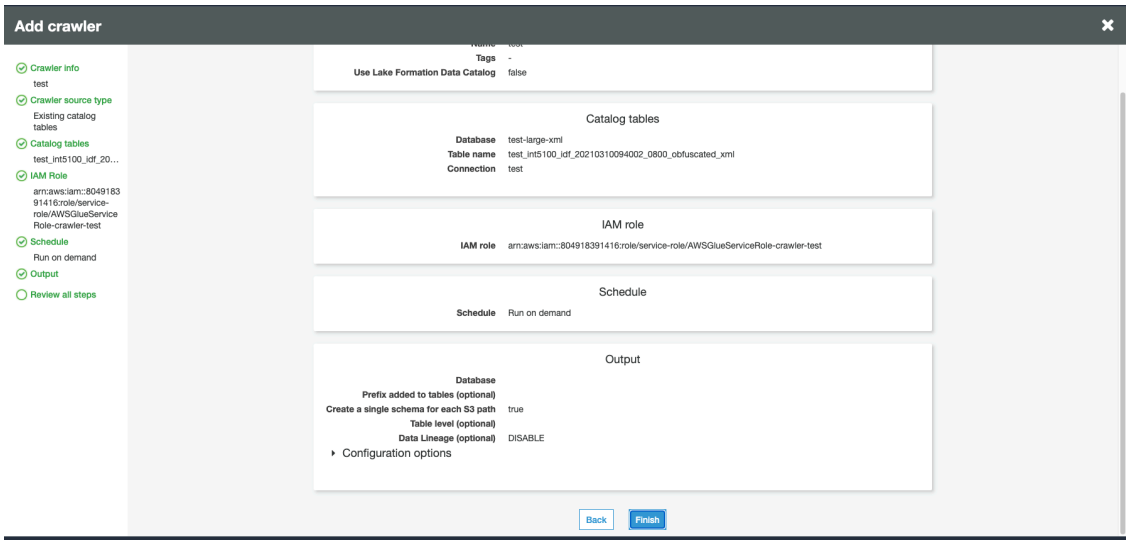
Création d'un crawler pour les tables du catalogue de données basées sur Amazon S3

Vous pouvez désormais créer un crawler qui spécifie la connexion `Network` que vous avez créée et un type de source du catalogue. Pour plus d'informations sur la création d'un crawler, veuillez consulter Configuration d'un robot.

1. Commencez par choisir Crawlers dans le volet de navigation de la AWS Glue console.
2. Choisissez Add crawler (Ajouter un crawler).
3. Indiquez le nom du crawler, puis choisissez Next (Suivant).
4. Lorsque vous êtes invité à indiquer le type de source du crawler, sélectionnez Existing catalog tables (Tables catalogue existantes), et indiquez les tables de catalogue existantes à analyser dans la liste des tables disponibles.

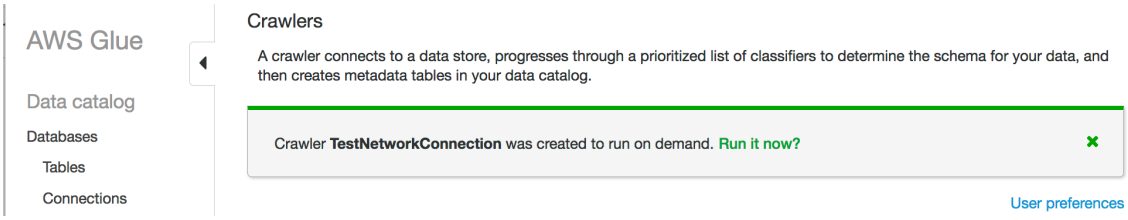


5. Choisissez le rôle IAM. Le rôle IAM doit autoriser l'accès au service AWS Glue et au compartiment Amazon S3. Pour de plus amples informations, veuillez consulter the section called “Configuration d'un robot”.
6. Définissez le planificateur pour le crawler.
7. Choisissez une base de données existante dans le catalogue de données ou créez-en une.
8. Terminez la configuration restante et passez en revue vos étapes.



Exécution d'un crawler

Exécutez votre crawler.



Résolution des problèmes

Pour résoudre les problèmes liés aux compartiments Amazon S3 à l'aide d'une passerelle VPC, veuillez consulter Pourquoi ne puis-je pas me connecter à un compartiment S3 à l'aide d'un point de terminaison d'un VPC passerelle ?

Travailler avec des AWS Lake Formation données protégées

Accès complet aux tables

AWS Glue prend en charge l'accès complet aux tables (FTA) pour les tables AWS Lake Formation protégées. Cela permet à vos tâches ETL de lire et d'écrire des données avec des autorisations complètes pour les tables.

Conditions préalables

- Rôles et autorisations IAM appropriées
- AWS Lake Formation configuré pour votre catalogue de données
- Types de table compatibles (Hive ou Iceberg)

Considérations clés

Autorisations requises

- Autorisation IAM `lakeformation:GetDataAccess`
- AWS Lake Formation autorisations de table
- Autorisations d'accès aux compartiments Amazon S3

Types de table pris en charge

- Tables Hive

- Tables Iceberg

Limitations

- Non compatible avec Spark Streaming
- Ne peut pas être utilisé simultanément avec le contrôle précis des accès
- Ne prend pas en charge les tables Delta ou Hudi

Bonnes pratiques

1. S'assurer que les rôles IAM sont correctement configurés avant l'exécution de la tâche
2. Tester les autorisations d'accès dans un environnement de développement
3. Surveiller les journaux d'exécution des tâches pour détecter les problèmes liés aux autorisations
4. Conserver une documentation claire des modèles d'accès

Résolution des problèmes

Les problèmes courants comprennent :

- Autorisations IAM manquantes
- AWS Lake Formation Configuration incorrecte
- Problèmes de compatibilité des types de table

Pour obtenir des instructions de configuration complètes et des détails de configuration, consultez la section Utilisation AWS Lake Formation avec un accès complet aux tables.

Résolution des problèmes de connexion dans AWS Glue

Lorsqu'un AWS Glue robot ou une tâche utilise les propriétés de connexion pour accéder à un magasin de données, vous pouvez rencontrer des erreurs lorsque vous essayez de vous connecter. AWS Glueutilise des adresses IP privées dans le sous-réseau lorsqu'il crée des interfaces réseau élastiques dans le cloud privé virtuel (VPC) et le sous-réseau que vous avez spécifiés. Les groupes de sécurité spécifiés dans la connexion sont appliqués sur chacune des interfaces réseau Elastic. Vérifiez si les groupes de sécurité permettent l'accès sortant et autorisent la connectivité au cluster de base de données.

En outre, Apache Spark nécessite une connexion bidirectionnelle entre nœuds de pilote et nœuds d'exécuteur. L'un des groupes de sécurité doit autoriser les règles entrantes sur tous les ports TCP. Vous pouvez l'empêcher d'être ouvert sur le monde en limitant la source du groupe de sécurité à elle-même avec un groupe de sécurité à auto-référencement.

Voici quelques actions typiques que vous pouvez adopter pour résoudre les problèmes de connexion :

- Vérifiez l'adresse du port de votre connexion.
- Vérifiez la chaîne de nom d'utilisateur et de mot de passe de votre connexion ou votre secret.
- Pour un magasin de données JDBC, vérifier qu'il autorise les connexions entrantes.
- Vérifiez que votre magasin de données peut être accessible au sein de votre VPC.
- Si vous stockez vos informations de connexion à l'aide de AWS Secrets Manager, assurez-vous que votre rôle IAM pour AWS Glue est autorisé à accéder à votre secret. Pour plus d'informations, consultez la rubrique Exemple : Autorisation pour récupérer des valeurs de secrets dans le Guide de l'utilisateur AWS Secrets Manager . En fonction de votre configuration réseau, vous devrez peut-être également créer un point de terminaison d'un VPC pour établir une connexion privée entre votre VPC et gestionaire de secrets Pour plus d'informations, consultez la section Utilisation d'un point de AWS Secrets Manager terminaison VPC.

Tutoriel : Utilisation du AWS Glue connecteur pour Elasticsearch

Elasticsearch est un célèbre moteur open source de recherche et d'analyse qui peut être utilisé, entre autres, pour l'analyse de journaux, la surveillance en temps réel des applications et l'analyse des flux de clics. Vous pouvez l'utiliser OpenSearch comme magasin de données pour vos tâches d'extraction, de transformation et de chargement (ETL) en configurant le AWS Glue connecteur pour Elasticsearch dans. AWS Glue Studio Ce connecteur est disponible gratuitement sur AWS Marketplace.

> **Note**
>
> Le connecteur Elasticsearch Spark AWS Marketplace est devenu obsolète. Veuillez utiliser le connecteur AWS Glue pour Elasticsearch à la place.

Dans ce didacticiel, nous allons vous montrer comment vous connecter à vos nœuds Amazon OpenSearch Service en un minimum d'étapes.

Rubriques

- Conditions préalables
- Étape 1 : (Facultatif) Créez un AWS secret pour les informations de votre OpenSearch cluster
- Étape 2 : S'abonner au connecteur
- Étape 3 : activer le connecteur AWS Glue Studio et créer une connexion
- Étape 4 : Configurer un rôle IAM pour votre tâche ETL
- Étape 5 : créer une tâche qui utilise la OpenSearch connexion
- Étape 6 : Exécuter la tâche

Conditions préalables

Pour utiliser ce didacticiel, vous devez disposer des éléments suivants :

- Accès à AWS Glue Studio
- Accès à un OpenSearch cluster dans le AWS cloud
- (Facultatif) Accès à AWS Secrets Manager.

Étape 1 : (Facultatif) Créez un AWS secret pour les informations de votre OpenSearch cluster

Pour stocker et utiliser en toute sécurité vos informations d'identification de connexion, enregistrez vos informations d'identification au format AWS Secrets Manager. Le secret que vous créez sera utilisé plus tard dans le didacticiel par la connexion. Les paires clé-valeur d'identification seront introduites dans le AWS Glue Connector for Elasticsearch en tant qu'options de connexion normales.

Pour plus d'informations sur la création de secrets, veuillez consulter la rubrique Création et gestion des secrets avec AWS Secrets Manager dans le Guide de l'utilisateur AWS Secrets Manager .

Pour créer un AWS secret

1. Connectez-vous à la console AWS Secrets Manager.
2. Sur la page de présentation du service ou la page de la liste Secrets, sélectionnez Store a new secret (Stocker un nouveau secret).

3. Sur la page Store a new secret (Stocker un nouveau secret), sélectionnez Other type of secret (Autre type de secret). Cette option signifie que vous devez fournir la structure et les détails de votre secret.
4. Ajoutez une paire clé/valeur pour le nom d'utilisateur du OpenSearch cluster. Par exemple :

   `es.net.http.auth.user`: *`username`*
5. Sélectionnez + Add row (+ Ajouter une ligne), puis saisissez une autre paire clé-valeur pour le mot de passe. Par exemple :

   `es.net.http.auth.pass`: *`password`*
6. Choisissez Suivant.
7. Saisissez un nom de secret. Par exemple : my-es-secret. Vous pouvez ajouter une description si vous le souhaitez.

   Enregistrez le nom secret, qui sera utilisé plus tard dans ce didacticiel, puis sélectionnez Next (Suivant).
8. Sélectionnez à nouveau Next (Suivant), puis Store (Stocker) pour créer le secret.

Étape suivante

Étape 2 : S'abonner au connecteur

Étape 2 : S'abonner au connecteur

Le AWS Glue connecteur pour Elasticsearch est disponible gratuitement à l'adresse. AWS Marketplace

Pour vous abonner au AWS Glue Connector for Elasticsearch sur AWS Marketplace

1. Si vous n'avez pas encore configuré votre AWS compte pour utiliser le License Manager, procédez comme suit :

   a. Ouvrez la AWS License Manager console à l'adresse https://console.aws.amazon.com/license-manager.

   b. Sélectionnez Create customer managed license (Créer une licence gérée par le client).

   c. Dans la fenêtre Autorisations IAM (configuration unique), choisissez J'accorde AWS License Manager les autorisations requises, puis choisissez Accorder les autorisations.

Si vous ne voyez pas cette fenêtre, cela signifie que vous avez déjà configuré les autorisations nécessaires.

2. Ouvrez la console AWS Glue Studio à l'adresse https://console.aws.amazon.com/gluestudio/.
3. Dans la AWS Glue Studio console, développez l'icône du menu (☰), puis choisissez Connectors dans le volet de navigation.
4. Sur la page Connectors (Connecteurs), sélectionnez Go to AWS Marketplace (Accéder à MKT).
5. Dans AWS Marketplace la section Rechercher des AWS Glue Studio produits, saisissez AWS Glue Connector for Elasticsearch dans le champ de recherche, puis appuyez sur Entrée.
6. Sélectionnez le nom du connecteur, Connecteur AWS Glue pour Elasticsearch.
7. Sur la page produit du connecteur, utilisez les onglets pour afficher des informations sur le connecteur. Lorsque vous êtes prêt à continuer, sélectionnez Continue to Subscribe (Continuer pour s'abonner).
8. Lisez les conditions d'utilisation du logiciel. Cliquez sur Accept Terms (Accepter les conditions d'utilisation).
9. Une fois le processus d'abonnement terminé, la notification suivante apparaît : « Merci de vous être abonné à ce produit ! Vous pouvez à présent configurer votre logiciel. » Au-dessus de la bannière se trouve le bouton Continue to Configuration (Continuer à configurer). Choisissez Continue to Configuration (Continuer vers Configuration).
10. Choisissez l'option remplissage sur la page Configure this software (Configurer ce logiciel). Vous pouvez choisir entre AWS Glue 1.0/2.0 ou AWS Glue 3.0. Puis sélectionnez Continue to Launch (Continuer pour lancer).

Étape suivante

Étape 3 : activer le connecteur AWS Glue Studio et créer une connexion

Étape 3 : activer le connecteur AWS Glue Studio et créer une connexion

Après avoir choisi Continuer au lancement, la page Lancer ce logiciel s'affiche dans AWS Marketplace. Après avoir utilisé le lien pour activer le connecteur AWS Glue Studio, vous créez une connexion.

Pour déployer le connecteur et créer une connexion dans AWS Glue Studio

1. Sur la page Lancer ce logiciel dans la AWS Marketplace console, choisissez Instructions d'utilisation, puis cliquez sur le lien dans la fenêtre qui apparaît.

   Votre navigateur est redirigé vers la page de connexion de la AWS Glue Studio console Créer une place de marché.
2. Saisissez un nom pour la connexion. Par exemple : my-es-connection.
3. Dans la section Connection access (Accès à la connexion), pour Connection credential type (Type d'informations d'identification de connexion), sélectionnez User name and password (Nom d'utilisateur et mot de passe).
4. Pour Secret AWS , saisissez le nom de votre secret. Par exemple : my-es-secret.
5. Dans la section Options réseau, entrez les informations VPC pour vous connecter au OpenSearch cluster.
6. Sélectionnez Create connection and activate connector (Créer une connexion et activer le connecteur).

Étape suivante

Étape 4 : Configurer un rôle IAM pour votre tâche ETL

Étape 4 : Configurer un rôle IAM pour votre tâche ETL

Lorsque vous créez la tâche AWS Glue ETL, vous spécifiez un rôle Gestion des identités et des accès AWS (IAM) pour la tâche à utiliser. Le rôle doit accorder l'accès à toutes les ressources utilisées par la tâche, y compris Amazon S3 (pour toutes les sources, cibles, scripts, fichiers de pilotes et répertoires temporaires), ainsi qu'aux AWS Glue Data Catalog objets.

Le rôle IAM assumé pour la tâche AWS Glue ETL doit également avoir accès au secret créé dans la section précédente. Par défaut, le rôle `AWSGlueServiceRole` géré par AWS n'a pas accès au secret. Pour configurer le contrôle d'accès pour vos secrets, consultez les rubriques Authentification et contrôle d'accès pour AWS Secrets Manager et Limiting Access to Specific Secrets.

Pour configurer un rôle IAM pour votre tâche ETL

1. Configurez les autorisations décrites dans the section called “Examinez les autorisations IAM nécessaires pour les tâches ETL”.

2. Configurez les autorisations supplémentaires nécessaires lors de l'utilisation de connecteurs avec AWS Glue Studio, comme décrit dansthe section called “Autorisations requises pour l'utilisation de connecteurs”.

Étape suivante

Étape 5 : créer une tâche qui utilise la OpenSearch connexion

Étape 5 : créer une tâche qui utilise la OpenSearch connexion

Après avoir créé un rôle pour votre tâche ETL, vous pouvez créer une tâche AWS Glue Studio qui utilise la connexion et le connecteur d'Open Spark ElasticSearch.

Si votre tâche est exécutée dans un Amazon Virtual Private Cloud (Amazon VPC), assurez-vous que le VPC est correctement configuré. Pour de plus amples informations, veuillez consulter the section called “Configuration d'un VPC pour votre tâche ETL”.

Pour créer une tâche qui utilise le connecteur Elasticsearch Spark

1. Dans AWS Glue Studio, choisissez Connectors.
2. Dans Your connections (Vos connexions), sélectionnez la connexion que vous venez de créer et sélectionnez Create job (Créer une tâche).
3. Dans l'éditeur de travail visuel, sélectionnez le nœud Data source (Source de données). Sur la droite, dans l'onglet Data source properties — Connector (Propriétés de la source de données — Connecteur), configurez des informations supplémentaires pour le connecteur.

   a. Sélectionnez Add schema (Ajouter un schéma) et saisissez le schéma de l'ensemble de données dans la source de données. Les connexions n'utilisent pas les tables stockées dans le catalogue de données, ce qui signifie que celui-ci AWS Glue Studio ne connaît pas le schéma des données. Vous devez fournir manuellement ces informations de schéma. Pour obtenir des instructions sur l'utilisation de l'éditeur de schéma, veuillez consulter the section called “Modifier le schéma d'un nœud de transformation personnalisé”.

   b. Développez Connection options (Options de connexion).

   c. Choisissez Ajouter une nouvelle option et entrez les informations requises pour le connecteur qui n'ont pas été saisies dans le AWS secret :

      - es.nodes : https ://< point de terminaison de OpenSearch domaine>
      - es.port : 443

- chemin : test
- es.nodes.wan.only. : true

Pour en savoir plus sur ces options de connexion, reportez-vous à https://www.elastic.co/guide/en/elasticsearch/hadoop/current/configuration.html.

4. Ajoutez un nœud cible au graphique.

   Votre cible de données peut être Amazon S3, ou elle peut utiliser les informations d'un AWS Glue Data Catalog ou d'un connecteur pour écrire des données dans un emplacement différent. Par exemple, vous pouvez utiliser une table Data Catalog pour écrire dans une base de données dans Amazon RDS, ou vous pouvez utiliser un connecteur comme cible de données pour écrire dans des magasins de données qui ne sont pas pris en charge de manière native dans AWS Glue.

   Si vous sélectionnez un connecteur pour votre cible de données, vous devez choisir une connexion créée pour ce connecteur. En outre, si le fournisseur de connecteurs l'exige, vous devez ajouter des options pour fournir des informations supplémentaires au connecteur. Si vous utilisez une connexion contenant des informations relatives à un AWS secret, il n'est pas nécessaire de fournir le nom d'utilisateur et le mot de passe d'authentification dans les options de connexion.

5. Ajoutez éventuellement des sources de données supplémentaires et un ou plusieurs nœuds de transformation, comme décrit dans the section called "Transformation de données avec des transformations gérées par AWS Glue".
6. Configurez les propriétés du travail comme décrit dans the section called "Modifier les propriétés de tâche", en commençant par l'étape 3, et enregistrez la tâche.

Étape suivante

Étape 6 : Exécuter la tâche

Étape 6 : Exécuter la tâche

Après avoir enregistré votre tâche, vous pouvez exécuter la tâche pour effectuer les opérations ETL.

Pour exécuter le job que vous avez créé pour le AWS Glue Connector for Elasticsearch

1. À l'aide de la AWS Glue Studio console, sur la page de l'éditeur visuel, choisissez Exécuter.

2. Dans la bannière de succès, sélectionnez Run Details (Détails de l'exécution), ou l'onglet Runs (Exécutions) de l'éditeur visuel pour afficher des informations sur l'exécution de la tâche.

Création AWS Glue d'emplois grâce à des sessions interactives

Les ingénieurs de données peuvent créer des tâches AWS Glue plus rapidement et plus facilement qu'avant en utilisant des séances interactives dans AWS Glue.

Rubriques

- Présentation de séances interactives AWS Glue
- Démarrage avec séances interactives AWS Glue
- Configuration des Séances interactives AWS Glue pour Jupyter et Blocs-notes AWS Glue Studio
- Conversion d'un script ou d'un bloc-notes en AWS Glue tâche
- Utilisation des opérations de streaming dans le cadre de sessions interactives AWS Glue
- Tarif des sessions interactives AWS Glue
- Développement et test de scripts AWS de tâches Glue localement
- Points de terminaison de développement

Présentation de séances interactives AWS Glue

Avec les séances interactives AWS Glue, vous pouvez rapidement créer, tester et exécuter des applications de préparation et d'analytique des données. Les séances interactives fournissent une interface programmatique et visuelle pour la création et le test de scripts d'extraction, transformation et chargement (ETL) pour la préparation de données. Les séances interactives exécutent les applications d'analytique Apache Spark et fournissent un accès à la demande à un environnement d'exécution Spark à distance. AWS Glue gère Spark sans serveur de manière transparente pour ces séances interactives.

Les sessions interactives sont flexibles, vous pouvez donc créer et tester des applications à partir de l'environnement de votre choix. Vous pouvez créer et utiliser des sessions interactives par le biais de l'API AWS Command Line Interface et de l'API. Vous pouvez utiliser des blocs-notes compatibles avec Jupyter pour créer et tester visuellement vos scripts de bloc-notes. Les sessions interactives fournissent un noyau Jupyter open source qui s'intègre presque partout où Jupyter le fait, y compris avec PyCharm IntelliJ et VS Code. IDEs Cela vous permet de coder dans votre environnement local et de l'exécuter de manière transparente sur le backend des séances interactives.

À l'aide de l'API des séances interactive, les clients peuvent exécuter par programmation des applications utilisant l'analyse Apache Spark sans avoir à gérer l'infrastructure Spark. Vous pouvez exécuter une ou plusieurs instructions Spark au cours d'une seule séance interactive.

Les séances interactives offrent donc un moyen plus rapide, moins coûteux et plus flexible de créer et d'exécuter des applications de préparation et d'analytique des données. Pour apprendre à utiliser les sessions interactives, consultez la documentation de cette section. Magics prises en charge par AWS Glue

Limitations

- Les signets de tâches ne sont pas pris en charge dans les séances interactives.
- La création de tâches de bloc-notes à l'aide du n' AWS Command Line Interface est pas prise en charge.
- AWS Glue Studio les ordinateurs portables ne supportent pas Scala.

Démarrage avec séances interactives AWS Glue

Ces sections décrivent comment exécuter des séances interactives AWS Glue localement.

Conditions préalables à la configuration locale des séances interactives

Les conditions suivantes sont requises pour installer des séances interactives :

- Les versions de Python prises en charge vont des versions 3.6 à 3.10 et ultérieures.
- Consultez les sections ci-dessous pour MacOS/Linux les instructions relatives à Windows.
- Consultez la documentation sur la tarification des sessions interactives pour comprendre la structure des coûts.

Installation de Jupyter et de sessions AWS Glue interactives (noyaux Jupyter)

Pour installer le noyau localement, procédez comme suit.

La commande `install-glue-kernels` installe les KernelSpec Jupyter pour les noyaux Pyspark et Spark et installe également les logos dans le bon répertoire.

```
pip3 install --upgrade jupyter boto3 aws-glue-sessions
```

```
install-glue-kernels
```

Exécution de Jupyter

Pour exécuter Jupyter Notebook, effectuez les étapes suivantes.

1. Pour lancer Jupyter Notebook, exécutez la commande suivante.

   ```
   jupyter notebook
   ```

2. Choisissez New (Nouveau), puis choisissez l'un des noyaux AWS Glue pour commencer à coder par rapport au AWS Glue.

Configuration des informations d'identification de séance et de région

Instructions MacOS/Linux

Les séances interactives AWS Glue requièrent les mêmes autorisations IAM que les tâches AWS Glue et les points de terminaison de développement. Spécifiez le rôle utilisé avec des séances interactives de l'une des deux manières suivantes :

1. Avec magics `%iam_role` et `%region`
2. Avec une ligne supplémentaire dans `~/.aws/config`

Configuration d'un rôle de séance avec magic

Dans la première cellule, saisissez `%iam_role <YourGlueServiceRole>` dans la première cellule exécutée.

Configuration d'un rôle de séance avec **`~/.aws/config`**

AWS GlueLe rôle de service pour les sessions interactives peut être spécifié dans le bloc-notes lui-même ou stocké à côté de la AWS CLI configuration. Si vous avez un rôle que vous utilisez généralement avec les tâches AWS Glue, ce sera ce rôle. Si vous n'avez pas de rôle que vous utilisez pour les tâches AWS Glue, veuillez suivre ce guide, Configuring IAM permissions for AWS Glue, pour en configurer un.

Pour définir ce rôle comme rôle par défaut pour les séances interactives, procédez comme suit :

1. Ouvrez `~/.aws/config` avec un éditeur de texte.
2. Recherchez le profil que vous utilisez pour AWS Glue. Si vous n'avez pas de profil, utilisez le profil `[Default]`.
3. Ajoutez une ligne dans le profil pour le rôle que vous avez l'intention d'utiliser comme `glue_role_arn=<AWSGlueServiceRole>`.
4. [Facultatif] : si votre profil ne possède pas de région par défaut, je vous recommande d'en ajouter une avec `region=us-east-1`, en remplaçant `us-east-1` par la région de votre choix.
5. Enregistrez la configuration.

Pour de plus amples informations, veuillez consulter Séances interactives avec IAM.

Instructions Windows

Les séances interactives AWS Glue requièrent les mêmes autorisations IAM que les tâches AWS Glue et les points de terminaison de développement. Spécifiez le rôle utilisé avec des séances interactives de l'une des deux manières suivantes :

1. Avec magics `%iam_role` et `%region`
2. Avec une ligne supplémentaire dans `~/.aws/config`

Configuration d'un rôle de séance avec magic

Dans la première cellule, saisissez `%iam_role <YourGlueServiceRole>` dans la première cellule exécutée.

Configuration d'un rôle de session avec **`~/.aws/config`**

AWS GlueLe rôle de service pour les sessions interactives peut être spécifié dans le bloc-notes lui-même ou stocké à côté de la AWS CLI configuration. Si vous avez un rôle que vous utilisez généralement avec les tâches AWS Glue, ce sera ce rôle. Si vous n'avez pas de rôle que vous utilisez pour les tâches AWS Glue, veuillez suivre ce guide, Configuration des autorisations IAM pour AWS Glue , pour en configurer un.

Pour définir ce rôle comme rôle par défaut pour les séances interactives, procédez comme suit :

1. Ouvrez `~/.aws/config` avec un éditeur de texte.

2. Recherchez le profil que vous utilisez pour AWS Glue. Si vous n'avez pas de profil, utilisez le profil `[Default]`.
3. Ajoutez une ligne dans le profil pour le rôle que vous avez l'intention d'utiliser comme `glue_role_arn=<AWSGlueServiceRole>`.
4. [Facultatif] : si votre profil ne possède pas de région par défaut, je vous recommande d'en ajouter une avec `region=us-east-1`, en remplaçant `us-east-1` par la région de votre choix.
5. Enregistrez la configuration.

Pour de plus amples informations, veuillez consulter Séances interactives avec IAM.

Mise à niveau à partir de l'aperçu des séances interactives

Le noyau a été mis à niveau avec de nouveaux noms lors de sa sortie avec la version 0.27. Pour nettoyer les versions d'aperçu des noyaux, exécutez ce qui suit depuis un terminal ou PowerShell.

> **Note**
>
> Si vous faites partie d'un autre aperçu AWS Glue qui nécessite un modèle de service personnalisé, la suppression du noyau supprimera le modèle de service personnalisé.

```
# Remove Old Glue Kernels
jupyter kernelspec remove glue_python_kernel
jupyter kernelspec remove glue_scala_kernel

# Remove Custom Model
cd ~/.aws/models
rm -rf glue/
```

Utilisation de sessions interactives avec SageMaker AI Studio

AWS Glue Interactive Sessions est un environnement d'exécution Apache Spark sans serveur à la demande que les data scientists et les ingénieurs peuvent utiliser pour créer, tester et exécuter rapidement des applications de préparation et d'analyse des données. Vous pouvez lancer une session AWS Glue interactive en démarrant un bloc-notes Amazon SageMaker AI Studio Classic.

Pour plus d'informations, consultez Prepare Data using AWS Glue interactive sessions.

Utilisation des séances interactives avec Microsoft Visual Studio Code

Conditions préalables

- Installez les séances interactives AWS Glue et vérifiez qu'elles fonctionnent avec Bloc-notes Jupyter.
- Téléchargez et installez Visual Studio Code avec Jupyter. Pour plus d'informations, consultez Bloc-notes Jupyter dans VS Code.

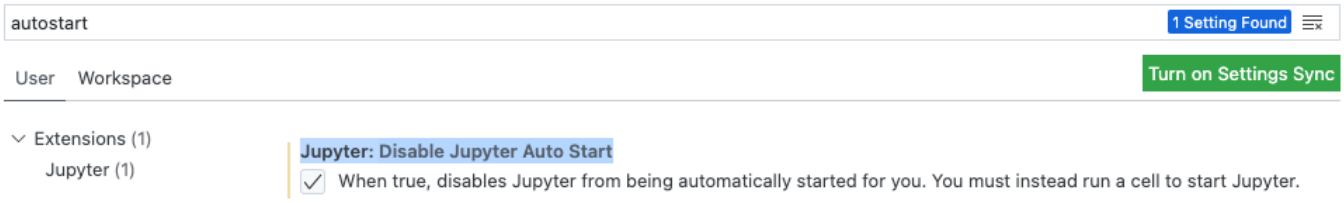
Pour démarrer avec des sessions interactives avec VSCode

1. Désactivez Jupyter AutoStart dans VS Code.

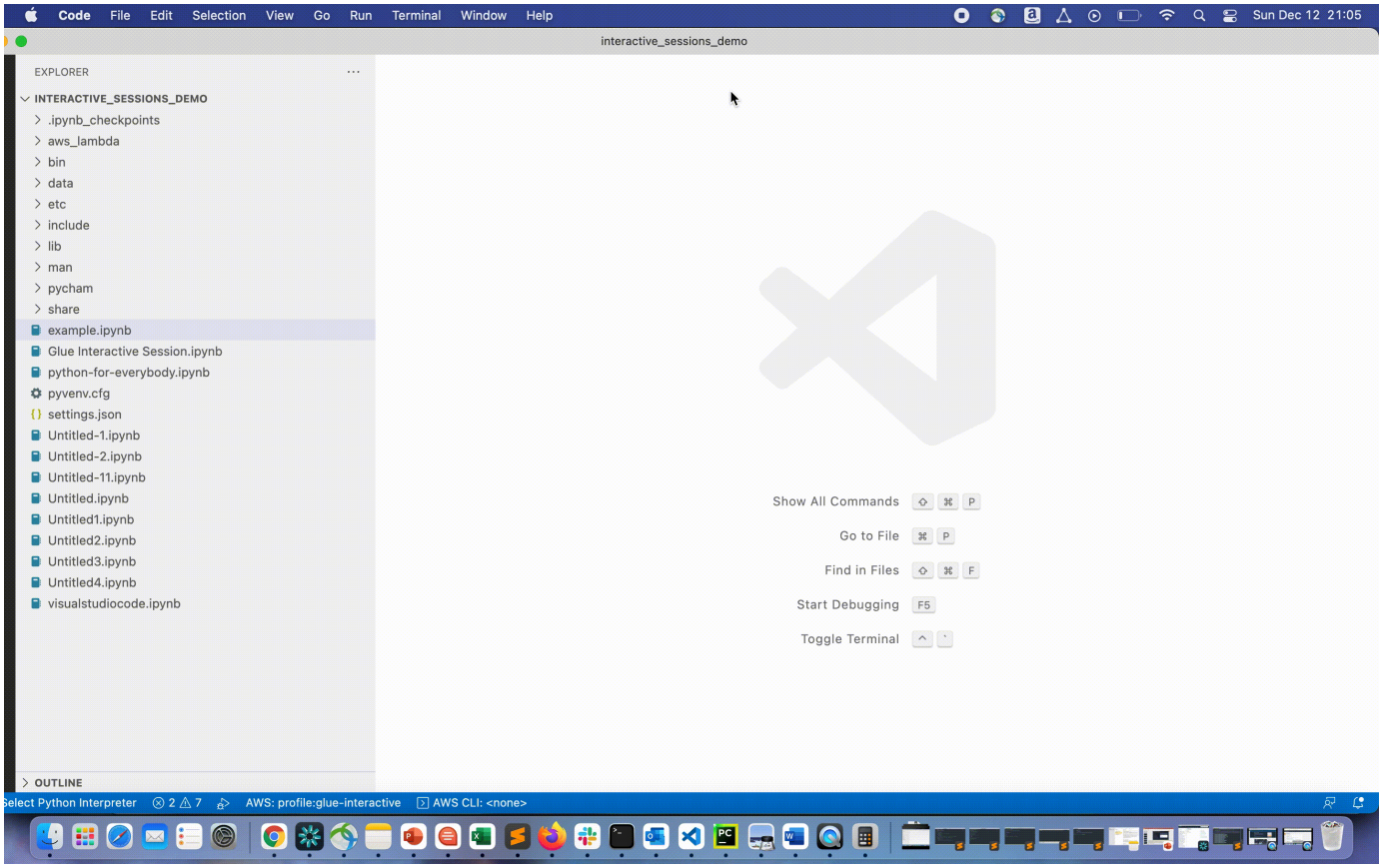
   Dans Visual Studio Code, les noyaux Jupyter démarreront automatiquement, ce qui empêchera vos magics de faire effet, car la session sera déjà lancée. Pour désactiver le Démarrage automatique sous Windows, accédez à Fichier > Préférences > Extensions > Jupyter > faites un clic droit sur Jupyter puis choisissez Paramètres d'extension.

   Sur macOS, accédez à Code > Paramètres > Extensions > Jupyter > faites un clic droit sur Jupyter puis choisissez Paramètres d'extension.

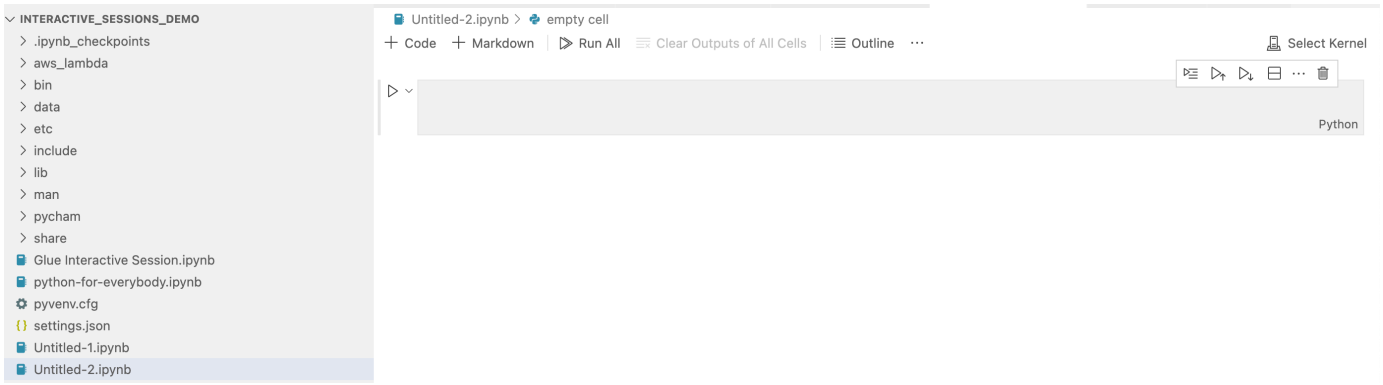
   Faites défiler la page vers le bas jusqu'à ce que vous voyiez Jupyter : désactiver le démarrage automatique de Jupyter. Cochez la case « Lorsque la valeur est vraie, le démarrage automatique de Jupyter est désactivé pour vous. Vous devez plutôt exécuter une cellule pour démarrer Jupyter. »



2. Accédez à Fichier > Nouveau fichier > Enregistrer pour enregistrer ce fichier avec le nom de votre choix en tant qu'une extension `.ipynb` ou sélectionnez jupyter sous select a language (Choisissez un langage), puis enregistrez le fichier.



3. Double-cliquez sur le fichier. Le shell Jupyter s'affiche et un bloc-notes s'ouvre.



4. Sous Windows, lorsque vous créez un fichier pour la première fois, par défaut, aucun noyau n'est sélectionné. Cliquez sur Select Kernel (Choisissez un noyau) et une liste des noyaux disponibles s'affiche. Choisissez Glue PySpark.

   Sur macOS, si le PySpark noyau Glue n'apparaît pas, essayez les étapes suivantes :

   1. Lancez une session Jupyter locale pour obtenir l'URL.

      Par exemple, pour lancer le bloc-notes Jupyter, exécutez la commande suivante.

```
jupyter notebook
```

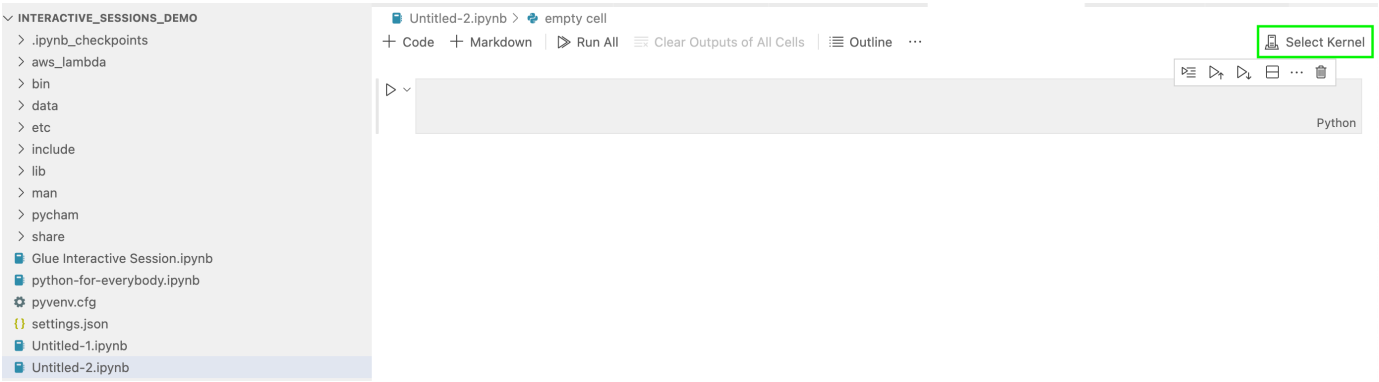
Lorsque le bloc-notes s'exécute pour la première fois, vous verrez une URL qui ressemble à `http://localhost:8888/?token=3398XXXXXXXXXXXXXXX`.

Copiez l'URL.

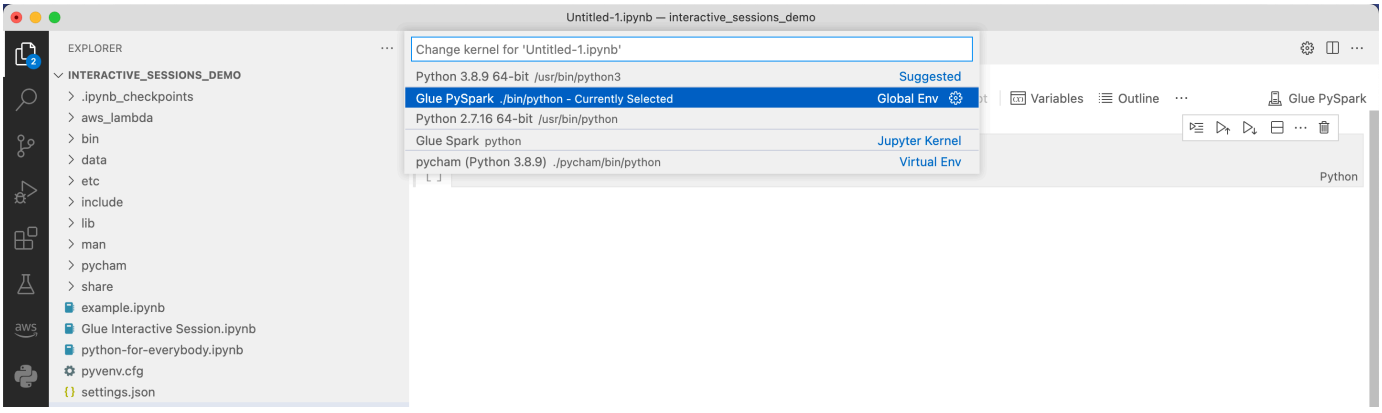
2. Dans VS Code, cliquez sur le noyau actuel, puis sur Sélectionnez un autre noyau…, puis sélectionnez Serveur Jupyter existant…. Collez l'URL que vous avez copiée à l'étape ci-dessus.

   Si vous recevez un message d'erreur, consultez le wiki VS Code Jupyter.

3. En cas de succès, le noyau sera défini sur Glue PySpark.



Choisissez le noyau Glue PySpark ou Glue Spark (pour Python et Scala respectivement).



Si aucun noyau AWS GlueSpark n'apparaît AWS Glue PySparkdans la liste déroulante, vérifiez que vous avez installé le AWS Glue noyau à l'étape ci-dessus ou que vos

`python.defaultInterpreterPath` paramètres dans Visual Studio Code sont corrects. Pour plus d'informations, consultez Python. defaultInterpreterPath description du réglage.

5. Création d'une séance interactive AWS Glue. Procédez à la création d'une session de la même manière que dans le bloc-notes Jupyter. Spécifiez n'importe quelle magie en haut de la première cellule et exécutez une instruction de code.

Séances interactives avec IAM

Ces sections décrivent les considérations relatives à la sécurité des séances interactives AWS Glue.

Rubriques

- Principaux IAM utilisés avec des séances interactives
- Paramétrer un client principal
- Paramétrer un rôle d'exécution
- Rendez votre session privée avec TagOnCreate
- Considérations de politique IAM

Principaux IAM utilisés avec des séances interactives

Vous utilisez deux principaux IAM utilisés avec des séances interactives AWS Glue.

- Client principal: Le principal client (un utilisateur ou un rôle) autorise les opérations API pour les séances interactives à partir d'un client AWS Glue configuré avec les informations d'identification basées sur l'identité du principal. Il peut s'agir par exemple d'un rôle IAM que vous utilisez généralement pour accéder à la console AWS Glue. Il peut également s'agir d'un rôle attribué à un utilisateur dans IAM dont les informations d'identification sont utilisées pour le noyau AWS Command Line Interface Jupyter ou d'un AWS Glue client utilisé par les sessions interactives du noyau Jupyter.
- Rôle d'exécution : le rôle d'exécution est un rôle IAM que le principal client transmet aux opérations API des séances interactives. AWS Glue utilise ce rôle pour exécuter des instructions dans votre séance. Par exemple, ce rôle peut être celui utilisé pour exécuter des tâches ETL AWS Glue.

  Pour plus d'informations, consultez Paramétrer un rôle d'exécution.

Paramétrer un client principal

Vous devez attacher une politique d'identité au principal du client pour lui permettre d'appeler l'API des séances interactives. Ce rôle doit avoir un accès `iam:PassRole` au rôle d'exécution que vous devez transmettre aux API des séances interactives telles que `CreateSession`. Par exemple, vous pouvez associer la politique AWSGlueConsoleFullAccessgérée à un rôle IAM qui permet aux utilisateurs de votre compte auxquels la politique est attachée d'accéder à toutes les sessions créées dans votre compte (comme la déclaration d'exécution ou la déclaration d'annulation).

Si vous souhaitez protéger votre session et la rendre privée uniquement pour certains rôles IAM, tels que ceux associés à l'utilisateur qui a créé la session, vous pouvez utiliser le contrôle d'autorisation basé sur les balises d'AWS GlueInteractive Session appelé TagOnCreate. Pour plus d'informations, découvrez Rendez votre session privée avec TagOnCreate comment une politique gérée délimitée basée sur le tag du propriétaire peut rendre votre session privée avec. TagOnCreate Pour plus d'informations sur les politiques basées sur l'identité, consultez Identity-based policies for AWS Glue.

Paramétrer un rôle d'exécution

Vous devez transmettre un rôle IAM à l'opération CreateSession API afin de permettre d'assumer et AWS Glue d'exécuter des instructions dans les sessions interactives. Le rôle doit avoir les mêmes autorisations IAM que celles requises pour exécuter une tâche AWS Glue typique. Par exemple, vous pouvez créer un rôle de service à l'aide de la AWSGlueServiceRolepolitique qui permet AWS Glue d'appeler AWS des services en votre nom. Si vous utilisez la console AWS Glue, il créera automatiquement une fonction du service en votre nom ou en utilisera une existante. Vous pouvez également créer votre propre rôle IAM et attacher votre propre politique IAM pour autoriser des autorisations similaires.

Si vous souhaitez protéger votre session et la rendre privée uniquement pour l'utilisateur qui l'a créée, vous pouvez utiliser le contrôle d'autorisation basé sur les balises d'AWS GlueInteractive Session appelé TagOnCreate. Pour plus d'informations, découvrez Rendez votre session privée avec TagOnCreate comment une politique gérée délimitée basée sur le tag du propriétaire peut rendre votre session privée avec. TagOnCreate Pour plus d'informations sur les stratégies basées sur l'identité, consultez Politiques basées sur l'identité pour Glue AWS. Si vous créez vous-même le rôle d'exécution à partir de la console IAM et que vous souhaitez rendre votre service privé grâce à TagOnCreate une fonctionnalité, suivez les étapes ci-dessous.

1. Créez un rôle IAM avec le type de rôle défini sur `Glue`.
2. Joignez cette politique AWS Glue gérée : AwsGlueSessionUserRestrictedServiceRole

3. Préfixez le nom du rôle par le nom de la politique. AwsGlueSessionUserRestrictedServiceRole Par exemple, vous pouvez créer un rôle nommé AwsGlueSessionUserRestrictedServiceRole-myrole et y associer une politique AWS Glue gérée. AwsGlueSessionUserRestrictedServiceRole
4. Attachez une politique d'approbation telle que la politique suivante pour autoriser AWS Glue à endosser le rôle :

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "glue.amazonaws.com"
        ]
      },
      "Action": [
        "sts:AssumeRole"
      ]
    }
  ]
}
```

Pour une session interactive (noyau Jupyter), vous pouvez spécifier la `iam_role` clé dans votre AWS Command Line Interface profil. Pour de plus amples informations, veuillez consulter Configuration de séances avec ~/.aws/config (langue Français non garantie). Si vous interagissez avec des séances interactives à l'aide d'un AWS Glue bloc-notes, vous pouvez ensuite passer le rôle d'exécution dans la magie `%iam_role` dans la première cellule que vous exécutez.

Rendez votre session privée avec TagOnCreate

Les séances interactives AWS Glue prennent en charge l'étiquette et l'autorisation basée sur les identifications (TBAC) pour les séances interactives en tant que ressource nommée. Outre l'utilisation du TBAC TagResource et UntagResource APIs, les sessions AWS Glue interactives prennent en charge la TagOnCreate fonctionnalité permettant de « baliser » une session avec une balise donnée uniquement lors de la création et de l' CreateSession exploitation de la session. Cela signifie également que ces balises seront supprimées le DeleteSession, c'est-à-dire UntagOnDelete.

TagOnCreate propose un puissant mécanisme de sécurité pour rendre votre session privée pour le créateur de la session. Par exemple, vous pouvez associer une politique IAM avec « owner » RequestTag et une valeur de $ {aws:userId} à un client principal (tel qu'un utilisateur) afin d'autoriser la création d'une session uniquement si une balise « owner » avec la valeur correspondante à l'identifiant de l'appelant est fournie en tant que balise userID dans la demande. CreateSession Cette politique autorise les séances interactives AWS Glue pour créer une ressource de séance et étiqueter la séance avec l'identification userId uniquement pendant la création de la séance. En outre, vous pouvez limiter l'accès (comme l'exécution d'instructions) à votre session uniquement au créateur (c'est-à-dire au tag owner avec la valeur $ {aws:userId}) de la session en attachant une politique IAM avec « owner » ResourceTag au rôle d'exécution que vous avez transmis au cours de cette session. CreateSession

Afin de faciliter l'utilisation de la TagOnCreate fonctionnalité permettant de rendre une session privée pour le créateur de session, AWS Glue fournit des politiques gérées et des rôles de service spécialisés.

Si vous souhaitez créer une session AWS Glue interactive à l'aide d'un AssumeRole principal IAM (c'est-à-dire en utilisant les informations d'identification fournies en assumant un rôle IAM) et que vous souhaitez rendre la session privée pour le créateur, utilisez des politiques similaires à celles de et respectivement. AWSGlueSessionUserRestrictedNotebookPolicyAWSGlueSessionUserRestrictedNotebookServiceRole Ces politiques permettent AWS Glue d'utiliser $ {aws :PrincipalTag} pour extraire la valeur de la balise du propriétaire. Cela nécessite que vous transmettiez une balise UserID avec la valeur $ {aws:userId} comme SessionTag dans les informations d'identification du rôle d'emprunt. Voir Balises d'identification de séance . Si vous utilisez une instance Amazon EC2 avec un profil d'instance qui vend les informations d'identification et que vous souhaitez créer une session ou interagir avec la session depuis l'instance Amazon EC2, vous devez transmettre une balise UserID avec la valeur $ {aws:userId} comme dans l'identifiant d'acceptation du rôle. SessionTag

Par exemple, si vous créez une session à l'aide d'un identifiant AssumeRole principal IAM et que vous souhaitez rendre votre service privé avec une TagOnCreate fonctionnalité, suivez les étapes ci-dessous.

1. Créez vous-même un rôle d'exécution à partir de la console IAM. Joignez cette politique AWS Glue gérée AwsGlueSessionUserRestrictedNotebookServiceRoleet préfixez le nom du rôle par le nom de la politique. AwsGlueSessionUserRestrictedNotebookServiceRole Par exemple, vous pouvez créer un rôle nommé AwsGlueSessionUserRestrictedNotebookServiceRole-myrole et y associer une politique AWS Glue gérée. AwsGlueSessionUserRestrictedNotebookServiceRole

2. Attachez une politique de confiance telle que celle ci-dessous pour autoriser AWS Glue à endosser le rôle ci-dessous.

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "glue.amazonaws.com"
        ]
      },
      "Action": [
        "sts:AssumeRole"
      ]
    }
  ]
}
```

3. Créez un autre rôle nommé avec un préfixe AwsGlueSessionUserRestrictedNotebookPolicyet attachez la politique AWS Glue gérée AwsGlueSessionUserRestrictedNotebookPolicypour rendre la session privée. Outre la politique gérée, veuillez joindre la politique en ligne suivante pour autoriser iam : au rôle que vous avez créé PassRole à l'étape 1.

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iam:PassRole"
      ],
      "Resource": [
        "arn:aws:iam::*:role/
AwsGlueSessionUserRestrictedNotebookServiceRole*"
      ],
```

```
      "Condition": {
        "StringLike": {
          "iam:PassedToService": [
            "glue.amazonaws.com"
          ]
        }
      }
    }
  ]
}
```

4. Attachez une politique de confiance telle que la suivante à IAM AWS Glue pour endosser ce rôle.

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "glue.amazonaws.com"
        ]
      },
      "Action": [
        "sts:AssumeRole",
        "sts:TagSession"
      ]
    }
  ]
}
```

> **Note**
>
> Vous pouvez éventuellement utiliser un seul rôle (par exemple, le rôle bloc-notes) et associer les deux politiques gérées ci-dessus AwsGlueSessionUserRestrictedNotebookServiceRoleet AwsGlueSessionUserRestrictedNotebookPolicy. Attachez également la politique en ligne supplémentaire pour autoriser `iam:passrole` de votre rôle à AWS Glue. Et

enfin, attachez la politique de confiance ci-dessus pour autoriser `sts:AssumeRole` et `sts:TagSession`.

AWSGlueSessionUserRestrictedNotebookPolicy

Permet AWSGlue SessionUserRestrictedNotebookPolicy de créer une session AWS Glue interactive à partir d'un bloc-notes uniquement si la clé de balise « propriétaire » et la valeur correspondent à AWS l'identifiant utilisateur du principal (utilisateur ou rôle). Pour plus d'informations, consultez Emplacement où vous pouvez utiliser les variables de politique. Cette politique est attachée au principal (utilisateur ou rôle) qui crée des blocs-notes de séance interactive AWS Glue à partir d'AWS Glue Studio. Cette politique permet également un accès suffisant au AWS Glue Studio bloc-notes pour interagir avec les ressources de session AWS Glue Studio interactive créées avec la valeur de balise « owner » correspondant à l'ID AWS utilisateur du principal. Cette politique rejette l'autorisation de modifier ou de supprimer les identifications « propriétaire » d'une ressource de séance AWS Glue après la création de la séance.

AWSGlueSessionUserRestrictedNotebookServiceRole

AWSGlueSessionUserRestrictedNotebookServiceRoleFournit un accès suffisant au AWS Glue Studio bloc-notes pour interagir avec les ressources de session AWS Glue interactive créées avec la valeur de balise « propriétaire » correspondant à l'ID AWS utilisateur du principal (utilisateur ou rôle) du créateur du bloc-notes. Pour plus d'informations, consultez Emplacement où vous pouvez utiliser les variables de politique. Cette politique de rôle de service est attachée au rôle qui est transmis comme par magie à un bloc-notes ou transmis en tant que rôle d'exécution à l' CreateSession API. Cette politique permet également de créer une session AWS Glue interactive à partir d'un bloc-notes uniquement si une balise clé « propriétaire » et une valeur correspondent à l'ID AWS utilisateur du principal. Cette politique rejette l'autorisation de modifier ou de supprimer une identification « propriétaire » d'une ressource de séance AWS Glue après la création de la séance. Cette politique inclut également les autorisations d'écriture et de lecture à partir de compartiments Amazon S3, de rédaction de CloudWatch journaux, de création et de suppression de balises pour les ressources Amazon EC2 utilisées par. AWS Glue

Rendre votre session privée avec les politiques utilisateur

Vous pouvez associer les rôles IAM AWSGlueSessionUserRestrictedPolicyà chacun des utilisateurs de votre compte pour les empêcher de créer une session uniquement avec un tag de propriétaire dont la valeur correspond à leur propre ${aws:userId}. Au lieu d'utiliser le AWSGlueSessionUserRestrictedNotebookPolicyet,

AWSGlueSessionUserRestrictedNotebookServiceRolevous devez utiliser des politiques similaires au AWSGlueSessionUserRestrictedPolicyet AWSGlueSessionUserRestrictedServiceRolerespectivement. Pour plus d'informations, consultez Using-identity based policies. Cette politique réduit l'accès à une session uniquement au créateur, l'${aws:userId} de l'utilisateur qui a créé la session avec la balise propriétaire portant son propre ${aws:userId}. Si vous avez créé vous-même le rôle d'exécution à l'aide de la console IAM en Paramétrer un rôle d'exécution suivant les étapes décrites dans la section, associez également la politique AwsGlueSessionUserRestrictedPolicygérée ci-dessous à chacun des utilisateurs de votre compte `iam:PassRole` pour autoriser le rôle d'exécution que vous avez créé précédemment.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iam:PassRole"
      ],
      "Resource": [
        "arn:aws:iam::*:role/AwsGlueSessionUserRestrictedServiceRole*"
      ],
      "Condition": {
        "StringLike": {
          "iam:PassedToService": [
            "glue.amazonaws.com"
          ]
        }
      }
    }
  ]
}
```

AWSGlueSessionUserRestrictedPolicy

AWSGlueSessionUserRestrictedPolicyPermet de créer une session AWS Glue interactive à l'aide de l' CreateSession API uniquement si une clé de balise « propriétaire » et une valeur correspondant à son ID AWS utilisateur sont fournies. Cette politique d'identité est attachée à l'utilisateur qui appelle

l' CreateSession API. Cette politique permet également d'interagir avec les ressources de session AWS Glue interactive créées avec une balise « owner » et une valeur correspondant à leur identifiant AWS utilisateur. Cette politique rejette l'autorisation de modifier ou de supprimer les identifications « propriétaire » d'une ressource de séance AWS Glue après la création de la séance.

AWSGlueSessionUserRestrictedServiceRole

AWSGlueSessionUserRestrictedServiceRoleFournit un accès complet à toutes les AWS Glue ressources, à l'exception des sessions, et permet aux utilisateurs de créer et d'utiliser uniquement les sessions interactives associées à l'utilisateur. Cette politique inclut également d'autres autorisations nécessaires AWS Glue pour gérer les ressources Glue dans d'autres AWS services. La politique permet également d'ajouter des balises aux AWS Glue ressources d'autres AWS services.

Considérations de politique IAM

Les séances interactives sont des ressources IAM dans AWS Glue. Étant donné qu'il s'agit de ressources IAM, l'accès et l'interaction à une séance sont régis par des politiques IAM. Sur la base des politiques IAM attachées à un client principal ou à un rôle d'exécution configuré par un administrateur, un client principal (utilisateur ou rôle) pourra créer de nouvelles séances et interagir avec ses propres séances et d'autres séances.

Si un administrateur a joint une politique IAM telle AWSGlue ServiceRole que AWSGlue ConsoleFullAccess ou qui autorise l'accès à toutes les AWS Glue ressources de ce compte, le client principal pourra collaborer entre eux. Par exemple, un utilisateur sera en mesure d'interagir avec des séances créées par d'autres utilisateurs si les politiques l'autorisent.

Si vous souhaitez configurer une politique adaptée à vos besoins spécifiques, veuillez consulter la documentation IAM sur la configuration des ressources pour une politique. Par exemple, pour isoler les sessions appartenant à un utilisateur, vous pouvez utiliser la TagOnCreate fonctionnalité prise en charge par les sessions AWS Glue interactives. Consultez Rendez votre session privée avec TagOnCreate.

Les sessions interactives prennent en charge la limitation de la création de sessions sur la base de certaines conditions VPC. Consultez Contrôler des politiques qui contrôlent les paramètres à l'aide des clés de condition.

Configuration des Séances interactives AWS Glue pour Jupyter et Blocs-notes AWS Glue Studio

Présentation de Jupyter magics

Les Jupyter magics sont des commandes qui peuvent être exécutées au début d'une cellule ou en tant que corps cellulaire entier. Les magics commencent par `%` pour les magics linéaires et `%%` pour les magics cellulaires. Les magics linéaires telles que `%region` et `%connections` peuvent être exécutées avec plusieurs magics dans une cellule, ou avec du code inclus à l'intérieur de la cellule, comme dans l'exemple suivant.

```
%region us-east-2
%connections my_rds_connection
dy_f = glue_context.create_dynamic_frame.from_catalog(database='rds_tables',
 table_name='sales_table')
```

Les magics cellulaires doivent utiliser la cellule entière et leur commande peut s'étendre sur plusieurs lignes. Ci-dessous un exemple de `%%sql`.

```
%%sql
select * from rds_tables.sales_table
```

Des magics soutenus par les séances interactives AWS Glue pour Jupyter

Ci-après les magics que vous pouvez utiliser avec les séances interactives AWS Glue pour les blocs-notes Jupyter.

Sessions Magics (Séances Magics)

Nom	Type	Description
`%help`	N/A	Renvoie une liste de descriptions et de types d'entrée pour toutes les commandes magiques.

Nom	Type	Description
`%profile`	String	Spécifiez un profil dans votre AWS configuration à utiliser comme fournisseur d'informations d'identification.
`%region`	String	Spécifiez le Région AWS ; dans lequel vous souhaitez initialiser une session. Valeurs par défaut de `~/.aws/configure`. Exemple : `%region us-west-1`
`%idle_timeout`	Int	Nombre de minutes d'inactivité au bout desquelles une séance expire après l'exécution d'une cellule. La valeur du délai d'inactivité par défaut pour les séances ETL Spark est le délai d'inactivité par défaut, soit 2 880 minutes (48 heures). Pour les autres types de séances, consultez la documentation relative. Exemple : `%idle_timeout 3000`
`%session_id`	N/A	Retournez l'ID de séance de la séance en cours d'exécution.
`%session_id_prefix`	String	Définissez une chaîne qui précédera toutes les sessions IDs au format [session_id_prefix] - [session_id]. Si aucun ID de séance n'est fourni, un UUID aléatoire est généré. Ce magic n'est pas pris en charge lorsque vous exécutez un bloc-notes Jupyter dans AWS Glue Studio. Exemple : `%session_id_prefix 001`

Nom	Type	Description
`%status`		Renvoie le statut de la séance AWS Glue actuelle, y compris sa durée, sa configura tion et l'exécution de son rôle/utilisateur.
`%stop_session`		Arrête la séance en cours.
`%list_sessions`		Répertorie toutes les séances en cours d'exécution par nom et ID.
`%session_type`	String	Définit le type de session comme suit : Streaming, ETL ou Ray. Exemple : `%session_type Streaming`
`%glue_version`	String	La version de AWS Glue à utiliser par cette session. Exemple : `%glue_version 3.0`

Magics pour la sélection des types de tâches

Nom	Type	Description
`%streaming`	String	Modifie le type de séance en Streaming AWS Glue.
`%etl`	String	Modifie le type de séance en ETL AWS Glue.
%glue_ray	String	Modifie le type de session en « AWS Glue for Ray ». Découvrez les sessions interacti ves Magics soutenues par AWS Glue Ray.

Magics de configuration AWS Glue pour Spark.

Le magic `%%configure` est un dictionnaire au format JSON comprenant tous les paramètres de configuration d'une session. Chaque paramètre peut être spécifié ici ou par magies individuelles.

Nom	Type	Description
`%%configure`	Dictionnaire	Spécifiez un dictionnaire au format JSON comprenant tous les paramètres de configuration d'une séance. Chaque paramètre peut être spécifié ici ou par magies individuelles. Pour obtenir la liste des paramètres et des exemples d'utilisation de `%%configure` , consultez Arguments du magic cellulaire %%configure.
`%iam_role`	String	Spécifiez un ARN de rôle IAM avec lequel exécuter votre séance. Valeurs par défaut de ~/.aws/configure. Exemple : `%iam_role AWSGlueServiceRole`
`%number_of_workers`	Int	Le nombre d'employés d'un paramètre worker_type défini qui sont alloués lorsqu'une tâche est exécutée. `worker_type` doit également être défini. La valeur par défaut `number_of_workers` est 5. Exemple : `%number_of_workers 2`
`%additional_python_modules`	List	Liste de modules Python supplémentaires séparés par des virgules à inclure dans votre cluster (peut provenir de PyPI ou de S3).

Nom	Type	Description
		Exemple: `%additional_python_modules pandas, numpy` .

Nom	Type	Description

`%%tags`	String	Ajoute des balises à une session. Spécifiez les balises entre crochets {}. Chaque paire de noms de balises est placée entre parenthèses (« ») et séparée par une virgule (,).
		
		`%%tags`
		`{"billing":"Data-Platform",`
		`"team":"analytics"}`
		
		Utilisez le magic `%status` pour afficher les balises associées à la session.
		
		`%status`
		
		`Session ID: <sessionId>`
		`Status: READY`
		`Role: <example-role>`
		`CreatedOn: 2023-05-26 11:12:17.`
		`056000-07:00`
		`GlueVersion: 3.0`
		`Job Type: glueetl`
		`Tags: {'owner':'example-owner',`
		`'team':'analytics', 'billing'`
		`:'Data-Platform'}`
		`Worker Type: G.4X`
		`Number of Workers: 5`
		`Region: us-west-2`
		`Applying the following default`
		`arguments:`
		`--glue_kernel_version 0.38.0`
		`--enable-glue-datacatalog true`
		`Arguments Passed: ['--glue_`
		`kernel_version: 0.38.0', '--`
		`enable-glue-datacatalog: true']`

Nom	Type	Description

%%assume_role	Dictionnaire	Spécifiez un dictionnaire au format JSON ou une chaîne ARN de rôle IAM pour créer une session d'accès intercompte.
		
		Exemple avec ARN :
		
		`%%assume_role`
		`{`
		`  'arn:aws:iam::XXXXXXXXXXXX:`
		`role/AWSGlueServiceRole'`
		`}`
		
		Exemple avec informations d'identification :
		
		`%%assume_role`
		`{{`
		`  "aws_access_key_id" =`
		`"XXXXXXXXXXXX",`
		`  "aws_secret_access_key" =`
		`"XXXXXXXXXXXX",`
		`  "aws_session_token" =`
		`"XXXXXXXXXXXX"`
		`}}`

Arguments du magic cellulaire %%configure

Le magic `%%configure` est un dictionnaire au format JSON comprenant tous les paramètres de configuration d'une session. Chaque paramètre peut être spécifié ici ou par magies individuelles. Vous trouverez ci-dessous des exemples d'arguments soutenus par le magic cellulaire `%%configure`. Utilisez le préfixe `--` pour les arguments d'exécution spécifiés pour la tâche. Exemple :

```
%%configure
{
   "--user-jars-first": "true",
```

```
   "--enable-glue-datacatalog": "false"
}
```

Pour plus d'informations sur les paramètres des tâches, consultez Paramètres du job.

Configuration de session

Paramètre	Type	Description

`max_retries`	Int	Nombre maximum de tentatives de cette tâche en cas d'échec.
		`%%configure`
		`{`
		`  "max_retries": "0"`
		`}`

`max_concurrent_runs`	Int	Nombre maximal d'exécutions simultanées autorisées pour une tâche.
		Exemple :
		`%%configure`
		`{`
		`  "max_concurrent_runs": "3"`
		`}`

Paramètres de session

Paramètre	Type	Description
`--enable-spark-ui`	Booléen	Activez l'interface utilisateur Spark pour surveiller et déboguer les tâches ETL AWS Glue. `%%configure`

Paramètre	Type	Description

		<pre>{
		  "--enable-spark-ui": "true"
		}</pre>

`--spark-event-logs-path`	String	Spécifie un chemin d'accès Amazon S3. Lorsque vous utilisez la fonctionnalité de surveillance de l'interface utilisateur Spark.
		
		Exemple :
		
		<pre>%%configure
		{
		  "--spark-event-logs-path":
		 "s3://path/to/event/logs/"
		}</pre>

`--script_location`	String	Spécifie le chemin d'accès S3 à un script qui exécute une tâche.
		
		Exemple :
		
		<pre>%%configure
		{
		  "script_location": "s3://new-
		folder-here"
		}</pre>

Paramètre	Type	Description

`--SECURITY_CONFIGURATION`	String	Le nom d'une configuration AWS Glue de sécurité
		
		Exemple :
		
		`%%configure`
		`{`
		`"--security_configuration": { "encryption_type": "kms", "kms_key_id": "YOUR_KMS_KEY_ARN" }`
		`}`

`--job-language`	String	Langage de programmation des scripts. Accepte la valeur « scala » ou « python ». La valeur par défaut est « python ».
		
		Exemple :
		
		`%%configure`
		`{`
		`"--job-language": "scala"`
		`}`

Paramètre	Type	Description

`--class`	String	La classe Scala qui sert de point d'entrée à votre script Scala. La valeur par défaut est nulle.
		
		Exemple :
		
		`%%configure`
		`{`
		`  "--class": "className"`
		`}`

`--user-jars-first`	Booléen	Donne la priorité aux fichiers JAR supplémentaires du client dans le chemin de classe. La valeur par défaut est nulle.
		
		Exemple :
		
		`%%configure`
		`{`
		`  "--user-jars-first": "true"`
		`}`

`--use-postgres-driver`	Booléen	Priorise le pilote JDBC Postgres dans le chemin de classe afin d'éviter tout conflit avec le pilote JDBC. Amazon Redshift La valeur par défaut est nulle.
		
		Exemple :
		
		`%%configure`
		`{`
		`  "--use-postgres-driver": "true"`
		`}`

Paramètre	Type	Description

`--extra-files`	List(string)	Les chemins Amazon S3 vers des fichiers supplémentaires (par exemple, des fichiers de configuration) que AWS Glue copie vers le répertoire de travail de votre script avant de l'exécuter.
		
		Exemple :
		
		`%%configure`
		`{`
		`  "--extra-files": "s3://path/to/additional/files/"`
		`}`

`--job-bookmark-option`	String	Contrôle le comportement d'un signet de tâche. Accepte les valeurs « job-bookmark-enable », « job-bookmark-disable » ou « job-bookmark-pause ». La valeur par défaut est job-bookmark-disable « ».
		
		Exemple :
		
		`%%configure`
		`{`
		`  "--job-bookmark-option": "job-bookmark-enable"`
		`}`

Paramètre	Type	Description

`--TempDir`	String	Spécifie un chemin Amazon S3 vers un compartiment qui peut être utilisé comme répertoire temporaire pour la tâche. La valeur par défaut est nulle.
		
		Exemple :
		
		`%%configure`
		`{`
		`  "--TempDir": "s3://path/to/temp/dir"`
		`}`

`--enable-s3-parquet-optimized-committer`	Booléen	Active le validateur EMRFS optimisé pour Amazon S3 pour l'écriture de données Parquet dans Amazon S3. La valeur par défaut est « true ».
		
		Exemple :
		
		`%%configure`
		`{`
		`  "--enable-s3-parquet-optimized-committer": "false"`
		`}`

Paramètre	Type	Description

`--enable-rename-algorithm-v2`	Booléen	Définit la version de l'algorithme de renommage EMRFS sur la version 2. La valeur par défaut est « true ».
		
		Exemple :
		
		`%%configure`
		`{`
		`  "--enable-rename-algorithm-v2": "true"`
		`}`

`--enable-glue-datacatalog`	Booléen	Vous permet d'utiliser le catalogue de données AWS Glue en tant que métastore Apache Hive Spark.
		
		Exemple :
		
		`%%configure`
		`{`
		`  "--enable-glue-datacatalog": "true"`
		`}`

`--enable-metrics`	Booléen	Permet de collecter des métriques de profilage de tâche pour l'exécution de la tâche. La valeur par défaut est « false ».
		
		Exemple :
		
		`%%configure`
		`{`
		`  "--enable-metrics": "true"`
		`}`

Paramètre	Type	Description

`--enable-continuous-cloudwatch-log`	Booléen	Active la journalisation continue en temps réel des tâches AWS Glue. La valeur par défaut est « false ».
		
		Exemple :
		
		`%%configure`
		`{`
		`  "--enable-continuous-cloudwatch-log": "true"`
		`}`

`--enable-continuous-log-filter`	Booléen	Spécifie un filtre standard ou aucun filtre lorsque vous créez ou modifiez une tâche activée pour la journalisation continue. La valeur par défaut est « true ».
		
		Exemple :
		
		`%%configure`
		`{`
		`  "--enable-continuous-log-filter": "true"`
		`}`

Paramètre	Type	Description

`--continuous-log-stream-prefix`	String	Spécifie un préfixe de flux de Amazon CloudWatch journal personnalisé pour une tâche activée pour la journalisation continue. La valeur par défaut est nulle.
		
		Exemple :
		
		`%%configure`
		`{`
		`  "--continuous-log-stream-prefix": "prefix"`
		`}`

`--continuous-log-conversionPattern`	String	Spécifie un modèle de journal de conversion personnalisé pour une tâche activée pour la journalisation en continu. La valeur par défaut est nulle.
		
		Exemple :
		
		`%%configure`
		`{`
		`  "--continuous-log-conversionPattern": "pattern"`
		`}`

Paramètre	Type	Description

`--conf`	String	Contrôle les paramètres de configuration de Spark. Elle est destinée aux cas d'utilisation avancés. Utilisez `--conf` avant chaque paramètre. Exemple :
		`%%configure`
		`{`
		`"--conf": "spark.hadoop.hive.metastore.glue.catalogid=123456789012 --conf hive.metastore.client.factory.class=com.amazonaws.glue.catalog.metastore.AWSGlueDataCatalogHiveClientFactory --conf hive.metastore.schema.verification=false"`
		`}`

timeout	Int	Détermine la durée maximale pendant laquelle la session Spark doit attendre qu'une instruction n'expire avant d'y mettre fin.
		`%%configure`
		`{`
		`"timeout": "30"`
		`}`

autoscaling	Booléen	Détermine s'il faut ou non utiliser l'autoscaling.
		`%%configure`
		`{`
		`"--enable-auto-scaling": "true"`
		`}`

Magics de tâches Spark (ETL et streaming)

Nom	Type	Description
`%worker_type`	String	Standard, G.1X, G.2X, G.4X, G.8X, G.12X, G.16X, R.1X, R.2X, R.4X ou R.8X. `number_of_workers` doit également être défini. Le paramètre worker_type par défaut est G.1X.

`%connections`	List	Spécifiez une liste de connexions séparées par des virgules à utiliser dans la séance.
		
		Exemple :
		
		`%connections my_rds_connection`
		`dy_f =`
		`glue_context.create_dynamic`
		`_frame.from_catalog(databas`
		`e='rds_tables', table_nam`
		`e='sales_table')`

`%extra_py_files`	List	Liste de fichiers Python supplémentaires séparée par des virgules provenant d'Amazon S3.
`%extra_jars`	List	Liste de pots supplémentaires séparés par des virgules à inclure dans le cluster.
`%spark_conf`	String	Spécifiez des configurations Spark personnalisées pour votre session. Par exemple, `%spark_conf spark.ser` `ializer=org.apache.spark.se` `rializer.KryoSerializer` .

Magics pour les tâches Ray

Nom	Type	Description
`%min_workers`	Int	Le nombre minimum de travailleurs alloués à une tâche Ray. Par défaut : 1. Exemple : `%min_workers 2`
`%object_memory_head`	Int	Le pourcentage de mémoire disponible sur le nœud principal de l'instance après un démarrage à chaud. Minimum : 0. Maximum : 100. Exemple : `%object_memory_head 100`
%object_memory_worker	Int	Le pourcentage de mémoire disponibl e sur le composant master de l'instance après un démarrage à chaud. Minimum : 0. Maximum : 100. Exemple : `%object_memory_worker 100`

Action Magics (Action Magics)

Nom	Type	Description
`%%sql`	String	Exécutez le code SQL. Toutes les lignes après magic `%%sql` initial comme faisant partie du code SQL. Exemple : `%%sql select * from rds_tables.sales_table`
`%matplot`	Illustration Matplotlib	Visualisez vos données à l'aide de la bibliothèque Matplotlib.

Nom	Type	Description

		Exemple :
		`import matplotlib.pyplot as plt`
		
		`# Set X-axis and Y-axis values`
		`x = [5, 2, 8, 4, 9]`
		`y = [10, 4, 8, 5, 2]`
		
		`# Create a bar chart`
		`plt.bar(x, y)`
		
		`# Show the plot`
		`%matplot plt`

`%plotly`	Illustration Plotly	Visualisez vos données à l'aide de la bibliothèque Plotly.
		
		Exemple :
		`import plotly.express as px`
		
		`#Create a graphical figure`
		`fig = px.line(x=["a","b","c"],`
		` y=[1,3,2], title="sample`
		` figure")`
		
		`#Show the figure`
		`%plotly fig`

Désignation des séances

AWS Glueles sessions interactives sont AWS des ressources qui nécessitent un nom. Les noms doivent être uniques pour chaque session et peuvent être restreints par vos administrateurs IAM. Pour de plus amples informations, veuillez consulter Séances interactives avec IAM. Le noyau Jupyter génère automatiquement des noms de session uniques pour vous. Toutefois, les séances peuvent être nommées manuellement de deux manières :

1. À l'aide du fichier de AWS Command Line Interface configuration situé à l'adresse`~.aws/config`. Voir Configuration de la AWS configuration avec le AWS Command Line Interface.
2. Utilisation des magics `%session_id_prefix`. Consultez Des magics soutenus par les séances interactives AWS Glue pour Jupyter.

Un nom de séance est généré comme suit :

- Lorsque le préfixe et l'identifiant de session sont fournis : le nom de la séance sera {prefix}-{UUID}.
- Lorsque rien n'est fourni : le nom de la séance sera {UUID}.

Le préfixe des noms de session vous permet de reconnaître votre session lorsque vous la listez dans la console AWS CLI or.

Spécification d'un rôle IAM pour les séances interactives

Vous devez spécifier un rôle AWS Identity and Access Management (IAM) à utiliser avec le code AWS Glue ETL que vous exécutez lors de sessions interactives.

Le rôle requiert les mêmes autorisations IAM que celles requises pour exécuter les tâches AWS Glue. Voir Création d'un rôle IAM pour AWS Glue pour plus d'informations sur la création d'un rôle pour les tâches AWS Glue et les séances interactives.

Les rôles IAM peuvent être spécifiés de deux manières :

- À l'aide du fichier de AWS Command Line Interface configuration situé à l'`~.aws/config`adresse (recommandé). Pour de plus amples informations, veuillez consulter Configuration de séances avec ~/.aws/config (langue Français non garantie).

  > **Note**
  >
  > Lorsque magic de `%profile` est utilisée, la configuration pour `glue_iam_role` de ce profil est respectée.

- Utilisation du magic %iam_role. Pour plus d'informations, consultez Des magics soutenus par les séances interactives AWS Glue pour Jupyter.

Configuration des séances avec des profils nommés

AWS Glueles sessions interactives utilisent les mêmes informations d'identification que le AWS Command Line Interface ou boto3, et les sessions interactives respectent et fonctionnent avec des profils nommés tels que ceux AWS CLI trouvés dans `~/.aws/config` (Linux et macOS) ou `%USERPROFILE%\.aws\config` (Windows). Pour de plus amples informations, consultez Using named profiles.

Les séances interactives tirent parti des profils nommés en permettant au préfixe de fonction du service et aux ID de séance AWS Glue d'être spécifiés dans un profil. Pour configurer un rôle de profil, ajoutez une ligne pour la `iam_role` clé and/or `session_id_prefix` de votre profil nommé, comme indiqué ci-dessous. Le `session_id_prefix` ne nécessite pas de guillemets. Par exemple, si vous souhaitez ajouter un `session_id_prefix`, saisissez la valeur du `session_id_prefix=myprefix`.

```
[default]
region=us-east-1
aws_access_key_id=AKIAIOSFODNN7EXAMPLE
aws_secret_access_key=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
glue_iam_role=arn:aws:iam::<AccountID>:role/<GlueServiceRole>
session_id_prefix=<prefix_for_session_names>

[user1]
region=eu-west-1
aws_access_key_id=AKIAI44QH8DHBEXAMPLE
aws_secret_access_key=je7MtGbClwBF/2Zp9Utk/h3yCo8nvbEXAMPLEKEY
glue_iam_role=arn:aws:iam::<AccountID>:role/<GlueServiceRoleUser1>
session_id_prefix=<prefix_for_session_names_for_user1>
```

Si vous disposez d'une méthode personnalisée pour générer des informations d'identification, vous pouvez également configurer votre profil pour utiliser le paramètre `credential_process` dans votre fichier `~/.aws/config`. Par exemple :

```
[profile developer]
region=us-east-1
credential_process = "/Users/Dave/generate_my_credentials.sh" --username helen
```

Vous trouverez plus de détails sur la source d'approvisionnement des informations d'identification via le paramètre `credential_process` ici : Source d'approvisionnement des informations d'identification avec un processus externe.

Si une région ou `iam_role` ne sont pas définis dans le profil que vous utilisez, vous devez les spécifier à l'aide des magics `%region` et `%iam_role` dans la première cellule que vous exécutez.

Conversion d'un script ou d'un bloc-notes en AWS Glue tâche

Il existe deux manières de convertir un script ou un bloc-notes en une tâche AWS Glue :

- Utilisez nbconvert pour convertir votre fichier de document de bloc-notes `.ipynb` Jupyter dans un fichier `.py`. Pour plus d'informations, consultez nbconvert : Convertir les blocs-notes dans d'autres formats.
- Chargez le fichier dans Blocs-notes AWS Glue Studio.
  - Dans la console AWS Glue Studio, choisissez Jobs (Tâches) dans le menu de navigation.
  - Dans la section Create job (Créer une tâche), choisissez Jupyter notebook (Bloc-notes Jupyter).
  - Dans la section Options, choisissez Upload and edit an existing notebook (Charger et modifier un bloc-notes existant).
  - Sélectionnez Choose file (Choisir un fichier) pour charger un fichier `.ipynb`.

Utilisation des opérations de streaming dans le cadre de sessions interactives AWS Glue

Changement de type de séance de streaming

Utilisez magic de configuration des séances interactives AWS Glue et `%streaming` pour définir la tâche que vous exécutez et initialisez une séance interactive de streaming.

Échantillonnage du flux d'entrée pour un développement interactif

L'un des outils que nous avons développés pour améliorer l'expérience AWS Glue interactive dans les sessions interactives est l'ajout d'une nouvelle méthode ci-dessous `GlueContext` pour obtenir un instantané d'un flux dans un fichier statique DynamicFrame. `GlueContext`vous permet d'inspecter, d'interagir et de mettre en œuvre votre flux de travail.

Avec l'instance de classe `GlueContext`, vous serez en mesure de localiser la méthode `getSampleStreamingDynamicFrame`. Les arguments requis pour cette méthode sont les suivants :

- `dataFrame`: The Spark Streaming DataFrame
- `options` : voir les options disponibles ci-dessous

Les options disponibles sont les suivantes :

- windowSize : ceci est également appelé Durée du micro-lot. Ce paramètre déterminera la durée d'attente d'une requête en streaming après le déclenchement du lot précédent. La valeur de ce paramètre doit être inférieure à `pollingTimeInMs`.
- pollingTimeInMme : La durée totale pendant laquelle la méthode sera exécutée. Il déclenche au moins un micro-lot pour obtenir des échantillons de registre à partir du flux d'entrée.
- recordPollingLimit: Ce paramètre vous permet de limiter le nombre total d'enregistrements que vous allez interroger à partir du flux.
- (Facultatif) Vous pouvez également utiliser `writeStreamFunction` pour appliquer cette fonction personnalisée à chaque fonction d'échantillonnage de registre. Voir ci-dessous des exemples dans Scala et Python.

Scala

```
val sampleBatchFunction = (batchDF: DataFrame, batchId: Long) => {//Optional but
 you can replace your own forEachBatch function here}
val jsonString: String = s"""{"pollingTimeInMs": "10000", "windowSize": "5
 seconds"}"""
val dynFrame = glueContext.getSampleStreamingDynamicFrame(YOUR_STREAMING_DF,
 JsonOptions(jsonString), sampleBatchFunction)
dynFrame.show()
```

Python

```
def sample_batch_function(batch_df, batch_id):
       //Optional but you can replace your own forEachBatch function here
options = {
            "pollingTimeInMs": "10000",
```

```
            "windowSize": "5 seconds",
        }
glue_context.getSampleStreamingDynamicFrame(YOUR_STREAMING_DF, options,
 sample_batch_function)
```

> **Note**
>
> Lorsque le paramètre `DynFrame` échantillonné est vide, cela peut être causé par certaines des raisons suivantes :
>
> - La source Streaming est définie sur « Dernier » et aucune nouvelle donnée n'a été ingérée pendant la période d'échantillonnage.
> - Le temps d'interrogation n'est pas suffisant pour traiter les registres qu'il a ingérés. Les données ne s'afficheront pas à moins que l'ensemble du lot n'ait été traité.

Exécution d'applications de streaming dans des séances interactives

Dans les séances interactives AWS Glue, vous pouvez exécuter l'application de streaming AWS Glue comme comment créer une application de streaming dans la console AWS Glue. Étant donné que les séances interactives sont basées sur une séance, la présence d'exceptions dans le moteur d'exécution ne provoque pas l'arrêt de la séance. Nous avons maintenant l'avantage supplémentaire de développer votre fonction par lots de manière itérative. Par exemple :

```
def batch_function(data_frame, batch_id):
    log.info(data_frame.count())
    invalid_method_call()
glueContext.forEachBatch(frame=streaming_df, batch_function = batch_function, options =
 {**})
```

Dans l'exemple ci-dessus, nous avons inclus une utilisation non valide d'une méthode et contrairement aux tâches AWS Glue habituelles qui quitteront l'application entière, le contexte de codage et les définitions de l'utilisateur sont entièrement préservés, et la séance est toujours opérationnelle. Il n'est pas nécessaire d'amorcer un nouveau cluster et de ré-exécuter toute la transformation précédente. Cela vous autorise à vous concentrer sur l'itération rapide de vos implémentations de fonctions par lots pour obtenir des résultats souhaitables.

Il est important de noter que la séance interactive évalue chaque instruction de manière bloquante afin que la séance n'exécute qu'une seule instruction à la fois. Étant donné que les requêtes de streaming sont continues et ne se terminent jamais, les séances avec des requêtes de streaming actives ne seront pas en mesure de gérer les instructions de suivi à moins qu'elles ne soient interrompues. Vous pouvez émettre la commande d'interruption directement à partir de Bloc-notes Jupyter et notre noyau s'occupera de l'annulation pour vous.

Prenons l'exemple de la séquence suivante d'instructions qui attendent d'être exécutées :

```
Statement 1:
      val number = df.count()
      #Spark Action with deterministic result
      Result: 5

Statement 2:
      streamingQuery.start().awaitTermination()
      #Spark Streaming Query that will be executing continously
      Result: Constantly updated with each microbatch

Statement 3:
      val number2 = df.count()
      #This will not be executed as previous statement will be running indefinitely
```

Tarif des sessions interactives AWS Glue

AWS les frais de session AWS Glue interactive sont calculés en fonction de la durée pendant laquelle la session est active et du nombre d'unités de traitement des données (DPU) utilisées. Un taux horaire vous est facturé en fonction du nombre d'heures DPUs utilisées pour exécuter vos charges de travail, facturé par tranches d'une seconde. AWS Glue les sessions interactives attribuent une valeur par défaut de 5 DPUs et en nécessitent un minimum de 2. DPUs Il existe également une durée de facturation minimale d'une minute pour chaque session interactive. Pour consulter les AWS Glue taux et les exemples de tarification, ou pour estimer vos coûts à l'aide du calculateur de AWS prix, consultez la section AWS Glue tarification.

Configuration de vos sessions interactives AWS Glue

Vous pouvez utiliser la magie de Jupyter dans votre session AWS interactive pour modifier vos paramètres de session et de configuration. Les commandes magiques sont de courtes commandes

préfixées par `%` au début des cellules Jupyter qui fournissent un moyen simple et rapide de vous aider à contrôler votre environnement. Par exemple, si vous souhaitez modifier le nombre de travailleurs alloués à votre tâche pour le faire passer de cinq (nombre par défaut) à dix, vous pouvez spécifier `%number_of_workers 10`. Si vous souhaitez configurer votre session pour qu'elle s'arrête après 10 minutes d'inactivité au lieu des 2 880 par défaut, vous pouvez spécifier `%idle_timeout 10`.

Pour la liste complète des AWS magies disponibles, consultez Configuration de sessions AWS interactives pour Jupyter et les ordinateurs portables. AWS Glue Studio

Développement et test de scripts AWS de tâches Glue localement

Lorsque vous développez et testez vos scripts de travail AWS Glue for Spark, plusieurs options sont disponibles :

- AWS Console Glue Studio
  - Visual editor (Éditeur visuel)
  - Éditeur de script
  - AWS Carnet Glue Studio
- Sessions interactives
  - Bloc-notes Jupyter
- Image Docker
  - Développement local
  - Développement à distance

Vous pouvez choisir l'une des options ci-dessus en fonction de vos besoins.

Si vous préférez ne pas coder ou moins d'expérience en matière de code, l'éditeur visuel AWS Glue Studio est un bon choix.

Si vous préférez une expérience interactive, le bloc-notes AWS Glue Studio est un bon choix. Pour plus d'informations, consultez la section Utilisation de Notebooks avec AWS Glue Studio et AWS Glue. Si vous souhaitez utiliser votre propre environnement local, les sessions interactives constituent un bon choix. Pour plus d'informations, consultez la section Utilisation de sessions interactives avec AWS Glue.

Si vous préférez une expérience de local/remote développement, l'image Docker est un bon choix. Cela vous permet de développer et de tester des scripts de travail AWS Glue for Spark où vous le souhaitez, sans encourir de frais liés à AWS Glue.

Si vous préférez le développement local sans Docker, l'installation du répertoire de la bibliothèque AWS Glue ETL localement est un bon choix.

Développement à l'aide AWS de Glue Studio

L'éditeur visuel AWS Glue Studio est une interface graphique qui facilite la création, l'exécution et le suivi de tâches d'extraction, de transformation et de chargement (ETL) dans AWS Glue. Vous pouvez composer visuellement des flux de travail de transformation des données et les exécuter de manière fluide sur le moteur AWS ETL sans serveur basé sur Apache Spark de Glue. Vous pouvez inspecter le schéma et les résultats des données à chaque étape de la tâche. Pour plus d'informations, veuillez consulter le Guide de l'utilisateur AWS Glue Studio.

Développement à l'aide de sessions interactives

Les sessions interactives vous permettent de créer et de tester des applications à partir de l'environnement de votre choix. Pour plus d'informations, consultez la section Utilisation de sessions interactives avec AWS Glue.

Développez et testez AWS Glue des jobs localement à l'aide d'une image Docker

Pour une plateforme de données prête à être mise en production, le processus de développement et le CI/CD pipeline de AWS Glue jobs constituent un sujet clé. Vous pouvez développer et tester des AWS Glue tâches de manière flexible dans un conteneur Docker. AWS Glue héberge des images Docker sur Docker Hub pour configurer votre environnement de développement avec des utilitaires supplémentaires. Vous pouvez utiliser votre IDE, bloc-notes ou REPL préféré à l'aide de la bibliothèque AWS Glue ETL. Cette rubrique explique comment développer et tester des jobs de AWS Glue la version 5.0 dans un conteneur Docker à l'aide d'une image Docker.

Images Docker disponibles

Les images Docker suivantes sont disponibles pour AWS Glue Amazon ECR :.

- Pour AWS Glue la version 5.0 : `public.ecr.aws/glue/aws-glue-libs:5`

- Pour AWS Glue la version 4.0 : `public.ecr.aws/glue/aws-glue-libs:glue_libs_4.0.0_image_01`
- Pour AWS Glue la version 3.0 : `public.ecr.aws/glue/aws-glue-libs:glue_libs_3.0.0_image_01`
- Pour AWS Glue la version 2.0 : `public.ecr.aws/glue/aws-glue-libs:glue_libs_2.0.0_image_01`

> **Note**
>
> AWS Glue Les images Docker sont compatibles avec x86_64 et arm64.

Dans cet exemple, nous utilisons `public.ecr.aws/glue/aws-glue-libs:5` et nous exécutons le conteneur sur une machine locale (Mac, Windows ou Linux). Cette image de conteneur a été testée pour les tâches Spark de la AWS Glue version 5.0. L'image contient les éléments suivants :

- Amazon Linux 2023
- AWS Glue Bibliothèque ETL
- Apache Spark 3.5.4
- Bibliothèques de formats de table ouverte ; Apache Iceberg 1.7.1, Apache Hudi 0.15.0 et Delta Lake 3.3.0
- AWS Glue Client de catalogue de données
- Amazon Redshift connecteur pour Apache Spark
- Amazon DynamoDB connecteur pour Apache Hadoop

Pour configurer votre conteneur, extrayez l'image de la galerie publique ECR, puis exécutez le conteneur. Cette rubrique explique comment exécuter votre conteneur selon les méthodes suivantes, en fonction de vos besoins :

- `spark-submit`
- Shell REPL (`pyspark`)
- `pytest`
- Code Visual Studio

Conditions préalables

Avant de commencer, assurez-vous que Docker est installé et que le démon Docker est en cours d'exécution. Pour obtenir des instructions d'installation, consultez la documentation Docker pour Mac ou Linux. La machine qui exécute le Docker héberge le AWS Glue conteneur. Assurez-vous également que vous disposez d'au moins 7 Go d'espace disque pour l'image sur l'hôte exécutant Docker.

Pour plus d'informations sur les restrictions applicables au développement de AWS Glue code en local, consultez la section Restrictions de développement locales.

Configuration AWS

Pour activer les appels AWS d'API depuis le conteneur, configurez les AWS informations d'identification en suivant les étapes suivantes. Dans les sections suivantes, nous utiliserons ce profil AWS nommé.

1. Créez un profil AWS nommé.
2. Ouvrez cmd sous Windows ou sur un terminal Mac/Linux et exécutez la commande suivante dans un terminal :

```
PROFILE_NAME="<your_profile_name>"
```

Dans les sections suivantes, nous utilisons ce profil AWS nommé.

Si vous exécutez Docker sur Windows, choisissez l'icône Docker (clic droit) et choisissez Basculer vers les conteneurs Linux avant d'extraire l'image.

Exécutez la commande suivante pour extraire l'image d'ECR Public :

```
docker pull public.ecr.aws/glue/aws-glue-libs:5
```

Exécution du conteneur

Vous pouvez maintenant exécuter un conteneur en utilisant cette image. Vous pouvez choisir l'une des options suivantes en fonction de vos besoins.

spark-submit

Vous pouvez exécuter un script de AWS Glue tâche en exécutant la `spark-submit` commande sur le conteneur.

1. Rédigez votre script, enregistrez-le en tant que `sample.py` dans l'exemple ci-dessous, puis enregistrez-le dans le répertoire `/local_path_to_workspace/src/` à l'aide des commandes suivantes :

```
$ WORKSPACE_LOCATION=/local_path_to_workspace
$ SCRIPT_FILE_NAME=sample.py
$ mkdir -p ${WORKSPACE_LOCATION}/src
$ vim ${WORKSPACE_LOCATION}/src/${SCRIPT_FILE_NAME}
```

2. Ces variables sont utilisées dans la commande docker run ci-dessous. L'exemple de code (sample.py) utilisé dans la commande spark-submit ci-dessous est inclus dans l'annexe à la fin de cette rubrique.

   Exécutez la commande suivante pour exécuter la commande `spark-submit` sur le conteneur afin de soumettre une nouvelle application Spark :

```
$ docker run -it --rm \
    -v ~/.aws:/home
    /hadoop/.aws \
    -v $WORKSPACE_LOCATION:/home/hadoop/workspace/ \
    -e AWS_PROFILE=$PROFILE_NAME \
    --name glue5_spark_submit \
    public.ecr.aws/glue/aws-glue-libs:5 \
    spark-submit /home/hadoop/workspace/src/$SCRIPT_FILE_NAME
```

3. (Facultatif) Configurez `spark-submit` pour qu'il corresponde à votre environnement. Par exemple, vous pouvez transmettre vos dépendances à la configuration `--jars`. Pour plus d'informations, consultez Dynamically Loading Spark Properties dans la documentation Spark.

Shell REPL (Pyspark)

Vous pouvez exécuter le shell REPL (`read-eval-print loops`) pour un développement interactif. Exécutez la commande suivante pour exécuter la PySpark commande sur le conteneur afin de démarrer le shell REPL :

```
$ docker run -it --rm \
```

```
    -v ~/.aws:/home/hadoop/.aws \
    -e AWS_PROFILE=$PROFILE_NAME \
    --name glue5_pyspark \
    public.ecr.aws/glue/aws-glue-libs:5 \
    pyspark
```

Vous obtenez la sortie suivante :

```
Python 3.11.6 (main, Jan  9 2025, 00:00:00) [GCC 11.4.1 20230605 (Red Hat 11.4.1-2)] on
 linux
Type "help", "copyright", "credits" or "license" for more information.
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use
 setLogLevel(newLevel).
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /__ / .__/\_,_/_/ /_/\_\   version 3.5.4-amzn-0
      /_/

Using Python version 3.11.6 (main, Jan  9 2025 00:00:00)
Spark context Web UI available at None
Spark context available as 'sc' (master = local[*], app id = local-1740643079929).
SparkSession available as 'spark'.
>>>
```

Avec ce shell REPL, vous pouvez coder et tester de manière interactive.

Pytest

Pour les tests unitaires, vous pouvez utiliser `pytest` des scripts de travail AWS Glue Spark. Exécutez les commandes suivantes pour la préparation.

```
$ WORKSPACE_LOCATION=/local_path_to_workspace
$ SCRIPT_FILE_NAME=sample.py
$ UNIT_TEST_FILE_NAME=test_sample.py
$ mkdir -p ${WORKSPACE_LOCATION}/tests
$ vim ${WORKSPACE_LOCATION}/tests/${UNIT_TEST_FILE_NAME}
```

Exécutez la commande suivante pour exécuter `pytest` à l'aide de `docker run` :

```
$ docker run -i --rm \
```

```
    -v ~/.aws:/home/hadoop/.aws \
    -v $WORKSPACE_LOCATION:/home/hadoop/workspace/ \
    --workdir /home/hadoop/workspace \
    -e AWS_PROFILE=$PROFILE_NAME \
    --name glue5_pytest \
    public.ecr.aws/glue/aws-glue-libs:5 \
    -c "python3 -m pytest --disable-warnings"
```

Une fois l'exécution des tests unitaires terminée par `pytest`, votre sortie ressemblera à ceci :

```
============================= test session starts ==============================
platform linux -- Python 3.11.6, pytest-8.3.4, pluggy-1.5.0
rootdir: /home/hadoop/workspace
plugins: integration-mark-0.2.0
collected 1 item

tests/test_sample.py .                                                   [100%]

======================== 1 passed, 1 warning in 34.28s =========================
```

Configuration du conteneur pour utiliser Visual Studio Code

Pour configurer le conteneur à l'aide de Visual Studio Code, procédez comme suit :

1. Installez Visual Studio Code.
2. Installez Python.
3. Installez Visual Studio Code Remote - Containers
4. Ouvrez le dossier de l'espace de travail dans Visual Studio Code.
5. Appuyez sur `Ctrl+Shift+P` (Windows/Linux) ou sur `Cmd+Shift+P` (Mac).
6. Tapez `Preferences: Open Workspace Settings (JSON)`.
7. Appuyez sur Entrée.
8. Collez le JSON suivant et enregistrez-le.

```
{
    "python.defaultInterpreterPath": "/usr/bin/python3.11",
    "python.analysis.extraPaths": [
        "/usr/lib/spark/python/lib/py4j-0.10.9.7-src.zip:/usr/lib/spark/python/:/usr/
lib/spark/python/lib/",
    ]
```

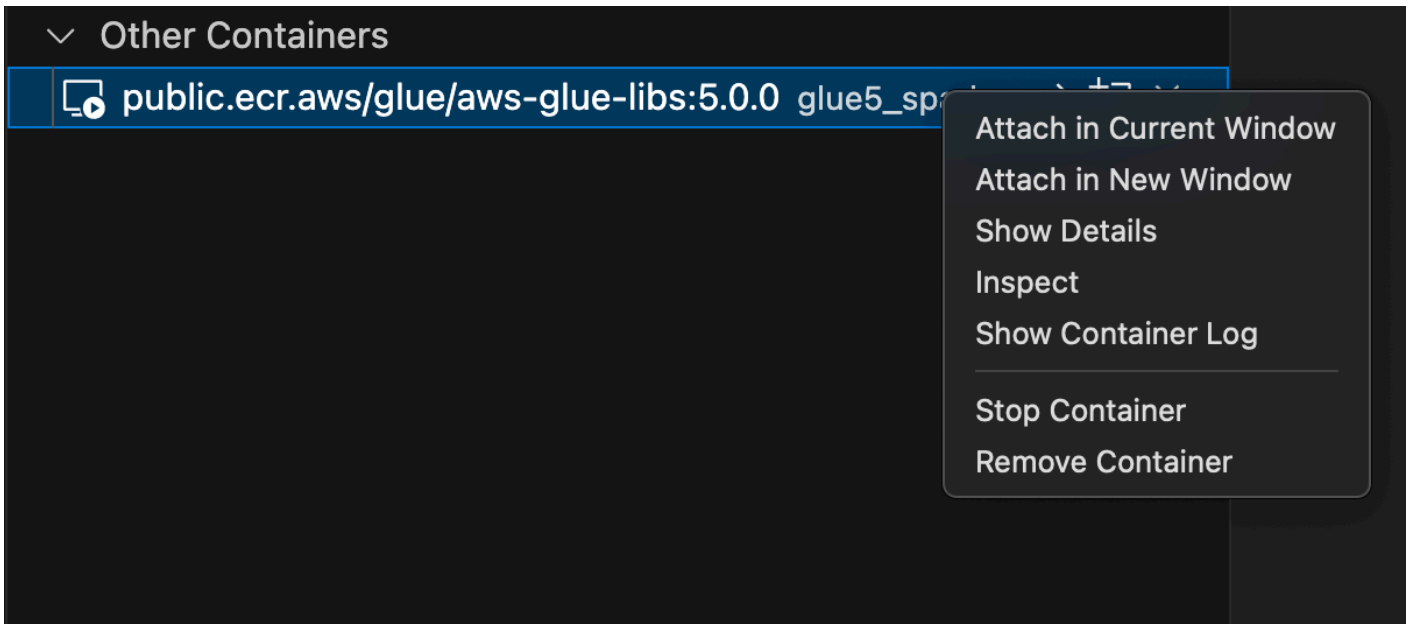
```
}
```

Pour configurer le conteneur :

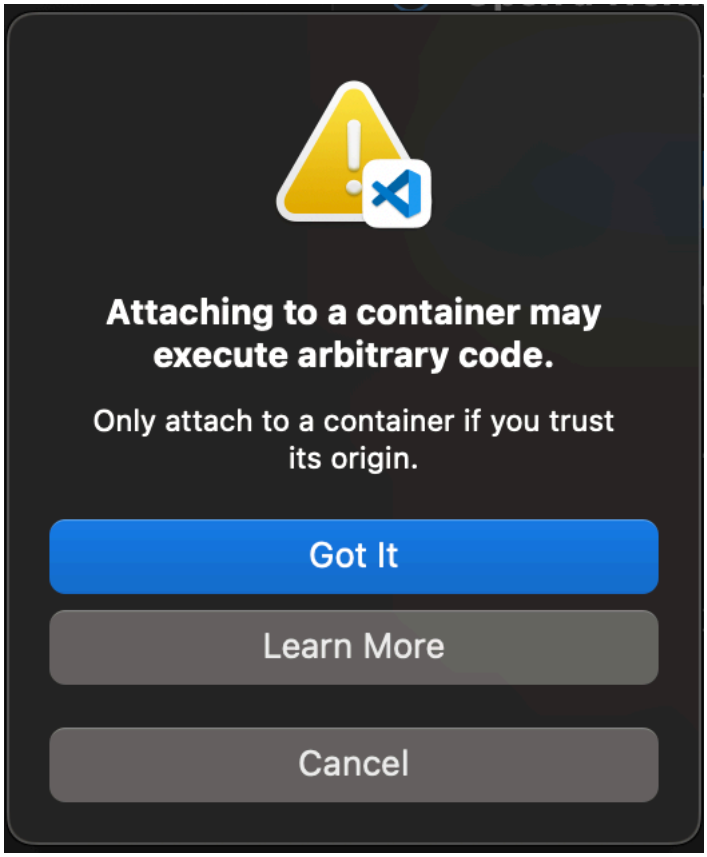
1. Exécutez le conteneur Docker.

```
$ docker run -it --rm \
    -v ~/.aws:/home/hadoop/.aws \
    -v $WORKSPACE_LOCATION:/home/hadoop/workspace/ \
    -e AWS_PROFILE=$PROFILE_NAME \
    --name glue5_pyspark \
    public.ecr.aws/glue/aws-glue-libs:5 \
    pyspark
```

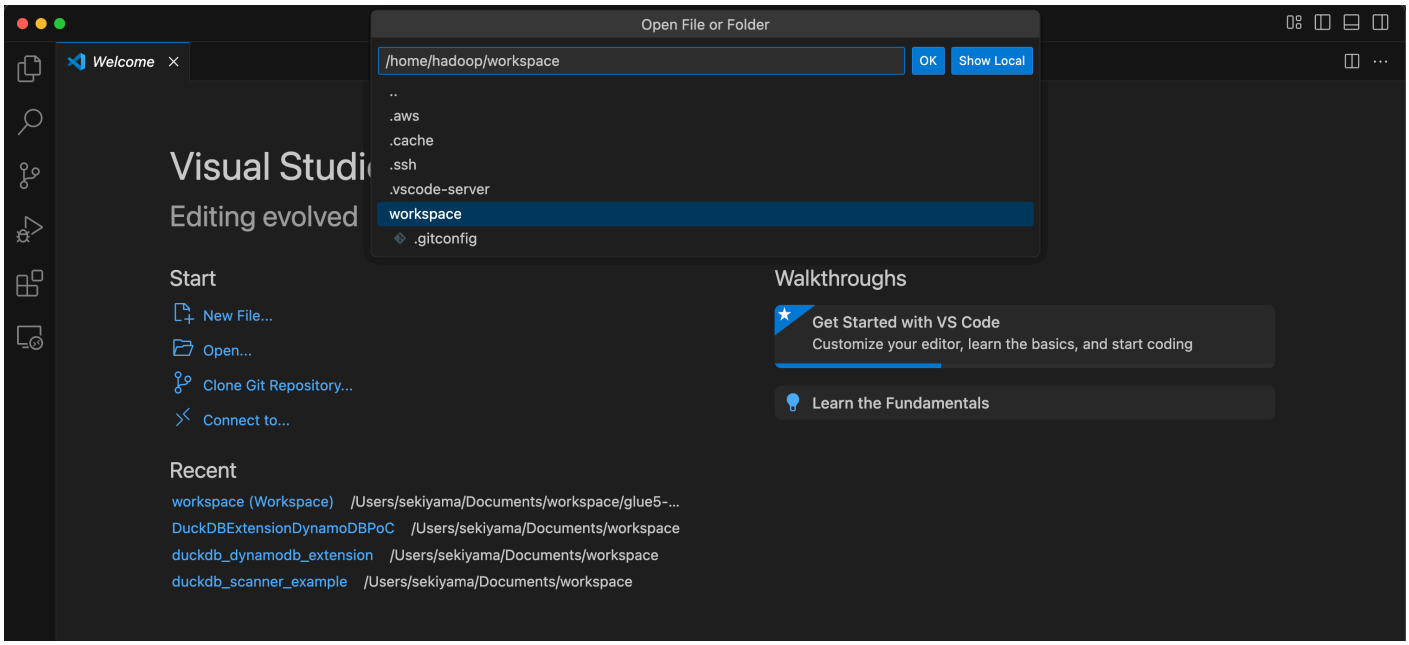
2. Démarrez Visual Studio Code.
3. Choisissez Remote Explorer (Explorateur distant) dans le menu de gauche, puis `amazon/aws-glue-libs:glue_libs_4.0.0_image_01`.
4. Cliquez avec le bouton droit de la souris et choisissez Joindre dans la fenêtre actuelle.



5. Si la boîte de dialogue suivante apparaît, choisissez J'ai compris.

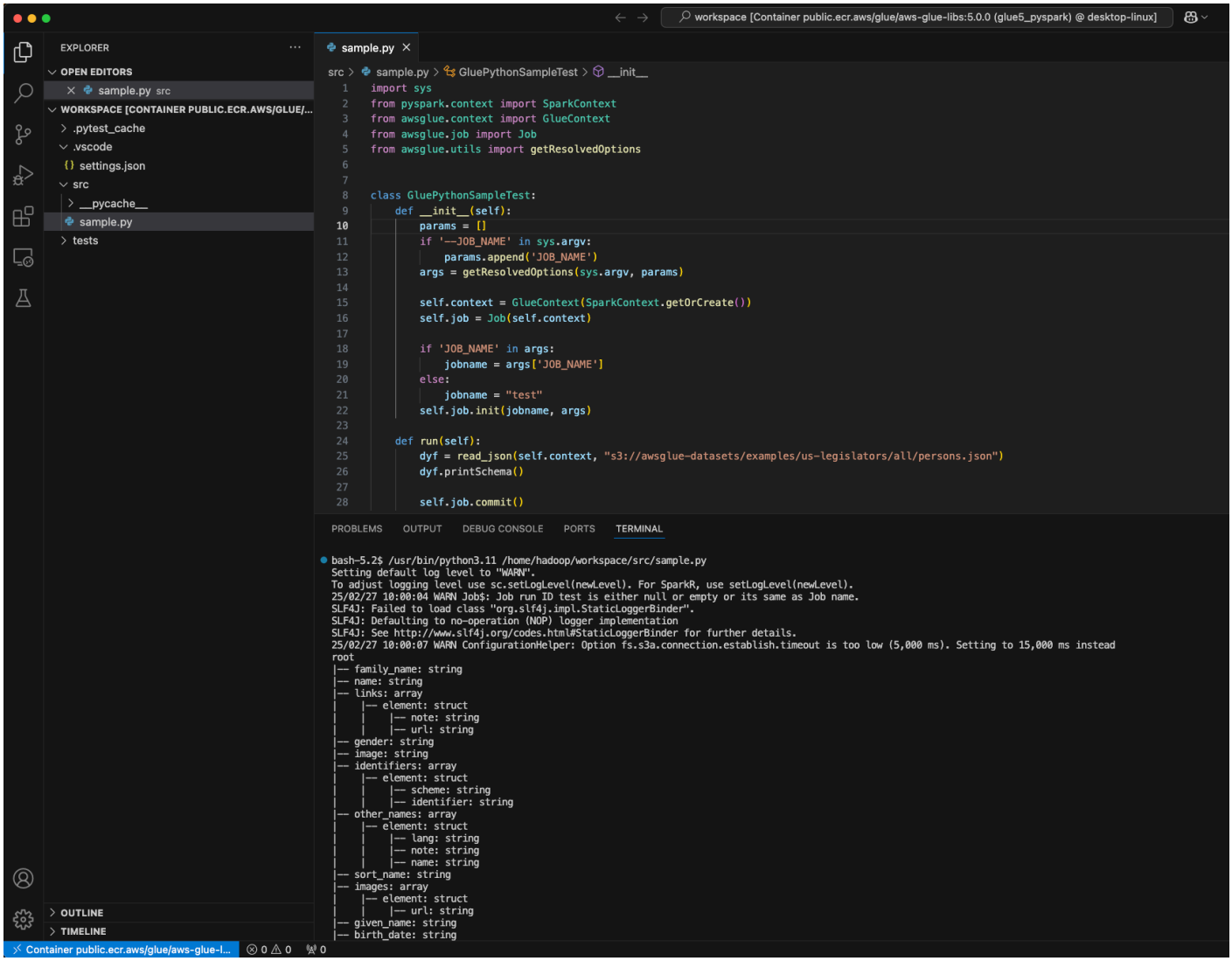


6. Ouvrir `/home/handoop/workspace/`.



7. Créez un AWS Glue PySpark script et choisissez Exécuter.

   Vous verrez l'exécution réussie du script.



Changements entre l'image Docker AWS Glue 4.0 et AWS Glue 5.0

Les principaux changements entre l'image Docker AWS Glue 4.0 et AWS Glue 5.0 :

- Dans la AWS Glue version 5.0, il existe une seule image de conteneur pour les tâches par lots et les tâches de streaming. Cela diffère de Glue 4.0, où il y avait une image pour le lot et une autre pour le streaming.
- Dans la AWS Glue version 5.0, le nom d'utilisateur par défaut du conteneur est`hadoop`. Dans la AWS Glue version 4.0, le nom d'utilisateur par défaut était`glue_user`.
- Dans la AWS Glue version 5.0, plusieurs bibliothèques supplémentaires, dont Livy JupyterLab et Livy, ont été supprimées de l'image. Vous pouvez les installer manuellement.

- Dans la AWS Glue version 5.0, toutes les bibliothèques Iceberg, Hudi et Delta sont préchargées par défaut, et la variable d'environnement n'`DATALAKE_FORMATS`est plus nécessaire. Avant la AWS Glue version 4.0, la variable d'`DATALAKE_FORMATS`environnement variable d'environnement était utilisée pour spécifier les formats de table spécifiques à charger.

La liste ci-dessus est spécifique à l'image Docker. Pour en savoir plus sur les mises à jour AWS Glue 5.0, consultez Présentation de la AWS Glue version 5.0 pour Apache Spark et Migration des AWS Glue tâches Spark vers AWS Glue la version 5.0.

Considérations

N'oubliez pas que les fonctionnalités suivantes ne sont pas prises en charge lorsque vous utilisez l'image du AWS Glue conteneur pour développer des scripts de travail localement.

- Signets de tâche
- AWS Glue Rédacteur de parquet (utilisation du format Parquet dans AWS Glue)
- FillMissingValues transformer
- FindMatches transformer
- Lecteur CSV SIMD vectorisé
- La propriété customJdbcDriverS3Path pour charger le pilote JDBC depuis le chemin Amazon S3
- AWS Glue Qualité des données
- Détection des données sensibles
- AWS Lake Formation vente d'informations d'identification basée sur des autorisations

Annexe : Ajout de pilotes JDBC et de bibliothèques Java

Pour ajouter un pilote JDBC qui n'est pas actuellement disponible dans le conteneur, vous pouvez créer un répertoire sous votre espace de travail avec les fichiers JAR dont vous avez besoin et monter le répertoire sur `/opt/spark/jars/` dans la commande docker run. Les fichiers JAR situés sous `/opt/spark/jars/` au sein du conteneur sont automatiquement ajoutés à Spark Classpath et pourront être utilisés pendant l'exécution de la tâche.

Par exemple, utilisez la commande docker run suivante pour ajouter des fichiers JAR de pilotes JDBC au shell REPL. PySpark

```
docker run -it --rm \
```

```
    -v ~/.aws:/home/hadoop/.aws \
    -v $WORKSPACE_LOCATION:/home/hadoop/workspace/ \
    -v $WORKSPACE_LOCATION/jars/:/opt/spark/jars/ \
    --workdir /home/hadoop/workspace \
    -e AWS_PROFILE=$PROFILE_NAME \
    --name glue5_jdbc \
    public.ecr.aws/glue/aws-glue-libs:5 \
    pyspark
```

Comme indiqué dans Considérations, l'option de `customJdbcDriverS3Path` connexion ne peut pas être utilisée pour importer un pilote JDBC personnalisé depuis Amazon S3 dans des images de AWS Glue conteneur.

Points de terminaison de développement

> **Note**
>
> L'expérience de la console pour les points de terminaison de développement a été supprimée à compter du 31 mars 2023. La création, la mise à jour et la surveillance des points de terminaison de développement sont toujours disponibles via API de points de terminaison de développement et AWS Glue CLI.

Nous vous recommandons vivement de migrer des points de terminaison de développement vers des séances interactives pour les raisons énumérées ci-dessous. Pour connaître les actions requises en matière de migration des points de terminaison de développement vers des séances interactives, veuillez consulter la rubrique Migration des points de terminaison de développement vers des séances interactives.

Description	Points d'extrémité du développeur	Sessions interactives
Prise en charge de la version Glue	Prend en charge les versions de AWS Glue 0.9 et 1.0	Prend en charge AWS Glue version 2.0 et ultérieures.
Les points de terminaison de développement ne sont désormais pas disponibles	Les séances interactives ne sont actuellement pas disponibles dans la région	

Description	Points d'extrémité du développeur	Sessions interactives
dans les régions Asie-Pacifique (Jakarta) (`ap-southeast-3`), Moyen-Orient (EAU) (`me-central-1`) Europe (Espagne) (`eu-south-2`), Europe (Zurich) (`eu-central-2`) ou dans les autres nouvelles régions.	Moyen-Orient (EAU) (`me-central-1`), mais le seront peut-être ultérieurement.	
Méthode d'accès au cluster Spark	Supporte SSH, le shell REPL, le bloc-notes Jupyter, l'IDE (par exemple) PyCharm	prend en charge le AWS Glue Studio bloc-notes, le bloc-notes Jupyter, divers IDEs (par exemple, Visual Studio Code PyCharm) et SageMaker le bloc-notes AI
Délai jusqu'à la première requête	La configuration d'un cluster Spark prend 10 à 15 minutes	La configuration d'un cluster Spark éphémère peut prendre jusqu'à 1 minute

Description	Points d'extrémité du développeur	Sessions interactives
Modèle de tarification	AWS frais pour les points de terminaison de développe ment en fonction de l'heure à laquelle le point de terminais on est provisionné et du nombre de. DPUs Les points de terminaison du développe ment n'arrivent pas à expiratio n. Un temps de facturation minimal de 10 minutes est calculé pour chaque point de terminaison de développement mis en service. En outre, les AWS frais pour le bloc-notes Jupyter sur les EC2 instances Amazon et les blocs-notes SageMaker AI lorsque vous les configurez avec des points de terminaison de développe ment.	AWS frais pour les sessions interactives en fonction de la durée pendant laquelle la session est active et du nombre de DPUs. Les sessions interactives ont des délais d'inactivité configura bles. AWS Glue Studioles ordinateurs portables fournissent une interface intégrée pour les sessions interactives et sont proposés sans frais supplémentaires. Un temps de facturation minimal d'une minute est calculé pour chaque session interactive. Les blocs-notes AWS Glue Studio disposent d'une interface intégrée pour les sessions interactives et sont proposés sans frais supplémentaires
Expérience de la console	Disponible uniquement via l'interface de ligne de commande et l'API	Disponible via la AWS Glue console, la CLI et APIs

Migration des points de terminaison de développement vers des séances interactives

Utilisez la liste de contrôle suivante pour déterminer la méthode appropriée permettant de migrer des points de terminaison de développement vers des séances interactives.

Votre script dépend-il de fonctionnalités spécifiques à AWS Glue version 0.9 ou 1.0 (par exemple, HDFS, YARN, etc.) ?

Si la réponse est oui, consultez Migrating AWS Glue jobs to AWS Glue version 3.0 pour savoir comment migrer de Glue 0.9 ou 1.0 vers Glue 3.0 et versions ultérieures.

Quelle méthode vous permet d'accéder à votre point de terminaison de développement ?

Si vous utilisez cette méthode,	Procédez comme suit
SageMaker Carnet AI, bloc-notes Jupyter ou JupyterLab	Migrez vers le bloc-notes AWS Glue Studio en téléchargeant les fichiers `.ipynb` sur Jupyter et créez une nouvelle tâche de bloc-notes AWS Glue Studio en téléchargeant le fichier `.ipynb`. Vous pouvez également utiliser SageMaker AI Studio et sélectionner le AWS Glue noyau.
Zeppelin notebook	Convertissez manuellement le bloc-notes en bloc-notes Jupyter en copiant et collant du code ou automatiquement à l'aide d'un convertisseur tiers tel que ze2nb. Utilisez ensuite le bloc-notes dans AWS Glue Studio Notebook ou SageMaker AI Studio.
IDE	Voir AWS GlueTâches d'auteur PyCharm utilisant des sessions AWS Glue interactives ou Utilisation de sessions interactives avec Microsoft Visual Studio Code.
REPL	Installez le `aws-glue-session package` en local, puis exécutez la commande suivante : • Pour Python : `jupyter console --kernal glue_pyspark` • Pour Scala : `jupyter console --kernal glue_spark`

Si vous utilisez cette méthode,	Procédez comme suit
SSH	Il n'existe pas d'option correspondante sur les séances interactives. Vous pouvez également utiliser une image Docker. Pour en savoir plus, veuillez consulter la rubrique Développement à l'aide d'une image Docker.

Les sections suivantes renseignent sur l'utilisation des points de terminaison du développeur pour développer des tâches dans AWS Glue (version 1.0).

Rubriques

- Développement de scripts à l'aide de points de terminaison de développement
- Gestion des blocs-notes

Développement de scripts à l'aide de points de terminaison de développement

> **Note**
>
> Les points de terminaison de développement ne sont pris en charge que pour les versions AWS Glue antérieures à 2.0. Pour un environnement interactif dans lequel vous pouvez créer et tester des scripts ETL, utilisez Notebooks on AWS Glue Studio.

AWS Glue peut créer un environnement (appelé point de terminaison de développement) que vous pouvez utiliser pour développer et tester de façon itérative vos scripts d'extraction, transformation et chargement (ETL). Vous pouvez créer, modifier et supprimer des points de terminaison de développement à l'aide de l'API ou de la console AWS Glue.

Gestion de votre environnement de développement

Lorsque vous créez un point de terminaison de développement, vous fournissez des valeurs de configuration pour mettre en service l'environnement de développement. Ces valeurs indiquent à AWS Glue comment configurer le réseau afin que vous puissiez accéder en toute sécurité au point de terminaison et que ce dernier puisse accéder à vos magasins de données.

Vous pouvez alors créer un bloc-notes qui se connecte au point de terminaison et utiliser votre bloc-notes pour créer et tester votre script ETL. Lorsque vous êtes satisfait des résultats de votre processus de développement, vous pouvez créer une tâche ETL qui exécute votre script. Grâce à ce processus, vous pouvez ajouter des fonctions et déboguer vos scripts de manière interactive.

Suivez les didacticiels de cette section pour apprendre à utiliser votre point de terminaison de développement avec des blocs-notes.

Rubriques

- Flux de travail de point de terminaison de développement
- Comment les terminaux AWS Glue de développement fonctionnent avec les ordinateurs portables SageMaker
- Ajout d'un point de terminaison de développement
- Accès à votre point de terminaison de développement
- Tutoriel : Configuration d'un bloc-notes Jupyter JupyterLab pour tester et déboguer des scripts ETL
- Tutoriel : Utiliser un bloc-notes SageMaker AI avec votre terminal de développement
- Didacticiel : Utilisation d'un shell REPL avec votre point de terminaison de développement
- Tutoriel : Configuration d'un PyCharm environnement professionnel avec un terminal de développement
- Configuration avancée : partage de points de terminaison de développement entre plusieurs utilisateurs

Flux de travail de point de terminaison de développement

Pour utiliser un point de terminaison de développement AWS Glue, vous pouvez suivre ce flux de travail :

1. Créez un point de terminaison de développement à l'aide de l'API. Ce point de terminaison est lancé dans un Virtual Private Cloud (VPC) avec vos groupes de sécurité définis.
2. L'API interroge le point de terminaison de développement jusqu'à ce qu'il soit alloué et prêt pour le travail. Lorsqu'il est prêt, connectez-vous au point de terminaison de développement à l'aide d'une des méthodes suivantes pour créer et tester les scripts AWS Glue.

    - Créez un bloc-notes basé sur l' SageMaker IA dans votre compte. Pour plus d'informations sur la création d'un bloc-notes, consultez the section called “Créer du code avec des blocs-notes AWS Glue Studio”.

- Ouvrez une fenêtre de terminal pour vous connecter directement à un point de terminaison de développement.
- Si vous disposez de l'édition professionnelle de l'IDE JetBrains PyCharm Python, connectez-la à un point de développement et utilisez-la pour développer de manière interactive. Si vous insérez `pydevd` des instructions dans votre script, il PyCharm peut prendre en charge les points d'arrêt distants.

3. Une fois que vous avez terminé le débogage et les tests sur votre point de terminaison de développement, vous pouvez le supprimer.

Comment les terminaux AWS Glue de développement fonctionnent avec les ordinateurs portables SageMaker

L'un des moyens les plus courants d'accéder à vos terminaux de développement consiste à utiliser Jupyter sur des ordinateurs portables. SageMaker Le bloc-notes Jupyter est une application web open source largement utilisée pour la visualisation, l'analytique, le machine learning, etc. Un AWS Glue SageMaker bloc-notes vous offre une expérience de bloc-notes Jupyter avec des terminaux de AWS Glue développement. Dans le AWS Glue SageMaker bloc-notes, l'environnement du bloc-notes Jupyter est préconfiguré avec SparkMagicun plugin Jupyter open source permettant de soumettre des tâches Spark à un cluster Spark distant. Apache Livy est un service qui permet l'interaction avec un cluster Spark distant via une API REST. Dans le AWS Glue SageMaker bloc-notes, SparkMagic est configuré pour appeler l'API REST sur un serveur Livy exécuté sur un point de AWS Glue développement.

Le flux de texte suivant explique le fonctionnement de chaque composant :

AWS Glue SageMaker bloc-notes : (Jupyter → SparkMagic) → (réseau) → point de terminaison de AWS Glue développement : (Apache Livy → Apache Spark)

Une fois que vous avez exécuté votre script Spark écrit dans chaque paragraphe d'un bloc-notes Jupyter, le code Spark est soumis au serveur Livy via SparkMagic, puis une tâche Spark nommée « Livy-session-n » s'exécute sur le cluster Spark. Cette tâche s'appelle une séance Livy. La tâche Spark sera exécutée pendant que la séance du bloc-notes sera active. La tâche Spark sera terminée lorsque vous arrêterez le noyau Jupyter à partir du bloc-notes ou lorsque la séance aura expiré. Une tâche Spark est lancée par fichier (.ipynb) de bloc-notes.

Vous pouvez utiliser un seul point de AWS Glue développement avec plusieurs instances de SageMaker bloc-notes. Vous pouvez créer plusieurs fichiers de bloc-notes dans chaque instance

de SageMaker bloc-notes. Lorsque vous ouvrez un fichier de bloc-notes et que vous exécutez les paragraphes, une session Livy est lancée par fichier de bloc-notes sur le cluster Spark via SparkMagic. Chaque séance Livy correspond à une seule tâche Spark.

Comportement par défaut pour les terminaux AWS Glue de développement et les blocs-notes SageMaker

Les tâches Spark s'exécutent en fonction de la configuration Spark. Il existe plusieurs manières de définir la configuration de Spark (par exemple, la configuration du cluster Spark, SparkMagic la configuration de Spark, etc.).

Par défaut, Spark alloue des ressources de cluster à une séance Livy en fonction de la configuration du cluster Spark. Dans les points de terminaison de développement AWS Glue, la configuration du cluster dépend du type de processus. Voici un tableau qui explique les configurations les plus courantes par type de tâche.

	Standard	G.1X	G.2X
`spark.driver.memory`	5G	10G	20G
`spark.executor.memory`	5G	10G	20G
`spark.executor.cores`	4	8	16
`spark.dynamicAllocation.enabled`	TRUE	TRUE	TRUE

Le nombre maximum de programmes d'exécution Spark est calculé automatiquement par combinaison de DPU (ou `NumberOfWorkers`) et de type de processus.

	Standard	G.1X	G.2X
Nombre maximal de programmes d'exécution Spark	`(DPU - 1) * 2 - 1`	`(NumberOfWorkers - 1)`	`(NumberOfWorkers - 1)`

Par exemple, si votre point de terminaison de développement a 10 processus et que le type de processus est `G.1X`, alors vous aurez 9 programmes d'exécution Spark et l'ensemble du cluster aura 90 Go de mémoire d'exécuteur puisque chaque exécuteur aura 10 Go de mémoire.

Quel que soit le type de nœud de processus spécifié, l'allocation dynamique des ressources Spark sera activée. Si un jeu de données est suffisamment volumineux, Spark peut allouer tous les programmes d'exécution à une seule séance Livy, car `spark.dynamicAllocation.maxExecutors` n'est pas défini par défaut. Cela signifie que d'autres séances Livy sur le même point de terminaison de développement attendront le lancement de nouveaux programmes d'exécution. Si le jeu de données est peu volumineux, Spark pourra allouer des programmes d'exécution à plusieurs séances Livy en même temps.

> **Note**
>
> Pour plus d'informations sur la façon dont les ressources sont allouées dans différents cas d'utilisation et sur la façon dont vous définissez une configuration pour modifier le comportement, consultez Configuration avancée : partage de points de terminaison de développement entre plusieurs utilisateurs.

Ajout d'un point de terminaison de développement

Utilisez les points de terminaison de développement pour développer et tester de façon itérative vos scripts ETL (extraction, transformation et chargement) dans AWS Glue. L'utilisation des points de terminaison de développement n'est disponible que via l' AWS Command Line Interface.

1. Dans une fenêtre de ligne de commande, entrez une commande similaire à ce qui suit.

```
aws glue create-dev-endpoint --endpoint-name "endpoint1" --role-arn
 "arn:aws:iam::account-id:role/role-name" --number-of-nodes "3" --glue-version
 "1.0" --arguments '{"GLUE_PYTHON_VERSION": "3"}' --region "region-name"
```

Cette commande spécifie AWS Glue version 1.0. Comme cette version prend en charge Python 2 et Python 3, vous pouvez utiliser le paramètre `arguments` pour indiquer la version Python souhaitée. Si le paramètre `glue-version` est omis, AWS Glue version 0.9 est utilisé. Pour de plus amples informations sur les versions d'AWS Glue, veuillez consulter Glue version job property.

Pour plus d'informations sur les paramètres de ligne de commande supplémentaires, reportez-vous create-dev-endpointà la référence des AWS CLI commandes.

2. (Facultatif) Entrez la commande suivante pour vérifier le statut du point de terminaison de développement. Lorsque le statut passe à `READY`, le point de terminaison de développement est prêt à être utilisé.

```
aws glue get-dev-endpoint --endpoint-name "endpoint1"
```

Accès à votre point de terminaison de développement

Lorsque vous créez un point de terminaison de développement dans un Virtual Private Cloud (VPC), AWS Glue renvoie uniquement une adresse IP privée. Le champ de l'adresse IP publique n'est pas renseigné. Lorsque vous créez un point de terminaison de développement non VPC, AWS Glue renvoie uniquement une adresse IP publique.

Si votre point de terminaison de développement possède une adresse publique, vérifiez qu'elle est accessible avec la clé privée SSH pour le point de terminaison de développement, comme dans l'exemple suivant.

```
ssh -i dev-endpoint-private-key.pem glue@public-address
```

Supposons que votre point de terminaison de développement possède une adresse privée, que votre sous-réseau VPC soit routable à partir de l'Internet public et que ses groupes de sécurité autorisent l'accès entrant depuis votre client. Dans ce cas, suivez les étapes ci-dessous pour attacher une adresse IP Elastic à un point de terminaison de développement afin d'autoriser l'accès depuis Internet.

> **Note**
>
> Pour utiliser des adresses IP Elastic, le sous-réseau utilisé nécessite une passerelle Internet associée via la table de routage.

Pour accéder à un point de terminaison de développement en attachant une adresse IP Elastic

1. Ouvrez la AWS Glue console à l'adresse https://console.aws.amazon.com/glue/.
2. Dans le panneau de navigation, sélectionnez Dev endpoints (Points de terminaison de dév.) et accédez à la page des détails des points de terminaison de développement. Enregistrez l'adresse privée pour l'utiliser à l'étape suivante.
3. Ouvrez la EC2 console Amazon à l'adresse https://console.aws.amazon.com/ec2/.
4. Dans le panneau de navigation, sous Network & Security (Réseau et sécurité), choisissez Network Interfaces (Interfaces réseau).
5. Recherchez le DNS privé (IPv4) correspondant à l'adresse privée sur la page de détails du point de terminaison de développement de la AWS Glue console.

   Vous devrez peut-être modifier les colonnes affichées sur votre EC2 console Amazon. Notez l'ID d'interface réseau (ENI) de cette adresse (par exemple, `eni-12345678`).
6. Sur la EC2 console Amazon, sous Réseau et sécurité, sélectionnez Elastic IPs.
7. Choisissez Allocate new address (Allouer une nouvelle adresse), puis Allocate (Allouer) pour allouer une nouvelle adresse IP Elastic.
8. Sur la IPs page Elastic, choisissez la nouvelle adresse IP Elastic allouée. Puis, choisissez Actions, Associate Address (Associer l'adresse).
9. Dans la page Associer l'adresse, procédez comme suit :

   - Pour Resource type (Type de ressource), choisissez Network interface (Interface réseau).
   - Dans la zone Interface réseau, entrez l'ID d'interface réseau (ENI) de l'adresse privée.
   - Choisissez Associer.
10. Vérifiez que l'adresse IP Elastic nouvellement associée est accessible avec la clé privée SSH associée au point de terminaison de développement, comme dans l'exemple suivant.

```
ssh -i dev-endpoint-private-key.pem glue@elastic-ip
```

Pour plus d'informations sur l'utilisation d'un hôte bastion pour obtenir un accès SSH à l'adresse privée du point de terminaison de développement, consultez le billet de blog sur la AWS sécurité Connectez-vous en toute sécurité aux instances Linux exécutées dans un Amazon VPC privé.

Tutoriel : Configuration d'un bloc-notes Jupyter JupyterLab pour tester et déboguer des scripts ETL

Dans ce didacticiel, vous allez connecter un bloc-notes Jupyter en JupyterLab cours d'exécution sur votre machine locale à un point de terminaison de développement. Vous procédez ainsi pour pouvoir exécuter, déboguer et tester de manière interactive les scripts Extract-transform-load (ETL) AWS Glue avant de les déployer. Ce didacticiel utilise le réacheminement de port Secure Shell (SSH) pour connecter votre ordinateur local à un point de terminaison de développement AWS Glue. Pour plus d'informations, consultez la page Wikipedia Redirection de port.

Étape 1 : Installation JupyterLab et Sparkmagic

Vous pouvez effectuer l'installation JupyterLab en utilisant `conda` ou`pip`. `conda`est un système de gestion de paquets open source et un système de gestion d'environnement qui fonctionne sous Windows, macOS et Linux. `pip`est le programme d'installation du package pour Python.

Si vous effectuez l'installation sur macOS, vous devez avoir installé Xcode avant de pouvoir installer Sparkmagic.

1. Install JupyterLab, Sparkmagic et les extensions associées.

```
$ conda install -c conda-forge jupyterlab
$ pip install sparkmagic
$ jupyter nbextension enable --py --sys-prefix widgetsnbextension
$ jupyter labextension install @jupyter-widgets/jupyterlab-manager
```

2. Vérifiez le répertoire `sparkmagic` depuis `Location`.

```
$ pip show sparkmagic | grep Location
Location: /Users/username/.pyenv/versions/anaconda3-5.3.1/lib/python3.7/site-
packages
```

3. Remplacez votre répertoire par celui renvoyé pour`Location`, et installez les noyaux pour Scala et. PySpark

```
$ cd /Users/username/.pyenv/versions/anaconda3-5.3.1/lib/python3.7/site-packages
$ jupyter-kernelspec install sparkmagic/kernels/sparkkernel
$ jupyter-kernelspec install sparkmagic/kernels/pysparkkernel
```

4. Télécharger un exemple de fichier `config`

```
$ curl -o ~/.sparkmagic/config.json https://raw.githubusercontent.com/jupyter-incubator/sparkmagic/master/sparkmagic/example_config.json
```

Dans ce fichier de configuration, vous pouvez configurer les paramètres liés à Spark tels que `driverMemory` et `executorCores`.

Étape 2 : Démarrer JupyterLab

Lorsque vous démarrez JupyterLab, votre navigateur Web par défaut s'ouvre automatiquement et l'URL `http://localhost:8888/lab/workspaces/{workspace_name}` s'affiche.

```
$ jupyter lab
```

Étape 3 : lancer le transfert de port SSH pour vous connecter à votre point de terminaison de développement

Ensuite, utilisez la redirection de port local SSH pour transférer un port local (ici, 8998) vers la destination distante définie par AWS Glue (`169.254.76.1:8998`).

1. Ouvrez une fenêtre de terminal séparée qui vous donne accès à SSH. Dans Microsoft Windows, vous pouvez utiliser le shell BASH fourni par Git pour Windows ou installer Cygwin.
2. Exécutez la commande SSH suivante, modifiée comme suit :

   - Remplacez *`private-key-file-path`* par un chemin vers le fichier `.pem` qui contient la clé privée correspondant à la clé publique que vous avez utilisée pour créer le point de terminaison de développement.
   - Si vous réacheminez un autre port que 8998, remplacez 8998 par le numéro de port que vous utilisez localement. L'adresse, `169.254.76.1:8998`, est le port distant et vous ne la modifiez pas.
   - Remplacez *`dev-endpoint-public-dns`* par l'adresse DNS publique du point de terminaison de développement. Pour trouver cette adresse, accédez à votre point de

terminaison dans la console AWS Glue, choisissez le nom et copiez la Public address (Adresse publique) qui apparaît dans la page Endpoint details (Détails de point de terminaison).

```
ssh -i private-key-file-path -NTL 8998:169.254.76.1:8998 glue@dev-endpoint-public-dns
```

Vous verrez probablement un message d'avertissement, tel que le suivant :

```
The authenticity of host 'ec2-xx-xxx-xxx-xx.us-west-2.compute.amazonaws.com
 (xx.xxx.xxx.xx)'
can't be established.  ECDSA key fingerprint is SHA256:4e97875Brt+1wKzRko
+JflSnp21X7aTP3BcFnHYLEts.
Are you sure you want to continue connecting (yes/no)?
```

Entrez **yes** et laissez la fenêtre du terminal ouverte pendant que vous l'utilisez JupyterLab.

3. Vérifiez que le réacheminement du port SSH fonctionne correctement avec le point de terminaison de développement.

```
$ curl localhost:8998/sessions
{"from":0,"total":0,"sessions":[]}
```

Étape 4 : exécuter un fragment de script simple dans un paragraphe de bloc-notes

Votre bloc-notes JupyterLab devrait désormais fonctionner avec votre terminal de développement. Saisissez le fragment de script ci-après dans votre bloc-notes et exécutez-le.

1. Vérifiez que Spark fonctionne correctement. La commande suivante demande à Spark de calculer 1, puis d'imprimer la valeur.

```
spark.sql("select 1").show()
```

2. Vérifiez si AWS Glue Data Catalog l'intégration fonctionne. La commande suivante répertorie les tables de Data Catalog.

```
spark.sql("show tables").show()
```

3. Vérifiez qu'un simple fragment de script qui utilise des bibliothèques AWS Glue fonctionne.

Le script suivant utilise les métadonnées de la table `persons_json` dans AWS Glue Data Catalog pour créer un `DynamicFrame` à partir de vos exemples de données. Il affiche ensuite le nombre d'éléments et le schéma des données.

```
import sys
from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create a Glue context
glueContext = GlueContext(SparkContext.getOrCreate())

# Create a DynamicFrame using the 'persons_json' table
persons_DyF = glueContext.create_dynamic_frame.from_catalog(database="legislators",
 table_name="persons_json")

# Print out information about *this* data
print("Count:  ", persons_DyF.count())
persons_DyF.printSchema()
```

La sortie du script est la suivante.

```
Count:  1961
root
|-- family_name: string
|-- name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- gender: string
|-- image: string
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- name: string
|    |    |-- lang: string
```

```
|-- sort_name: string
|-- images: array
|    |-- element: struct
|    |    |-- url: string
|-- given_name: string
|-- birth_date: string
|-- id: string
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
|-- death_date: string
```

Résolution des problèmes

- Au cours de l'installation de JupyterLab, si votre ordinateur se trouve derrière un proxy ou un pare-feu d'entreprise, vous risquez de rencontrer des erreurs HTTP et SSL en raison de profils de sécurité personnalisés gérés par les services informatiques de l'entreprise.

  Voici un exemple d'erreur typique qui se produit lorsque `conda` ne peut pas se connecter à ses propres référentiels :

  ```
  CondaHTTPError: HTTP 000 CONNECTION FAILED for url <https://repo.anaconda.com/pkgs/
  main/win-64/current_repodata.json>
  ```

  Cela peut se produire parce que votre entreprise peut bloquer les connexions aux référentiels les plus utilisés en Python et dans JavaScript les communautés. Pour plus d'informations, consultez la section Problèmes d'installation sur le JupyterLab site Web.
- Si vous obtenez un message d'erreur connection refused (connexion refusée) lorsque vous essayez de vous connecter à votre point de terminaison de développement, vous utilisez peut-être un point de terminaison de développement obsolète. Essayez de créer un nouveau point de terminaison de développement et de vous reconnecter.

Tutoriel : Utiliser un bloc-notes SageMaker AI avec votre terminal de développement

DansAWS Glue, vous pouvez créer un point de terminaison de développement, puis créer un bloc-notes d' SageMaker IA pour vous aider à développer vos scripts ETL et d'apprentissage automatique. Un bloc-notes SageMaker AI est une instance de calcul d'apprentissage automatique entièrement gérée qui exécute l'application Jupyter Notebook.

1. Dans la console AWS Glue, choisissez Dev endpoints (Points de terminaisons de dév.) pour accéder à la liste des points de terminaison de développement.
2. Cochez la case à côté du nom du point de terminaison de développement que vous souhaitez utiliser, puis dans le menu Action, choisissez Créer un SageMaker bloc-notes.
3. Complétez la page Create and configure a notebook (Créer et configurer un bloc-notes) comme suit :

   a. Entrez un nom de bloc-notes.

   b. Sous Attach to development endpoint (Attacher au point de terminaison de développement), vérifiez le point de terminaison de développement.

   c. Créez ou choisissez un rôle Gestion des identités et des accès AWS (IAM).

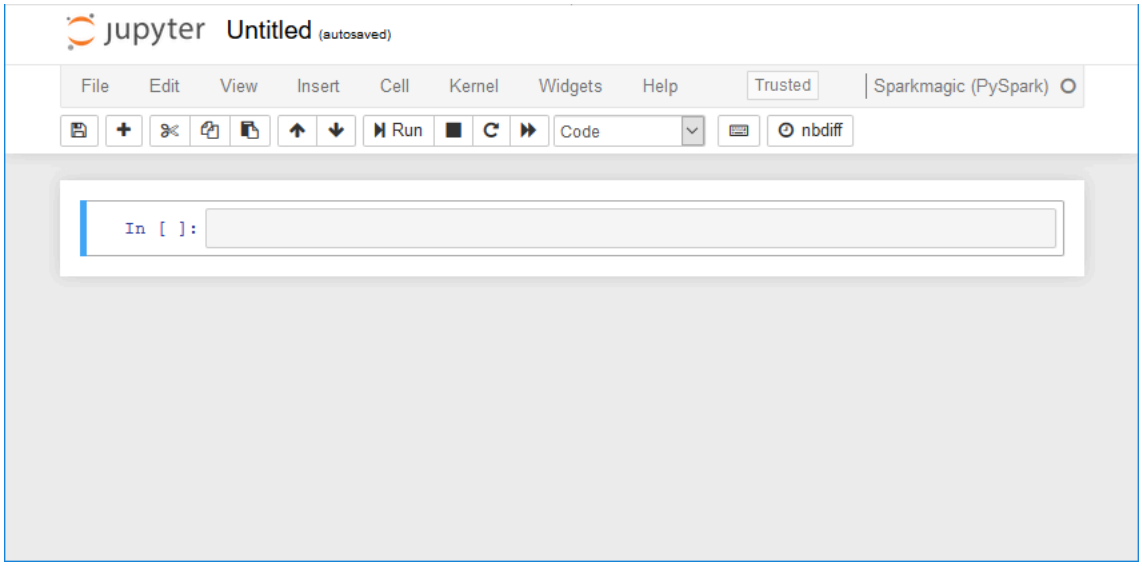
      La création d'un rôle est recommandée. Si vous utilisez un rôle existant, assurez-vous qu'il dispose des autorisations requises. Pour plus d'informations, consultez the section called “Étape 6 : créer une politique IAM pour les ordinateurs portables dotés d' SageMaker intelligence artificielle”.

   d. (Facultatif) Choisissez un VPC, un sous-réseau et un ou plusieurs groupes de sécurité.

   e. (Facultatif) Choisissez une clé AWS Key Management Service de chiffrement.

   f. (Facultatif) Ajoutez des balises pour l'instance de bloc-notes.
4. Choisissez Create Notebook (Créer un bloc-notes). Sur la page Notebooks (Bloc-notes), choisissez l'icône d'actualisation en haut à droite, puis continuez jusqu'à ce que l'état `Ready` s'affiche.
5. Sélectionnez la case à cocher en regard du nom du nouveau bloc-notes, puis choisissez Open notebook (Ouvrir le bloc-notes).
6. Créez un nouveau bloc-notes : sur la page Jupyter, choisissez Nouveau, puis Sparkmagic (). PySpark

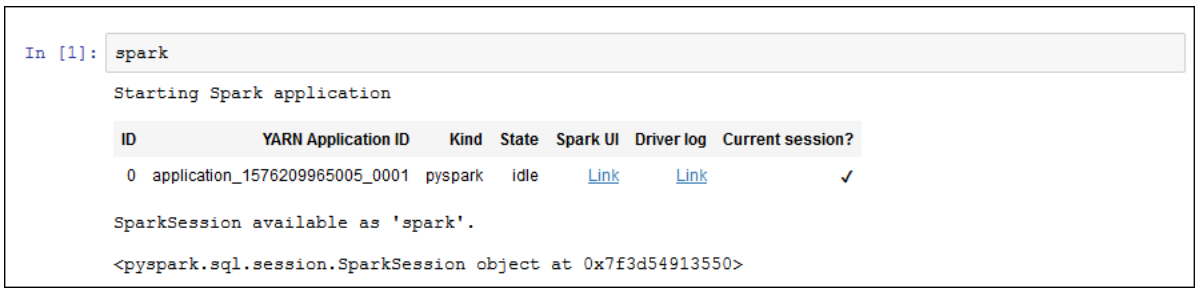
   Votre écran doit maintenant avoir l'aspect suivant :



7. (Facultatif) En haut de la page, choisissez Untitled (Sans titre), et donnez un nom au bloc-notes.
8. Pour démarrer une application Spark, entrez la commande suivante dans le bloc-notes, puis dans la barre d'outils, choisissez Run (Exécuter).

```
spark
```

Après un court délai, vous devriez voir la réponse suivante :



9. Créez un cadre dynamique et exécutez une requête sur celui-ci : copiez, collez et exécutez le code suivant, qui génère le nombre et le schéma de la table `persons_json`.

```
import sys
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.transforms import *
glueContext = GlueContext(SparkContext.getOrCreate())
persons_DyF = glueContext.create_dynamic_frame.from_catalog(database="legislators",
 table_name="persons_json")
print ("Count:  ", persons_DyF.count())
```

```
persons_DyF.printSchema()
```

Didacticiel : Utilisation d'un shell REPL avec votre point de terminaison de développement

DansAWS Glue, vous pouvez créer un point de terminaison de développement, puis invoquer un shell REPL (Read—Evaluate—Print Loop) pour exécuter le PySpark code de manière incrémentielle afin de pouvoir déboguer de manière interactive vos scripts ETL avant de les déployer.

Pour utiliser un REPL sur un point de terminaison de développement, vous devez être autorisé à accéder au point de terminaison par SSH.

1. Sur votre ordinateur local, ouvrez une fenêtre de terminal pouvant exécuter des commandes SSH, et collez-y la commande SSH modifiée. Exécutez la commande .

   En supposant que vous avez accepté la version 1.0 de AWS Glue avec Python 3 pour le point de terminaison de développement, la sortie se présente comme suit :

```
Python 3.6.8 (default, Aug  2 2019, 17:42:44)
[GCC 4.8.5 20150623 (Red Hat 4.8.5-28)] on linux
Type "help", "copyright", "credits" or "license" for more information.
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/share/aws/glue/etl/jars/glue-assembly.jar!/
org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/spark/jars/slf4j-log4j12-1.7.16.jar!/
org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use
 setLogLevel(newLevel).
2019-09-23 22:12:23,071 WARN  [Thread-5] yarn.Client (Logging.scala:logWarning(66))
 - Neither spark.yarn.jars nor spark.yarn.archive is set, falling back to uploading
 libraries under SPARK_HOME.
2019-09-23 22:12:26,562 WARN  [Thread-5] yarn.Client (Logging.scala:logWarning(66))
 - Same name resource file:/usr/lib/spark/python/lib/pyspark.zip added multiple
 times to distributed cache
2019-09-23 22:12:26,580 WARN  [Thread-5] yarn.Client (Logging.scala:logWarning(66))
 - Same path resource file:///usr/share/aws/glue/etl/python/PyGlue.zip added
 multiple times to distributed cache.
```

```
2019-09-23 22:12:26,581 WARN  [Thread-5] yarn.Client (Logging.scala:logWarning(66))
 - Same path resource file:///usr/lib/spark/python/lib/py4j-src.zip added multiple
 times to distributed cache.
2019-09-23 22:12:26,581 WARN  [Thread-5] yarn.Client (Logging.scala:logWarning(66))
 - Same path resource file:///usr/share/aws/glue/libs/pyspark.zip added multiple
 times to distributed cache.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /__ / .__/\_,_/_/ /_/\_\   version 2.4.3
      /_/

Using Python version 3.6.8 (default, Aug  2 2019 17:42:44)
SparkSession available as 'spark'.
>>>
```

2. Vérifiez que le shell de la boucle REPL fonctionne correctement en tapant la déclaration, `print(spark.version)`. Si la version Spark s'affiche, votre REPL est prêt à être utilisé.
3. Vous pouvez maintenant essayer d'exécuter le script simple suivant, ligne par ligne, dans le shell :

```
import sys
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.transforms import *
glueContext = GlueContext(SparkContext.getOrCreate())
persons_DyF = glueContext.create_dynamic_frame.from_catalog(database="legislators",
 table_name="persons_json")
print ("Count:  ", persons_DyF.count())
persons_DyF.printSchema()
```

Tutoriel : Configuration d'un PyCharm environnement professionnel avec un terminal de développement

Ce didacticiel explique comment connecter l'IDE Python PyCharm professionnel exécuté sur votre machine locale à un point de terminaison de développement afin de pouvoir exécuter, déboguer et tester des scripts AWS Glue ETL (extraction, transfert et chargement) de manière interactive avant de les déployer. Les instructions et les captures d'écran du didacticiel sont basées sur la version PyCharm professionnelle 2019.3.

Pour vous connecter à un point de développement de manière interactive, PyCharm Professional doit être installé sur votre ordinateur. Vous ne pouvez pas le faire avec la version gratuite.

> **Note**
>
> Le didacticiel utilise Amazon S3 comme source de données. Si vous souhaitez utiliser une source de données JDBC à la place, vous devez exécuter votre point de terminaison de développement dans un cloud privé virtuel (VPC). Pour vous connecter avec SSH à un point de terminaison de développement dans un VPC, vous devez créer un tunnel SSH. Ce didacticiel ne contient pas d'instructions pour la création d'un tunnel SSH. Pour plus d'informations sur l'utilisation de SSH pour se connecter à un point de terminaison de développement dans un VPC, consultez la section Connexion sécurisée aux instances Linux exécutées dans un Amazon VPC privé sur le blog de sécurité. AWS

Rubriques

- Connecter un PyCharm professionnel à un terminal de développement
- Déploiement du script sur votre point de terminaison de développement
- Configuration d'un interpréteur distant
- Exécution de votre script sur le point de terminaison de développement

Connecter un PyCharm professionnel à un terminal de développement

1. Créez un nouveau projet en Python pur dans named. PyCharm `legislators`
2. Créez un fichier nommé `get_person_schema.py` dans le projet avec le contenu suivant :

```
from pyspark.context import SparkContext
from awsglue.context import GlueContext


def main():
    # Create a Glue context
    glueContext = GlueContext(SparkContext.getOrCreate())

    # Create a DynamicFrame using the 'persons_json' table
    persons_DyF =
 glueContext.create_dynamic_frame.from_catalog(database="legislators",
 table_name="persons_json")
```

```
    # Print out information about this data
    print("Count:  ", persons_DyF.count())
    persons_DyF.printSchema()


if __name__ == "__main__":
    main()
```

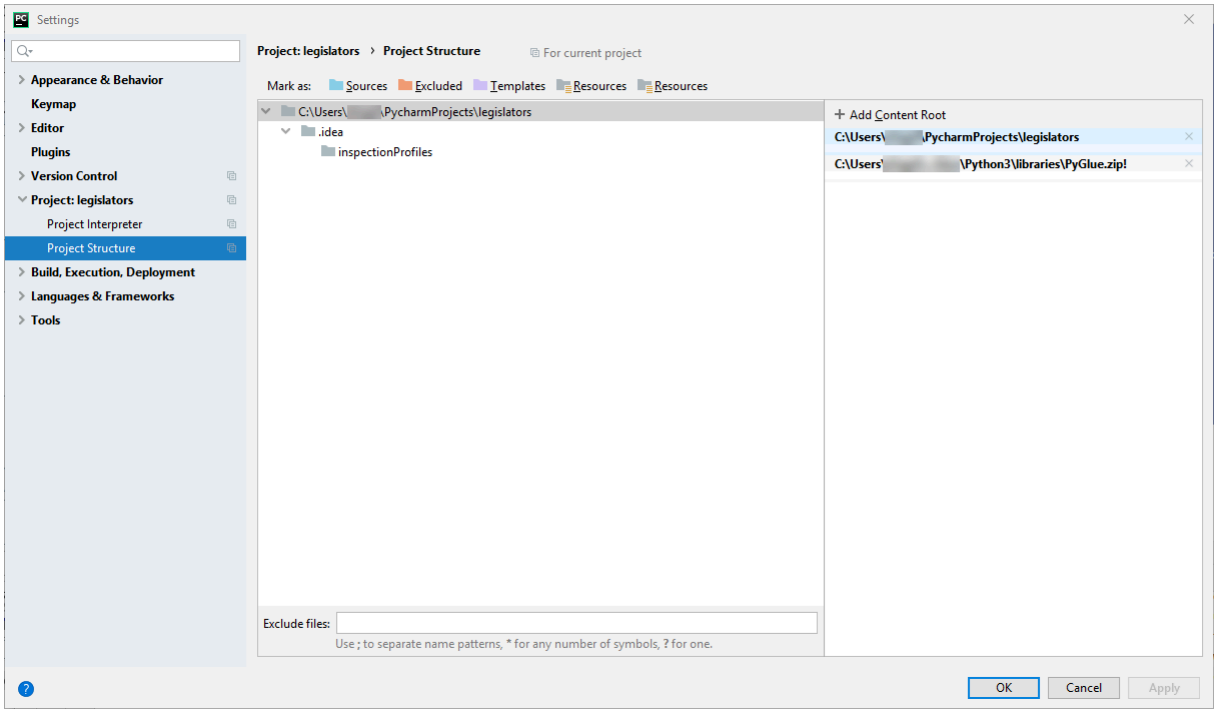
3. Effectuez l'une des actions suivantes :

   - Pour la version 0.9 de AWS Glue, téléchargez le fichier de bibliothèque Python AWS Glue, `PyGlue.zip` à l'adresse `https://s3.amazonaws.com/aws-glue-jes-prod-us-east-1-assets/etl/python/PyGlue.zip`, vers un emplacement approprié sur votre ordinateur local.
   - Pour les versions 1.0 et ultérieures de AWS Glue, téléchargez le fichier de bibliothèque Python AWS Glue, `PyGlue.zip` à l'adresse `https://s3.amazonaws.com/aws-glue-jes-prod-us-east-1-assets/etl-1.0/python/PyGlue.zip`, vers un emplacement approprié sur votre ordinateur local.

4. Ajoutez `PyGlue.zip` en tant que racine de contenu pour votre projet dans PyCharm :

   - Dans PyCharm, choisissez Fichier, Paramètres pour ouvrir la boîte de dialogue Paramètres. (Vous pouvez également appuyer sur `Ctrl+Alt+S`.)
   - Développez le projet `legislators` et choisissez Project Structure (Structure du projet). Ensuite, dans le panneau droit, sélectionnez + Add Content Root (+ Ajouter le contenu racine).
   - Accédez à l'emplacement où vous avez enregistré `PyGlue.zip`, sélectionnez-le, puis choisissez Apply (Appliquer).

   L'écran Settings (Paramètres) doit ressembler à ceci :



Laissez la boîte de dialogue Settings (Paramètres) ouverte après avoir choisi Apply (Appliquer).

5. Configurez les options de déploiement pour télécharger le script local sur votre terminal de développement à l'aide du protocole SFTP (cette fonctionnalité n'est disponible que dans PyCharm Professional) :

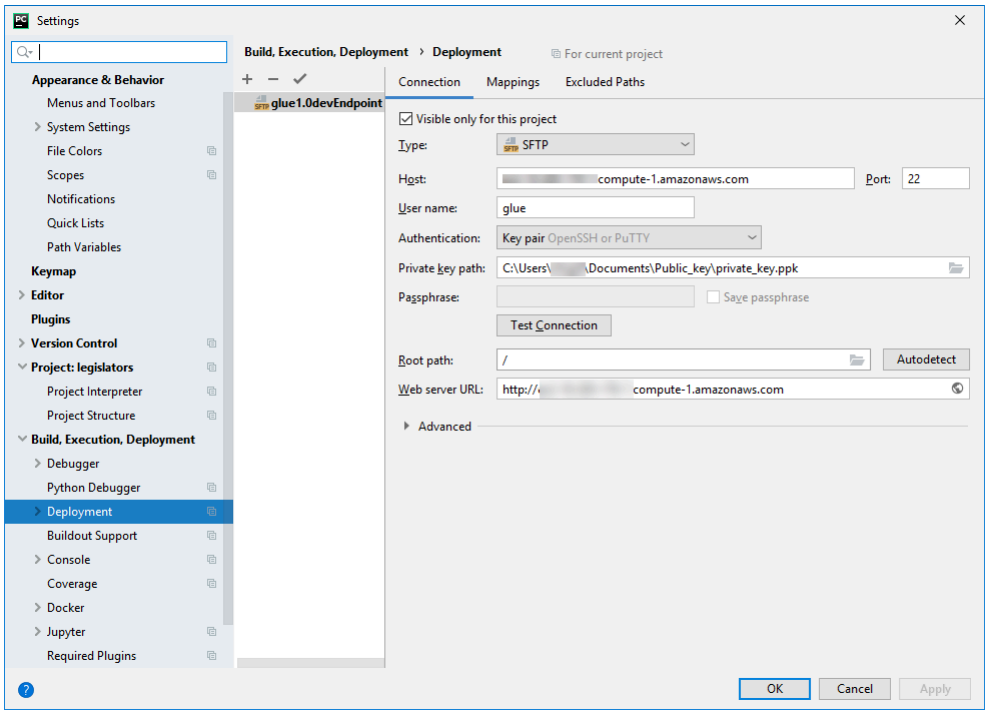
   - Dans la boîte de dialogue Settings (Paramètres), développez la section Build, Execution, Deployment (Création, exécution, déploiement). Choisissez la sous-section Deployment (Déploiement).
   - Sélectionnez l'icône + en haut du panneau central pour ajouter un nouveau serveur. Définissez Type sur `SFTP` et donnez-lui un nom.
   - Définissez SFTP host (Hôte SFTP) sur l'adresse publique de votre point de terminaison de développement, comme indiqué sur sa page de détails. (Choisissez le nom de votre point de terminaison de développement dans la console AWS Glue pour afficher la page de détails). Pour un point de terminaison de développement s'exécutant dans un VPC, définissez SFTP host (Hôte SFTP) sur l'adresse hôte et le port local de votre tunnel SSH sur le point de terminaison de développement.
   - Définissez le champ User name (Nom d'utilisateur) sur `glue`.
   - Définissez le champ Auth type (Type d'autorisation) sur Key pair (OpenSSH or Putty) (Paire de clés (OpenSSH ou Putty)). Définissez le champ Private key file (Fichier de clé privée) en accédant à l'emplacement du fichier de clé privée de votre point de terminaison de

développement. Notez que PyCharm seuls les types de clés DSA, RSA et ECDSA OpenSSH sont pris en charge et que les clés au format privé de Putty ne sont pas acceptées. Vous pouvez utiliser une up-to-date version de `ssh-keygen` pour générer un type de paire de clés qui PyCharm accepte, en utilisant une syntaxe similaire à la suivante :

```
ssh-keygen -t rsa -f <key_file_name> -C "<your_email_address>"
```

- Choisissez Test connection (Tester la connexion) et autorisez le test de la connexion. Si la connexion est établie, choisissez Apply (Appliquer).

L'écran Settings (Paramètres) doit désormais ressembler à ceci :



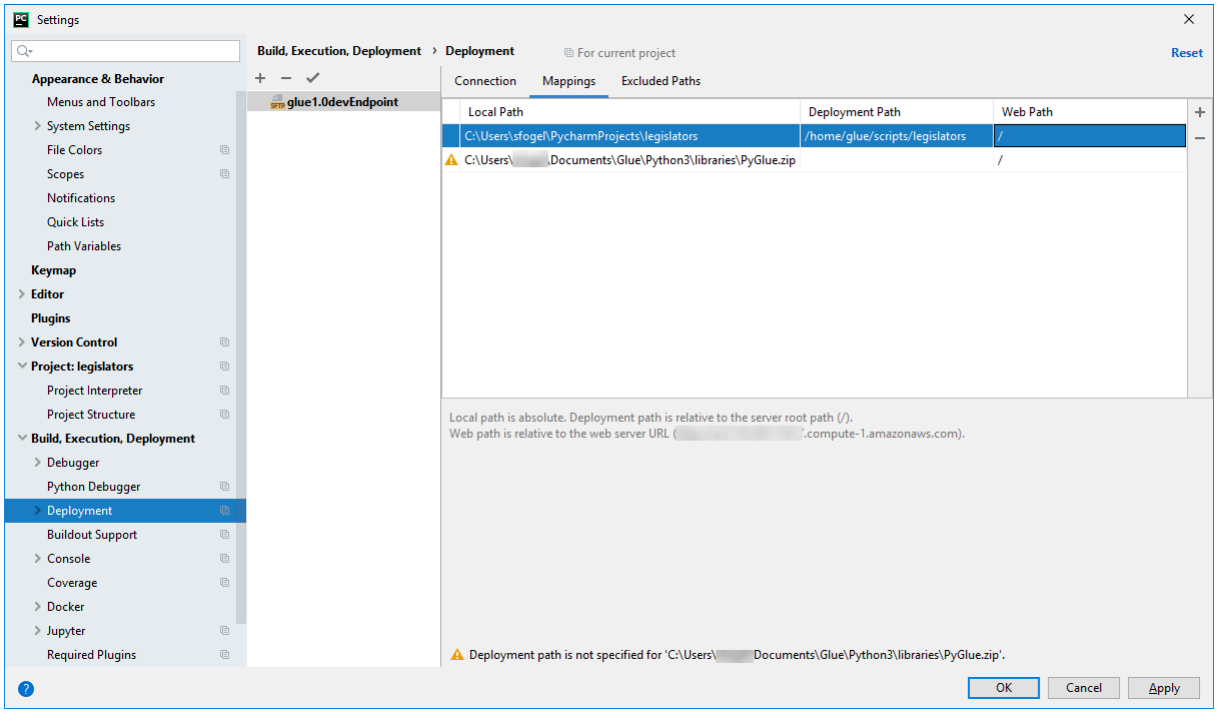
Laissez à nouveau la boîte de dialogue Settings (Paramètres) ouverte après avoir choisi Apply (Appliquer).

6. Mappez le répertoire local à un répertoire distant pour le déploiement :

   - Dans le panneau de droite de la page Deployment (Déploiement), sélectionnez l'onglet central en haut, intitulé Mappings (Mappages).
   - Dans la colonne Deployment Path (Chemin d'accès du déploiement), saisissez un chemin sous `/home/glue/scripts/` pour le déploiement du chemin de votre projet. Par exemple : `/home/glue/scripts/legislators`.

- Cliquez sur Appliquer.

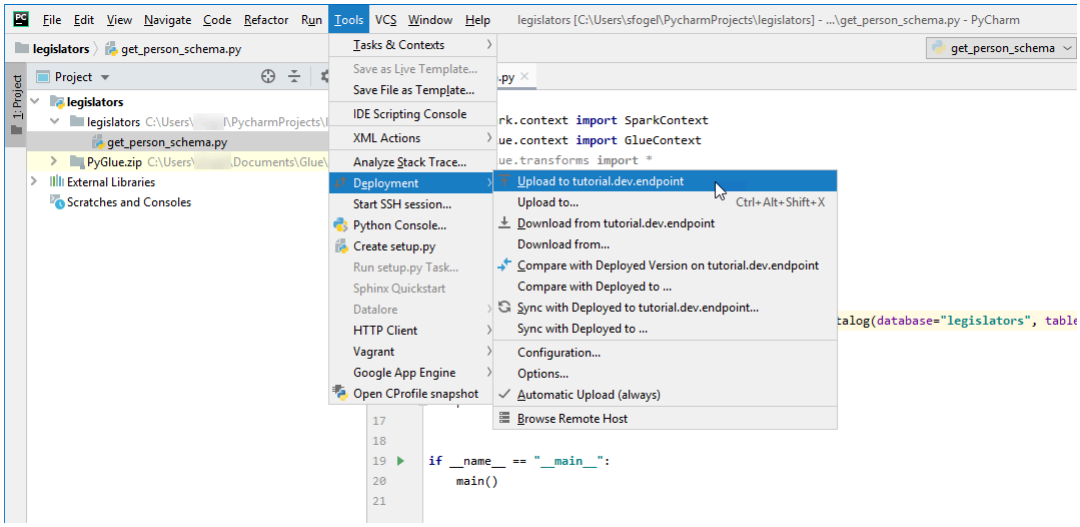
L'écran Settings (Paramètres) doit désormais ressembler à ceci :



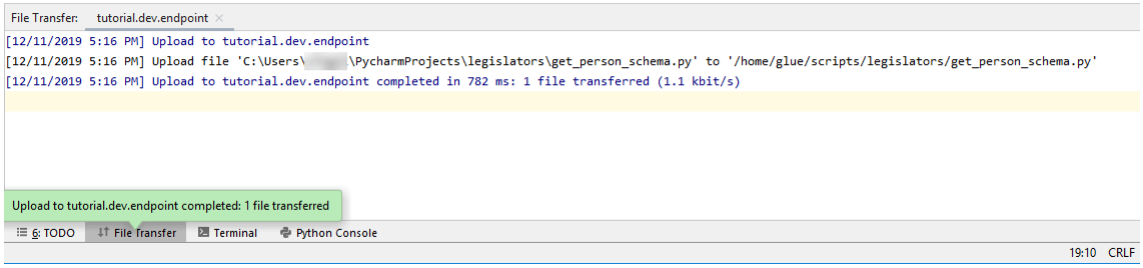
Choisissez OK pour fermer la boîte de dialogue Settings (Paramètres).

Déploiement du script sur votre point de terminaison de développement

1. Choisissez Tools (Outils), Deployment (Déploiement), puis choisissez le nom sous lequel vous configurez votre point de terminaison de développement, comme illustré dans l'image suivante :



Une fois que votre script a été déployé, le bas de l'écran doit ressembler à ce qui suit :



2. Dans la barre de menus, choisissez Tools (Outils), Deployment (Déploiement), Automatic Upload (always) [Téléchargement automatique (toujours)]. Assurez-vous qu'une coche s'affiche en regard de Automatic Upload (always) [Téléchargement automatique (toujours)].

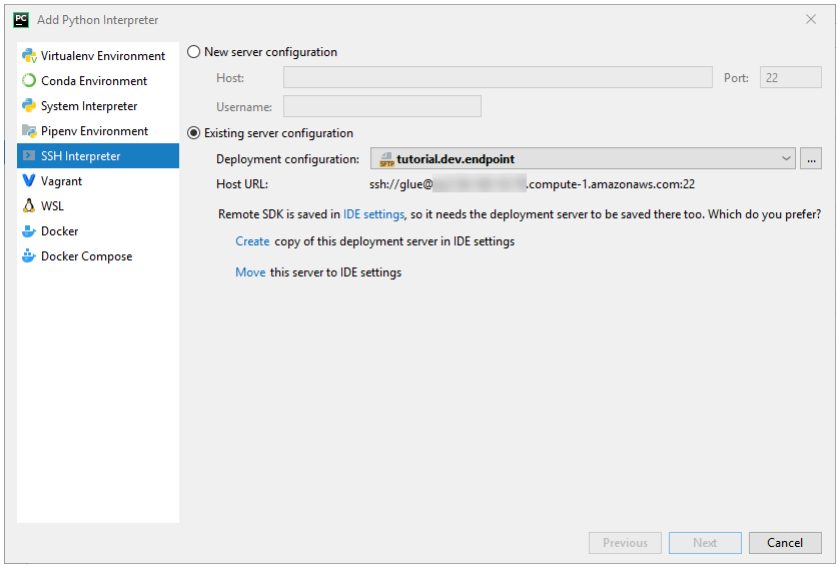
   Lorsque cette option est activée, les fichiers modifiés PyCharm sont automatiquement téléchargés sur le terminal de développement.

Configuration d'un interpréteur distant

Configurez PyCharm pour utiliser l'interpréteur Python sur le point de terminaison de développement.

1. Dans le menu File (Fichier), choisissez Settings (Paramètres).
2. Développez les législateurs du projet et choisissez Project Interpreter (Interpréteur de projet).
3. Choisissez l'icône représentant un engrenage en regard de la liste Project Interpreter (Interpréteur de projet), puis choisissez Add (Ajouter).
4. Dans la boîte de dialogue Add Python Interpreter (Ajouter un interpréteur Python) dans le panneau gauche, sélectionnez SSH Interpreter (Interpréteur SSH).
5. Choisissez Existing server configuration (Configuration du serveur existant), et, dans la liste Deployment configuration (Configuration du déploiement), choisissez votre configuration.

   Votre écran doit ressembler à l'image suivante.



6. Choisissez Move this server to IDE settings (Déplacer ce serveur dans les paramètres IDE, puis choisissez Next (Suivant).
7. Dans le champ Interpreter (Interpréteur) modifiez le chemin d'accès en `/usr/bin/gluepython` si vous utilisez Python 2, ou en `/usr/bin/gluepython3` si vous utilisez Python 3. Choisissez ensuite Finish (Terminer).

Exécution de votre script sur le point de terminaison de développement

Pour exécuter le script :

- Dans le volet de gauche, cliquez avec le bouton droit sur le nom du fichier et choisissez Exécuter ***<filename>*** « ».

  Après une série de messages, la sortie finale doit afficher le nombre et le schéma.

```
Count:  1961
root
|-- family_name: string
|-- name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- gender: string
|-- image: string
|-- identifiers: array
|    |-- element: struct
```

```
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- lang: string
|    |    |-- note: string
|    |    |-- name: string
|-- sort_name: string
|-- images: array
|    |-- element: struct
|    |    |-- url: string
|-- given_name: string
|-- birth_date: string
|-- id: string
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
|-- death_date: string


Process finished with exit code 0
```

Vous êtes maintenant en mesure de déboguer votre script à distance sur votre point de terminaison de développement.

Configuration avancée : partage de points de terminaison de développement entre plusieurs utilisateurs

Cette section explique comment vous pouvez tirer parti des terminaux de développement à l'aide de SageMaker blocs-notes dans des cas d'utilisation typiques pour partager des points de développement entre plusieurs utilisateurs.

Configuration à location unique

Dans les cas d'utilisation à locataire unique, pour simplifier l'expérience du développeur et éviter les conflits de ressources, il est recommandé que chaque développeur utilise son propre point de terminaison de développement dimensionné pour le projet sur lequel il travaille. Cela simplifie également les décisions liées au type de travailleur et au nombre de DPU, les laissant à la discrétion du développeur et du projet sur lequel ils travaillent.

Vous n'aurez pas besoin de vous occuper de l'allocation des ressources, sauf si vous exécutez plusieurs fichiers de bloc-notes simultanément. Si vous exécutez du code dans plusieurs fichiers de bloc-notes simultanément, plusieurs séances Livy seront lancées simultanément. Pour séparer les configurations de cluster Spark afin d'exécuter plusieurs séances Livy simultanément, vous pouvez suivre les étapes qui sont introduites dans les cas d'utilisation multi-locataire.

Par exemple, si votre point de terminaison de développement a 10 processus et que le type de processus est `G.1X`, alors vous aurez 9 programmes d'exécution Spark et l'ensemble du cluster aura 90 Go de mémoire d'exécuteur puisque chaque exécuteur aura 10 Go de mémoire.

Quel que soit le type de nœud de processus spécifié, l'allocation dynamique des ressources Spark sera activée. Si un jeu de données est suffisamment volumineux, Spark peut allouer tous les programmes d'exécution à une seule séance Livy, car `spark.dynamicAllocation.maxExecutors` n'est pas défini par défaut. Cela signifie que d'autres séances Livy sur le même point de terminaison de développement attendront le lancement de nouveaux programmes d'exécution. Si le jeu de données est peu volumineux, Spark pourra allouer des programmes d'exécution à plusieurs séances Livy en même temps.

> **Note**
>
> Pour plus d'informations sur la façon dont les ressources sont allouées dans différents cas d'utilisation et sur la façon dont vous définissez une configuration pour modifier le comportement, consultez Configuration avancée : partage de points de terminaison de développement entre plusieurs utilisateurs.

Configuration multilocataire

> **Note**
>
> Veuillez noter que les points de terminaison de développement sont destinés à émuler l'environnement ETL AWS Glue en tant qu'environnement à utilisateur unique. Bien que l'utilisation multi-locataire soit possible, il s'agit d'un cas d'utilisation avancé et il est recommandé à la plupart des utilisateurs de maintenir un modèle de locataire unique pour chaque point de terminaison de développement.

Dans les cas d'utilisation multilocataires, vous devrez peut-être vous occuper de l'allocation des ressources. Le facteur clé est le nombre d'utilisateurs qui utilisent un bloc-notes Jupyter

simultanément. Si votre équipe travaille dans un flux de travail follow-the-sun « » et qu'il n'y a qu'un seul utilisateur Jupyter par fuseau horaire, le nombre d'utilisateurs simultanés est d'un seul. Vous n'aurez donc pas à vous soucier de l'allocation des ressources. Cependant, si votre bloc-notes est partagé entre plusieurs utilisateurs et que chaque utilisateur soumet du code de manière ad hoc, vous devrez alors prendre en compte les points ci-dessous.

Pour partitionner les ressources du cluster Spark entre plusieurs utilisateurs, vous pouvez utiliser SparkMagic des configurations. Il existe deux manières différentes de configurer SparkMagic.

(A) Utiliser la directive %%configure -f

Si vous souhaitez modifier la configuration par séance Livy à partir du bloc-notes, vous pouvez exécuter la directive `%%configure -f` sur le paragraphe du bloc-notes.

Par exemple, si vous souhaitez exécuter l'application Spark sur 5 programmes d'exécution, vous pouvez exécuter la commande suivante sur le paragraphe du bloc-notes.

```
%%configure -f
{"numExecutors":5}
```

Ensuite, vous ne verrez que 5 programmes d'exécution en cours d'exécution pour le travail sur l'interface utilisateur Spark.

Nous vous recommandons de limiter le nombre maximum d'exécuteurs pour l'allocation dynamique des ressources.

```
%%configure -f
{"conf":{"spark.dynamicAllocation.maxExecutors":"5"}}
```

(B) Modifier le fichier de SparkMagic configuration

SparkMagic fonctionne sur la base de l'API Livy. SparkMagic crée des sessions Livy avec des configurations telles que`driverMemory`, `driverCores`,`executorMemory`,`executorCores`,, `numExecutors`,`conf`, etc. Ce sont les facteurs clés qui déterminent la quantité de ressources consommées par l'ensemble du cluster Spark. SparkMagic vous permet de fournir un fichier de configuration pour spécifier les paramètres qui sont envoyés à Livy. Pour voir un exemple de fichier de configuration, consultez ce référentiel Github.

Si vous souhaitez modifier la configuration de toutes les séances Livy à partir d'un bloc-notes, vous pouvez modifier `/home/ec2-user/.sparkmagic/config.json` pour ajouter `session_config`.

Pour modifier le fichier de configuration sur une instance de SageMaker bloc-notes, vous pouvez suivre ces étapes.

1. Ouvrez un SageMaker bloc-notes.
2. Ouvrez le noyau Terminal.
3. Exécutez les commandes suivantes :

```
sh-4.2$ cd .sparkmagic
sh-4.2$ ls
config.json logs
sh-4.2$ sudo vim config.json
```

Par exemple, vous pouvez ajouter ces lignes à `/home/ec2-user/.sparkmagic/config.json` et redémarrer le noyau Jupyter à partir du bloc-notes.

```
  "session_configs": {
    "conf": {
      "spark.dynamicAllocation.maxExecutors":"5"
    }
  },
```

Consignes et bonnes pratiques

Pour éviter ce type de conflit de ressources, vous pouvez utiliser des approches de base telles que :

- Accroître la taille d'un cluster Spark en augmentant `NumberOfWorkers` (mise à l'échelle horizontale) et en mettant à niveau `workerType` (mise à l'échelle verticale)
- Allouer moins de ressources par utilisateur (moins de ressources par séance Livy)

Votre approche dépendra de votre cas d'utilisation. Si vous avez un point de terminaison de développement plus important et qu'il n'y a pas une énorme quantité de données, la possibilité d'un conflit de ressources diminuera considérablement, car Spark peut allouer des ressources en fonction d'une stratégie d'allocation dynamique.

Comme décrit ci-dessus, le nombre de programmes d'exécution Spark peut être calculé automatiquement en fonction d'une combinaison de DPU (ou `NumberOfWorkers`) et de type de processus. Chaque application Spark lance un pilote et plusieurs programmes d'exécution. Pour effectuer un calcul, `NumberOfWorkers` devra être égal à `NumberOfExecutors + 1`. La

matrice ci-dessous explique la capacité dont vous avez besoin dans votre point de terminaison de développement en fonction du nombre d'utilisateurs simultanés.

Nombre d'utilisateurs simultanés de blocs-notes	Nombre d'exécuteurs Spark que vous souhaitez allouer par utilisateur	Total NumberOfWorkers pour votre point de terminaison de développement
3	5	18
10	5	60
50	5	300

Si vous souhaitez allouer moins de ressources par utilisateur, `spark.dynamicAllocation.maxExecutors` (ou `numExecutors`) serait le paramètre le plus simple à configurer en tant que paramètre de séance Livy. Si vous définissez la configuration ci-dessous`/home/ec2-user/.sparkmagic/config.json`, vous SparkMagic attribuerez un maximum de 5 exécuteurs par session Livy. Cela aidera à répartir les ressources par séance Livy.

```
"session_configs": {
    "conf": {
      "spark.dynamicAllocation.maxExecutors":"5"
    }
  },
```

Supposons qu'il y ait un point de terminaison de développement avec 18 processus (G.1X) et qu'il y ait 3 utilisateurs de blocs-notes simultanément. Si votre configuration de séance dispose de `spark.dynamicAllocation.maxExecutors=5`, chaque utilisateur peut utiliser 1 pilote et 5 programmes d'exécution. Il n'y aura pas de conflits de ressources même si vous exécutez plusieurs paragraphes de bloc-notes en même temps.

Compromis

Avec cette configuration de séance `"spark.dynamicAllocation.maxExecutors":"5"`, vous pourrez éviter les erreurs de conflit de ressources et vous n'aurez pas besoin d'attendre l'allocation des ressources lorsqu'il y a des accès utilisateurs simultanés. Cependant, même lorsqu'il existe de nombreuses ressources gratuites (par exemple, il n'y a pas d'autres utilisateurs simultanés), Spark ne peut pas affecter plus de 5 programmes d'exécution pour votre séance Livy.

Autres remarques

Il est recommandé d'arrêter le noyau Jupyter lorsque vous arrêtez d'utiliser un bloc-notes. Cela libérera des ressources et les autres utilisateurs de blocs-notes pourront utiliser ces ressources immédiatement sans attendre l'expiration du noyau (arrêt automatique).

Problèmes courants

Même en suivant les directives, vous pouvez rencontrer certains problèmes.

Séance introuvable

Lorsque vous essayez d'exécuter un paragraphe de bloc-notes même si votre séance Livy est déjà résiliée, vous verrez le message ci-dessous. Pour activer la séance Livy, vous devez redémarrer le noyau Jupyter en sélectionnant Kernel (Noyau) > Restart (Redémarrer) dans le menu Jupyter, puis exécuter à nouveau le paragraphe du bloc-notes.

```
An error was encountered:
Invalid status code '404' from http://localhost:8998/sessions/13 with error payload:
 "Session '13' not found."
```

Nombre de ressources YARN insuffisant

Lorsque vous essayez d'exécuter un paragraphe de bloc-notes même si votre cluster Spark ne dispose pas de suffisamment de ressources pour démarrer une nouvelle séance Livy, le message ci-dessous s'affiche. Vous pouvez souvent éviter ce problème en suivant les instructions. Cependant, il est possible que vous soyez confronté à ce problème. Pour contourner le problème, vous pouvez vérifier s'il existe des séances Livy actives inutiles. S'il existe des séances Livy inutiles, vous devrez les résilier pour libérer les ressources du cluster. Pour en savoir plus, consultez la section suivante.

```
Warning: The Spark session does not have enough YARN resources to start.
The code failed because of a fatal error:
    Session 16 did not start up in 60 seconds..

Some things to try:
a) Make sure Spark has enough available resources for Jupyter to create a Spark
 context.
b) Contact your Jupyter administrator to make sure the Spark magics library is
 configured correctly.
c) Restart the kernel.
```

Surveillance et débogage

Cette section décrit les techniques de surveillance des ressources et des séances.

Surveillance et débogage de l'allocation des ressources du cluster

Vous pouvez consulter l'interface utilisateur de Spark pour contrôler le nombre de ressources allouées par séance Livy et quelles sont les configurations Spark efficaces sur la tâche. Pour activer l'interface utilisateur de Spark, consultez Activation de l'interface utilisateur web Apache Spark pour les points de terminaison de développement.

(Facultatif) Si vous avez besoin d'une vue en temps réel de l'interface utilisateur Spark, vous pouvez configurer un tunnel SSH sur le serveur d'historique Spark exécuté sur le cluster Spark.

```
ssh -i <private-key.pem> -N -L 8157:<development endpoint public address>:18080
 glue@<development endpoint public address>
```

Vous pouvez alors ouvrir la page http://localhost:8157 dans votre navigateur pour afficher localement l'interface utilisateur Spark.

Séances Livy gratuites inutiles

Passez en revue ces procédures pour arrêter toutes les séances Livy inutiles à partir d'un bloc-notes ou d'un cluster Spark.

(a). Résilier les séances Livy à partir d'un bloc-notes

Vous pouvez arrêter le noyau sur un bloc-notes Jupyter pour résilier aux séances Livy inutiles.

(b). Résilier les séances Livy à partir d'un cluster Spark

Si des séances Livy inutiles sont toujours en cours d'exécution, vous pouvez les arrêter sur le cluster Spark.

Avant d'effectuer cette procédure, vous devez configurer votre clé publique SSH pour votre point de terminaison de développement.

Pour vous connecter au cluster Spark, vous pouvez exécuter la commande suivante :

```
$ ssh -i <private-key.pem> glue@<development endpoint public address>
```

Vous pouvez exécuter la commande suivante pour voir les séances Livy actives :

```
$ yarn application -list
20/09/25 06:22:21 INFO client.RMProxy: Connecting to ResourceManager at
 ip-255-1-106-206.ec2.internal/172.38.106.206:8032
Total number of applications (application-types: [] and states: [SUBMITTED, ACCEPTED,
 RUNNING]):2
Application-Id Application-Name Application-Type User Queue State Final-State Progress
 Tracking-URL
application_1601003432160_0005 livy-session-4 SPARK livy default RUNNING UNDEFINED 10%
 http://ip-255-1-4-130.ec2.internal:41867
application_1601003432160_0004 livy-session-3 SPARK livy default RUNNING UNDEFINED 10%
 http://ip-255-1-179-185.ec2.internal:33727
```

Vous pouvez ensuite fermer la séance Livy avec la commande suivante :

```
$ yarn application -kill application_1601003432160_0005
20/09/25 06:23:38 INFO client.RMProxy: Connecting to ResourceManager at
 ip-255-1-106-206.ec2.internal/255.1.106.206:8032
Killing application application_1601003432160_0005
20/09/25 06:23:39 INFO impl.YarnClientImpl: Killed application
 application_1601003432160_0005
```

Gestion des blocs-notes

> **Note**
>
> Les points de terminaison de développement ne sont pris en charge que pour les versions AWS Glue antérieures à 2.0. Pour un environnement interactif dans lequel vous pouvez créer et tester des scripts ETL, utilisez Notebooks on AWS Glue Studio.

Un bloc-notes permet de développer et de tester de manière interactive vos scripts ETL (extraction, transformation et chargement) sur un terminal de développement. AWS Gluefournit une interface pour les ordinateurs SageMaker portables AI Jupyter. AvecAWS Glue, vous créez et gérez des blocs-notes SageMaker dotés d'intelligence artificielle. Vous pouvez également ouvrir des blocs-notes SageMaker AI depuis la AWS Glue console.

En outre, vous pouvez utiliser Apache Spark avec l' SageMaker IA sur les terminaux de AWS Glue développement qui prennent en charge l' SageMaker IA (mais pas les tâches AWS Glue ETL).

SageMaker Spark est une bibliothèque Apache Spark open source pour l' SageMaker IA. Pour plus d'informations, consultez la section Utilisation d'Apache Spark avec Amazon SageMaker.

Important

La gestion des blocs-notes SageMaker IA dotés de points de terminaison de AWS Glue développement est disponible dans les régions suivantes : AWS

Région	Code
USA Est (Ohio)	`us-east-2`
USA Est (Virginie du Nord)	`us-east-1`
USA Ouest (Californie du Nord)	`us-west-1`
USA Ouest (Oregon)	`us-west-2`
Asie Pacifique (Tokyo)	`ap-northeast-1`
Asie-Pacifique (Séoul)	`ap-northeast-2`
Asie-Pacifique (Mumbai)	`ap-south-1`
Asie-Pacifique (Singapour)	`ap-southeast-1`
Asie-Pacifique (Sydney)	`ap-southeast-2`
Canada (Centre)	`ca-central-1`
Europe (Francfort)	`eu-central-1`
Europe (Irlande)	`eu-west-1`
Europe (Londres)	`eu-west-2`

Création de tâches ETL visuelles

Créez des tâches ETL visuelles avec AWS Glue Studio

AWS Glue Studio fournit une interface visuelle pour créer, exécuter et surveiller des tâches Extract/Transform/Load (ETL) dans AWS Glue. Une tâche AWS Glue comprend la logique métier qui exécute les tâches d'extraction, de transformation et de chargement (ETL). Vous pouvez ainsi composer visuellement des flux de travail de transformation des données et les exécuter de manière fluide sur le moteur AWS Glue ETL sans serveur basé sur Apache Spark. AWS Glue Studio Vous pouvez créer des tâches qui déplacent et transforment des données entre différents magasins de données et flux à l'aide d'une drag-and-drop interface sans avoir à apprendre à utiliser Spark ou à écrire du code.

Une tâche AWS Glue encapsule un script qui se connecte à vos données source, les traite, puis les écrit dans votre cible de données. En général, une tâche exécute les scripts d'extraction, de transformation et de chargement (ETL). Les tâches peuvent exécuter des scripts conçus pour les environnements d'exécution Apache Spark et Ray. Les tâches peuvent également exécuter des scripts Python à usage général (tâches shell Python). Les déclencheurs AWS Glue peuvent démarrer des tâches en fonction d'une planification, d'un événement ou à la demande. Vous pouvez surveiller les exécutions de tâche pour comprendre les métriques d'exécution telles que le statut d'achèvement, la durée et l'heure de début.

Vous pouvez utiliser des scripts générés par AWS Glue ou fournir les vôtres. Avec un schéma source et un emplacement ou un schéma cible, le générateur de AWS Glue Studio code peut créer automatiquement un script d'API Apache Spark (PySpark). Vous pouvez utiliser ce script comme point de départ et le modifier en fonction de vos objectifs.

AWS Glue peut écrire des fichiers de sortie dans plusieurs formats de données. Chaque type de tâche peut prendre en charge différents formats de sortie. Pour certains formats de données, des formats de compression courants peuvent être écrits.

Gestion des AWS Glue tâches dans la AWS console

Pour consulter les tâches existantes, connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/. Ensuite, choisissez l'onglet Tâches dans AWS Glue. La liste Jobs (Tâches) affiche l'emplacement du script associé à chaque tâche, quand la tâche a été modifiée pour la dernière fois et l'option de signet de la tâche actuelle.

Vous pouvez créer des tâches dans la section ETL de la console AWS Glue. Lors de la création d'une tâche, ou après avoir l'avoir enregistrée, vous pouvez utiliser AWS Glue Studio pour modifier vos tâches ETL. Vous pouvez le faire en éditant les nœuds dans l'éditeur visuel ou en modifiant le script de la tâche en mode développeur. Vous pouvez également ajouter et supprimer des nœuds dans l'éditeur visuel pour créer des tâches ETL plus compliquées.

Prochaines étapes de création d'une tâche dans AWS Glue Studio

Vous utilisez l'éditeur de tâches visuelles pour configurer les nœuds de votre tâche. Chaque nœud représente une action, telle que la lecture de données à partir de l'emplacement source ou l'application d'une transformation aux données. Chaque nœud que vous ajoutez à votre tâche possède des propriétés qui fournissent des informations sur l'emplacement des données ou la transformation.

Voici les étapes suivantes pour créer et gérer vos tâches :

- Démarrage de tâches ETL visuelles dans AWS Glue Studio
- Afficher le script de tâche
- Modifier les propriétés de tâche
- Sauvegarder la tâche
- Démarrer une exécution de tâche
- Afficher les informations sur les exécutions de tâche récentes
- Accès au tableau de bord de surveillance de tâche

Créez des flux ETL visuels avec Amazon SageMaker

Avec un flux de travail Amazon SageMaker Unified Studio, vous pouvez configurer et exécuter une série de tâches dans Amazon SageMaker Unified Studio. Les flux de travail Amazon SageMaker Unified Studio utilisent Apache Airflow pour modéliser les procédures de traitement des données et orchestrer vos artefacts de code Amazon SageMaker Unified Studio. Pour plus d'informations, consultez Utilisation des flux de travail dans Amazon SageMaker Unified Studio.

Démarrage de tâches ETL visuelles dans AWS Glue Studio

Vous pouvez utiliser l'interface visuelle simple dans AWS Glue Studio Glue Studio pour créer vos tâches ETL. Vous utilisez la page Tâches pour créer des tâches. Vous pouvez également utiliser

un éditeur de script pour travailler directement avec du code dans le script de tâche ETL AWS Glue Studio Glue Studio.

Sur la page Tâches, vous pouvez voir toutes les tâches que vous avez créées soit avec AWS Glue Studio Glue Studio, soit avec AWS Glue. Vous pouvez afficher, gérer et exécuter vos tâches sur cette page.

Consultez également le didacticiel de blog pour obtenir un autre exemple de création de tâches ETL avec AWS Glue Studio.

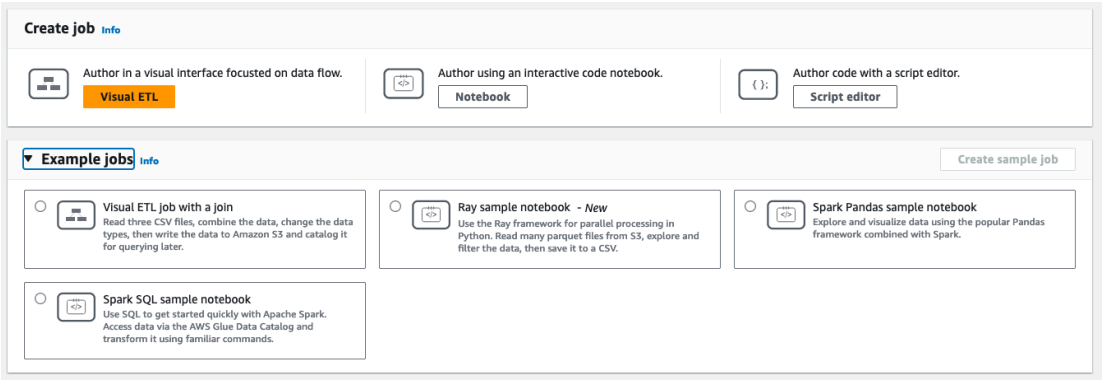
Démarrer une tâche dans AWS Glue Studio

AWS Glue vous permet de créer une tâche par le biais d'une interface visuelle, d'un bloc-notes de code interactif ou d'un éditeur de script. Vous pouvez démarrer une tâche en cliquant sur l'une des options ou créer une nouvelle tâche à partir d'un exemple de tâche.

Les exemples de tâches créent une tâche à l'aide de l'outil de votre choix. Par exemple, les exemples de tâches vous permettent de créer une tâche ETL visuelle qui joint des fichiers CSV dans une table de catalogue, de créer une tâche dans un bloc-notes de code interactif avec AWS Glue pour Ray ou AWS Glue pour Spark lorsque vous travaillez avec Pandas, ou de créer une tâche dans un bloc-notes de code interactif avec SparkSQL.

Création d'une tâche dans AWS Glue Studio à partir de zéro

1. Connectez-vous à la AWS Glue Studio console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/gluestudio/.
2. Dans le panneau de navigation, choisissez Tâches ETL.
3. Dans la section Créer une tâche, sélectionnez une option de configuration pour votre tâche.



Options pour créer une tâche à partir de zéro :

- ETL visuelle : créez une interface visuelle axée sur le flux de données
- Créer à l'aide d'un bloc-notes de code interactif : créez des tâches de manière interactive dans une interface de bloc-notes basée sur les blocs-notes Jupyter.

  Lorsque vous sélectionnez cette option, vous devez fournir des informations supplémentaires avant de créer une session de création de blocs-notes. Pour plus d'informations sur la manière de spécifier ces informations, veuillez consulter Mise en route avec les blocs-notes dans AWS Glue Studio.
- Créer du code avec un éditeur de script : pour ceux qui connaissent la programmation et l'écriture de scripts ETL, vous pouvez choisir cette option pour créer une nouvelle tâche ETL Spark. Choisissez le moteur (shell Python, Ray, Spark [Python] ou Spark [Scala]). Choisissez ensuite Redémarrer ou Charger un script pour charger un script existant à partir d'un fichier local. Si vous choisissez d'utiliser l'éditeur de script, vous ne pouvez pas utiliser l'éditeur de tâches visuel.

  Une tâche Spark est exécutée dans un environnement Apache Spark géré par AWS Glue. Par défaut, les nouveaux scripts sont codés en Python. Pour écrire un nouveau script Scala, veuillez consulter Création et modification de scripts Scala dans AWS Glue Studio.

Création d'une tâche dans AWS Glue Studio à partir d'un exemple de tâche

Vous pouvez choisir de créer une tâche à partir d'un exemple de tâche. Dans la section Exemples de tâches, choisissez un exemple de tâche, puis choisissez Créer un exemple de tâche. La création d'un exemple de tâche à partir de l'une des options fournit un modèle rapide à partir duquel vous pouvez travailler.

1. Connectez-vous à la AWS Glue Studio console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/gluestudio/.
2. Dans le panneau de navigation, choisissez Tâches ETL.
3. Sélectionnez une option pour créer une tâche à partir d'un exemple de tâche :

   - Tâche ETL visuelle pour de joindre plusieurs sources : lisez trois fichiers CSV, combinez les données, modifiez les types de données, puis écrivez les données dans Amazon S3 et cataloguez-les pour les interroger ultérieurement.
   - Bloc-notes Spark utilisant Pandas : explorez et visualisez les données à l'aide du célèbre cadre Pandas combiné à Spark.

- Bloc-notes Spark utilisant SQL : utilisez SQL pour démarrer rapidement avec Apache Spark. Accédez aux données via AWS Glue Data Catalogue données et transformez-le à l'aide de commandes familières.

4. Choisissez Créer un exemple de tâche.

Fonctionnalités de l'éditeur de tâche

L'éditeur de tâches fournit les fonctionnalités suivantes pour la création et la modification des tâches.

- Un diagramme visuel de votre tâche, avec un nœud pour chaque tâche : nœuds de source de données pour la lecture des données ; nœuds de transformation pour la modification des données ; nœuds de données cibles pour l'écriture des données.

  Vous pouvez afficher et configurer les propriétés de chaque nœud dans le diagramme de tâches. Vous pouvez également afficher le schéma et les échantillons de données pour chaque nœud dans le diagramme de tâche. Ces fonctionnalités vous aident à vérifier que votre tâche modifie et transforme les données de la bonne manière, sans avoir à exécuter la tâche.
- Un onglet d'affichage et d'édition de script, dans lequel vous pouvez modifier le code généré pour votre tâche.
- Un onglet Détails de la tâche, dans lequel vous pouvez configurer une variété de paramètres pour personnaliser l'environnement dans lequel votre tâche ETL AWS Glue s'exécute.
- Un onglet Exécutions, dans lequel vous pouvez afficher les exécutions actuelles et précédentes de la tâche, afficher l'état de l'exécution de celle-ci et accéder à ses journaux d'exécution.
- Un onglet Qualité des données, dans lequel vous pouvez appliquer des règles de qualité des données à votre tâche.
- Un onglet Planifications, dans lequel vous pouvez configurer l'heure de début de votre tâche ou configurer une exécution de tâche récurrente.
- Un onglet Contrôle de version, dans lequel vous pouvez configurer un service Git à utiliser avec votre tâche.

Utilisation des prévisualisations de schéma dans l'éditeur de tâches visuel

Lorsque vous créez ou modifiez votre tâche, vous pouvez utiliser l'onglet Output Schema (Schéma de sortie) pour afficher le schéma de vos données.

Avant de voir le schéma, l'éditeur de tâche a besoin d'autorisations pour accéder à la source de données. Vous pouvez spécifier un rôle IAM dans l'onglet Détails de la tâche de l'éditeur ou dans l'onglet Output Schema (Schéma de sortie) pour chaque nœud. Si le rôle IAM dispose de toutes les autorisations nécessaires pour accéder à la source de données, vous pouvez alors afficher le schéma dans l'onglet Schema pour chaque nœud.

Utilisation des prévisualisations de données dans l'éditeur de tâches visuel

Les prévisualisations des données vous aident à créer et à tester votre tâche à l'aide d'un échantillon de vos données, sans avoir à exécuter la tâche de manière répétée. En utilisant la prévisualisation des données, vous pouvez :

- Tester un rôle IAM pour vous assurer que vous avez accès à vos sources de données ou cibles de données.
- Vérifier que la transformation modifie les données de la manière prévue. Par exemple, si vous utilisez une transformation de filtre, vous pouvez vous assurer que le filtre sélectionne le sous-ensemble de données approprié.
- Vérifiez vos données. Si votre jeu de données contient des colonnes avec des valeurs de plusieurs types, l'aperçu des données affiche une liste de tuples pour ces colonnes. Chaque tuple contient le type de données et sa valeur.

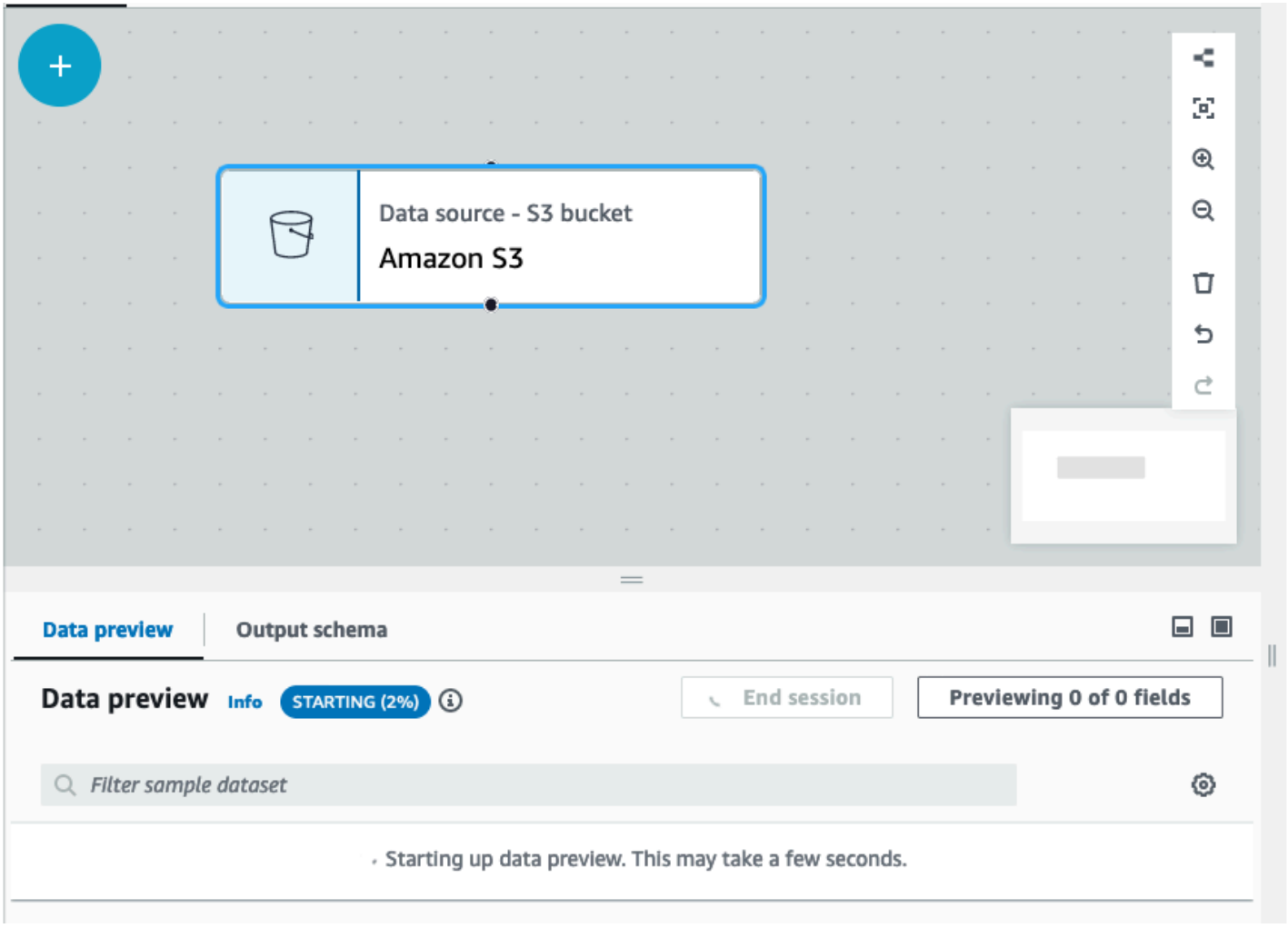
> **Note**
>
> Si vous utilisez une session d'aperçu des données et un code SQL personnalisé ou un nœud de code personnalisé, la session d'aperçu des données exécutera le code SQL ou le bloc de code tel quel pour tout le jeu de données.

Lors de la création ou de la modification de votre tâche, vous pouvez utiliser l'onglet Prévisualisation des données sous le canevas de la tâche pour afficher un échantillon de vos données. Une nouvelle session de prévisualisation des données démarre automatiquement lorsque le rôle est déjà configuré dans la tâche ou qu'un rôle IAM par défaut a été configuré dans le compte. Si aucun rôle n'a été configuré auparavant, vous pouvez démarrer une session en sélectionnant le rôle.

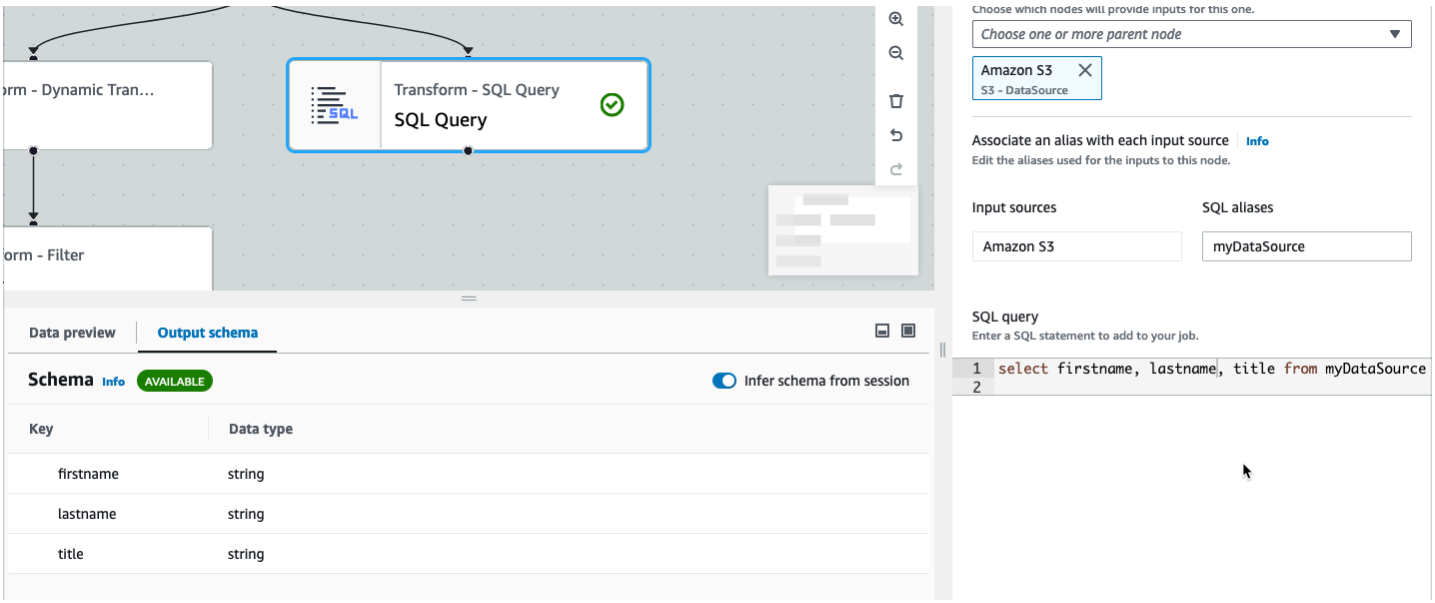
> **Note**
>
> Le rôle que vous choisissez pour la session de prévisualisation des données sera également utilisé pour la tâche.

Vous pouvez voir l'état et la progression de votre session ainsi que les détails de la session en cliquant sur l'icône d'information.

Lorsque la session est prête, AWS Glue Studio charge les données pour le nœud que vous avez sélectionné. Vous pouvez voir le % d'avancement au fur et à mesure de sa progression.



Lorsque vous créez votre tâche visuelle, AWS Glue Studio mettra automatiquement à jour le schéma du nœud sélectionné lorsque vous activez Déduire les schémas de la session sous l'onglet Schéma en sortie.



Pour configurer vos préférences pour les prévisualisations des données :

Choisissez l'icône des paramètres (symbole d'engrenage) pour configurer vos préférences pour les prévisualisations des données. Ces paramètres s'appliquent à tous les nœuds dans le diagramme de tâche. Vous pouvez effectuer les actions suivantes :

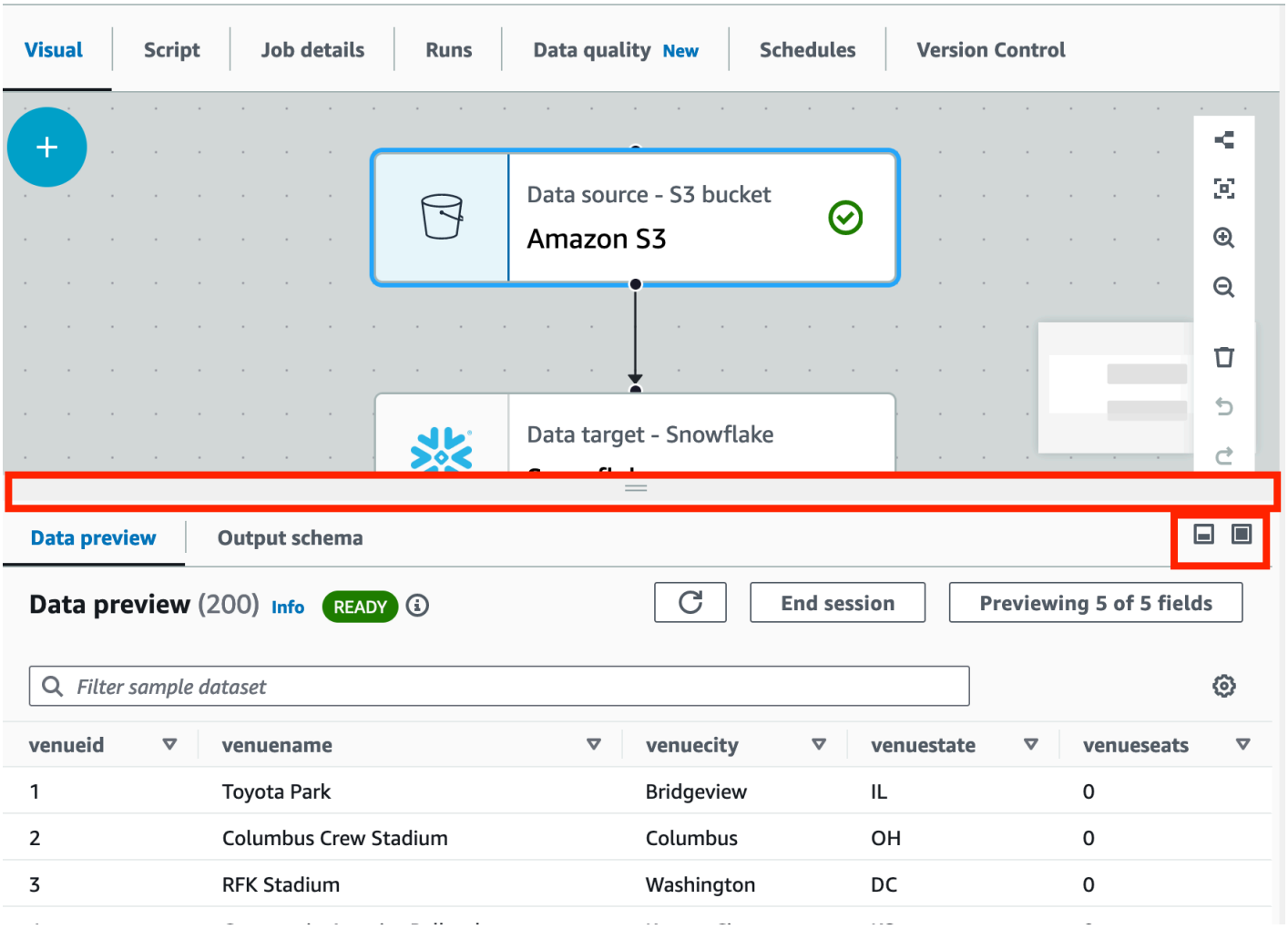
- Choisir d'envelopper le texte d'une ligne à l'autre. Cette option est activée par défaut
- Modifier le nombre de lignes (200 par défaut)
- Choisir un rôle IAM ou en créer un si nécessaire
- Choisir de démarrer automatiquement une nouvelle session lorsque vous créez une tâche. Cela permet d'ouvrir une nouvelle session interactive lors de la création de tâches. Ce paramètre s'applique au niveau du compte. Une fois défini, il s'appliquera à tous les utilisateurs de votre compte lors de la modification d'une tâche.
- Choisir de déduire automatiquement le schéma. Les schémas en sortie seront automatiquement déduits pour le nœud sélectionné
- Choisir d'importer automatiquement les bibliothèques AWS Glue. Ceci est utile pour empêcher la prévisualisation des données de redémarrer de nouvelles sessions lors de l'ajout de nouvelles transformations nécessitant un redémarrage de session

Les fonctionnalités supplémentaires incluent la possibilité de :

- Choisissez le bouton Previewing x of y fields (Prévisualiser x des y champs) pour sélectionner les colonnes (champs) à afficher. Lorsque vous prévisualisez vos données à l'aide des paramètres par

défaut, l'éditeur de tâches affiche les 5 premières colonnes de votre jeu de données. Vous pouvez le modifier pour afficher tout ou aucun (non recommandé).

- Faire défiler la fenêtre de prévisualisation des données horizontalement et verticalement.
- Utilisez le bouton d'agrandissement pour étendre l'onglet Prévisualisation des données et le superposer au graphique des tâches afin de mieux visualiser les données et les structures de données. De même, utilisez le bouton de réduction pour réduire l'onglet Prévisualisation des données. Vous pouvez également saisir le panneau de poignée et le faire glisser vers le haut pour développer l'onglet Prévisualisation des données.



- Cliquez sur Mettre fin à la session pour arrêter la prévisualisation des données. Lorsque vous arrêtez la session, vous pouvez choisir un nouveau rôle IAM et définir des paramètres supplémentaires (tels que les paramètres d'activation ou de désactivation) pour démarrer automatiquement une nouvelle session, déduire un schéma ou importer des bibliothèques AWS Glue, puis redémarrer la session.

Restrictions lors de l'utilisation de prévisualisations de données

Lorsque vous utilisez des prévisualisations de données, il se peut que vous disposiez des restrictions ou limitations suivantes.

- La première fois que vous choisissez l'onglet Data preview (Prévisualisation des données), vous devez choisir le rôle IAM. Ce rôle doit disposer des autorisations nécessaires pour accéder aux données et aux autres ressources nécessaires à la création des prévisualisations de données.
- Une fois que vous avez fourni un rôle IAM, il faut un certain temps avant que les données soient disponibles pour l'affichage. Pour les jeux de données avec moins de 1 Go de données, cela peut prendre jusqu'à une minute. Si vous disposez d'un jeu de données volumineux, vous devriez utiliser des partitions pour améliorer le temps de chargement. Le chargement des données directement à partir d'Amazon S3 offre les meilleures performances.
- Si vous disposez d'un jeu de données très volumineux et qu'il faut plus de 15 minutes pour interroger les données pour la prévisualisation des données, la requête expire. Les aperçus des données ont un délai d'inactivité de 30 minutes. Pour pallier ce problème, réduisez la taille du jeu de données pour utiliser des prévisualisations de données.
- Par défaut, les 50 premières colonnes s'affichent dans l'onglet Prévisualisation des données. Si les colonnes n'ont pas de valeurs de données, vous recevrez un message indiquant qu'il n'y a pas de données à afficher. Vous pouvez augmenter le nombre de lignes échantillonnées ou sélectionner différentes colonnes pour afficher les valeurs de données.
- Les prévisualisations de données ne sont actuellement pas prises en charge pour les sources de données en streaming ou pour les sources de données qui utilisent des connecteurs personnalisés.
- Les erreurs sur un nœud affectent l'ensemble de la tâche. Si un nœud a une erreur avec les prévisualisations de données, l'erreur apparaîtra sur tous les nœuds jusqu'à ce que vous le corrigiez.
- Si vous modifiez une source de données pour la tâche, il se peut que les nœuds enfants de cette source de données doivent être mis à jour pour correspondre au nouveau schéma. Par exemple, si un ApplyMapping nœud modifie une colonne et que celle-ci n'existe pas dans la source de données de remplacement, vous devez mettre à jour le nœud de ApplyMapping transformation.
- Si vous affichez l'onglet Prévisualisation des données pour un nœud de transformation de requête SQL et que la requête SQL utilise un nom de champ incorrect, l'onglet Prévisualisation des données affiche une erreur.

Génération de code de script

Lorsque vous utilisez l'éditeur visuel pour créer une tâche, le code ETL est automatiquement généré pour vous. AWS Glue Studio crée un script de tâche fonctionnel et complet, et l'enregistre dans un emplacement Amazon S3.

Il existe deux formes de code générées par AWS Glue Studio : la version originale, ou classique, et une version plus récente et simplifiée. Par défaut, le nouveau générateur de code est utilisé pour créer le script de tâche. Vous pouvez générer un script de tâche à l'aide du générateur de code classique dans l'onglet Script en sélectionnant le bouton à bascule Generate classic script (Générer un script classique).

Voici quelques-unes des différences de la nouvelle version du code généré :

- Les blocs de commentaires volumineux ne sont plus ajoutés au script
- Les structures de sortie du code utilisent le nom du nœud que vous spécifiez dans l'éditeur visuel. Dans le script de classe, les structures de sortie sont simplement nommées `DataSource0`, `DataSource1`, `Transform0`, `Transform1`, `DataSink0`, `DataSink1`, etc.
- Les commandes longues sont réparties sur plusieurs lignes pour supprimer le besoin de faire défiler la page pour voir l'ensemble de la commande.

Les nouvelles fonctions dans AWS Glue Studio nécessitent la nouvelle version de la génération de code et ne fonctionnent pas avec le script de code classique. Il vous est proposé de mettre à jour ces tâches lorsque vous tentez de les exécuter.

Transformation de données avec des transformations gérées par AWS Glue

AWS Glue Studio fournit deux types de transformations :

- Transformations AWS Glue natives : disponibles pour tous les utilisateurs et gérées par AWS Glue.
- Transformations visuelles personnalisées : permettent de télécharger vos propres transformations à utiliser dans AWS Glue Studio

Nœuds de transformation de données gérées par AWS Glue

AWS Glue Studio fournit un ensemble de transformations intégrées que vous pouvez utiliser pour traiter vos données. Vos données passent d'un nœud dans le diagramme de tâches à un autre, dans

une structure de données appelée `DynamicFrame`, qui est une extension d'un `DataFrame` SQL Apache Spark.

Dans le diagramme prérempli d'une tâche, entre la source de données et les nœuds de données cibles, se trouve le nœud de transformation Modifier le schéma. Vous pouvez configurer ce nœud de transformation pour modifier vos données ou utiliser des transformations supplémentaires.

Les transformations intégrées suivantes sont disponibles avec AWS Glue Studio :

- ChangeSchema: Mappez les clés de propriété des données de la source de données aux clés de propriété des données de la cible de données. Vous pouvez renommer les clés, modifier leur type de données et choisir les clés à supprimer du jeu de données.
- SelectFields: Choisissez les clés de propriété des données que vous souhaitez conserver.
- DropFields: Choisissez les clés de propriété des données que vous souhaitez supprimer.
- RenameField: renommez une seule clé de propriété de données.
- Spigot : écrivez des échantillons de données dans un compartiment Amazon S3.
- Join : joignez deux jeux de données dans un jeu de données à l'aide d'une phrase de comparaison sur les clés de propriété de données spécifiées. Vous pouvez utiliser des jointures internes (ou intérieures), externes (ou extérieures), gauche, droite, semi gauche et anti gauche.
- Union : combinez les lignes de plusieurs sources de données ayant le même schéma.
- SplitFields: Divisez les clés de propriété des données en deux`DynamicFrames`. Le résultat est une collection de `DynamicFrames` : une avec les clés de propriété de données sélectionnées, et une autre avec les clés de propriété de données restantes.
- SelectFromCollection: Choisissez-en un `DynamicFrame` parmi une collection de`DynamicFrames`. Le résultat est le `DynamicFrame` sélectionné.
- FillMissingValues: Localisez les enregistrements de l'ensemble de données qui contiennent des valeurs manquantes et ajoutez un nouveau champ avec une valeur suggérée déterminée par imputation
- Filter : divisez un jeu de données en deux, en fonction d'une condition de filtrage.
- Drop Null Fields : supprime les colonnes du jeu de données si toutes les valeurs de la colonne sont « null ».
- Drop Duplicates : supprime des lignes de votre source de données en choisissant de faire correspondre des lignes entières ou de spécifier des clés.
- SQL : entrez le code SparkSQL dans un champ de saisie de texte pour utiliser une requête SQL pour transformer les données. Le résultat est un `DynamicFrame` unique .

- Regrouper : effectue un calcul (tel que la moyenne, la somme, le minimum, le maximum) sur les champs et les lignes sélectionnés, et crée un nouveau champ avec la ou les valeurs nouvellement calculées.
- Aplatir : extraie les champs des structs dans les champs de niveau supérieur.
- UUID : ajoute une colonne avec un identifiant unique universel pour chaque ligne.
- Identifiant : ajoute une colonne avec un identifiant numérique pour chaque ligne.
- En horodatage : convertit une colonne en type horodatage.
- Formater l'horodatage : convertit une colonne d'horodatage en chaîne formatée.
- Transformation du routeur conditionnel : applique plusieurs conditions aux données entrantes. Chaque ligne des données entrantes est évaluée par une condition de filtre de groupe et traitée dans le groupe correspondant.
- Transformation Concaténer des colonnes : créez une colonne de chaîne en utilisant les valeurs d'autres colonnes avec un espacement facultatif.
- Transformation Diviser la chaîne : divisez une chaîne en un tableau de jetons à l'aide d'une expression régulière pour définir la manière dont la division est effectuée.
- Transformation Tableau vers colonnes : extrayez certains ou tous les éléments d'une colonne de type tableau dans de nouvelles colonnes.
- Transformation Ajouter un horodatage actuel : marquez les lignes avec l'heure à laquelle les données ont été traitées. Ceci est utile à des fins d'audit ou pour suivre la latence dans le pipeline de données.
- Transformation Faire pivoter les lignes en colonnes : agrégez une colonne numérique en faisant pivoter des valeurs uniques sur des colonnes sélectionnées qui deviennent de nouvelles colonnes. Si plusieurs colonnes sont sélectionnées, les valeurs sont concaténées pour nommer les nouvelles colonnes.
- Transformation Dépivoter les colonnes en lignes : convertissez des colonnes en valeurs de nouvelles colonnes en générant une ligne pour chaque valeur unique.
- Transformation Traitement de l'équilibre automatique : redistribuez mieux les données entre les travailleurs. Cela est utile dans les cas où les données sont déséquilibrées ou lorsque la source ne permet pas un traitement parallèle suffisant.
- Transformation Colonnes dérivées : définissez une nouvelle colonne basée sur une formule mathématique ou une expression SQL dans laquelle vous pouvez utiliser d'autres colonnes dans les données, ainsi que des constantes et des littéraux.

- Transformation Rechercher : ajoutez des colonnes à partir d'une table de catalogue définie lorsque les clés correspondent aux colonnes de recherche définies dans les données.
- Transformation Éclater le tableau ou la carte en lignes : extrayez les valeurs d'une structure imbriquée en lignes individuelles plus faciles à manipuler.
- Transformation Correspondance d'enregistrements : invoquez une transformation de classification de données de machine learning Correspondance des enregistrements existante.
- Transformation Supprimer les lignes nulles : supprimez du jeu de données les lignes dont toutes les colonnes sont nulles ou vides.
- Transformation Analyser la colonne JSON : analysez une colonne de chaîne contenant des données JSON et convertissez-la en une structure ou en une colonne de tableau, selon que le JSON est respectivement un objet ou un tableau.
- Transformation Extraire le chemin JSON : extrayez les nouvelles colonnes d'une colonne de chaîne JSON.
- Extraire des fragments de chaînes d'une expression régulière : extrayez des fragments de chaîne à l'aide d'une expression régulière et créez une colonne à partir de celle-ci, ou plusieurs colonnes si vous utilisez des groupes d'expressions régulières.
- Custom transform : saisissez du code dans un champ de saisie de texte pour utiliser des transformations personnalisées. Le résultat est une collection de `DynamicFrames`.

Utilisation d'une recette de préparation des données dans AWS Glue Studio

La transformation Recettes de préparation des données vous permet de créer une recette de préparation des données à partir de zéro à l'aide d'une interface de création interactive de type grille. Elle vous permet également d'importer une recette AWS Glue DataBrew existante, puis de la modifier dans AWS Glue Studio.

Le nœud Recette de préparation des données est disponible dans le panneau Ressources. Vous pouvez connecter le nœud Recette de préparation des données à un autre nœud du flux de travail visuel, qu'il s'agisse d'un nœud de source de données ou d'un autre nœud de transformation. Après avoir choisi une AWS Glue DataBrew recette et une version, les étapes appliquées dans la recette sont visibles dans l'onglet des propriétés du nœud.

Conditions préalables

- Si vous importez une AWS Glue DataBrew recette, vous disposez des autorisations IAM requises, comme décrit dansImporter une AWS Glue DataBrew recette dans AWS Glue Studio.

- Une session de prévisualisation des données doit être créée.

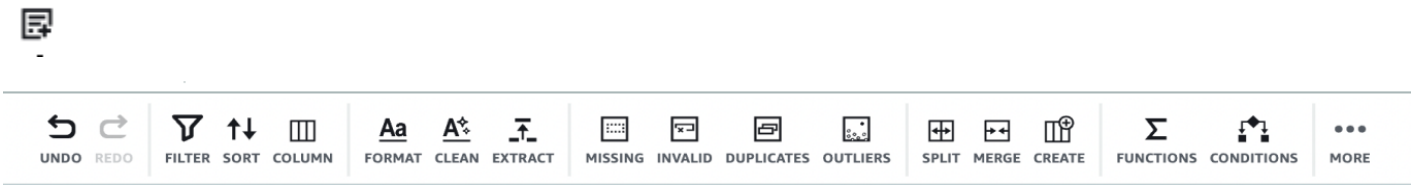
Limitations

- AWS Glue DataBrew les recettes ne sont prises en charge que dans DataBrew les régions commerciales.
- Toutes les AWS Glue DataBrew recettes ne sont pas prises en charge parAWS Glue. Certaines recettes ne pourront pas être exécutées dans AWS Glue Studio.
  - Les recettes avec des transformations de type `UNION` et `JOIN` ne sont pas prises en charge, mais AWS Glue Studio dispose déjà de nœuds de transformation « Join » et « Union » qui peuvent être utilisés avant ou après un nœud Data Preparation Recipe.
- Les nœuds Recette de préparation des données sont pris en charge pour les tâches à partir de la version 4.0 de AWS Glue. Cette version sera sélectionnée automatiquement après l'ajout d'un nœud Recette de préparation des données à la tâche.
- Les nœuds Recette de préparation des données nécessitent Python. Ceci est automatiquement défini lorsque le nœud Recette de préparation des données est ajouté à la tâche.
- L'ajout d'un nouveau nœud Data Preparation Recipe au graphique visuel redémarrera automatiquement votre session de prévisualisation des données avec les bibliothèques appropriées pour utiliser le nœud Data Preparation Recipe.
- Les transformations suivantes ne sont pas prises en charge pour l'importation ou la modification dans un nœud Data Preparation Recipe : `GROUP_BY`, `PIVOT`, `UNPIVOT` et `TRANSPOSE`.

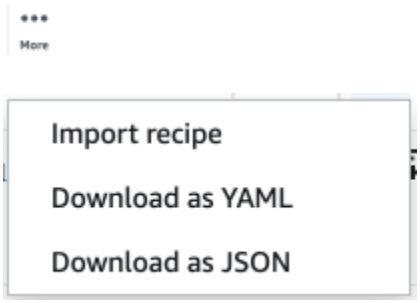
Fonctionnalités supplémentaires

Lorsque vous avez sélectionné la transformation de Recette de préparation des données, vous pouvez effectuer des actions supplémentaires après avoir sélectionné Créer une recette.

- Ajouter une étape : vous pouvez ajouter des étapes supplémentaires à une recette selon vos besoins en cliquant sur l'icône d'ajout d'étape, ou utiliser la barre d'outils du volet de prévisualisation en choisissant une action.



- Importer une recette : choisissez Plus, puis Importer une recette à utiliser dans le cadre de votre tâche AWS Glue Studio .

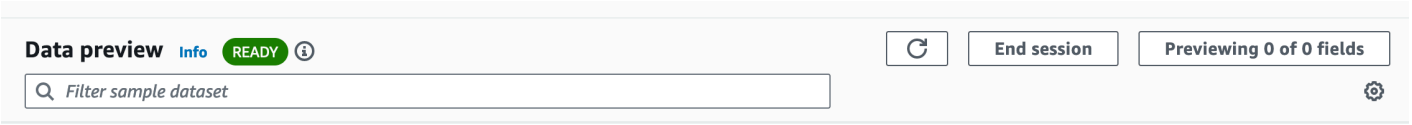


- Télécharger au format YAML : choisissez Plus puis Télécharger au format YAML pour télécharger votre recette et l'enregistrer en dehors de. AWS Glue Studio
- Télécharger au format JSON : choisissez Plus puis Télécharger au format JSON pour télécharger votre recette à enregistrer en dehors de AWS Glue Studio.
- Annuler et rétablir les étapes de la recette : vous pouvez annuler et rétablir les étapes de la recette dans le volet de prévisualisation lorsque vous travaillez avec des données dans la grille.

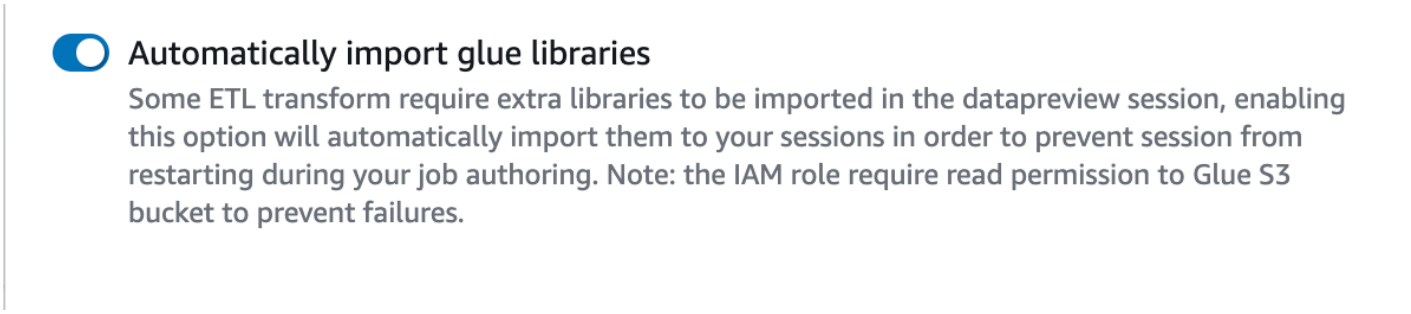
Créez et exécutez des recettes de préparation de données dans une AWS Glue tâche ETL visuelle

Dans ce scénario, vous pouvez créer des recettes de préparation des données sans avoir à les créer au préalable DataBrew. Avant de commencer à créer des recettes, vous devez respecter les conditions suivantes :

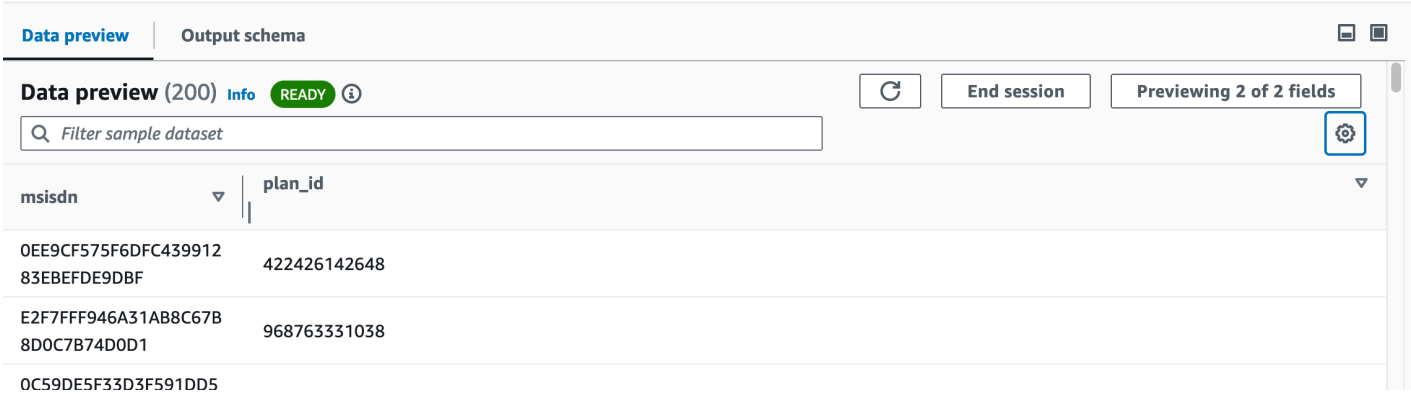
- Disposer d'une session de prévisualisation des données active et en cours d'exécution. Lorsque la session de prévisualisation des données est PRÊTE, l'option Créer une recette devient active et vous pouvez commencer à créer ou à modifier votre recette.



- Assurez-vous que l'option Importer automatiquement les bibliothèques Glue est activée.

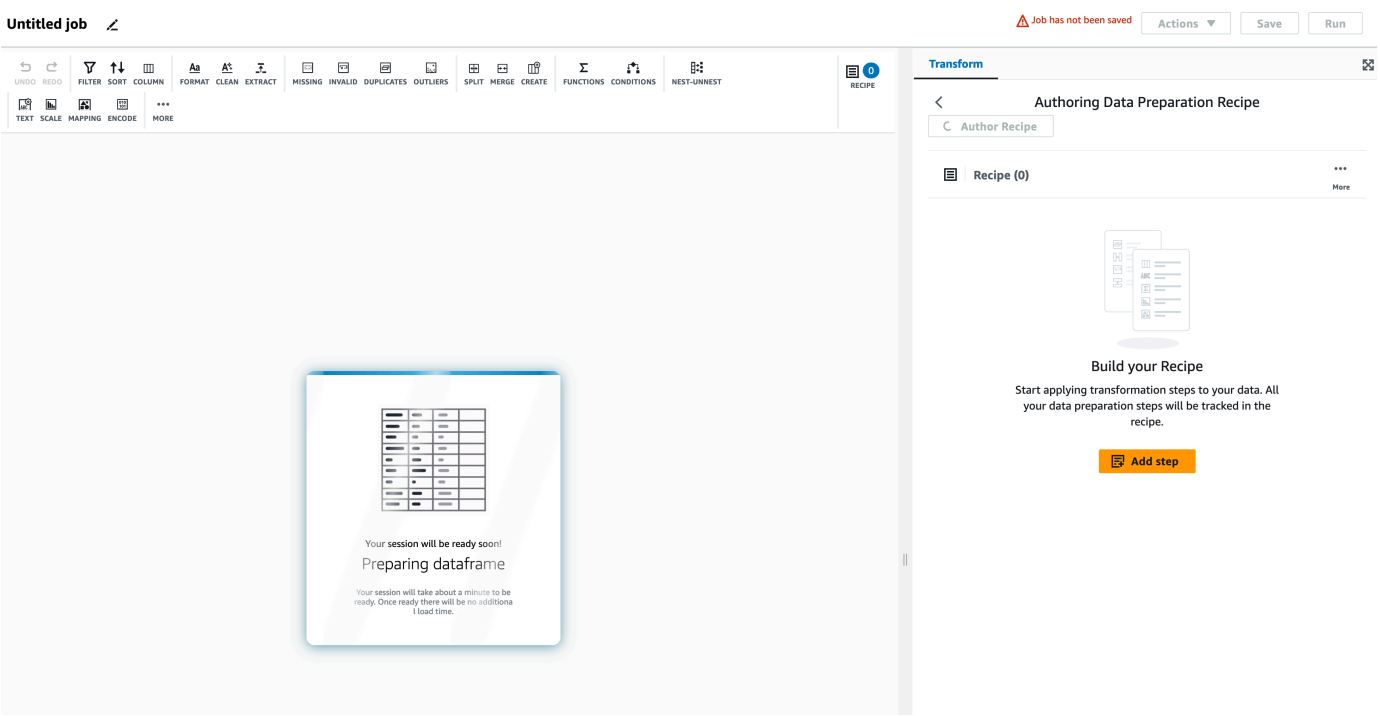


Vous pouvez procéder en choisissant l'icône d'engrenage dans le volet Prévisualisation des données.

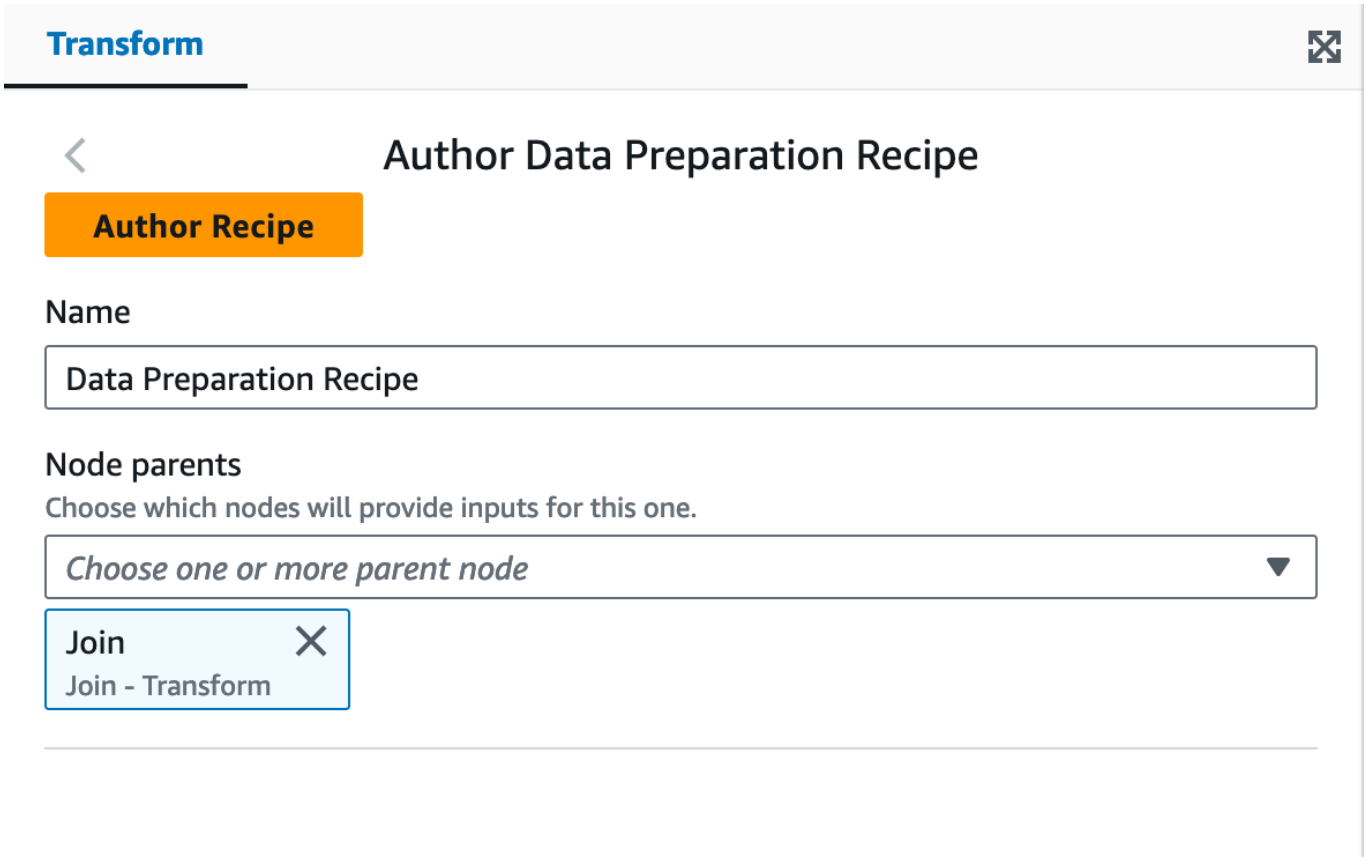


Pour créer une recette de préparation des données dans AWS Glue Studio :

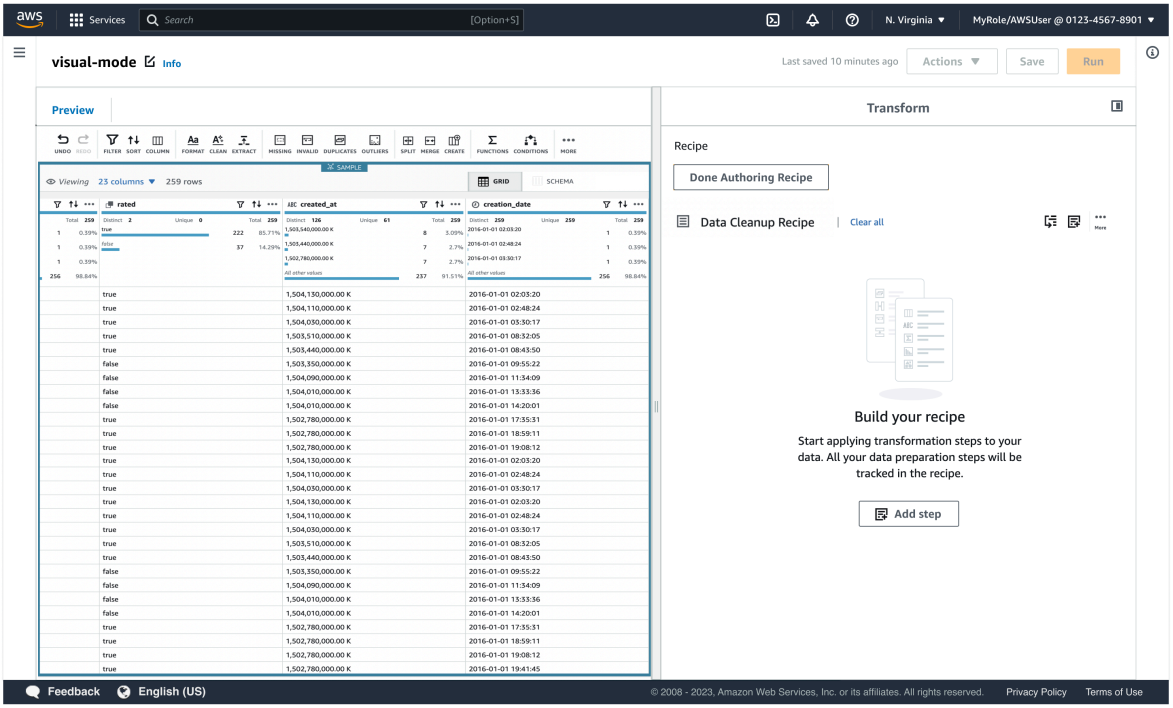
1. Ajoutez la transformation Recette de préparation des données à votre canevas de tâches. Votre transformation doit être connectée à un nœud de source de données parent. Lorsque vous ajoutez le nœud Data Preparation Recipe, celui-ci redémarrera avec les bibliothèques appropriées et vous verrez la trame de données en cours de préparation.



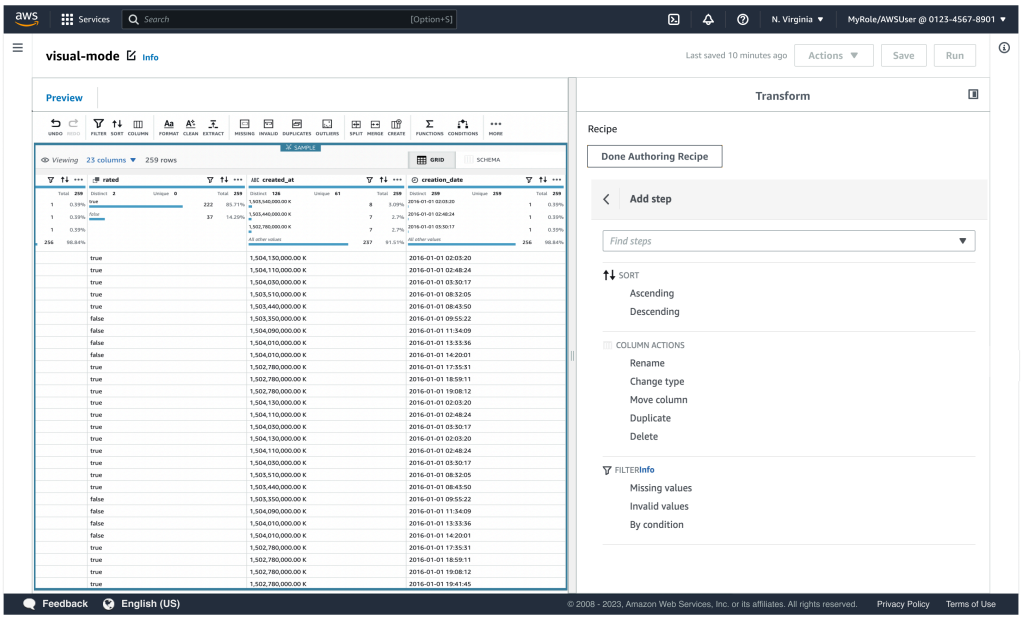
2. Une fois que la session de prévisualisation des données est prête, les données contenant les étapes précédemment appliquées apparaîtront en bas de l'écran.
3. Choisissez Créer une recette. Cela vous permettra de démarrer une nouvelle recette dans AWS Glue Studio.



4. Dans le panneau Transformation situé à droite du canevas de tâches, saisissez le nom de votre recette de préparation des données.
5. Sur le côté gauche, le canevas sera remplacé par une vue en grille de vos données. Sur la droite, le panneau Transformation changera pour vous montrer les étapes de votre recette. Choisissez Ajouter une étape pour ajouter la première étape à votre recette.

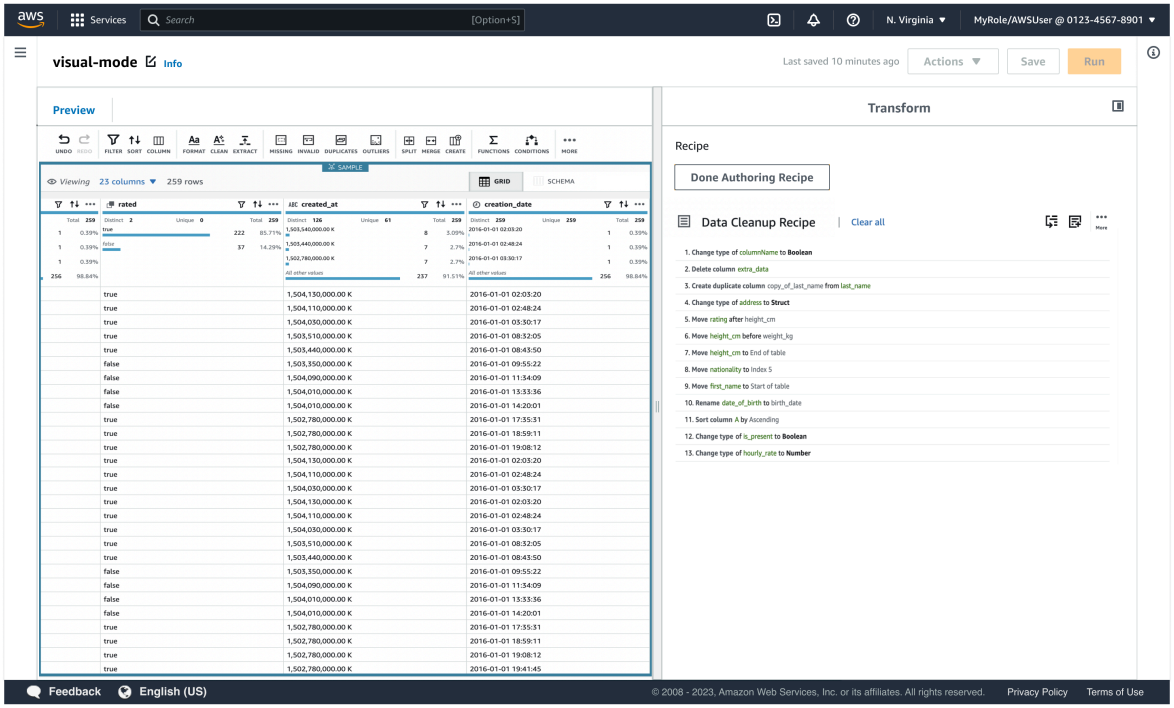


6. Dans le panneau Transformation, choisissez de trier, d'effectuer une action sur la colonne et de filtrer les valeurs. Par exemple, choisissez Renommer la colonne.



7. Dans le panneau Transformation sur le côté droit, les options visant à renommer une colonne vous permettent de choisir la colonne source à renommer et de saisir le nouveau nom de colonne. Une fois que vous l'avez fait, choisissez Appliquer.

   Vous pouvez prévisualiser chaque étape, annuler une étape, réorganiser les étapes et utiliser n'importe laquelle des icônes d'action, telles que Filtrer, Trier, Diviser, Fusionner, etc. Lorsque vous effectuez des actions dans la grille de données, les étapes sont ajoutées à la recette dans le panneau Transformation.

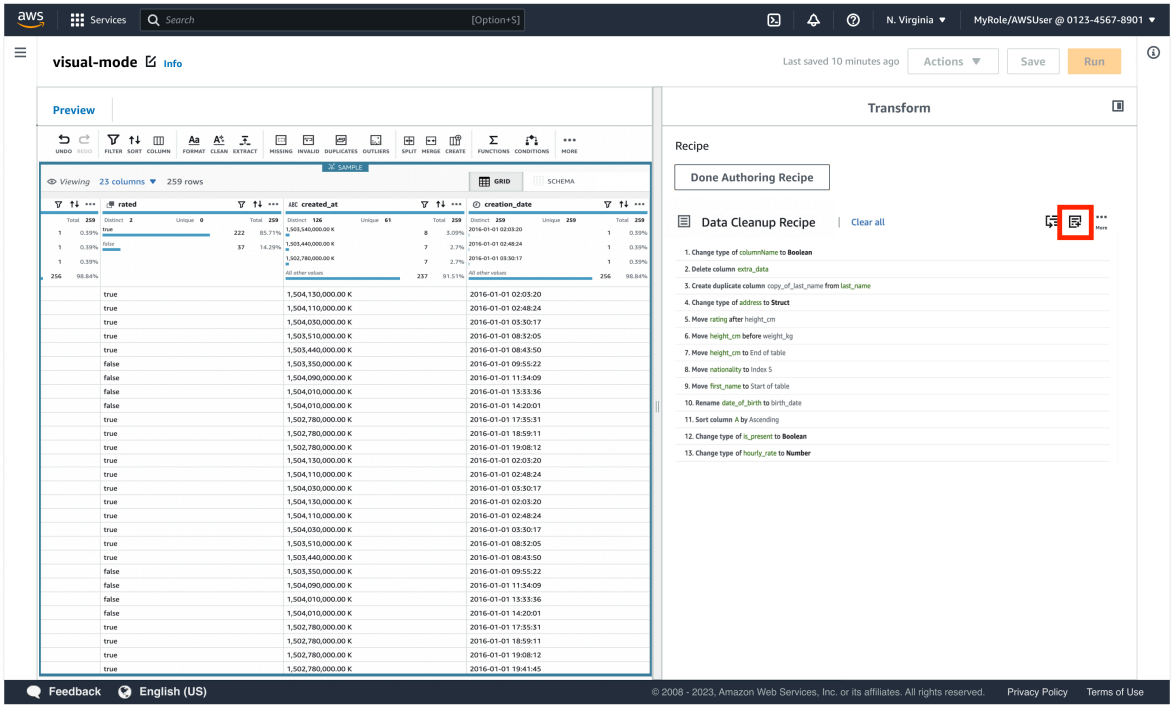


Si vous devez apporter une modification, vous pouvez le faire dans le volet d'aperçu en prévisualisant le résultat de chaque étape, en annulant une étape et en réorganisant les étapes. Par exemple :

- Undo/redo étape : annuler une étape en choisissant l'icône d'annulation. Vous pouvez répéter une étape en choisissant l'icône de rétablissement.

- Étape de réorganisation : lorsque vous réorganisez une étape, AWS Glue Studio chaque étape est validée et vous indique si l'étape n'est pas valide.

8. Une fois que vous avez appliqué une étape, le panneau Transformation affiche toutes les étapes de votre recette. Vous pouvez effacer toutes les étapes pour recommencer, ajouter d'autres étapes en cliquant sur l'icône d'ajout ou choisir Terminer la création de la recette.



9. Choisissez Enregistrer en haut à droite de votre écran. Les étapes de votre recette ne sont pas enregistrées tant que vous n'avez pas enregistré votre tâche.

Importer une AWS Glue DataBrew recette dans AWS Glue Studio

Dans AWS Glue DataBrew, une recette est un ensemble d'étapes de transformation de données. AWS Glue DataBrew recipes décrit comment transformer des données déjà lues et ne décrit pas où et comment lire les données, ni comment et où écrire des données. Ceci est configuré dans les nœuds source et cible dans AWS Glue Studio. Pour plus d'informations sur les recettes, voir Création et utilisation de AWS Glue DataBrew recettes.

Pour utiliser AWS Glue DataBrew des recettes dansAWS Glue Studio, commencez par créer des recettes dans AWS Glue DataBrew. Si vous avez des recettes que vous souhaitez utiliser, vous pouvez ignorer cette étape.

Autorisations IAM pour AWS Glue DataBrew

Cette rubrique fournit des informations pour vous aider à comprendre les actions et les ressources qu'un administrateur IAM peut utiliser dans une politique Gestion des identités et des accès AWS (IAM) pour la transformation de la recette de préparation des données.

Pour plus d'informations sur la sécurité dans AWS Glue, consultez la section Gestion des accès.

> **Note**
>
> Le tableau suivant répertorie les autorisations dont les utilisateurs ont besoin pour importer une recette AWS Glue DataBrew existante.

Actions de la transformation Recette de préparation des données

Action	Description
`databrew:ListRecipes`	Accorde l'autorisation de récupérer AWS Glue DataBrew des recettes.
`databrew:ListRecipeVersions`	Accorde l'autorisation de récupérer des versions de AWS Glue DataBrew recettes.
`databrew:DescribeRecipe`	Accorde l'autorisation de récupérer AWS Glue DataBrew la description de la recette.

Le rôle que vous utilisez pour accéder à cette fonctionnalité doit être doté d'une politique qui autorise plusieurs actions AWS Glue DataBrew . Vous pouvez y parvenir soit en utilisant la politique `AWSGlueConsoleFullAccess` qui inclut les actions nécessaires, soit en ajoutant la politique en ligne suivante à votre rôle :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
```

```
      "Effect": "Allow",
      "Action": [
        "databrew:ListRecipes",
        "databrew:ListRecipeVersions",
        "databrew:DescribeRecipe"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
```

Pour utiliser la transformation Recette de préparation des données, vous devez ajouter l'action `IAM:PassRole` à la politique d'autorisation.

Autorisations supplémentaires requises

Action	Description
`iam:PassRole`	Permet à IAM d'autoriser l'utilisateur à transmettre les rôles approuvés.

Sans ces autorisations, l'erreur suivante se produit :

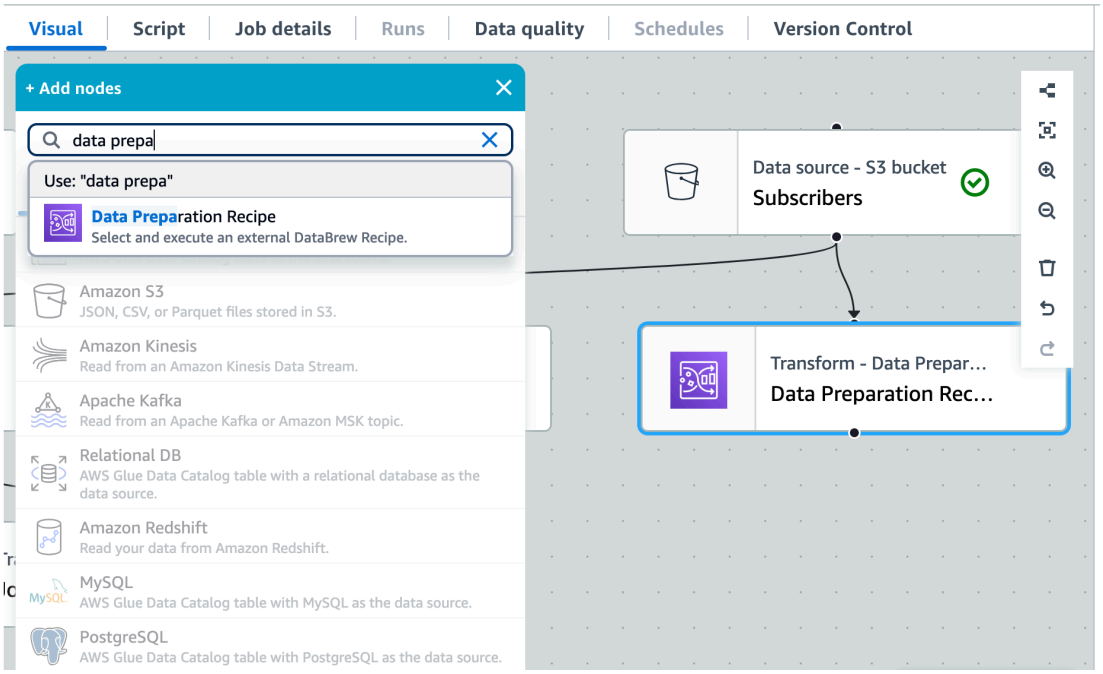
```
"errorCode": "AccessDenied"
"errorMessage": "User: arn:aws:sts::account_id:assumed-role/AWSGlueServiceRole is not
authorized to perform: iam:PassRole on resource: arn:aws:iam::account_id:role/service-
role/AWSGlueServiceRole
because no identity-based policy allows the iam:PassRole action"
```

Importer une AWS Glue DataBrew recette

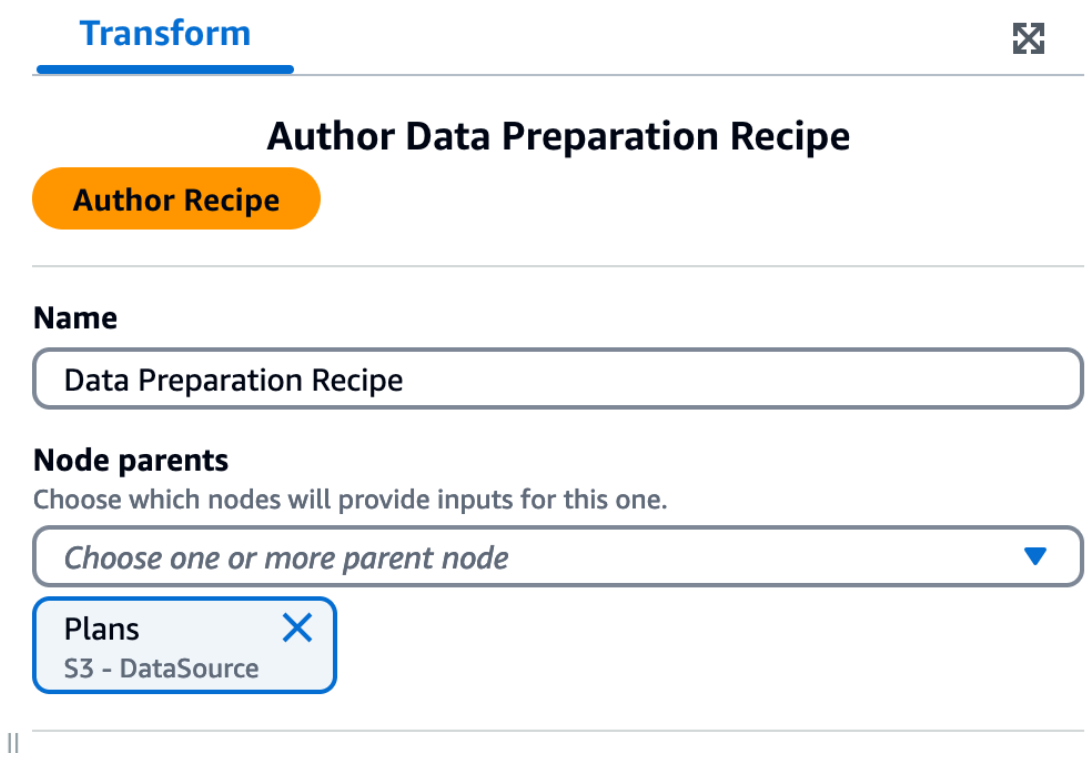
Pour importer une AWS Glue DataBrew recette et l'utiliser dans AWS Glue Studio :

Si vous possédez déjà un nœud de recette de préparation des données et que vous souhaitez modifier les étapes de la recette directement dans celui-ci AWS Glue Studio, vous devrez importer les étapes de la recette dans votre AWS Glue Studio tâche.

1. Démarrez une tâche AWS Glue dans AWS Glue Studio avec une source de données.
2. Ajoutez le nœud Data Preparation Recipe au canevas de tâches.

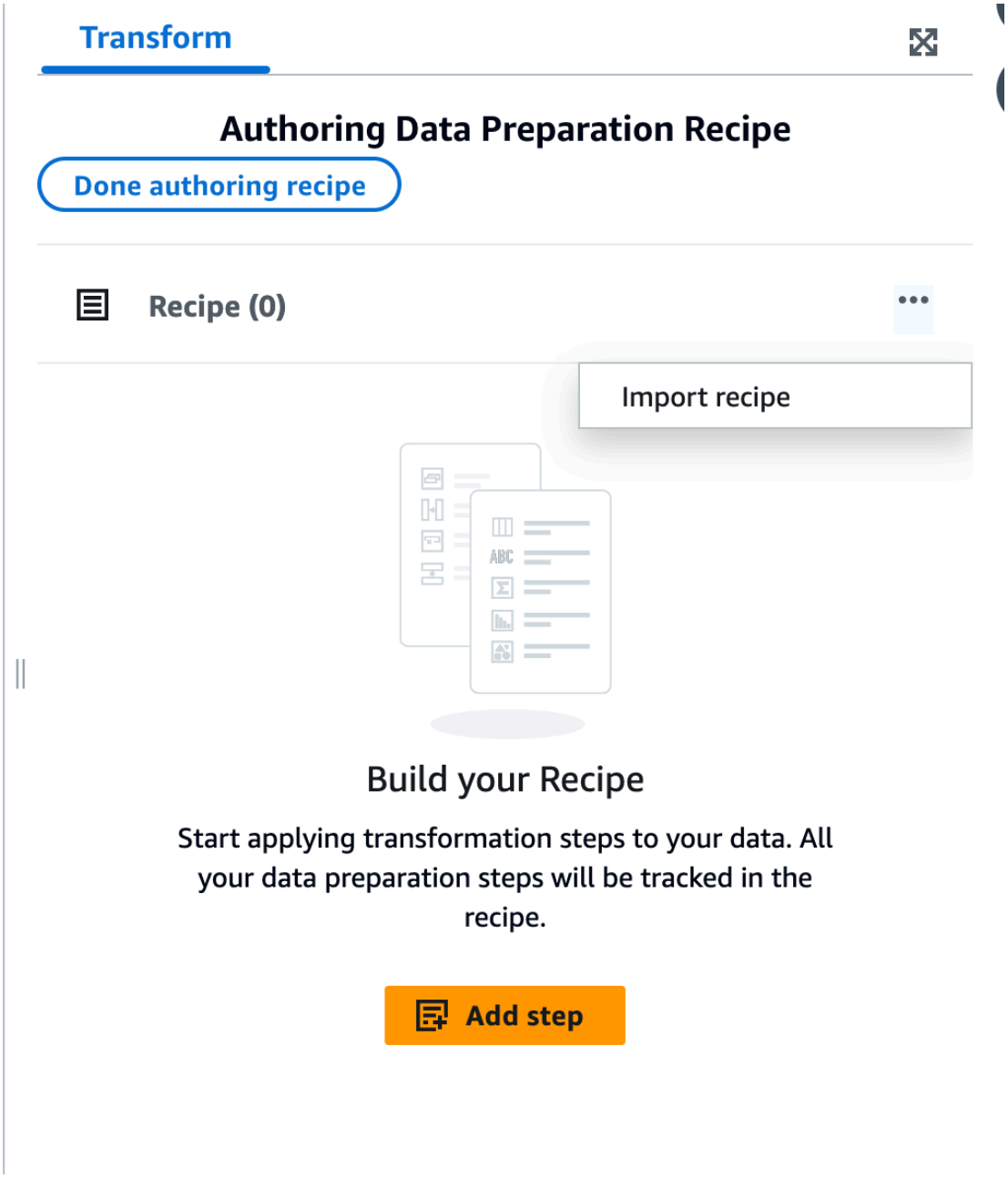


3. Dans le panneau Transformation, saisissez un nom pour votre recette.
4. Sélectionnez un ou plusieurs nœuds parent en sélectionnant les nœuds disponibles sur le canevas dans la liste déroulante.
5. Choisissez Créer une recette. Si l'option Créer une recette est grisée, elle n'est pas disponible tant que les nœuds parent n'ont pas été sélectionnés et qu'une session de prévisualisation des données ne s'est pas terminée.

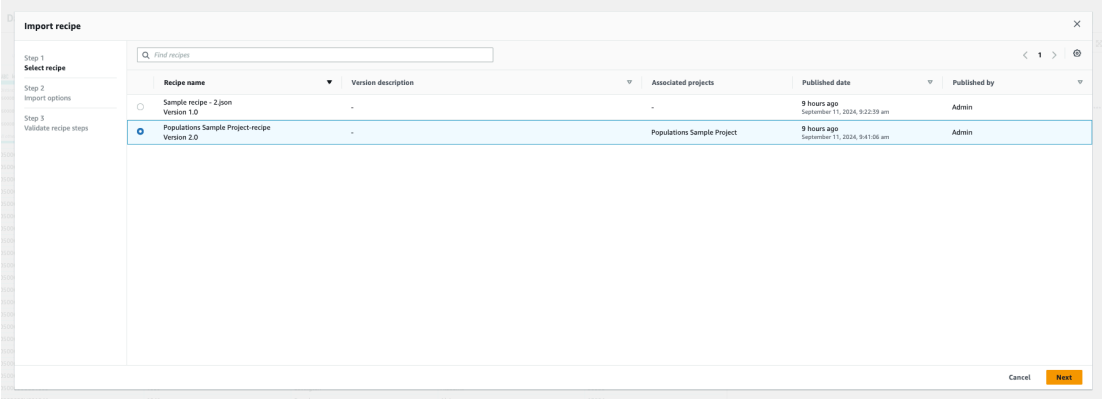


6. La trame de données se charge et affiche des informations détaillées sur vos données source.

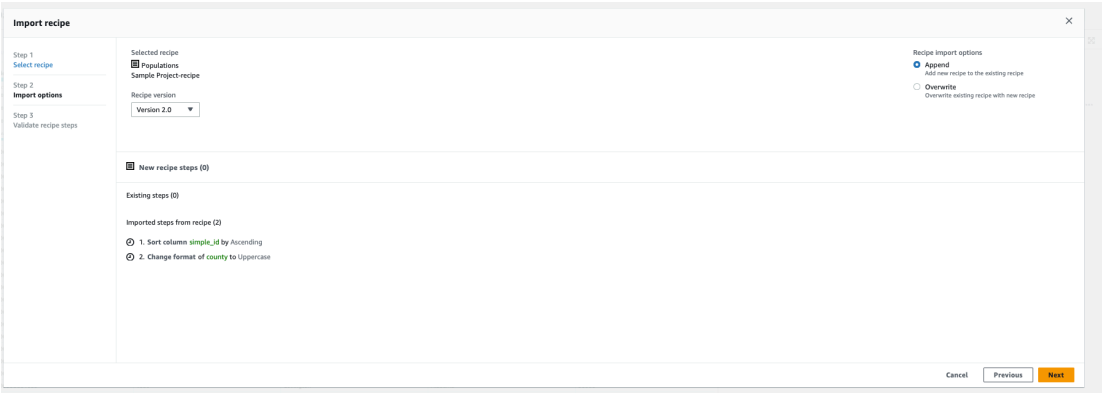
   Sélectionnez l'icône Autres actions et choisissez Importer la recette.



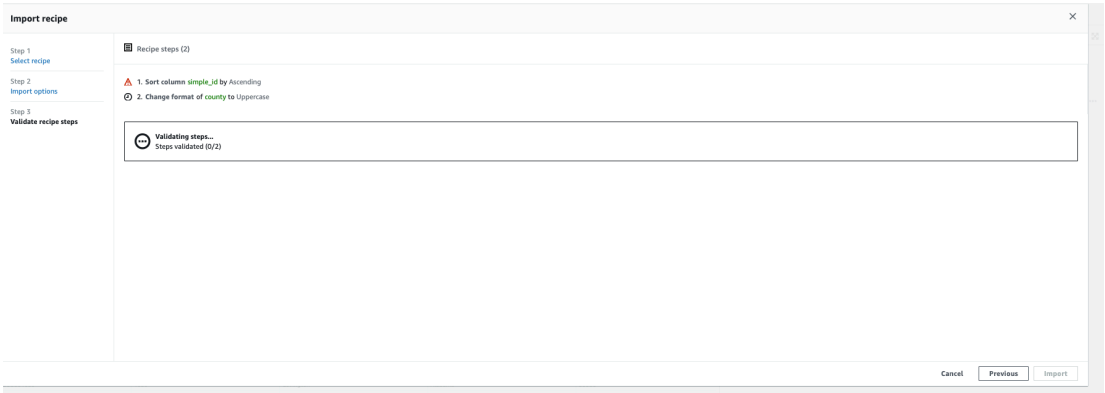
7. Utilisez l'assistant d'importation de recettes pour effectuer la procédure. À l'étape 1, recherchez votre recette, sélectionnez-la, puis cliquez sur Suivant.

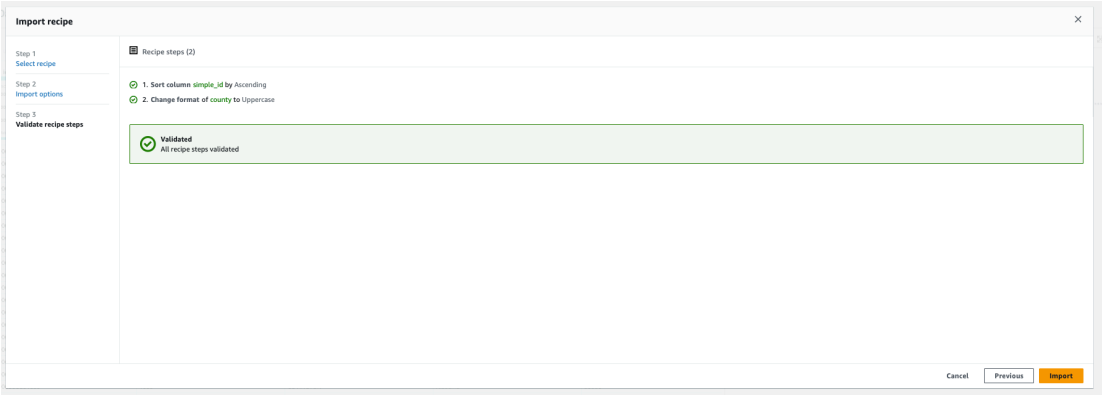


8. À l'étape 2, choisissez vos options d'importation. Vous pouvez choisir d'ajouter une nouvelle recette à une recette existante ou de remplacer une recette existante. Choisissez Suivant.



9. À l'étape 3, validez les étapes de la recette. Une fois que vous avez importé votre AWS Glue DataBrew recette, vous pouvez la modifier directement dans AWS Glue Studio.





10. Ensuite, les étapes seront importées dans le cadre de votre AWS Glue travail. Apportez les modifications de configuration nécessaires dans l'onglet Informations de la tâche, par exemple en nommant votre tâche et en ajustant la capacité allouée selon les besoins. Choisissez Enregistrer pour enregistrer votre tâche et votre recette.

    > **Note**
    >
    > JOIN, UNION, GROUP_BY, PIVOT, UNPIVOT, TRANSPOSE ne sont pas pris en charge pour l'importation de recettes et ne seront pas non plus disponibles en mode de création de recettes.

11. En option, vous pouvez terminer la création de la tâche en ajoutant d'autres nœuds de transformation selon les besoins et ajouter des nœuds Cible de données.

    Si vous réorganisez les étapes après avoir importé une recette, AWS Glue effectue la validation de ces étapes. Par exemple, si vous renommez puis supprimez une colonne et que vous déplacez l'étape de suppression vers le haut, l'étape de renommage ne sera pas valide. Vous pouvez ensuite modifier les étapes pour corriger l'erreur de validation.

Migration depuis AWS Glue DataBrew AWS Glue Studio

Si vous avez des recettes AWS Glue DataBrew, utilisez la liste de contrôle suivante pour migrer vos recettes vers AWS Glue Studio.

Si vous souhaitez	Procédez comme suit
Permettez aux utilisateurs de récupérer AWS Glue DataBrew des recettes, des versions de recettes et des descriptions de recettes.	Ajouter des autorisations IAM à une politique qui permet à votre rôle d'accéder aux actions

Si vous souhaitez	Procédez comme suit
	nécessaires. Consultez Autorisations IAM pour AWS Glue DataBrew.
Importez une AWS Glue DataBrew recette existante dans AWS Glue Studio.	Suivez les étapes de Importer une AWS Glue DataBrew recette.
Importer une recette avec JOIN et UNION.	Les recettes comportant des transformations UNION et JOIN ne sont pas prises en charge. Utilisez les transformations Join et Union dans AWS Glue Studio avant ou après un nœud Data Preparation Recipe.

Utilisation de Modifier le schéma pour remapper les clés de propriétés de données

Une transformation Modifier le schéma remet les clés de propriété de données source dans la configuration souhaitée pour les données cibles. Dans un nœud de transformation Modifier le schéma, vous pouvez :

- Modifiez le nom de plusieurs clés de propriété de données.
- Modifiez le type de données des clés de propriété de données, si le nouveau type de données est pris en charge et qu'il existe un chemin de transformation entre les deux types de données.
- Choisir un sous-ensemble de clés de propriété de données en indiquant les clés de propriété de données que vous souhaitez supprimer.

Vous pouvez aussi ajouter d'autres nœuds Modifier le schéma au schéma de tâches, selon les besoins, par exemple, pour modifier des sources de données supplémentaires ou à la suite d'une transformation de Jointure.

Utilisation de Modifier le schéma avec un type de données décimal

Lorsque vous utilisez la transformation Modifier le schéma avec un type de données décimal, la transformation Modifier le schéma modifie la précision à la valeur par défaut de (10,2). Pour modifier cela et définir la précision adaptée à votre cas d'utilisation, vous pouvez utiliser la transformation Requête SQL et convertir les colonnes avec une précision spécifique.

Par exemple, si vous avez une colonne d'entrée nommée « DecimalCol » de type Décimal et que vous souhaitez la remapper en une colonne de sortie nommée « OutputDecimalCol » avec une précision spécifique de (18,6), vous devez :

1. Ajoutez une transformation Requête SQL ultérieure après la transformation Modifier le schéma.
2. Dans la transformation Requête SQL, utilisez une requête SQL pour convertir la colonne remappée avec la précision souhaitée. La requête SQL se présente comme suit :

   ```
   SELECT col1, col2, CAST(DecimalCol AS DECIMAL(18,6)) AS OutputDecimalCol
   FROM __THIS__
   ```

   Dans la requête SQL ci-dessus :

   - `col1` et `col2` sont d'autres colonnes de vos données que vous souhaitez transmettre sans modification.
   - `DecimalCol` est le nom de colonne d'origine issu des données d'entrée.
   - `CAST (DecimalCol AS DECIMAL (18,6)) `convertit le` DecimalCol `en un type décimal avec une précision de 18 chiffres et 6 décimales.
   - `AS OutputDecimalCol `renomme la colonne castée en` `. OutputDecimalCol

En utilisant la transformation Requête SQL, vous pouvez remplacer la précision par défaut définie par la transformation Modifier le schéma et convertir explicitement les colonnes décimales à la précision souhaitée. Cette approche vous permet de tirer parti de la transformation Modifier le schéma pour renommer et restructurer vos données tout en gérant les exigences de précision pour les colonnes décimales via la transformation Requête SQL ultérieure.

Ajout d'une transformation Modifier le schéma à votre tâche

> **Note**
>
> La transformation Modifier le schéma n'est pas sensible à la casse.

Pour ajouter un nœud de transformation Modifier le schéma à votre diagramme de tâche

1. (Facultatif) Ouvrez le panneau Ressources, puis choisissez Modifier le schéma pour ajouter une nouvelle transformation à votre diagramme de tâches, si nécessaire.

2. Dans le panneau des propriétés du nœud, saisissez un nom pour le nœud dans le diagramme de tâches. Si aucun parent de nœud n'est déjà sélectionné, choisissez un nœud dans la liste de Node parents (Parents de nœud) à utiliser comme source pour la transformation.
3. Choisissez l'onglet Transformer dans le volet de propriétés du nœud.
4. Modifiez le schéma d'entrée :
   - Pour renommer une clé de propriété de données, saisissez le nouveau nom de la clé dans le champ Target key (Clé cible).
   - Pour changer le type de données d'une clé de propriété de données, choisissez le nouveau type de données de la clé dans la liste déroulante Data type (Type de données).
   - Pour supprimer une clé de propriété de données du schéma cible, choisissez la case à cocher Drop (Supprimer) pour cette clé.
5. (Facultatif) Après avoir configuré les propriétés du nœud de transformation, vous pouvez afficher le schéma modifié pour vos données en sélectionnant l'onglet Output Schema (Schéma de sortie) dans le volet de détails du nœud. La première fois que vous choisissez cet onglet pour un nœud de votre tâche, vous êtes invité à fournir un rôle IAM pour accéder aux données. Si vous n'avez pas spécifié de rôle IAM dans le Job détails (Informations de la tâche), vous y êtes invité à ce stade.
6. (Facultatif) Après avoir configuré les propriétés du nœud et les propriétés de transformation, vous pouvez prévisualiser le jeu de données modifié en sélectionnant l'onglet Prévisualisation des données dans le volet de détails du nœud. La première fois que vous choisissez cet onglet pour un nœud de votre tâche, vous êtes invité à fournir un rôle IAM pour accéder aux données. Il y a un coût associé à l'utilisation de cette fonction, et la facturation commence dès que vous fournissez le rôle IAM.

Utilisation de Supprimer les doublons

La transformation Supprimer les doublons supprime des lignes de votre source de données en vous proposant deux options. Vous pouvez choisir de supprimer les lignes dupliquées qui sont complètement identiques, ou vous pouvez choisir les champs à faire correspondre et ne supprimer que les lignes basées sur ces champs.

Par exemple, dans ce jeu de données, vous avez des lignes dupliquées où toutes les valeurs de certaines lignes sont exactement identiques à celles d'une autre ligne, et certaines valeurs des lignes sont identiques ou différentes.

Row	Name	E-mail	Age	State	Remarque
1	Joy	joy@gmail	33	NY	
2	Tim	tim@gmail	45	OH	
3	Rose	rose@gmail	23	NJ	
4	Tim	tim@gmail	42	OH	
5	Rose	rose@gmail	23	NJ	
6	Tim	tim@gmail	42	OH	Il s'agit d'une ligne dupliquée qui correspond parfaitement à toutes les valeurs de la ligne n° 4
7	Rose	rose@gmail	23	NJ	Il s'agit d'une ligne dupliquée qui correspond parfaitement à toutes les valeurs de la ligne n° 5

Si vous choisissez de faire correspondre des lignes entières, les lignes 6 et 7 seront supprimées du jeu de données. Le jeu de données est désormais le suivant :

Row	Name	E-mail	Age	State
1	Joy	joy@gmail	33	NY

Row	Name	E-mail	Age	State
2	Tim	tim@gmail	45	OH
3	Rose	rose@gmail	23	NJ
4	Tim	tim@gmail	42	OH
5	Rose	rose@gmail	23	NJ

Si vous avez choisi de spécifier des clés, vous pouvez choisir de supprimer les lignes correspondant à « name » et « email » Cela vous permet de mieux contrôler ce qu'est une « ligne dupliquée » pour votre jeu de données. En spécifiant « name » et « email », le jeu de données est désormais le suivant :

Row	Name	E-mail	Age	State
1	Joy	joy@gmail	33	NY
2	Tim	tim@gmail	45	OH
3	Rose	rose@gmail	23	NJ

Quelques points à garder à l'esprit :

- Pour que les lignes soient reconnues comme des doublons, les valeurs sont sensibles à la casse. Toutes les valeurs des lignes doivent avoir la même casse. Cela s'applique à l'une ou l'autre des options que vous choisissez (Faire correspondre des lignes entières ou Spécifier des clés).
- Toutes les valeurs sont lues sous forme de chaînes.
- La transformation Supprimer les doublons utilise la commande Spark dropDuplicates.
- Lorsque vous utilisez la transformation Supprimer les doublons, la première ligne est conservée et les autres lignes sont supprimées.
- La transformation Supprimer les doublons ne modifie pas le schéma de la trame de données. Si vous choisissez de spécifier des clés, tous les champs sont conservés dans la trame de données obtenue.

Utilisation SelectFields pour supprimer la plupart des clés de propriété des données

Vous pouvez créer un sous-ensemble de clés de propriétés de données à partir de l'ensemble de données à l'aide de la SelectFieldstransformation. Vous indiquez quelles clés de propriété de données vous souhaitez conserver et les autres sont supprimées du jeu de données.

> **Note**
>
> La SelectFieldstransformation distingue les majuscules et minuscules. À utiliser ApplyMappingsi vous avez besoin d'une méthode qui ne distingue pas les majuscules des minuscules pour sélectionner les champs.

Pour ajouter un nœud de SelectFields transformation à votre diagramme de tâches

1. (Facultatif) Ouvrez le panneau Ressources, puis choisissez SelectFieldsd'ajouter une nouvelle transformation à votre diagramme de tâches, si nécessaire.
2. Sur la page Node properties (Propriétés de nœud) au cours de la tâche, saisissez un nom pour le nœud dans le diagramme de tâche. Si aucun parent de nœud n'est déjà sélectionné, choisissez un nœud dans la liste Node parents (Parents de nœud) à utiliser comme source pour la transformation.
3. Choisissez l'onglet Transformation dans le volet de détails du nœud.
4. Sous le titre SelectFields, choisissez les clés de propriété des données de l'ensemble de données que vous souhaitez conserver. Toutes autres clés de propriété de données non sélectionnées sont supprimées du jeu de données.

   Vous pouvez également cocher la case en regard de l'en-tête de colonne Field (Champ) pour choisir automatiquement toutes les clés de propriété de données dans le jeu de données. Vous pouvez ensuite désélectionner des clés de propriété de données individuelles pour les supprimer du jeu de données.
5. (Facultatif) Après avoir configuré les propriétés du nœud de transformation, vous pouvez afficher le schéma modifié pour vos données en sélectionnant l'onglet Output Schema (Schéma de sortie) dans le volet de détails du nœud. La première fois que vous choisissez cet onglet pour un nœud de votre tâche, vous êtes invité à fournir un rôle IAM pour accéder aux données. Si vous n'avez pas spécifié de rôle IAM dans le Job détails (Informations de la tâche), vous y êtes invité à ce stade.

6. (Facultatif) Après avoir configuré les propriétés du nœud et les propriétés de transformation, vous pouvez prévisualiser le jeu de données modifié en sélectionnant l'onglet Prévisualisation des données dans le volet de détails du nœud. La première fois que vous choisissez cet onglet pour un nœud de votre tâche, vous êtes invité à fournir un rôle IAM pour accéder aux données. Il y a un coût associé à l'utilisation de cette fonction, et la facturation commence dès que vous fournissez le rôle IAM.

Utilisation DropFields pour conserver la plupart des clés de propriété des données

Vous pouvez créer un sous-ensemble de clés de propriétés de données à partir de l'ensemble de données à l'aide de la DropFieldstransformation. Vous indiquez les clés de propriété des données que vous souhaitez supprimer du jeu de données et le reste des clés est conservé.

> **Note**
>
> La DropFieldstransformation distingue les majuscules et minuscules. Utilisez Modifier le schéma si vous avez besoin d'une méthode non sensible à la casse pour sélectionner des champs.

Pour ajouter un nœud de DropFields transformation à votre diagramme de tâches

1. (Facultatif) Ouvrez le panneau Ressources, puis choisissez DropFieldsd'ajouter une nouvelle transformation à votre diagramme de tâches, si nécessaire.
2. Sur la page Node properties (Propriétés de nœud) au cours de la tâche, saisissez un nom pour le nœud dans le diagramme de tâche. Si aucun parent de nœud n'est déjà sélectionné, choisissez un nœud dans la liste Node parents (Parents de nœud) à utiliser comme source pour la transformation.
3. Choisissez l'onglet Transformation dans le volet de détails du nœud.
4. Sous le titre DropFields, choisissez les clés de propriété des données à supprimer de la source de données.

   Vous pouvez également cocher la case en regard de l'en-tête de colonne Field (Champ) pour choisir automatiquement toutes les clés de propriété de données dans le jeu de données. Vous pouvez ensuite désélectionner les clés de propriété de données individuelles afin qu'elles soient conservées dans le jeu de données.

5. (Facultatif) Après avoir configuré les propriétés du nœud de transformation, vous pouvez afficher le schéma modifié pour vos données en sélectionnant l'onglet Output Schema (Schéma de sortie) dans le volet de détails du nœud. La première fois que vous choisissez cet onglet pour un nœud de votre tâche, vous êtes invité à fournir un rôle IAM pour accéder aux données. Si vous n'avez pas spécifié de rôle IAM dans le Job détails (Informations de la tâche), vous y êtes invité à ce stade.
6. (Facultatif) Après avoir configuré les propriétés du nœud et les propriétés de transformation, vous pouvez prévisualiser le jeu de données modifié en sélectionnant l'onglet Prévisualisation des données dans le volet de détails du nœud. La première fois que vous choisissez cet onglet pour un nœud de votre tâche, vous êtes invité à fournir un rôle IAM pour accéder aux données. Il y a un coût associé à l'utilisation de cette fonction, et la facturation commence dès que vous fournissez le rôle IAM.

Renommer un champ dans le jeu de données

Vous pouvez utiliser la RenameFieldtransformation pour modifier le nom d'une clé de propriété individuelle dans l'ensemble de données.

> **Note**
>
> La RenameFieldtransformation distingue les majuscules et minuscules. À utiliser ApplyMappingsi vous avez besoin d'une transformation qui ne distingue pas les majuscules et minuscules.

> **Tip**
>
> Si vous utilisez la transformation Modifier le schéma, vous pouvez renommer plusieurs clés de propriété de données dans le jeu de données avec une seule transformation.

Pour ajouter un nœud de RenameField transformation à votre diagramme de tâches

1. (Facultatif) Ouvrez le panneau Ressources, puis choisissez RenameFieldd'ajouter une nouvelle transformation à votre diagramme de tâches, si nécessaire.
2. Sur la page Node properties (Propriétés de nœud) au cours de la tâche, saisissez un nom pour le nœud dans le diagramme de tâche. Si aucun parent de nœud n'est déjà sélectionné,

choisissez un nœud dans la liste Node parents (Parents de nœud) à utiliser comme source pour la transformation.

3. Choisissez l'onglet Transformation.
4. Sous l'en-tête Data field (Champ de données), choisissez une clé de propriété dans les données source, puis entrez un nouveau nom dans le champ New field name (Nom de champ).
5. (Facultatif) Après avoir configuré les propriétés du nœud de transformation, vous pouvez afficher le schéma modifié pour vos données en sélectionnant l'onglet Output Schema (Schéma de sortie) dans le volet de détails du nœud. La première fois que vous choisissez cet onglet pour un nœud de votre tâche, vous êtes invité à fournir un rôle IAM pour accéder aux données. Si vous n'avez pas spécifié de rôle IAM dans le Job détails (Informations de la tâche), vous y êtes invité à ce stade.
6. (Facultatif) Après avoir configuré les propriétés du nœud et les propriétés de transformation, vous pouvez prévisualiser le jeu de données modifié en sélectionnant l'onglet Prévisualisation des données dans le volet de détails du nœud. La première fois que vous choisissez cet onglet pour un nœud de votre tâche, vous êtes invité à fournir un rôle IAM pour accéder aux données. Il y a un coût associé à l'utilisation de cette fonction, et la facturation commence dès que vous fournissez le rôle IAM.

Utilisation de Spigot pour échantillonner votre jeu de données

Pour tester les transformations effectuées par votre tâche, vous pouvez obtenir un échantillon de données afin de vérifier que la transformation fonctionne comme prévu. La transformation Spigot écrit un sous-ensemble d'enregistrements à partir du jeu de données dans un fichier JSON situé dans un compartiment Amazon S3. La méthode d'échantillonnage des données peut être soit un nombre spécifique d'enregistrements à partir du début du fichier, soit un facteur de probabilité utilisé pour choisir les enregistrements.

Pour ajouter un nœud de transformation Spigot à votre diagramme de tâche

1. (Facultatif) Ouvrez le panneau Ressources, puis choisissez Spigot pour ajouter une nouvelle transformation à votre diagramme de tâches, si nécessaire.
2. Sur la page Node properties (Propriétés de nœud) au cours de la tâche, saisissez un nom pour le nœud dans le diagramme de tâche. Si aucun parent de nœud n'est déjà sélectionné, choisissez un nœud dans la liste Node parents (Parents de nœud) à utiliser comme source pour la transformation.
3. Choisissez l'onglet Transformation dans le volet de détails du nœud.

4. Saisissez un chemin d'accès Amazon S3 ou choisissez Browse S3 (Parcourir S3) pour choisir un emplacement dans Amazon S3. C'est l'emplacement où la tâche écrit le fichier JSON qui contient l'échantillon de données.
5. Saisissez les informations relatives à la méthode d'échantillonnage. Vous pouvez spécifier une valeur pour le Number of records (Nombre d'enregistrements) à écrire en commençant par le début du jeu de données et un Probability threshold (Seuil de probabilité) (saisi comme une valeur décimale avec une valeur maximale de 1) permettant de choisir un enregistrement donné.

   Par exemple, pour écrire les 50 premiers enregistrements du jeu de données, vous devez définir le Number of records (Nombre d'enregistrements) à 50 et le Probability threshold (Seuil de probabilité) à 1 (100%).

Joindre des jeux de données

La transformation Join permet de joindre deux jeux de données en un seul. Vous spécifiez les noms de clé dans le schéma de chaque jeu de données à comparer. Le résultat `DynamicFrame` contient des lignes où les clés répondent à la condition de jointure. Les lignes de chaque jeu de données qui répondent à la condition de jointure sont combinées en une seule ligne dans le résultat en sortie `DynamicFrame`, qui contient toutes les colonnes de l'un ou l'autre des jeux de données.

Pour ajouter un nœud de transformation Join à votre diagramme de tâche

1. Si une seule source de données est disponible, vous devez ajouter un nouveau nœud de source de données au diagramme de tâche.
2. Choisissez l'un des nœuds source pour la jointure. Ouvrez le panneau Ressources, puis choisissez Jointure pour ajouter une nouvelle transformation à votre diagramme de tâches.
3. Sur la page Node properties (Propriétés de nœud) au cours de la tâche, saisissez un nom pour le nœud dans le diagramme de tâche.
4. Dans l'onglet Node properties (Propriétés de nœud), sous l'en-tête Node parents (Parents de nœud), ajoutez un nœud parent de sorte qu'il y ait deux jeux de données fournissant des entrées pour la jointure. Le parent peut être un nœud de source de données ou un nœud de transformation.

> **Note**
>
> Une jointure ne peut avoir que deux nœuds parents.

5. Choisissez l'onglet Transformation.

   Si vous voyez un message indiquant qu'il y a des noms de clés en conflit, vous pouvez :

   - Choisissez Resolve it pour ajouter automatiquement un nœud de ApplyMappingtransformation à votre diagramme de tâches. Le ApplyMapping nœud ajoute un préfixe à toutes les clés de l'ensemble de données portant le même nom qu'une clé de l'autre ensemble de données. Par exemple, si vous utilisez la valeur par défaut **`right`**, toutes les clés du jeu de données de droite ayant le même nom qu'une clé dans le jeu de données de gauche seront renommées en `(right)`*`key name`*.
   - Ajouter manuellement un nœud de transformation plus tôt dans le diagramme de tâches pour supprimer ou renommer les clés en conflit.

6. Choisir le type de jointure dans la liste Join type (Type de jointure).

   - Inner join (Jointure interne ou intérieure) : renvoie une ligne contenant des colonnes des deux jeux de données pour chaque correspondance en fonction de la condition de jointure. Les lignes qui ne satisfont pas à la condition de jointure ne sont pas renvoyées.
   - Left join (Jointure gauche) : toutes les lignes du jeu de données de gauche et seules les lignes du jeu de données de droite qui satisfont à la condition de jointure.
   - Right join (Joindre à droite) : toutes les lignes du jeu de données de droite et uniquement les lignes du jeu de données de gauche qui satisfont à la condition de jointure.
   - Outer join (Jointure externe ou extérieure) : toutes les lignes des deux jeux de données.
   - Left semi join (Semi-jointure gauche) : toutes les lignes du jeu de données de gauche qui ont une correspondance dans le jeu de données de droite en fonction de la condition de jointure.
   - Left anti join (Anti-jointure gauche) : toutes les lignes du jeu de données de gauche qui n'ont pas de correspondance dans le jeu de données de droite en fonction de la condition de jointure.

7. Sur l'onglet Transformation, sous l'en-tête Join conditions (Conditions de jointure), choisissez Add condition (Ajouter une condition). Choisissez une clé de propriété dans chaque jeu de données à comparer. Les clés de propriété sur le côté gauche de l'opérateur de comparaison sont appelées le jeu de données gauche et les clés de propriété à droite sont appelées le jeu de données de droite.

   Pour des conditions de jointure plus complexes, vous pouvez ajouter des clés correspondantes supplémentaires en sélectionnant Add condition (Ajouter une condition) plus d'une fois. Si vous ajoutez accidentellement une condition, vous pouvez choisir l'icône de suppression

(🗑)
pour la supprimer.

8. (Facultatif) Après avoir configuré les propriétés du nœud de transformation, vous pouvez afficher le schéma modifié pour vos données en sélectionnant l'onglet Output Schema (Schéma de sortie) dans le volet de détails du nœud. La première fois que vous choisissez cet onglet pour un nœud de votre tâche, vous êtes invité à fournir un rôle IAM pour accéder aux données. Si vous n'avez pas spécifié de rôle IAM dans le Job détails (Informations de la tâche), vous y êtes invité à ce stade.
9. (Facultatif) Après avoir configuré les propriétés du nœud et les propriétés de transformation, vous pouvez prévisualiser le jeu de données modifié en sélectionnant l'onglet Prévisualisation des données dans le volet de détails du nœud. La première fois que vous choisissez cet onglet pour un nœud de votre tâche, vous êtes invité à fournir un rôle IAM pour accéder aux données. Il y a un coût associé à l'utilisation de cette fonctionnalité, et la facturation commence dès que vous fournissez le rôle IAM.

Pour un exemple de schéma de sortie de jointure, considérez une jointure entre deux jeux de données avec les clés de propriété suivantes :

```
Left: {id, dept, hire_date, salary, employment_status}
Right: {id, first_name, last_name, hire_date, title}
```

La jointure est configurée pour correspondre aux clés `id` et `hire_date` en utilisant l'opérateur de comparaison =.

Parce que les deux jeux de données contiennent les clés `id` et `hire_date`, vous avez choisi Resolve it (Résoudre les problèmes) pour ajouter automatiquement le préfixe **`right`** aux clés du jeu de données approprié.

Les clés dans le schéma de sortie seraient :

```
{id, dept, hire_date, salary, employment_status,
(right)id, first_name, last_name, (right)hire_date, title}
```

Utiliser Union pour combiner des lignes

Vous utilisez le nœud de transformation Union lorsque vous souhaitez combiner des lignes provenant de plusieurs sources de données ayant le même schéma.

Il existe deux types de transformations Union :

1. ALL : lorsque vous appliquez ALL, l'union qui en résulte ne supprime pas les lignes en double.
2. DISTINCT : lorsque vous appliquez DISTINCT, l'union qui en résulte supprime les lignes dupliquées.

Unions vs. jointures

Vous utilisez Union pour combiner des lignes. Vous utilisez Jointure pour combiner des colonnes.

Utilisation de la transformation Union dans le canevas ETL visuel

1. Ajoutez plusieurs sources de données pour effectuer une transformation Union. Pour ajouter une source de données, ouvrez le panneau Ressources, puis sélectionnez la source de données dans l'onglet Sources. Avant d'utiliser la transformation Union, vous devez vous assurer que toutes les sources de données impliquées dans l'union ont le même schéma et la même structure.
2. Lorsque vous souhaitez combiner au moins deux sources de données à l'aide de la transformation Union, créez la transformation Union en l'ajoutant au canevas. Ouvrez le panneau Ressources sur le canevas et recherchez « Union ». Vous pouvez également choisir l'onglet Transformer dans le panneau Ressources et faire défiler la page vers le bas jusqu'à ce que vous trouviez la transformation Union, puis choisir Union.
3. Sélectionnez le nœud Union sur le canevas de la tâche. Dans la fenêtre Propriétés du nœud, choisissez les nœuds parents à connecter à la transformation Union.
4. AWS Glue vérifie la compatibilité pour s'assurer que la transformation Union peut être appliquée à toutes les sources de données. Si le schéma des sources de données est identique, l'opération sera autorisée. Si les sources de données n'ont pas le même schéma, un message d'erreur non valide s'affiche : « Les schémas d'entrée de cette union ne sont pas identiques. Envisagez ApplyMapping de l'utiliser pour faire correspondre les schémas. » Pour résoudre ce problème, choisissez Utiliser ApplyMapping.
5. Choisissez le type d'union.
    1. All : par défaut, le type d'union All est sélectionné ; cela entraînera des lignes dupliquées s'il y en a dans la combinaison de données.
    2. Distinct : choisissez Distinct si vous souhaitez que les lignes dupliquées soient supprimées de la combinaison de données résultante.

Utilisation SplitFields pour diviser un ensemble de données en deux

La SplitFieldstransformation vous permet de choisir certaines des clés de propriété des données dans le jeu de données en entrée et de les placer dans un ensemble de données et les clés non sélectionnées dans un ensemble de données distinct. Le résultat de cette transformation est une collection de `DynamicFrames`.

> **Note**
>
> Vous devez utiliser une SelectFromCollectiontransformation pour convertir la collection de `DynamicFrames` en une seule `DynamicFrame` avant de pouvoir envoyer la sortie vers un emplacement cible.

La SplitFieldstransformation distingue les majuscules et minuscules. Ajoutez une ApplyMappingtransformation en tant que nœud parent si vous avez besoin de noms de clés de propriété qui ne distinguent pas les majuscules et minuscules.

Pour ajouter un nœud de SplitFields transformation à votre diagramme de tâches

1. (Facultatif) Ouvrez le panneau Ressources, puis choisissez SplitFieldsd'ajouter une nouvelle transformation à votre diagramme de tâches, si nécessaire.
2. Sur la page Node properties (Propriétés de nœud) au cours de la tâche, saisissez un nom pour le nœud dans le diagramme de tâche. Si aucun parent de nœud n'est déjà sélectionné, choisissez un nœud dans la liste Node parents (Parents de nœud) à utiliser comme source pour la transformation.
3. Choisissez l'onglet Transformation.
4. Choisissez les clés de propriété que vous souhaitez placer dans le premier jeu de données. Les clés que vous ne choisissez pas sont placées dans le deuxième jeu de données.
5. (Facultatif) Après avoir configuré les propriétés du nœud de transformation, vous pouvez afficher le schéma modifié pour vos données en sélectionnant l'onglet Output Schema (Schéma de sortie) dans le volet de détails du nœud. La première fois que vous choisissez cet onglet pour un nœud de votre tâche, vous êtes invité à fournir un rôle IAM pour accéder aux données. Si vous n'avez pas spécifié de rôle IAM dans le Job détails (Informations de la tâche), vous y êtes invité à ce stade.
6. (Facultatif) Après avoir configuré les propriétés du nœud et les propriétés de transformation, vous pouvez prévisualiser le jeu de données modifié en sélectionnant l'onglet Prévisualisation

des données dans le volet de détails du nœud. La première fois que vous choisissez cet onglet pour un nœud de votre tâche, vous êtes invité à fournir un rôle IAM pour accéder aux données. Il y a un coût associé à l'utilisation de cette fonction, et la facturation commence dès que vous fournissez le rôle IAM.

7. Configurez un nœud de SelectFromCollectiontransformation pour traiter les ensembles de données résultants.

Présentation d'un transformation SelectFromCollection

Certaines transformations ont plusieurs ensembles de données en sortie au lieu d'un seul ensemble de données, SplitFieldspar exemple. La SelectFromCollectiontransformation sélectionne un ensemble de données (`DynamicFrame`) parmi une collection d'ensembles de données (un tableau de`DynamicFrames`). Le résultat de la transformation est la `DynamicFrame`.

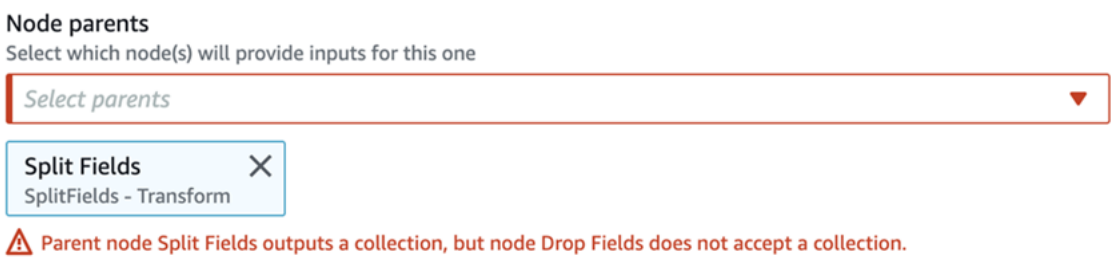
Vous devez utiliser cette transformation après avoir utilisé une transformation qui crée une collection de `DynamicFrames`, tels que :

- Transformations de code personnalisé
- SplitFields

Si vous n'ajoutez aucun nœud de SelectFromCollectiontransformation à votre diagramme de tâches après l'une de ces transformations, vous obtiendrez une erreur pour votre tâche.

Le nœud parent de cette transformation doit être un nœud qui renvoie une collection de `DynamicFrames`. Si vous choisissez un parent pour ce nœud de transformation qui renvoie un seul `DynamicFrame`, comme Join, votre tâche renvoie une erreur.

De même, si vous utilisez un SelectFromCollectionnœud dans votre diagramme de tâches comme parent pour une transformation qui attend un nœud unique `DynamicFrame` en entrée, votre tâche renvoie une erreur.



Utilisation SelectFromCollection pour choisir le jeu de données à conserver

Utilisez la SelectFromCollectiontransformation pour convertir une collection de `DynamicFrames` en une seule`DynamicFrame`.

Pour ajouter un nœud de SelectFromCollection transformation à votre diagramme de tâches

1. (Facultatif) Ouvrez le panneau Ressources, puis choisissez SelectFromCollectiond'ajouter une nouvelle transformation à votre diagramme de tâches, si nécessaire.
2. Sur la page Node properties (Propriétés de nœud) au cours de la tâche, saisissez un nom pour le nœud dans le diagramme de tâche. Si aucun parent de nœud n'est déjà sélectionné, choisissez un nœud dans la liste Node parents (Parents de nœud) à utiliser comme source pour la transformation.
3. Choisissez l'onglet Transformation.
4. Sous l'en-tête Frame index (Index de l'image), choisissez le numéro d'index du tableau qui correspond au `DynamicFrame` que vous souhaitez sélectionner dans la collection de `DynamicFrames`.

   Par exemple, si le nœud parent de cette transformation est une SplitFieldstransformation, dans l'onglet Schéma de sortie de ce nœud, vous pouvez voir le schéma de chaque transformation`DynamicFrame`. Si vous souhaitez conserver le `DynamicFrame` associé au schéma de Sortie 2, vous pouvez choisir **1** pour la valeur de Frame index (Index de l'image), qui est la deuxième valeur de la liste.

   Seul le `DynamicFrame` que vous choisissez est inclus dans le résultat.
5. (Facultatif) Après avoir configuré les propriétés du nœud de transformation, vous pouvez afficher le schéma modifié pour vos données en sélectionnant l'onglet Output Schema (Schéma de sortie) dans le volet de détails du nœud. La première fois que vous choisissez cet onglet pour un nœud de votre tâche, vous êtes invité à fournir un rôle IAM pour accéder aux données. Si vous n'avez pas spécifié de rôle IAM dans le Job détails (Informations de la tâche), vous y êtes invité à ce stade.
6. (Facultatif) Après avoir configuré les propriétés du nœud et les propriétés de transformation, vous pouvez prévisualiser le jeu de données modifié en sélectionnant l'onglet Prévisualisation des données dans le volet de détails du nœud. La première fois que vous choisissez cet onglet pour un nœud de votre tâche, vous êtes invité à fournir un rôle IAM pour accéder aux données. Il y a un coût associé à l'utilisation de cette fonction, et la facturation commence dès que vous fournissez le rôle IAM.

Rechercher et remplir les valeurs manquantes dans un jeu de données

Vous pouvez utiliser la FillMissingValuestransformation pour localiser les enregistrements de l'ensemble de données qui contiennent des valeurs manquantes et ajouter un nouveau champ dont la valeur est déterminée par imputation. Le jeu de données source est utilisé pour entraîner le modèle de machine learning (ML) qui détermine la valeur manquante. Si vous utilisez des jeux de données incrémentiels, chacun d'entre eux est utilisé comme données d'entraînement pour le modèle ML, de sorte que les résultats peuvent ne pas être aussi précis.

Pour utiliser un nœud de FillMissingValues transformation dans votre diagramme de tâches

1. (Facultatif) Ouvrez le panneau Ressources, puis choisissez FillMissingValuesd'ajouter une nouvelle transformation à votre diagramme de tâches, si nécessaire.
2. Sur la page Node properties (Propriétés de nœud) au cours de la tâche, saisissez un nom pour le nœud dans le diagramme de tâche. Si aucun parent de nœud n'est déjà sélectionné, choisissez un nœud dans la liste de Node parents (Parents de nœud) à utiliser comme source pour la transformation.
3. Choisissez l'onglet Transformation.
4. Pour Data field (Champ de données), choisissez le nom de la colonne ou du champ dans les données source que vous souhaitez analyser pour les valeurs manquantes.
5. (Facultatif) Dans la section New field name (Nouveau nom de champ), saisissez un nom pour le champ ajouté à chaque enregistrement qui contiendra la valeur de remplacement estimée du champ analysé. Si le champ analysé n'a pas de valeur manquante, la valeur du champ analysé est copiée dans le nouveau champ.

   Si vous ne spécifiez pas de nom pour le nouveau champ, le nom par défaut est le nom de la colonne analysée, auquel est ajouté `_filled`. Par exemple, si vous entrez **`Age`** pour Data field (Champ de données) et ne spécifiez pas de valeur pour New field name (Nouveau nom de champ), un nouveau champ nommé **`Age_filled`** est ajouté à chaque enregistrement.
6. (Facultatif) Après avoir configuré les propriétés du nœud de transformation, vous pouvez afficher le schéma modifié pour vos données en sélectionnant l'onglet Output Schema (Schéma de sortie) dans le volet de détails du nœud. La première fois que vous choisissez cet onglet pour un nœud de votre tâche, vous êtes invité à fournir un rôle IAM pour accéder aux données. Si vous n'avez pas spécifié de rôle IAM dans le Job détails (Informations de la tâche), vous y êtes invité à ce stade.
7. (Facultatif) Après avoir configuré les propriétés du nœud et les propriétés de transformation, vous pouvez prévisualiser le jeu de données modifié en sélectionnant l'onglet Prévisualisation

des données dans le volet de détails du nœud. La première fois que vous choisissez cet onglet pour un nœud de votre tâche, vous êtes invité à fournir un rôle IAM pour accéder aux données. Il y a un coût associé à l'utilisation de cette fonction, et la facturation commence dès que vous fournissez le rôle IAM.

Filtrage des clés dans un jeu de données

Utiliser la transformation Filter (Filtrer) pour créer un jeu de données en filtrant les enregistrements du jeu de données source en fonction d'une expression régulière. Les lignes qui ne satisfont pas à la condition de filtre sont retirées du résultat.

- Pour les types de données de chaîne, vous pouvez filtrer les lignes où la valeur clé correspond à une chaîne spécifiée.
- Pour les types de données numériques, vous pouvez filtrer les lignes en comparant la valeur clé à une valeur spécifiée, à l'aide des opérateurs de comparaison <, >, =, !=, <= et >=.

Si vous spécifiez plusieurs conditions de filtrage, les résultats sont combinés à l'aide d'un opérateur `AND` par défaut, mais vous pouvez choisir `OR` à la place.

La transformation Filter (Filtrer) est sensible à la casse. Ajoutez une ApplyMappingtransformation en tant que nœud parent si vous avez besoin de noms de clés de propriété qui ne distinguent pas les majuscules et minuscules.

Pour ajouter un nœud de transformation de filtre à votre diagramme de tâche

1. (Facultatif) Ouvrez le panneau Ressources, puis choisissez Filtrer pour ajouter une nouvelle transformation à votre diagramme de tâches, si nécessaire.
2. Sur la page Node properties (Propriétés de nœud) au cours de la tâche, saisissez un nom pour le nœud dans le diagramme de tâche. Si aucun parent de nœud n'est déjà sélectionné, choisissez un nœud dans la liste Node parents (Parent de nœud) à utiliser comme source pour la transformation.
3. Choisissez l'onglet Transformation.
4. Choisissez soit ET global, soit OU global. Cela détermine la façon dont plusieurs conditions de filtrage sont combinées. Toutes les conditions sont combinées en utilisant les opérations `AND` ou `OR`. Si vous n'avez qu'une seule condition de filtre, vous pouvez choisir l'une ou l'autre.
5. Cliquez sur le bouton Add condition (Ajouter une condition) dans la section Filter condition (Condition de filtrage) pour ajouter une condition de filtre.

Dans le champ Key (Clé), choisissez un nom de clé de propriété dans le jeu de données. Dans le champ Opération, choisissez l'opérateur de comparaison. Dans le champ Valeur, saisissez la valeur de comparaison. Voici quelques exemples de conditions de filtrage :

- `year >= 2018`
- `State matches 'CA*'`

Lorsque vous filtrez des valeurs de chaîne, assurez-vous que la valeur de comparaison utilise un format d'expression régulière qui correspond au langage de script sélectionné dans les propriétés de la tâche (Python ou Scala).

6. Ajoutez des conditions de filtrage supplémentaires, si nécessaire.
7. (Facultatif) Après avoir configuré les propriétés du nœud de transformation, vous pouvez afficher le schéma modifié pour vos données en sélectionnant l'onglet Output Schema (Schéma de sortie) dans le volet de détails du nœud. La première fois que vous choisissez cet onglet pour un nœud de votre tâche, vous êtes invité à fournir un rôle IAM pour accéder aux données. Si vous n'avez pas spécifié de rôle IAM dans le Job détails (Informations de la tâche), vous y êtes invité à ce stade.
8. (Facultatif) Après avoir configuré les propriétés du nœud et les propriétés de transformation, vous pouvez prévisualiser le jeu de données modifié en sélectionnant l'onglet Prévisualisation des données dans le volet de détails du nœud. La première fois que vous choisissez cet onglet pour un nœud de votre tâche, vous êtes invité à fournir un rôle IAM pour accéder aux données. Il y a un coût associé à l'utilisation de cette fonction, et la facturation commence dès que vous fournissez le rôle IAM.

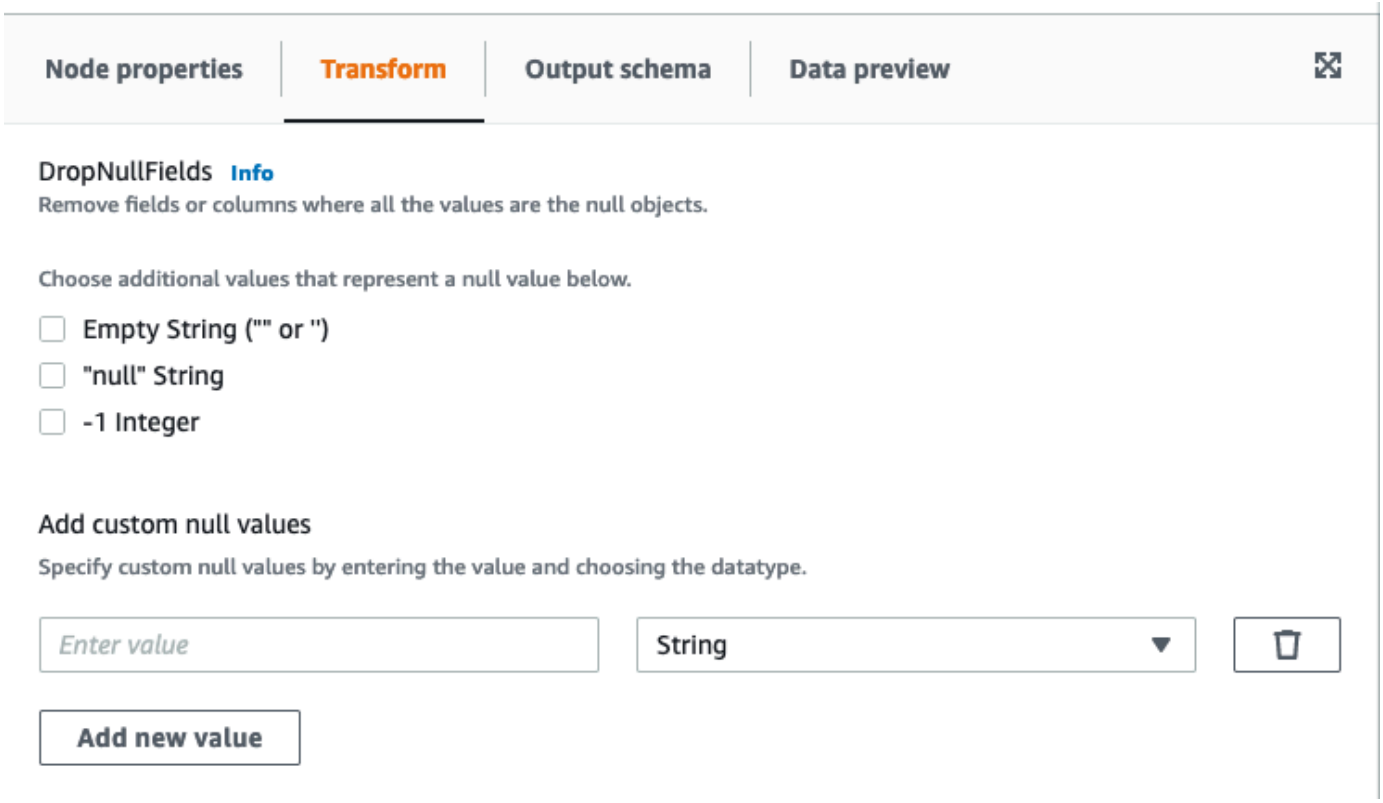
Utilisation DropNullFields pour supprimer des champs contenant des valeurs nulles

Utilisez la DropNullFieldstransformation pour supprimer des champs de l'ensemble de données si toutes les valeurs du champ sont « nulles ». Par défaut, AWS Glue Studio reconnaîtra les objets null, mais certaines valeurs telles que des chaînes vides, des chaînes « null », des entiers -1 ou d'autres valeurs de remplacement tels que des zéros ne sont pas automatiquement reconnues comme null.

Pour utiliser le DropNullFields

1. Ajoutez un DropNullFields nœud au diagramme des tâches.

2. Dans la page Node properties (Propriétés de nœud), choisissez des valeurs supplémentaires qui représentent une valeur null. Vous pouvez choisir de sélectionner aucune valeur ou la totalité des valeurs :



   - Chaîne vide («» ou '') – les champs contenant des chaînes vides seront supprimés
   - « Chaîne null » – les champs contenant la chaîne avec le mot « null » seront supprimés
   - Entier -1 – les champs contenant un entier -1 (un négatif) seront supprimés
3. Au besoin, vous pouvez également spécifier des valeurs null personnalisées. Il s'agit de valeurs null qui peuvent être uniques à votre jeu de données. Pour ajouter une valeur null personnalisée, choisissez Add new value (Ajouter une nouvelle valeur).
4. Entrez la valeur null personnalisée. Par exemple, il peut s'agir de zéro ou de toute valeur utilisée pour représenter une valeur null dans le jeu de données.
5. Choisissez le type de données dans le champ déroulant. Les types de données peuvent être soit des chaînes, soit des nombres entiers.

> **Note**
>
> Les valeurs null personnalisées et leurs types de données doivent correspondre exactement pour que les champs soient reconnus comme des valeurs null et que

les champs soient supprimés. Les correspondances partielles où seule la valeur null personnalisée correspond, mais pas le type de données, n'entraîneront pas la suppression des champs.

Utilisation d'une requête SQL pour transformer des données

Vous pouvez utiliser une transformation SQL pour écrire votre propre transformation sous la forme d'une requête SQL.

Un nœud de transformation SQL peut avoir plusieurs jeux de données source, mais ne produit qu'un seul jeu de données en sortie. Dans contient un champ de texte, dans lequel vous saisir la requête Apache SparkSQL. Vous pouvez attribuer des alias à chaque jeu de données utilisé en entrée, pour contribuer à simplifier la requête SQL. Pour plus d'informations sur la syntaxe SQL, veuillez consulter la documentation Spark SQL.

> **Note**
>
> Si vous utilisez une transformation SQL Spark avec une source de données située dans un VPC, ajoutez un point de terminaison d'un VPC AWS Glue au VPC qui contient la source de données. Pour plus d'informations sur la configuration des points de terminaison de développement, veuillez consulter les rubriques Ajout d'un point de terminaison de développement, Configuration de votre environnement pour les points de terminaison de développement, et Accès à votre point de terminaison de développement dans le Guide du développeur AWS Glue .

Pour utiliser un nœud de transformation SQL dans votre diagramme de tâche

1. (Facultatif) Ajoutez un nœud de transformation au diagramme de tâche, si nécessaire. Choisissez Requête SQL comme type de nœud.

   > **Note**
   >
   > Si vous utilisez une session d'aperçu des données et un code SQL personnalisé ou un nœud de code personnalisé, la session d'aperçu des données exécutera le code SQL ou le bloc de code tel quel pour tout le jeu de données.

2. Sur la page Node properties (Propriétés de nœud) au cours de la tâche, saisissez un nom pour le nœud dans le diagramme de tâche. Si un parent de nœud n'est pas déjà sélectionné ou si vous souhaitez plusieurs entrées pour la transformation SQL, choisissez un nœud dans la liste Node parents (Parents de nœud) à utiliser comme source pour la transformation. Ajoutez d'autres nœuds parents si nécessaire.
3. Choisissez l'onglet Transformation dans le volet de détails du nœud.
4. Les jeux de données source de la requête SQL sont identifiés par les noms que vous avez spécifiés dans le champ Name (Nom) de chaque nœud. Si vous ne souhaitez pas utiliser ces noms, ou si les noms ne conviennent pas à une requête SQL, vous pouvez associer un nom à chaque jeu de données. La console fournit des alias par défaut, tels que `MyDataSource`.

   Par exemple, si un nœud parent du nœud de transformation SQL est nommé `Rename Org PK field`, vous pouvez associer le nom `org_table` à ce jeu de données. Cet alias peut ensuite être utilisé dans la requête SQL à la place du nom du nœud.
5. Dans le champ de saisie de texte sous l'en-tête Bloc de code, collez ou saisissez la requête SQL. Le champ de texte affiche la mise en évidence de la syntaxe SQL et des suggestions de mots-clés.
6. Lorsque le nœud de transformation SQL est sélectionné, choisissez l'option Schema (Schéma), puis choisissez Edit (Modifier). Indiquez les colonnes et les types de données qui décrivent les champs de sortie de la requête SQL.

   Spécifiez le schéma à l'aide des actions suivantes dans la section Output Schema (Schéma de sortie) de la page :

   - Pour renommer une colonne, placez le curseur dans la zone Key (Clé) pour la colonne (également appelée field (champ) ou property key (clé de propriété)) et entrez le nouveau nom.
   - Pour modifier le type de données d'une colonne, sélectionnez le nouveau type de données de la colonne dans la liste déroulante.
   - Pour ajouter une nouvelle colonne de niveau supérieur au schéma, choisissez le bouton Overflow (Surcharger) (⋯), puis choisissez Add root key (Ajouter une clé racine). De nouvelles colonnes sont ajoutées en haut du schéma.

- Pour supprimer une colonne du schéma, choisissez l'icône de suppression
  ( )
  à l'extrême droite du nom de la clé.

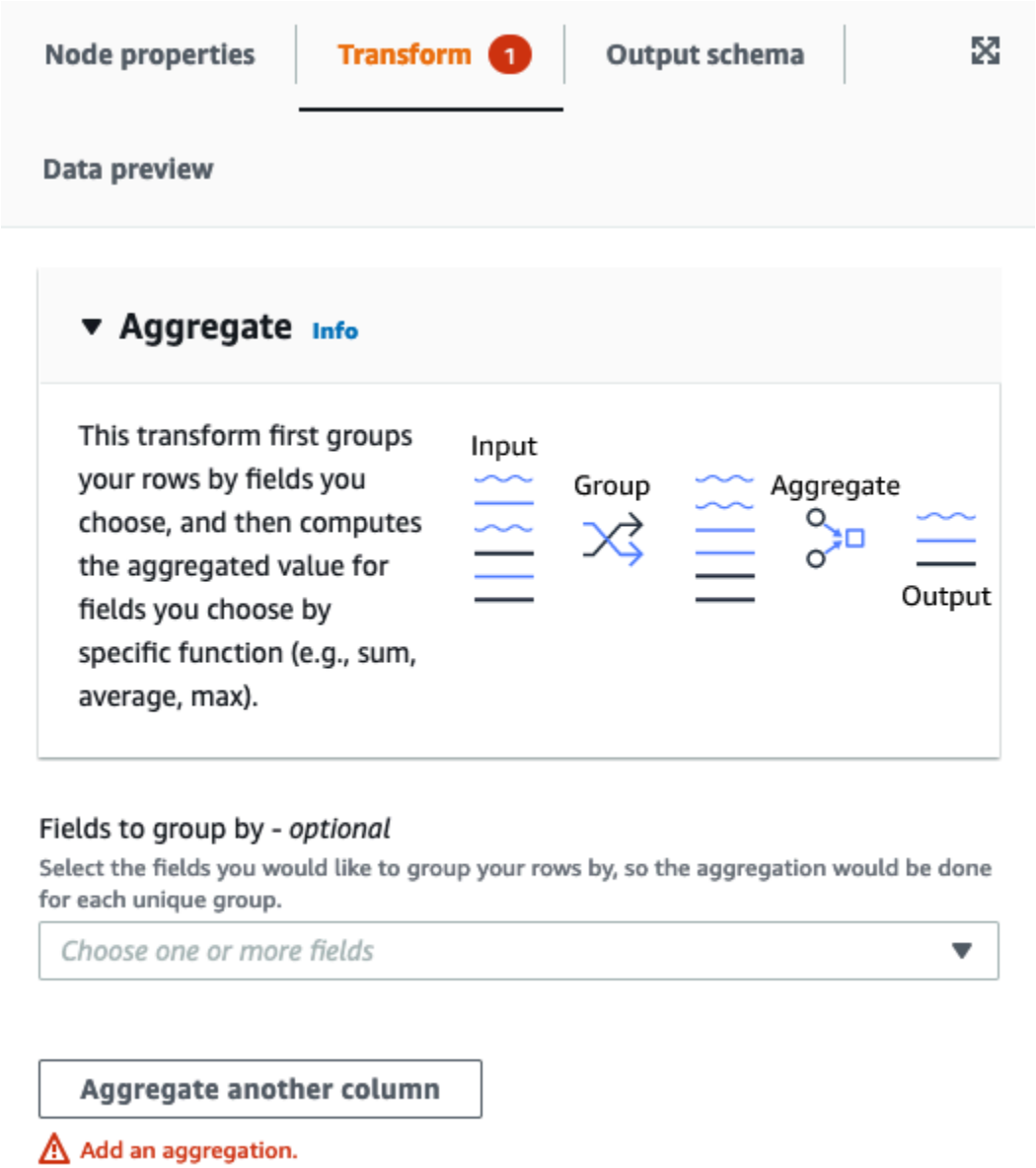
7. Lorsque vous avez terminé de spécifier le schéma en sortie, choisissez Apply (Appliquer) pour enregistrer vos modifications et quittez l'éditeur de schéma. Si vous ne souhaitez pas enregistrer de modifications, choisissez Cancel (Annuler) pour quitter l'éditeur de schéma.
8. (Facultatif) Après avoir configuré les propriétés du nœud et les propriétés de transformation, vous pouvez prévisualiser le jeu de données modifié en sélectionnant l'onglet Prévisualisation des données dans le volet de détails du nœud. La première fois que vous choisissez cet onglet pour un nœud de votre tâche, vous êtes invité à fournir un rôle IAM pour accéder aux données. Il y a un coût associé à l'utilisation de cette fonction, et la facturation commence dès que vous fournissez le rôle IAM.

Utilisation de Agrégation pour effectuer des calculs récapitulatifs sur des champs sélectionnés

Pour utiliser Aggregate transform (transformation d'agrégation)

1. Ajoutez le Aggregate node (nœud d'agrégation) au diagramme de tâches.
2. Dans l'onglet Node properties (Propriétés de nœud), choisissez les champs à regrouper en sélectionnant le champ déroulant (facultatif). Vous pouvez sélectionner plusieurs champs à la fois ou rechercher un nom de champ en saisissant dans la barre de recherche.

   Lorsque les champs sont sélectionnés, le nom et le type de données sont affichés. Pour supprimer un champ, choisissez « X » sur le champ.



3. Choisissez Aggregate another column (Agréger une autre colonne). Il est nécessaire de sélectionner au moins un champ.

4. Choisissez un champ dans le champ déroulant Field to aggregate (Champ à agréger).

5. Choisissez la fonction d'agrégation à appliquer au champ choisi :

   - avg – calcule la moyenne
   - countDistinct – calcule le nombre de valeurs uniques non null
   - count – calcule le nombre de valeurs non null
   - first – renvoie la première valeur qui répond aux critères de « groupe par »
   - last – renvoie la dernière valeur qui répond aux critères de « groupe par »
   - kurtosis – calcule la netteté du pic d'une courbe de distribution de fréquences
   - max – renvoie la valeur la plus élevée qui répond aux critères de « groupe par »
   - min – renvoie la valeur la plus basse qui répond aux critères de « groupe par »
   - asymétrie – mesure de l'asymétrie de la distribution de probabilité d'une distribution normale
   - stddev_pop – calcule l'écart type de population et renvoie la racine carrée de la variance de la population
   - sum – la somme de toutes les valeurs dans le groupe
   - sumDistinct – la somme des valeurs distinctes dans le groupe
   - var_samp – la variance d'échantillon du groupe (ignore les valeurs nulles)
   - var_pop – la variance de la population du groupe (ignore les valeurs nulles)

Aplatissement de structs imbriqués

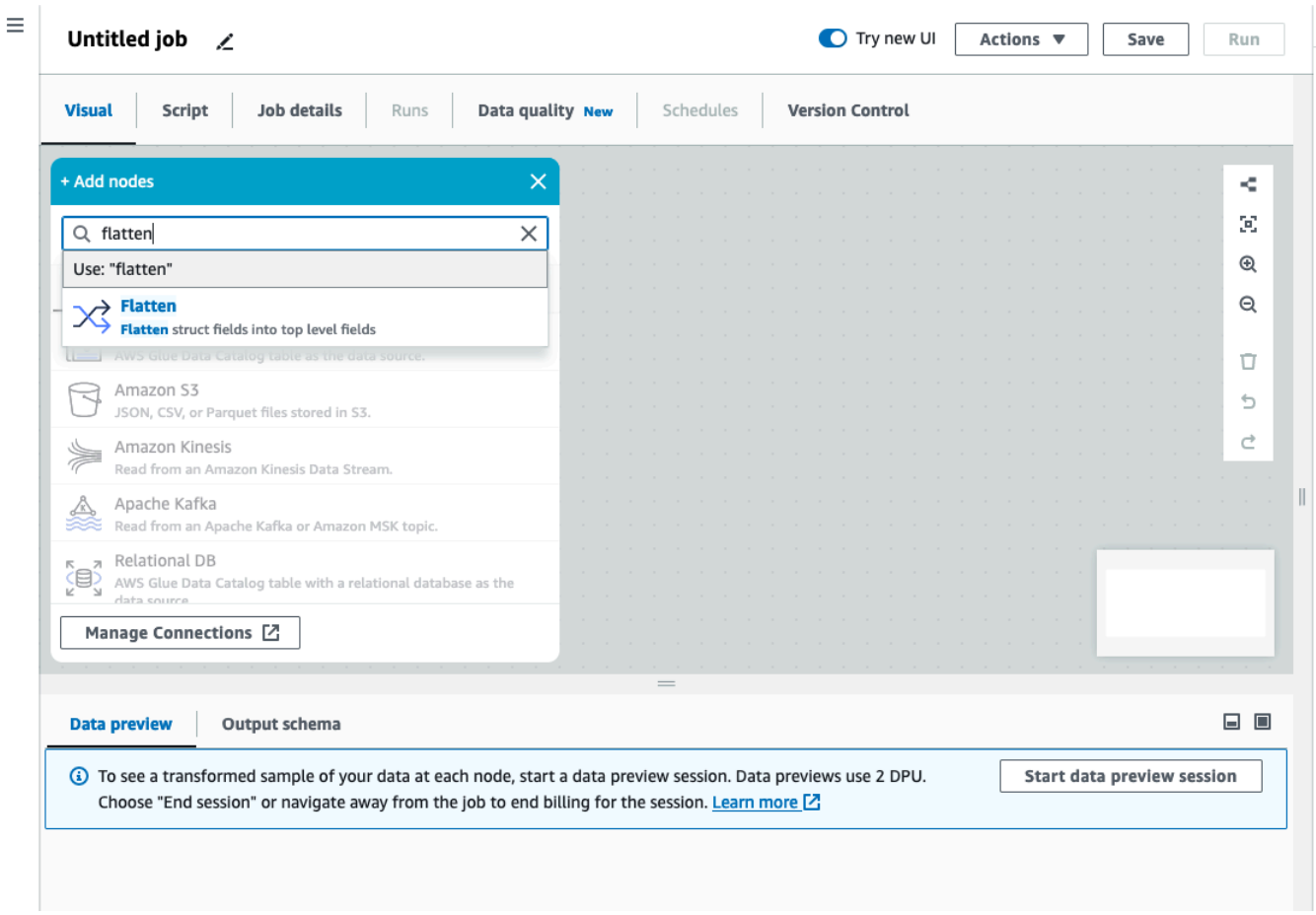
Aplatissez les champs des structs imbriqués dans les données afin qu'ils deviennent des champs de niveau supérieur. Les nouveaux champs sont nommés en utilisant le nom du champ préfixé par les noms des champs de struct pour y accéder, séparés par des points.

Par exemple, si les données contiennent un champ de type Struct nommé « phone_numbers », celui-ci contient, entre autres, un champ de type « Struct » nommé « home_phone » avec deux champs : « country_code » et « number ». Une fois aplatis, ces deux champs deviendront des champs de niveau supérieur nommés respectivement : « phone_numbers.home_phone.country_code » et « phone_numbers.home_phone.number ».

Ajouter un nœud de transformation Aplatir à votre diagramme de tâche

1. Ouvrez le panneau Ressources, puis choisissez l'onglet Transformer, puis Aplatir pour ajouter une nouvelle transformation à votre diagramme de tâches. Vous pouvez également utiliser la

barre de recherche en saisissant « Aplatir », puis en cliquant sur le nœud Aplatir. Le nœud sélectionné au moment de l'ajout du nœud sera son parent.



2. (Facultatif) Dans l'onglet Propriétés de nœud, vous pouvez saisir un nom pour le nœud dans le diagramme de tâches. Si aucun parent de nœud n'est déjà sélectionné, choisissez un nœud dans la liste Node parents (Parents de nœud) à utiliser comme source pour la transformation.
3. (Facultatif) Dans l'onglet Transformation, vous pouvez limiter le niveau d'imbrication maximal à aplatir. Par exemple, définir cette valeur sur 1 signifie que seuls les structs de niveau supérieur seront aplatis. Le réglage de la valeur maximale sur 2 aplatira le niveau supérieur et les structs situés directement en dessous.

Ajouter une colonne UUID

Lorsque vous ajoutez une colonne UUID (identifiant unique universel), une chaîne unique de 36 caractères est attribuée à chaque ligne.

Pour ajouter un nœud de transformation UUID à votre diagramme de tâche

1. Ouvrez le panneau Ressources, puis choisissez UUID pour ajouter une nouvelle transformation à votre diagramme de tâches. Le nœud sélectionné au moment de l'ajout du nœud sera son parent.
2. (Facultatif) Dans l'onglet Propriétés de nœud, vous pouvez saisir un nom pour le nœud dans le diagramme de tâches. Si aucun parent de nœud n'est déjà sélectionné, choisissez un nœud dans la liste Node parents (Parents de nœud) à utiliser comme source pour la transformation.
3. (Facultatif) Dans l'onglet Transformation, vous pouvez personnaliser le nom de la nouvelle colonne. Par défaut, son nom sera « uuid ».

Ajouter une colonne d'identifiant

Attribuez un identifiant numérique à chaque ligne du jeu de données.

Ajouter un nœud de transformation d'identifiant à votre diagramme de tâches

1. Ouvrez le panneau Ressources, puis choisissez Identifiant pour ajouter une nouvelle transformation à votre diagramme de tâches. Le nœud sélectionné au moment de l'ajout du nœud sera son parent.
2. (Facultatif) Dans l'onglet Propriétés de nœud, vous pouvez saisir un nom pour le nœud dans le diagramme de tâches. Si aucun parent de nœud n'est déjà sélectionné, choisissez un nœud dans la liste Node parents (Parents de nœud) à utiliser comme source pour la transformation.
3. (Facultatif) Dans l'onglet Transformation, vous pouvez personnaliser le nom de la nouvelle colonne. Par défaut, son nom sera « id ».
4. (Facultatif) Si la tâche traite et stocke les données de manière incrémentielle, vous devez éviter de réutiliser les mêmes ID entre les exécutions de tâches.

   Dans l'onglet Transformation, cochez la case unique. Cela inclura l'horodatage de la tâche dans l'identifiant, le rendant ainsi unique entre plusieurs exécutions. Pour tenir compte d'un nombre plus élevé, la colonne sera décimale plutôt que de type LONG.

Convertir une colonne en type d'horodatage

Vous pouvez utiliser la transformation En horodatage pour modifier le type de données d'une colonne numérique ou de chaîne en horodatage, afin qu'elle puisse être stockée avec ce type de données ou appliquée à d'autres transformations nécessitant un horodatage.

Ajouter un nœud de transformation En horodatage à votre diagramme de tâches

1. Ouvrez le panneau Ressources, puis choisissez En horodatage pour ajouter une nouvelle transformation à votre diagramme de tâches. Le nœud sélectionné au moment de l'ajout du nœud sera son parent.
2. (Facultatif) Dans l'onglet Propriétés de nœud, vous pouvez saisir un nom pour le nœud dans le diagramme de tâches. Si aucun parent de nœud n'est déjà sélectionné, choisissez un nœud dans la liste Node parents (Parents de nœud) à utiliser comme source pour la transformation.
3. Dans l'onglet Transformation, saisissez le nom de la colonne à convertir.
4. Dans l'onglet Transformation, définissez le mode d'analyse de la colonne sélectionnée en choisissant le type.

   Si la valeur est un nombre, elle peut être exprimée en secondes (horodatage Unix/Python), en millisecondes ou en microsecondes. Choisissez l'option correspondante.

   Si la valeur est une chaîne formatée, choisissez le type « iso ». La chaîne doit être conforme à l'une des variantes du format ISO, par exemple : « 2022-11-02T14:40:59.915Z ».

   Si vous ne connaissez pas le type à ce stade, ou si différentes lignes utilisent des types différents, vous pouvez choisir « détection automatique » et le système fera sa meilleure estimation, avec un faible coût en termes de performances.
5. (Facultatif) Dans l'onglet Transformation, au lieu de convertir la colonne sélectionnée, vous pouvez en créer une nouvelle et conserver l'originale en saisissant un nom pour la nouvelle.

Convertir une colonne d'horodatage en chaîne formatée

Formatez une colonne d'horodatage en chaîne basée sur un modèle. Vous pouvez utiliser Formater l'horodatage pour obtenir la date et l'heure sous forme de chaîne au format souhaité. Vous pouvez définir le format à l'aide de la syntaxe de date Spark ainsi que de la plupart des codes de date Python.

Par exemple, si vous souhaitez que votre chaîne de date soit formatée comme « 2023-01-01 00:00 », vous pouvez définir ce format en utilisant la syntaxe Spark « yyyy-MM-dd HH:mm » ou les codes de date Python équivalents tels que « %Y-%m-%d %H : %M »

Ajouter un nœud de transformation Formater l'horodatage à votre diagramme de tâches

1. Ouvrez le panneau Ressources, puis choisissez Formater l'horodatage pour ajouter une nouvelle transformation à votre diagramme de tâches. Le nœud sélectionné au moment de l'ajout du nœud sera son parent.
2. (Facultatif) Dans l'onglet Propriétés de nœud, vous pouvez saisir un nom pour le nœud dans le diagramme de tâches. Si aucun parent de nœud n'est déjà sélectionné, choisissez un nœud dans la liste Node parents (Parents de nœud) à utiliser comme source pour la transformation.
3. Dans l'onglet Transformation, saisissez le nom de la colonne à convertir.
4. Dans l'onglet Transformation, saisissez le modèle de Format d'horodatage à utiliser, exprimé à l'aide de la syntaxe de date Spark ou de codes de date Python.
5. (Facultatif) Dans l'onglet Transformation, au lieu de convertir la colonne sélectionnée, vous pouvez en créer une nouvelle et conserver l'originale en saisissant un nom pour la nouvelle.

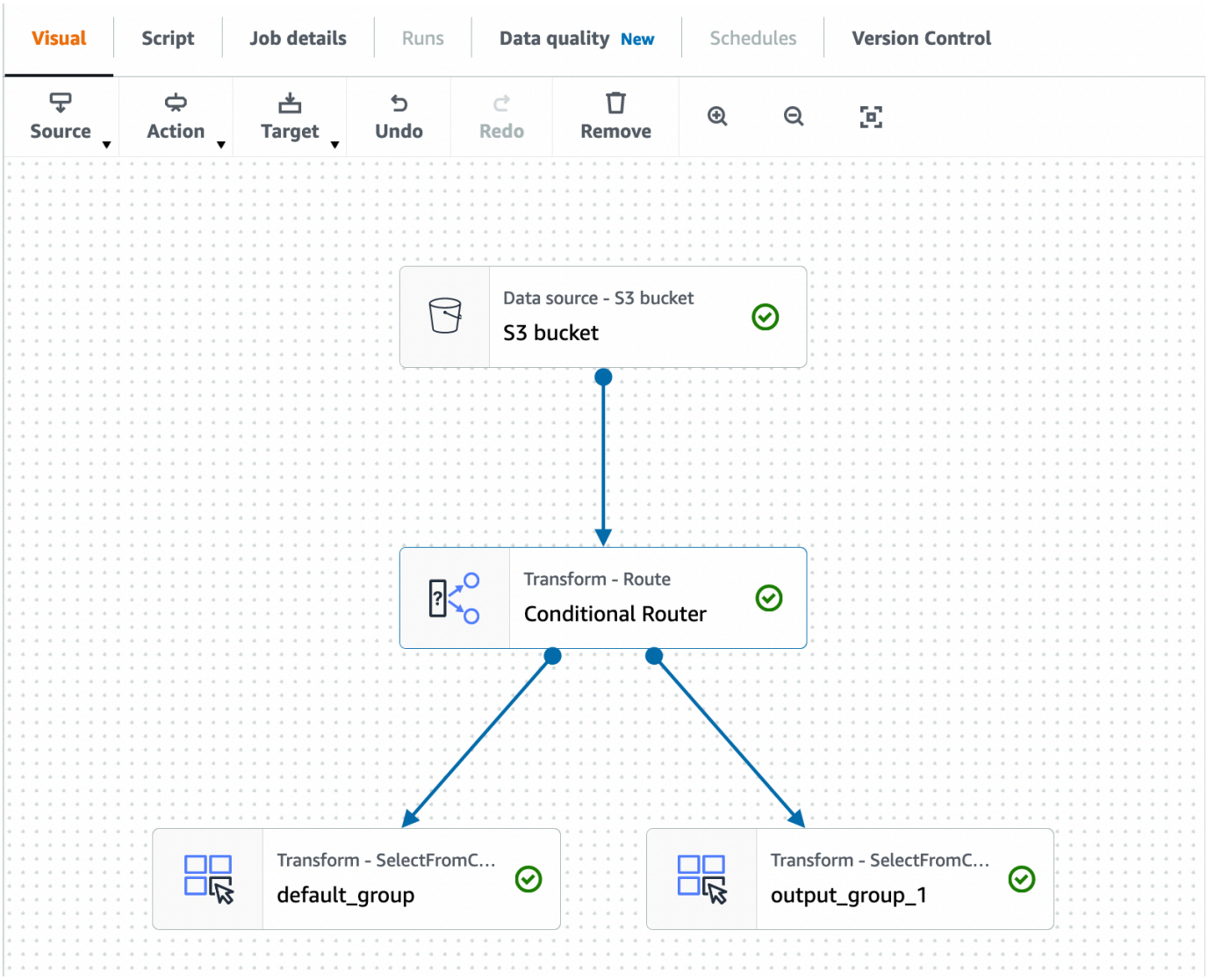
Création d'une transformation du routeur conditionnel

La transformation du routeur conditionnel vous permet d'appliquer plusieurs conditions aux données entrantes. Chaque ligne des données entrantes est évaluée par une condition de filtre de groupe et traitée dans le groupe correspondant. Si une ligne répond à plusieurs conditions de filtre de groupe, la transformation transmet la ligne à plusieurs groupes. Si une ligne ne répond à aucune condition, elle peut être supprimée ou acheminée vers un groupe de sortie par défaut.

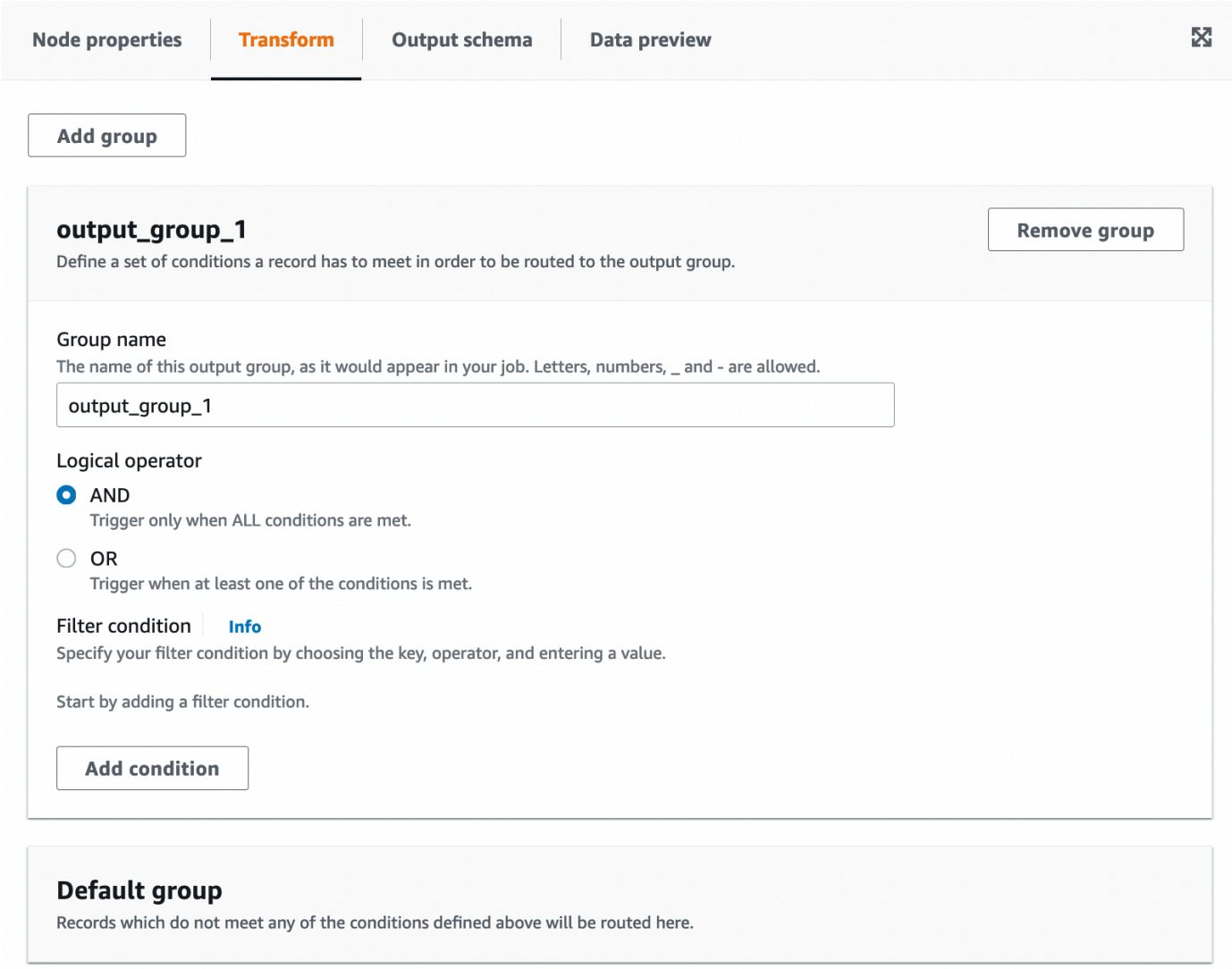
Cette transformation est similaire à la transformation par filtre, mais elle est utile pour les utilisateurs qui souhaitent tester les mêmes données d'entrée sous plusieurs conditions.

Pour ajouter une transformation de routeur conditionnel :

1. Choisissez un nœud sur lequel vous allez effectuer la transformation du routeur conditionnel. Il peut s'agir d'un nœud source ou d'une autre transformation.
2. Choisissez Action, puis utilisez la barre de recherche pour rechercher et choisir « Routeur conditionnel ». Une transformation de Routeur conditionnel est ajoutée avec deux nœuds de sortie. Un nœud de sortie, « Groupe par défaut », contient des enregistrements qui ne répondent à aucune des conditions définies dans les autres nœuds de sortie. Le groupe par défaut ne peut pas être modifié.

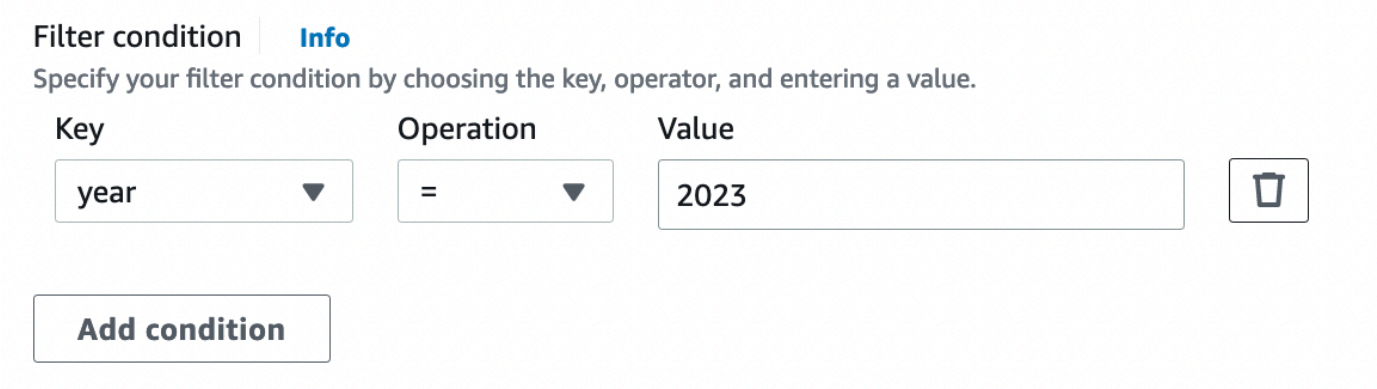


Vous pouvez ajouter des groupes de sortie supplémentaires en choisissant Ajouter un groupe. Pour chaque groupe de sortie, vous pouvez nommer le groupe et ajouter des conditions de filtre et un opérateur logique.



3. Renommez le groupe de sortie en saisissant un nouveau nom. AWS Glue Studio nommera automatiquement vos groupes pour vous (par exemple, « output_group_1 »).
4. Choisissez un opérateur logique (ET, OU) et ajoutez une Condition de filtre en spécifiant la Clé, l'Opération et la Valeur. Les opérateurs logiques vous permettent d'implémenter plusieurs conditions de filtre et d'exécuter l'opérateur logique sur chaque condition de filtre que vous spécifiez.

   Lorsque vous spécifiez la clé, vous pouvez choisir parmi les clés disponibles dans votre schéma. Vous pouvez ensuite choisir l'opération disponible en fonction du type de clé que vous avez sélectionné. Par exemple, si le type de clé est « chaîne », l'opération disponible à choisir est « correspondances ».



5. Dans le champ Valeur, saisissez la valeur. Choisissez Ajouter une condition pour ajouter des conditions de filtre supplémentaires. Pour supprimer les conditions de filtre, sélectionnez l'icône corbeille.

Utilisation de la transformation Concaténer des colonnes pour ajouter des colonnes

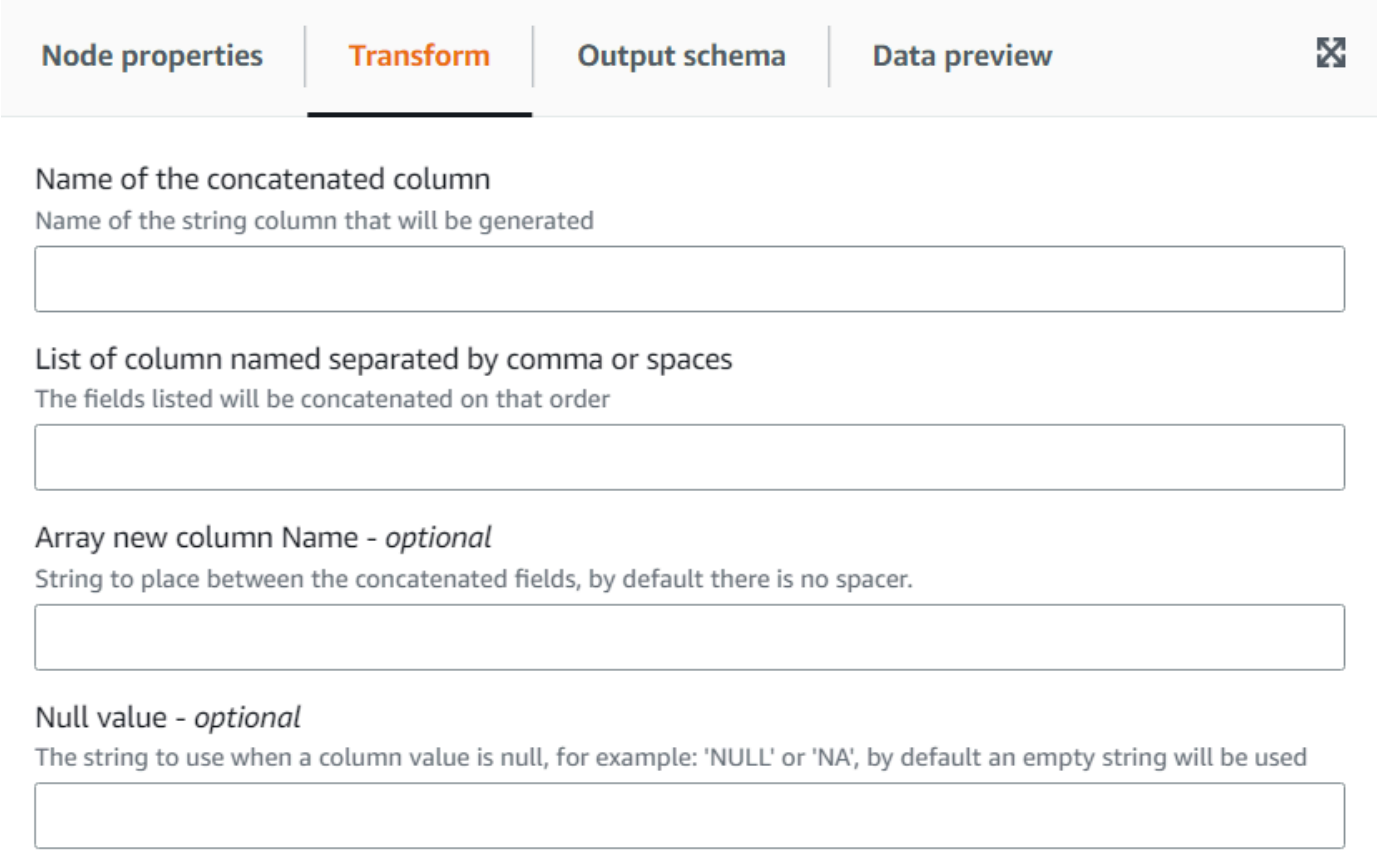
La transformation Concaténer vous permet de créer une nouvelle colonne de chaîne en utilisant les valeurs d'autres colonnes avec un espacement facultatif. Par exemple, si nous définissons une colonne concaténée « date » comme la concaténation de « année », « mois » et « jour » (dans cet ordre) avec « - » comme espacement, nous obtiendrions :

day	month	year	date
01	01	2020	01/01/2020
02	01	2020	02/01/2020
03	01	2020	03/01/2020
04	01	2020	04/01/2020

Pour ajouter une transformation Concaténer :

1. Ouvrez le panneau Ressources. Choisissez ensuite Concaténer des colonnes pour ajouter une nouvelle transformation à votre diagramme de tâches. Le nœud sélectionné au moment de l'ajout du nœud sera son parent.

2. (Facultatif) Dans l'onglet Propriétés de nœud, vous pouvez saisir un nom pour le nœud dans le diagramme de tâches. Si aucun parent de nœud n'est déjà sélectionné, choisissez un nœud dans la liste Node parents (Parents de nœud) à utiliser comme source pour la transformation.
3. Dans l'onglet Transformer, saisissez le nom de la colonne qui contiendra la chaîne concaténée ainsi que les colonnes à concaténer. L'ordre dans lequel vous cochez les colonnes de la liste déroulante sera l'ordre utilisé.



4. Espaceur (facultatif) : saisissez une chaîne à placer entre les champs concaténés. Par défaut, il n'y a pas d'espacement.
5. Valeur nulle (facultatif) : saisissez une chaîne à utiliser lorsqu'une valeur de colonne est nulle. Par défaut, dans les cas où les colonnes ont la valeur « NULL » ou « NA », une chaîne vide est utilisée.

Utilisation de la transformation Fractionner la chaîne pour diviser une colonne de chaînes

La transformation Fractionner la chaîne vous permet de diviser une chaîne en un tableau de jetons à l'aide d'une expression régulière pour définir la manière dont la division est effectuée. Vous pouvez

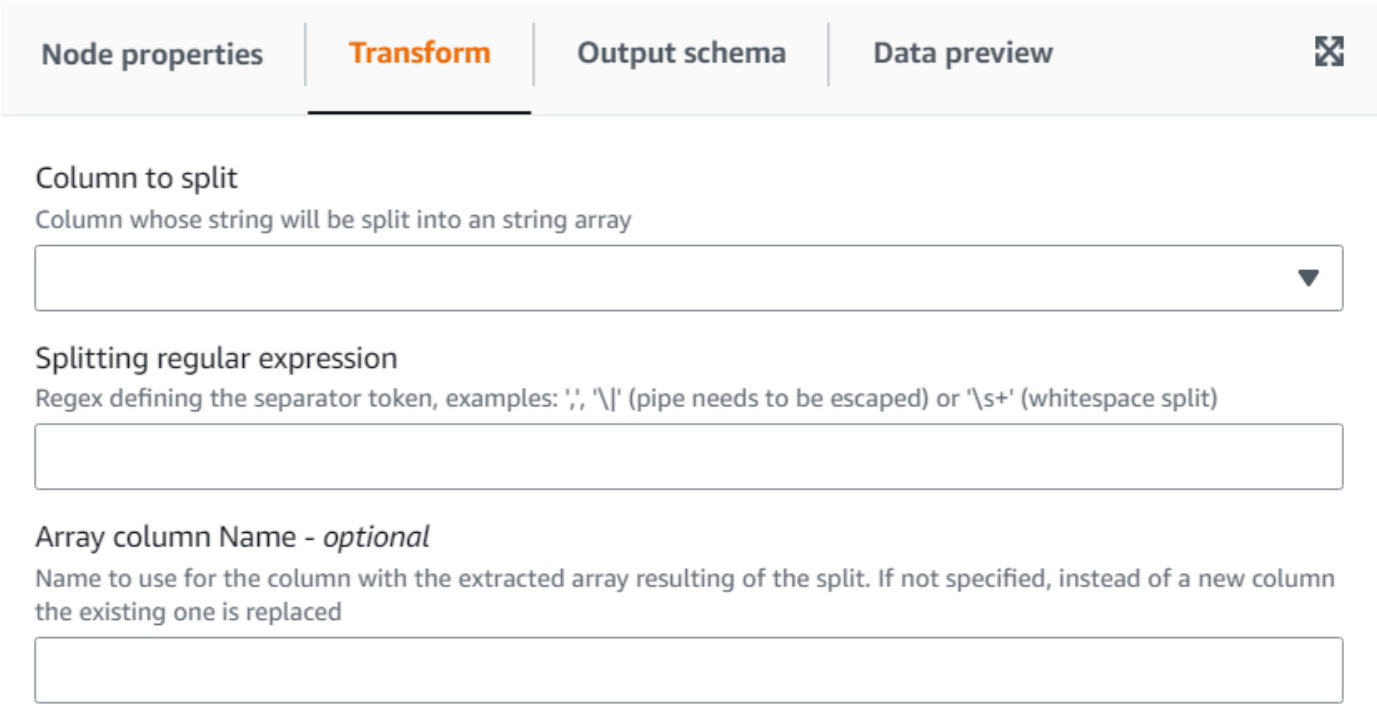
alors conserver la colonne sous forme de tableau ou appliquer une transformation Tableau vers colonnes après celle-ci, pour extraire les valeurs du tableau dans les champs de niveau supérieur, en supposant que chaque jeton a une signification que nous connaissons à l'avance. De plus, si l'ordre des jetons n'est pas pertinent (par exemple, un ensemble de catégories), vous pouvez utiliser la transformation Exploser pour générer une ligne distincte pour chaque valeur.

Par exemple, vous pouvez diviser la colonne « catégories » en utilisant une virgule comme modèle pour ajouter une colonne « categories_arr ».

product_id	categories	categories_arr
1	sports,hiver	[sports, hiver]
2	jardin,outils	[jardin, outils]
3	jeux vidéo	[jeux vidéo]
4	jeu,jeu de plateau,social	[jeu, jeu de plateau, social]

Pour ajouter une transformation Fractionner la chaîne :

1. Ouvrez le panneau Ressources, puis choisissez Fractionner la chaîne pour ajouter une nouvelle transformation à votre diagramme de tâches. Le nœud sélectionné au moment de l'ajout du nœud sera son parent.
2. (Facultatif) Dans l'onglet Propriétés de nœud, vous pouvez saisir un nom pour le nœud dans le diagramme de tâches. Si aucun parent de nœud n'est déjà sélectionné, choisissez un nœud dans la liste Node parents (Parents de nœud) à utiliser comme source pour la transformation.
3. Dans l'onglet Transformer, choisissez la colonne à fractionner et saisissez le modèle à utiliser pour fractionner la chaîne. Dans la plupart des cas, vous pouvez simplement saisir le ou les caractères, sauf s'ils ont une signification particulière en tant qu'expression régulière et doivent être échappés. Les caractères qui doivent être échappés sont les suivants : `\.[]{}()<>* +-=!?^$|` en ajoutant une barre oblique inverse devant le caractère. Par exemple, si vous souhaitez séparer par un point (« . »), vous devez saisir `\.`. Cependant, une virgule n'a pas de signification particulière et peut simplement être spécifiée telle quelle : `,`.



4. (Facultatif) Si vous souhaitez conserver la colonne de chaîne d'origine, vous pouvez saisir un nom pour une nouvelle colonne de tableau, en conservant ainsi à la fois la colonne de chaîne d'origine et la nouvelle colonne de tableau tokenisée.

Utilisation de la transformation Tableau vers colonnes pour extraire les éléments d'un tableau en colonnes de niveau supérieur

La transformation Tableau vers colonnes permet d'extraire certains ou tous les éléments d'une colonne de type tableau dans de nouvelles colonnes. La transformation remplira les nouvelles colonnes autant que possible si le tableau contient suffisamment de valeurs à extraire, en prenant éventuellement les éléments aux positions spécifiées.

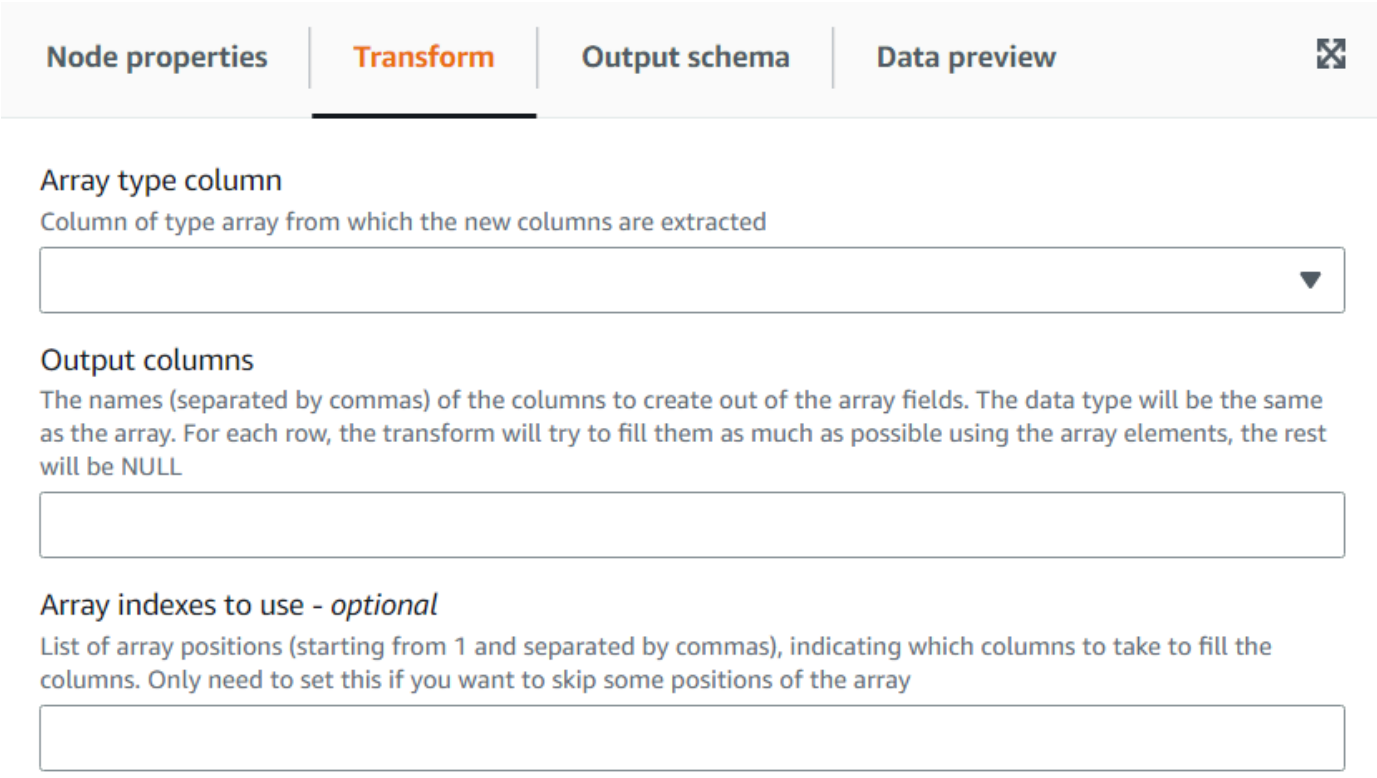
Par exemple, si vous avez une colonne de tableau « sous-réseau », qui est le résultat de la transformation « Diviser la chaîne » sur un sous-réseau IP v4, vous pouvez extraire les première et quatrième positions dans les nouvelles colonnes « first_octect » et « forth_octect ». Le résultat de la transformation dans cet exemple serait (remarquez que les deux dernières lignes ont des tableaux plus courts que prévu) :

sous-réseau	first_octect	fourth_octect
[54, 240, 197, 238]	54	238

sous-réseau	first_octect	fourth_octect
[192, 168, 0, 1]	192	1
[192, 168]	192	
[]		

Pour ajouter une transformation Tableau vers colonnes :

1. Ouvrez le panneau Ressources, puis choisissez Tableau vers colonnes pour ajouter une nouvelle transformation à votre diagramme de tâches. Le nœud sélectionné au moment de l'ajout du nœud sera son parent.
2. (Facultatif) Dans l'onglet Propriétés de nœud, vous pouvez saisir un nom pour le nœud dans le diagramme de tâches. Si aucun parent de nœud n'est déjà sélectionné, choisissez un nœud dans la liste Node parents (Parents de nœud) à utiliser comme source pour la transformation.
3. Dans l'onglet Transformer, choisissez la colonne du tableau à extraire et saisissez la liste des nouvelles colonnes pour les jetons extraits.



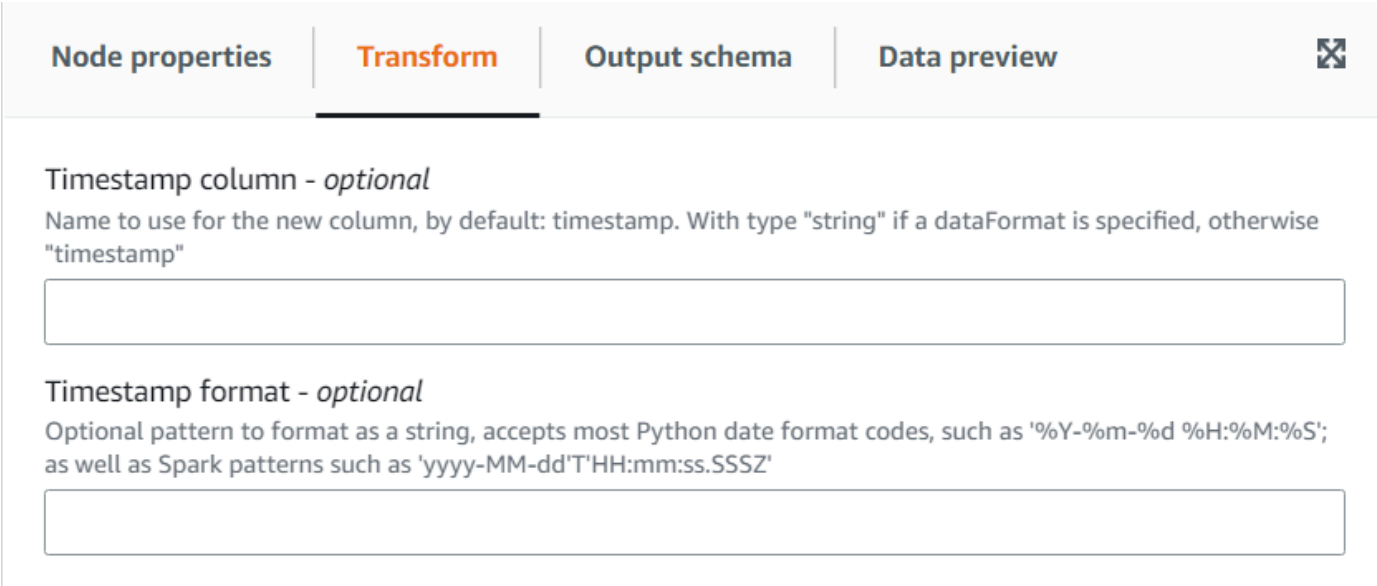
4. (Facultatif) Si vous ne voulez pas prendre les jetons du tableau pour les attribuer aux colonnes, vous pouvez spécifier les index à prendre qui seront attribués à la liste des colonnes dans le même ordre que celui indiqué. Par exemple, si les colonnes de sortie sont « colonne1, colonne2, colonne3 » et les index « 4, 1, 3 », le quatrième élément du tableau ira à la colonne1, le premier à la colonne2 et le troisième à la colonne3 (si le tableau est plus court que le nom d'index, une valeur NULL sera définie).

Utilisation de la transformation Ajouter un horodatage actuel

La transformation Ajouter un horodatage actuel vous permet de marquer les lignes avec l'heure à laquelle les données ont été traitées. Ceci est utile à des fins d'audit ou pour suivre la latence dans le pipeline de données. Vous pouvez ajouter cette nouvelle colonne sous forme de type de données d'horodatage ou de chaîne formatée.

Pour ajouter une transformation Ajouter un horodatage actuel :

1. Ouvrez le panneau Ressources, puis choisissez Ajouter un horodatage actuel pour ajouter une nouvelle transformation à votre diagramme de tâches. Le nœud sélectionné au moment de l'ajout du nœud sera son parent.
2. (Facultatif) Dans l'onglet Propriétés de nœud, vous pouvez saisir un nom pour le nœud dans le diagramme de tâches. Si aucun parent de nœud n'est déjà sélectionné, choisissez un nœud dans la liste Node parents (Parents de nœud) à utiliser comme source pour la transformation.



3. (Facultatif) Dans l'onglet Transformer, saisissez un nom personnalisé pour la nouvelle colonne et un format si vous préférez que la colonne soit une chaîne de date formatée.

Utilisation de la transformation Pivoter les lignes en colonnes

La transformation Pivoter les lignes en colonnes vous permet d'agréger une colonne numérique en faisant pivoter des valeurs uniques sur des colonnes sélectionnées qui deviennent de nouvelles colonnes (si plusieurs colonnes sont sélectionnées, les valeurs sont concaténées pour nommer les nouvelles colonnes). De cette façon, les lignes sont consolidées tout en ayant davantage de colonnes avec des agrégations partielles pour chaque valeur unique. Par exemple, si vous disposez de ce jeu de données de ventes par mois et par pays (trié pour faciliter l'illustration) :

year	month	country	montant
2020	Jan	uk	32
2020	Jan	de	42
2020	Jan	us	64
2020	Fév	uk	67
2020	Fév	de	4
2020	Fév	de	7
2020	Fév	us	6
2020	Fév	us	12
2020	Jan	us	90

Si vous faites pivoter le montant et le pays comme colonnes d'agrégation, de nouvelles colonnes sont créées à partir de la colonne du pays d'origine. Dans le tableau ci-dessous, vous avez de nouvelles colonnes pour de, uk et us au lieu de la colonne country.

year	month	de	uk	us
2020	Jan	42	32	64
2020	Jan	11	67	18

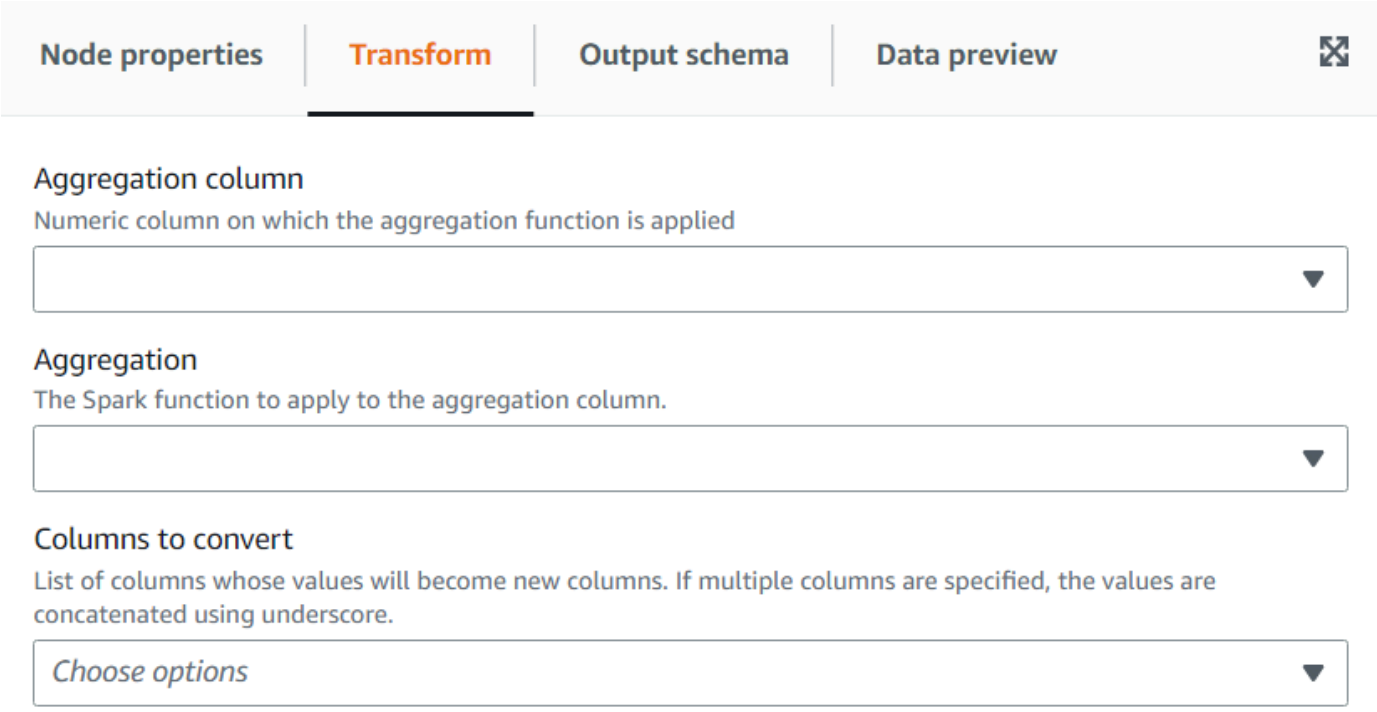
year	month	de	uk	us
2021	Jan			90

Si vous souhaitez plutôt faire pivoter à la fois le mois et le pays, vous obtenez une colonne pour chaque combinaison des valeurs de ces colonnes :

year	Jan_de	Jan_uk	Jan_us	Feb_de	Feb_uk	Feb_us
2020	42	32	64	11	67	18
2021			90			

Pour ajouter une transformation Pivoter les lignes en colonnes :

1. Ouvrez le panneau Ressources, puis choisissez Pivoter les lignes en colonnes pour ajouter une nouvelle transformation à votre diagramme de tâches. Le nœud sélectionné au moment de l'ajout du nœud sera son parent.
2. (Facultatif) Dans l'onglet Propriétés de nœud, vous pouvez saisir un nom pour le nœud dans le diagramme de tâches. Si aucun parent de nœud n'est déjà sélectionné, choisissez un nœud dans la liste Node parents (Parents de nœud) à utiliser comme source pour la transformation.
3. Dans l'onglet Transformer, choisissez la colonne numérique qui sera agrégée pour produire les valeurs des nouvelles colonnes, la fonction d'agrégation à appliquer et la ou les colonnes pour convertir leurs valeurs uniques en nouvelles colonnes.



Utilisation de la transformation Dépivoter les colonnes en lignes

La transformation Dépivoter vous permet de convertir des colonnes en valeurs de nouvelles colonnes en générant une ligne pour chaque valeur unique. C'est l'inverse du pivot, mais notez qu'il n'est pas équivalent puisqu'il ne peut pas séparer les lignes contenant des valeurs identiques qui ont été agrégées ou fractionnées en combinaisons dans les colonnes d'origine (vous pouvez le faire ultérieurement à l'aide d'une transformation Fractionner). Par exemple, si vous avez la table suivante :

year	month	de	uk	us
2020	Jan	42	32	64
2020	Fév	11	67	18
2021	Jan			90

Vous pouvez dépivoter les colonnes « de », « uk » et « us » dans une colonne « country » avec la valeur « amount », et obtenir ce qui suit (trié ici à des fins d'illustration) :

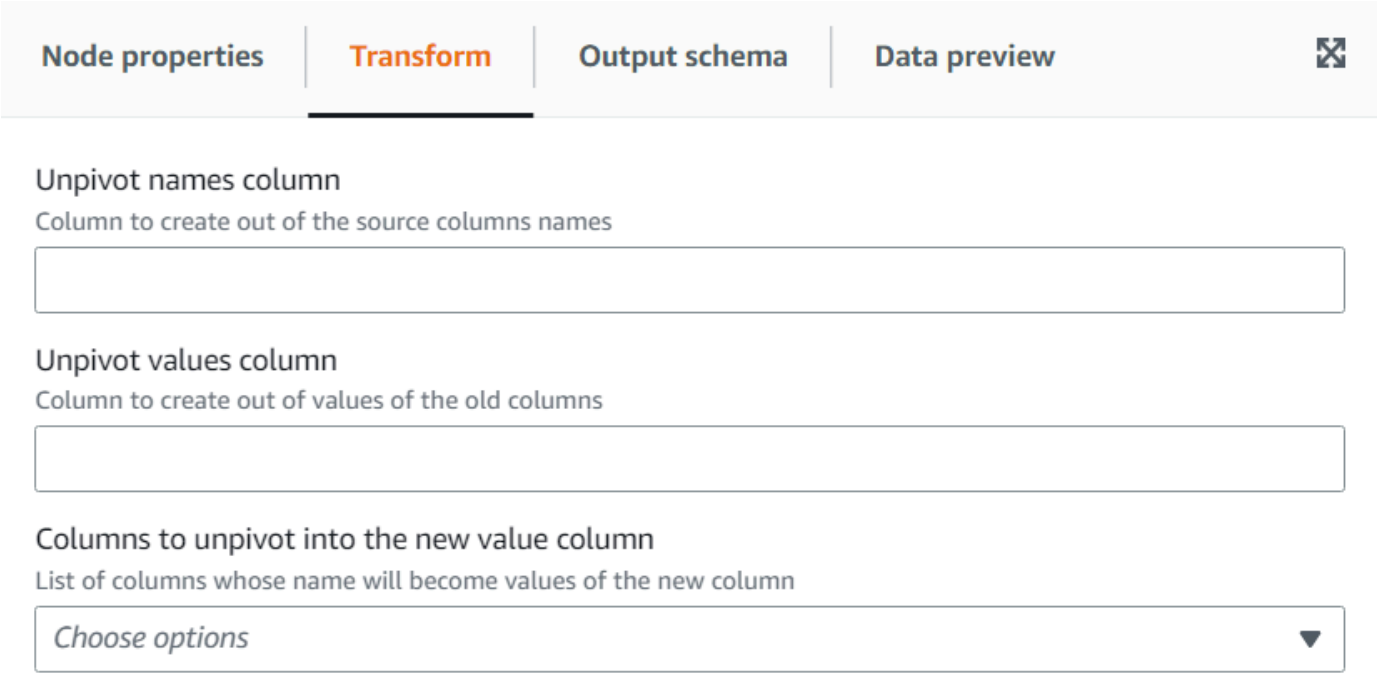
year	month	country	montant
2020	Jan	uk	32
2020	Jan	de	42
2020	Jan	us	64
2020	Fév	uk	67
2020	Fév	de	11
2020	Fév	us	18
2021	Jan	us	90

Notez que les colonnes qui ont une valeur NULL (« de » et « uk » de janvier 2021) ne sont pas générées par défaut. Vous pouvez activer cette option pour obtenir :

year	month	country	montant
2020	Jan	uk	32
2020	Jan	de	42
2020	Jan	us	64
2020	Fév	uk	67
2020	Fév	de	11
2020	Fév	us	18
2021	Jan	us	90
2021	Jan	de	
2021	Jan	uk	

Pour ajouter une transformation Dépivoter les colonnes en lignes :

1. Ouvrez le panneau Ressources, puis choisissez Dépivoter les colonnes en lignes pour ajouter une nouvelle transformation à votre diagramme de tâches. Le nœud sélectionné au moment de l'ajout du nœud sera son parent.
2. (Facultatif) Dans l'onglet Propriétés de nœud, vous pouvez saisir un nom pour le nœud dans le diagramme de tâches. Si aucun parent de nœud n'est déjà sélectionné, choisissez un nœud dans la liste Node parents (Parents de nœud) à utiliser comme source pour la transformation.
3. Dans l'onglet Transformer, saisissez les nouvelles colonnes à créer pour contenir les noms et les valeurs des colonnes choisies pour être dépivotées.

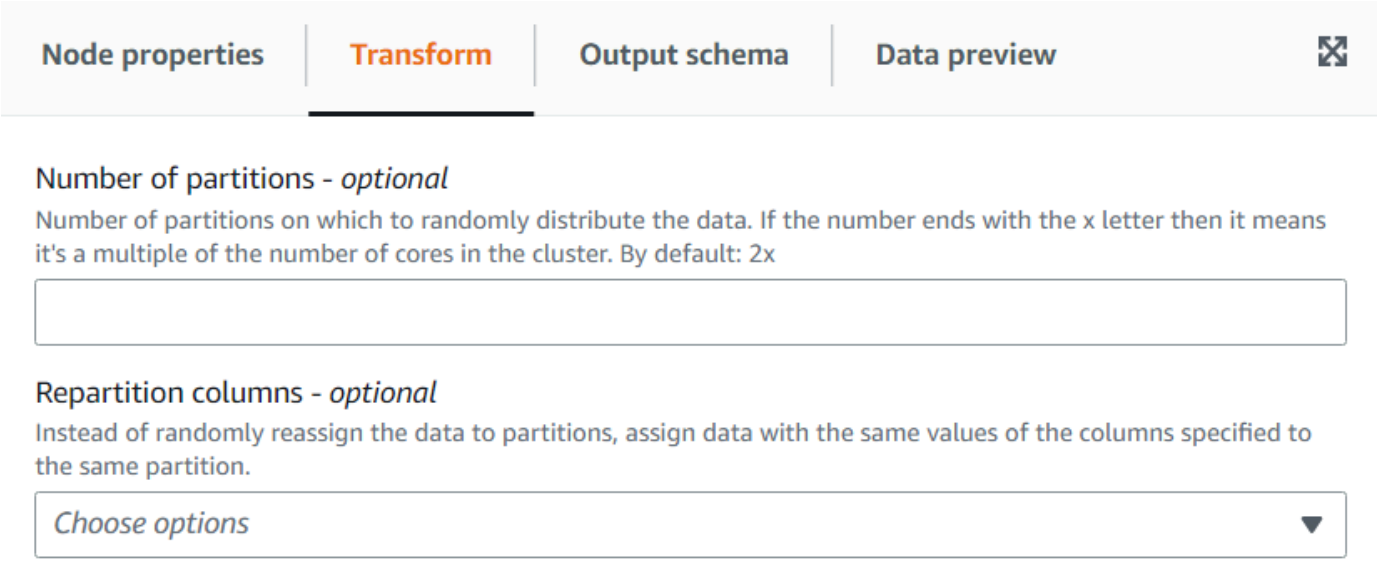


Utilisation de la transformation Traitement de l'équilibre automatique pour optimiser votre exécution

La transformation Traitement de l'équilibre automatique redistribue les données entre les travailleurs pour de meilleures performances. Cela est utile dans les cas où les données sont déséquilibrées ou lorsque la source ne permet pas un traitement parallèle suffisant. Cette situation est fréquente lorsque la source est compressée ou qu'il s'agit de JDBC. La redistribution des données a un coût de performance modeste, de sorte que l'optimisation ne compense pas toujours cet effort si les données étaient déjà correctement équilibrées. Ci-dessous, la transformation utilise la répartition Apache Spark pour réaffecter les données de manière aléatoire entre un certain nombre de partitions

optimales pour la capacité du cluster. Pour les utilisateurs avancés, il est possible de saisir un certain nombre de partitions manuellement. En outre, il peut être utilisé pour optimiser l'écriture des tables partitionnées en réorganisant les données en fonction des colonnes spécifiées. Cela permet d'obtenir des fichiers de sortie plus consolidés.

1. Ouvrez le panneau Ressources, puis choisissez Traitement de l'équilibre automatique pour ajouter une nouvelle transformation à votre diagramme de tâches. Le nœud sélectionné au moment de l'ajout du nœud sera son parent.
2. (Facultatif) Dans l'onglet Propriétés de nœud, vous pouvez saisir un nom pour le nœud dans le diagramme de tâches. Si aucun parent de nœud n'est déjà sélectionné, choisissez un nœud dans la liste Node parents (Parents de nœud) à utiliser comme source pour la transformation.
3. (Facultatif) Dans l'onglet Transformer, vous pouvez saisir un certain nombre de partitions. En général, il est recommandé de laisser le système décider de cette valeur, mais vous pouvez ajuster le multiplicateur ou saisir une valeur spécifique si vous avez besoin de la contrôler. Si vous souhaitez enregistrer les données partitionnées par colonnes, vous pouvez choisir les mêmes colonnes comme colonnes de répartition. De cette manière, le nombre de fichiers sur chaque partition sera minimisé et il n'y aura pas beaucoup de fichiers par partition, ce qui entraverait les performances des outils qui interrogent ces données.



Utilisation de la transformation Colonnes dérivées pour combiner d'autres colonnes

La transformation Colonnes dérivées vous permet de définir une nouvelle colonne basée sur une formule mathématique ou une expression SQL dans laquelle vous pouvez utiliser d'autres colonnes

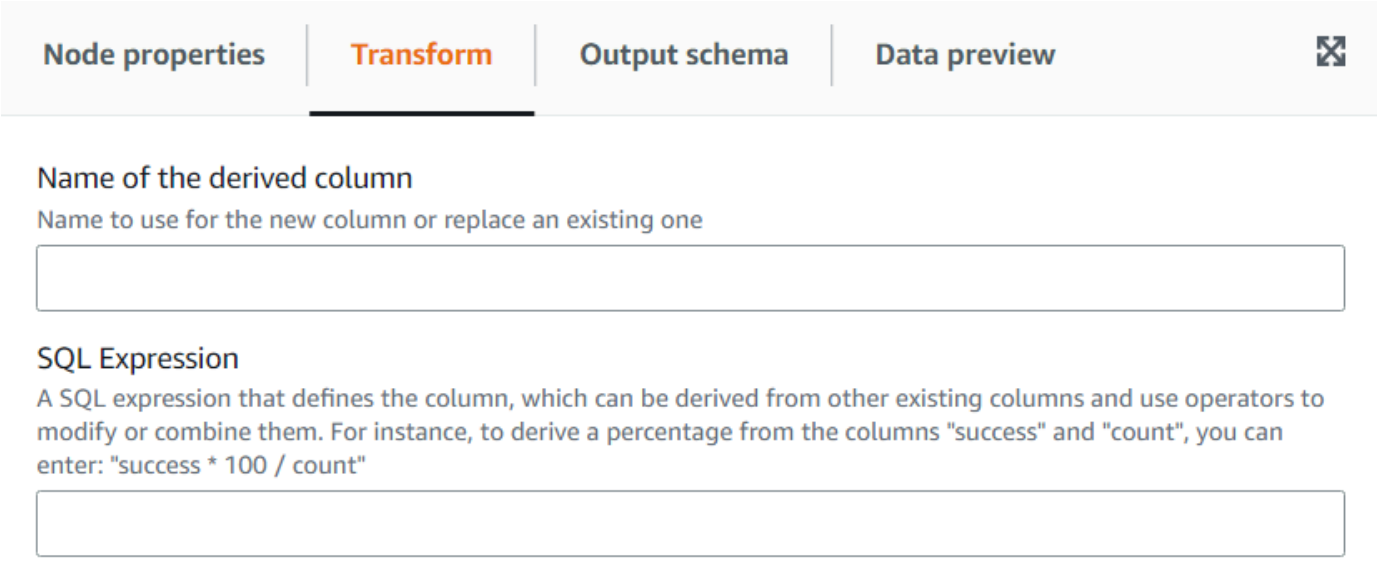
dans les données, ainsi que des constantes et des littéraux. Par exemple, pour dériver une colonne « percentage » à partir des colonnes « success » et « count », vous pouvez saisir l'expression SQL : « success * 100 / count || '% ».

Exemple de résultat :

success	count	percentage
14	100	14 %
6	20	3 %
3	40	7,5 %

Pour ajouter une transformation Colonnes dérivées :

1. Ouvrez le panneau Ressources, puis choisissez Colonnes dérivées pour ajouter une nouvelle transformation à votre diagramme de tâches. Le nœud sélectionné au moment de l'ajout du nœud sera son parent.
2. (Facultatif) Dans l'onglet Propriétés de nœud, vous pouvez saisir un nom pour le nœud dans le diagramme de tâches. Si aucun parent de nœud n'est déjà sélectionné, choisissez un nœud dans la liste Node parents (Parents de nœud) à utiliser comme source pour la transformation.
3. Dans l'onglet Transformer, saisissez le nom de la colonne et l'expression de son contenu.



Utilisation de la transformation Rechercher pour ajouter des données correspondantes à partir d'une table de catalogue

La transformation Rechercher vous permet d'ajouter des colonnes à partir d'une table de catalogue définie lorsque les clés correspondent aux colonnes de recherche définies dans les données. Cela revient à effectuer une jointure externe gauche entre les données et la table de recherche en utilisant comme condition les colonnes correspondantes.

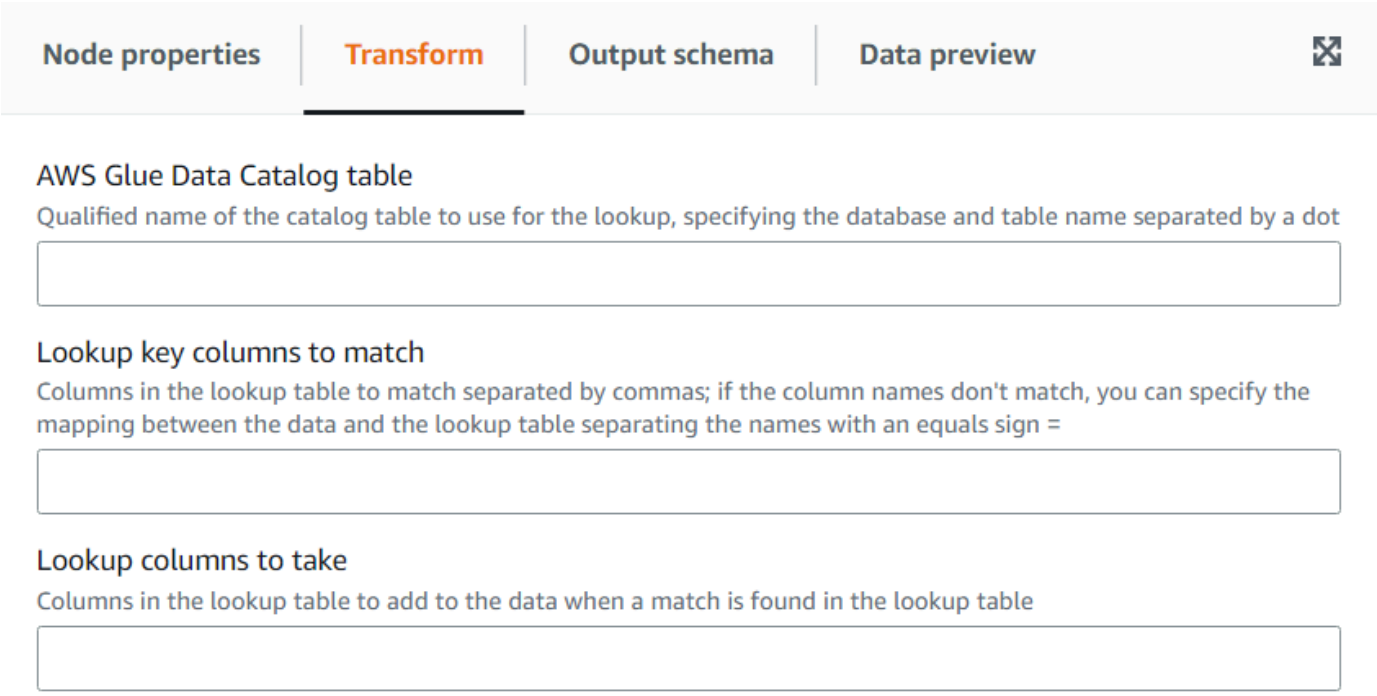
Pour ajouter une transformation Rechercher :

1. Ouvrez le panneau Ressources, puis choisissez Rechercher pour ajouter une nouvelle transformation à votre diagramme de tâches. Le nœud sélectionné au moment de l'ajout du nœud sera son parent.
2. (Facultatif) Dans l'onglet Propriétés de nœud, vous pouvez saisir un nom pour le nœud dans le diagramme de tâches. Si aucun parent de nœud n'est déjà sélectionné, choisissez un nœud dans la liste Node parents (Parents de nœud) à utiliser comme source pour la transformation.
3. Dans l'onglet Transformer, saisissez le nom de la table de catalogue entièrement qualifiée à utiliser pour effectuer les recherches. Par exemple, si votre base de données est « mydb » et votre table « mytable », saisissez « mydb.mytable ». Saisissez ensuite les critères permettant de trouver une correspondance dans la table de recherche, si la clé de recherche est composée. Saisissez la liste des colonnes clés séparées par des virgules. Si une ou plusieurs colonnes clés ne portent pas le même nom, vous devez définir le mappage des correspondances.

   Par exemple, si les colonnes de données sont « user_id » et « region » et que dans le tableau des utilisateurs les colonnes correspondantes sont nommées « id » et « region », alors dans le champ Colonne à faire correspondre, saisissez : « user_id=id, region ». Vous pouvez saisir region=region mais ce n'est pas nécessaire, car ce sont les mêmes.
4. Enfin, saisissez les colonnes à extraire de la ligne correspondant à la table de recherche afin de les intégrer aux données. Si aucune correspondance n'est trouvée, ces colonnes seront définies sur NULL.

> **Note**
>
> Sous la transformation Rechercher, elle utilise une jointure gauche pour être efficace. Si la table de recherche possède une clé composite, assurez-vous que les colonnes à associer sont configurées pour correspondre à toutes les colonnes clés afin qu'il n'y ait qu'une seule correspondance possible. Sinon, plusieurs lignes de recherche

correspondent, ce qui entraîne l'ajout de lignes supplémentaires pour chacune de ces correspondances.



Utilisation de la transformation Éclater le tableau ou la carte en lignes

La transformation Éclater vous permet d'extraire les valeurs d'une structure imbriquée en lignes individuelles plus faciles à manipuler. Dans le cas d'un tableau, la transformation génère une ligne pour chaque valeur du tableau, répliquant les valeurs des autres colonnes de la ligne. Dans le cas d'une carte, la transformation génère une ligne pour chaque entrée, avec la clé et la valeur comme colonnes, ainsi que toutes les autres colonnes de la ligne.

Par exemple, si nous avons ce jeu de données qui possède une colonne de type « category » contenant plusieurs valeurs.

product_id	category
1	[sports, hiver]
2	[jardin, outils]

product_id	category
3	[jeux vidéo]
4	[jeu, jeu de plateau, social]
5	[]

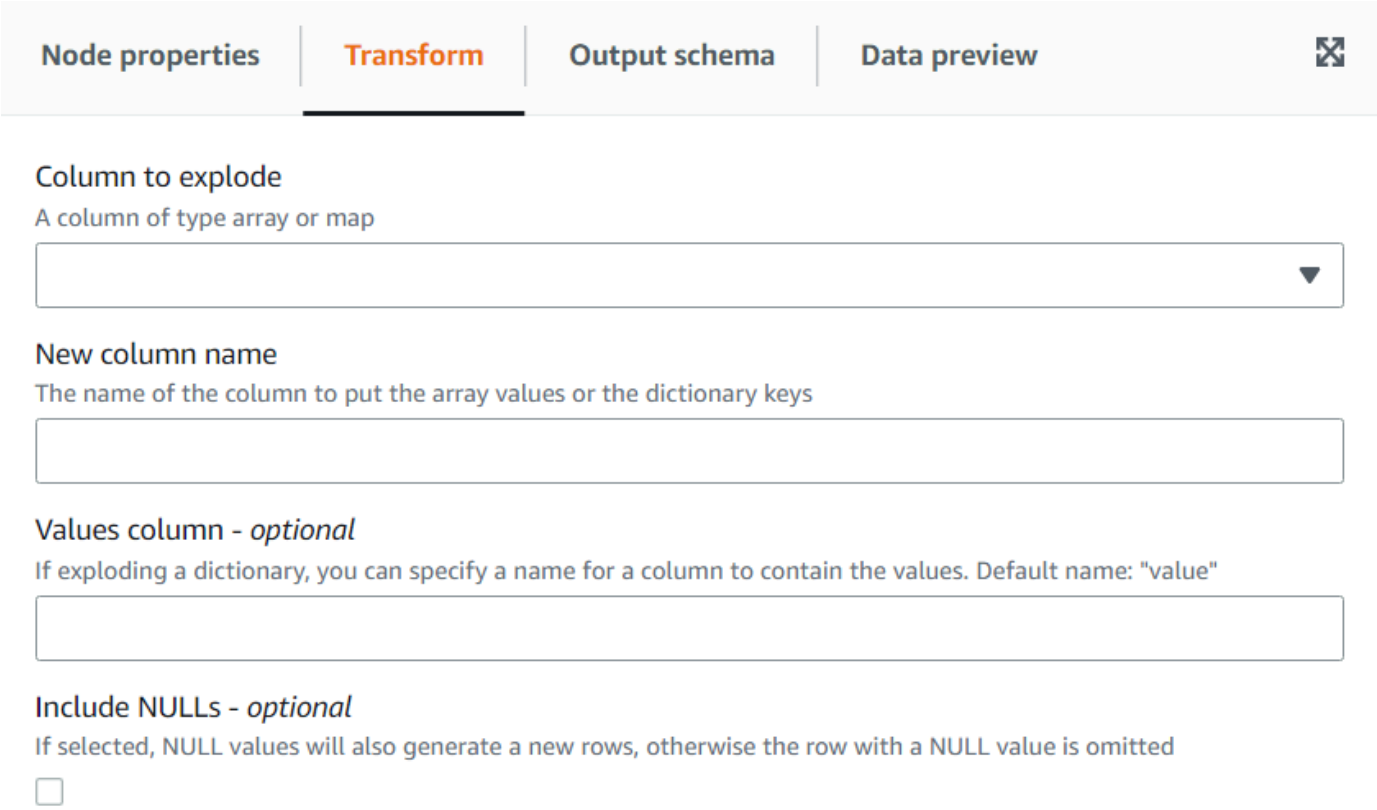
Si vous explosez la colonne « category » en une colonne portant le même nom, vous remplacerez la colonne. Vous pouvez sélectionner ce que vous souhaitez NULLs inclure pour obtenir les éléments suivants (commandés à des fins d'illustration) :

product_id	category
1	sports
1	hiver
2	jardin
2	outil
3	jeux vidéo
4	game
4	jeu de plateau
4	social
5	

Pour ajouter une transformation Éclater le tableau ou la carte en lignes :

1. Ouvrez le panneau Ressources, puis choisissez Éclater le tableau ou la carte en lignes pour ajouter une nouvelle transformation à votre diagramme de tâches. Le nœud sélectionné au moment de l'ajout du nœud sera son parent.

2. (Facultatif) Dans l'onglet Propriétés de nœud, vous pouvez saisir un nom pour le nœud dans le diagramme de tâches. Si aucun parent de nœud n'est déjà sélectionné, choisissez un nœud dans la liste Node parents (Parents de nœud) à utiliser comme source pour la transformation.
3. Dans l'onglet Transformer, choisissez la colonne à éclater (il doit s'agir d'un tableau ou d'un type de carte). Saisissez ensuite un nom pour la colonne des éléments du tableau ou les noms des colonnes des clés et des valeurs si vous éclatez une carte.
4. (Facultatif) Dans l'onglet Transformer, par défaut, si la colonne à éclater est NULL ou possède une structure vide, elle sera omise dans le jeu de données éclaté. Si vous souhaitez conserver la ligne (avec les nouvelles colonnes nulles), cochez « Inclure NULLs ».



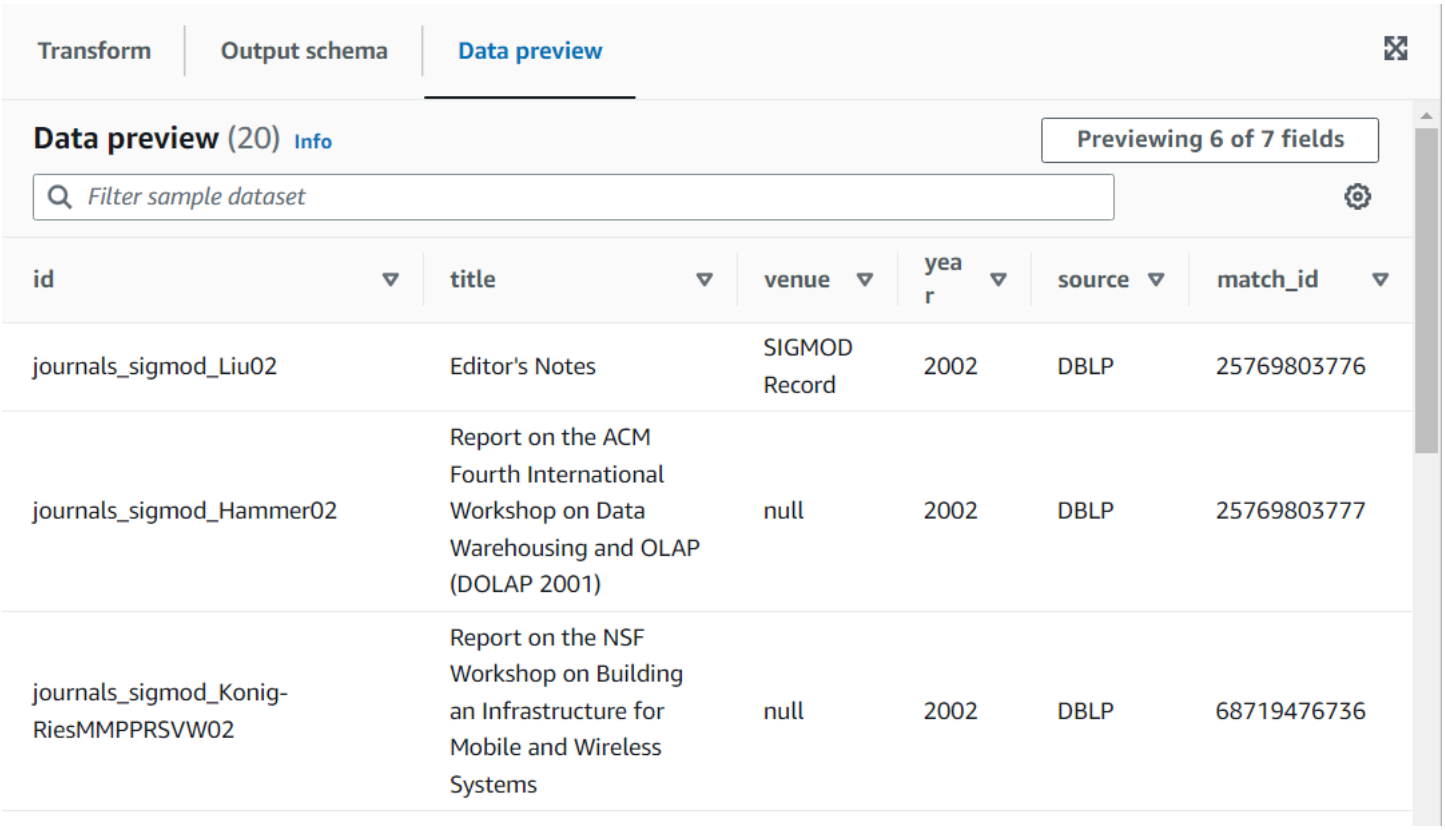
Utilisation de la transformation Correspondance des enregistrements pour invoquer une transformation de classification de données existante

Cette transformation invoque une transformation existante de classification de données de machine learning Correspondance des enregistrements.

La transformation évalue les données actuelles par rapport au modèle entraîné sur la base des étiquettes. Une colonne « match_id » est ajoutée pour attribuer chaque ligne à un groupe

d'éléments considérés comme équivalents sur la base de l'apprentissage de l'algorithme. Pour plus d'informations, voir Correspondance des enregistrements avec Lake Formation FindMatches.

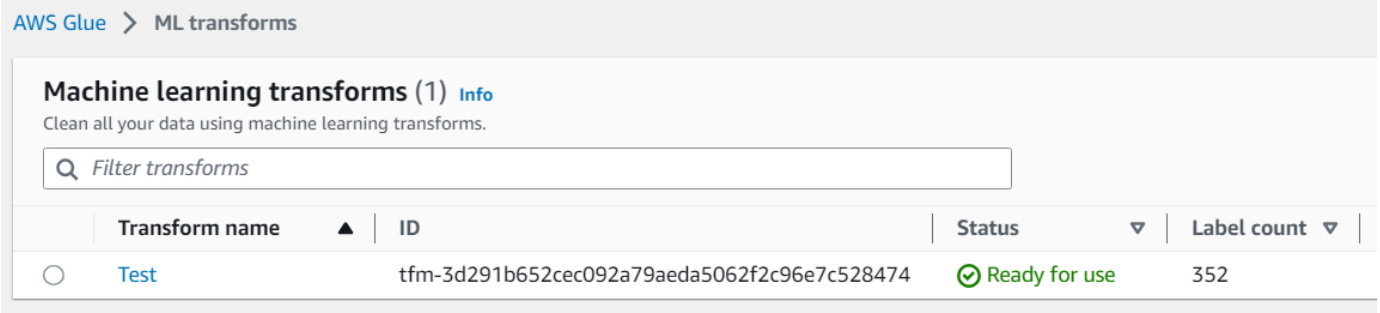
> **Note**
>
> La version AWS Glue utilisée par la tâche visuelle doit correspondre à la version AWS Glue utilisée pour créer la transformation Record Matching.



Pour ajouter un nœud de transformation Correspondance des enregistrements à votre diagramme de tâche

1. Ouvrez le panneau Ressources, puis choisissez Correspondance des enregistrements pour ajouter une nouvelle transformation à votre diagramme de tâches. Le nœud sélectionné au moment de l'ajout du nœud sera son parent.
2. Dans le panneau des propriétés du nœud, vous pouvez saisir un nom pour le nœud dans le diagramme de tâches. Si aucun parent de nœud n'est déjà sélectionné, choisissez un nœud dans la liste de Node parents (Parents de nœud) à utiliser comme source pour la transformation.

3. Dans l'onglet Transformer, saisissez l'ID extrait de la page des Transformations du machine learning :



4. (Facultatif) Dans l'onglet Transformer, vous pouvez cocher l'option permettant d'ajouter les scores de confiance. Au prix d'un calcul supplémentaire, le modèle estimera un score de confiance pour chaque correspondance dans une colonne supplémentaire.

Supprimer les lignes nulles

Cette transformation supprime du jeu de données les lignes dont toutes les colonnes sont nulles. En outre, vous pouvez étendre ce critère pour inclure les champs vides, afin de conserver les lignes dont au moins une colonne n'est pas vide.

Pour ajouter un nœud de transformation Supprimer les lignes nulles à votre diagramme de tâche

1. Ouvrez le panneau Ressources, puis choisissez Supprimer les lignes nulles pour ajouter une nouvelle transformation à votre diagramme de tâches. Le nœud sélectionné au moment de l'ajout du nœud sera son parent.
2. Dans le panneau des propriétés du nœud, vous pouvez saisir un nom pour le nœud dans le diagramme de tâches. Si aucun parent de nœud n'est déjà sélectionné, choisissez un nœud dans la liste de Node parents (Parents de nœud) à utiliser comme source pour la transformation.
3. (Facultatif) Dans l'onglet Transformer, cochez l'option Étendu si vous souhaitez que les lignes ne soient pas simplement nulles, mais également qu'elles ne soient pas vides. Ainsi, les chaînes, les tableaux ou les cartes vides seront considérés comme nuls dans le cadre de cette transformation.

Analyse d'une colonne de chaîne contenant des données JSON

Cette transformation analyse une colonne de chaîne contenant des données JSON et la convertit en une structure ou en une colonne de tableau, selon que le JSON est respectivement un objet ou un tableau. Vous pouvez éventuellement conserver la colonne analysée et la colonne d'origine.

Le schéma JSON peut être fourni ou déduit (dans le cas des objets JSON), avec un échantillonnage facultatif.

Pour ajouter un nœud de transformation Analyser la colonne JSON à votre diagramme de tâches

1. Ouvrez le panneau Ressources, puis choisissez Analyser la colonne JSON pour ajouter une nouvelle transformation à votre diagramme de tâches. Le nœud sélectionné au moment de l'ajout du nœud sera son parent.
2. Dans le panneau des propriétés du nœud, vous pouvez saisir un nom pour le nœud dans le diagramme de tâches. Si aucun parent de nœud n'est déjà sélectionné, choisissez un nœud dans la liste de Node parents (Parents de nœud) à utiliser comme source pour la transformation.
3. Dans l'onglet Transformer, sélectionnez la colonne contenant la chaîne JSON.
4. (Facultatif) Dans l'onglet Transformer, saisissez le schéma suivi par les données JSON en utilisant la syntaxe SQL, par exemple : « field1 STRING, field2 INT » dans le cas d'un objet ou « ARRAY<STRING> » dans le cas d'un tableau.

   Dans le cas d'un tableau, le schéma est obligatoire, mais dans le cas d'un objet, si le schéma n'est pas spécifié, il sera déduit à l'aide des données. Pour réduire l'impact de l'inférence du schéma (en particulier sur un jeu de données volumineux), vous pouvez éviter de lire le jeu de données deux fois en saisissant un Ratio d'échantillons à utiliser pour déduire le schéma. Si la valeur est inférieure à 1, le ratio correspondant d'échantillons aléatoires est utilisé pour déduire le schéma. Si les données sont fiables et que l'objet est cohérent entre les lignes, vous pouvez utiliser un faible ratio tel que 0,1 pour améliorer les performances.
5. (Facultatif) Dans l'onglet Transformer, vous pouvez saisir un nouveau nom de colonne si vous souhaitez conserver à la fois la colonne de chaîne d'origine et la colonne analysée.

Extraction d'un chemin JSON

Cette transformation extrait les nouvelles colonnes d'une colonne de chaîne JSON. Cette transformation est utile lorsque vous n'avez besoin que de quelques éléments de données et que vous ne souhaitez pas importer l'intégralité du contenu JSON dans le schéma de la table.

Pour ajouter un nœud de transformation Extraire un chemin JSON à votre diagramme de tâches

1. Ouvrez le panneau Ressources, puis choisissez Extraire un chemin JSON pour ajouter une nouvelle transformation à votre diagramme de tâches. Le nœud sélectionné au moment de l'ajout du nœud sera son parent.
2. Dans le panneau des propriétés du nœud, vous pouvez saisir un nom pour le nœud dans le diagramme de tâches. Si aucun parent de nœud n'est déjà sélectionné, choisissez un nœud dans la liste de Node parents (Parents de nœud) à utiliser comme source pour la transformation.
3. Dans l'onglet Transformer, sélectionnez la colonne contenant la chaîne JSON. Saisissez une ou plusieurs expressions de chemin JSON séparées par des virgules, chacune indiquant comment extraire une valeur du tableau ou de l'objet JSON. Par exemple, si la colonne JSON contient un objet avec les propriétés « prop_1 » et « prop2 », vous pouvez extraire les deux en spécifiant leurs noms « prop_1, prop_2 ».

   Si le champ JSON contient des caractères spéciaux, par exemple pour extraire la propriété de ce `{"a. a": 1}` JSON, vous pouvez utiliser le chemin `$['a. a']`. La virgule est l'exception, car elle est réservée à la séparation des chemins. Saisissez ensuite les noms de colonnes correspondants pour chaque chemin, en les séparant par des virgules.
4. (Facultatif) Dans l'onglet Transformer, vous pouvez cocher la case pour supprimer la colonne JSON une fois extraite. Cela est utile lorsque vous n'avez pas besoin du reste des données JSON une fois que vous avez extrait les parties dont vous avez besoin.

Extraction de fragments de chaîne à l'aide d'une expression régulière

Cette transformation extrait des fragments de chaîne à l'aide d'une expression régulière et crée une nouvelle colonne à partir de celle-ci, ou plusieurs colonnes si vous utilisez des groupes d'expressions régulières.

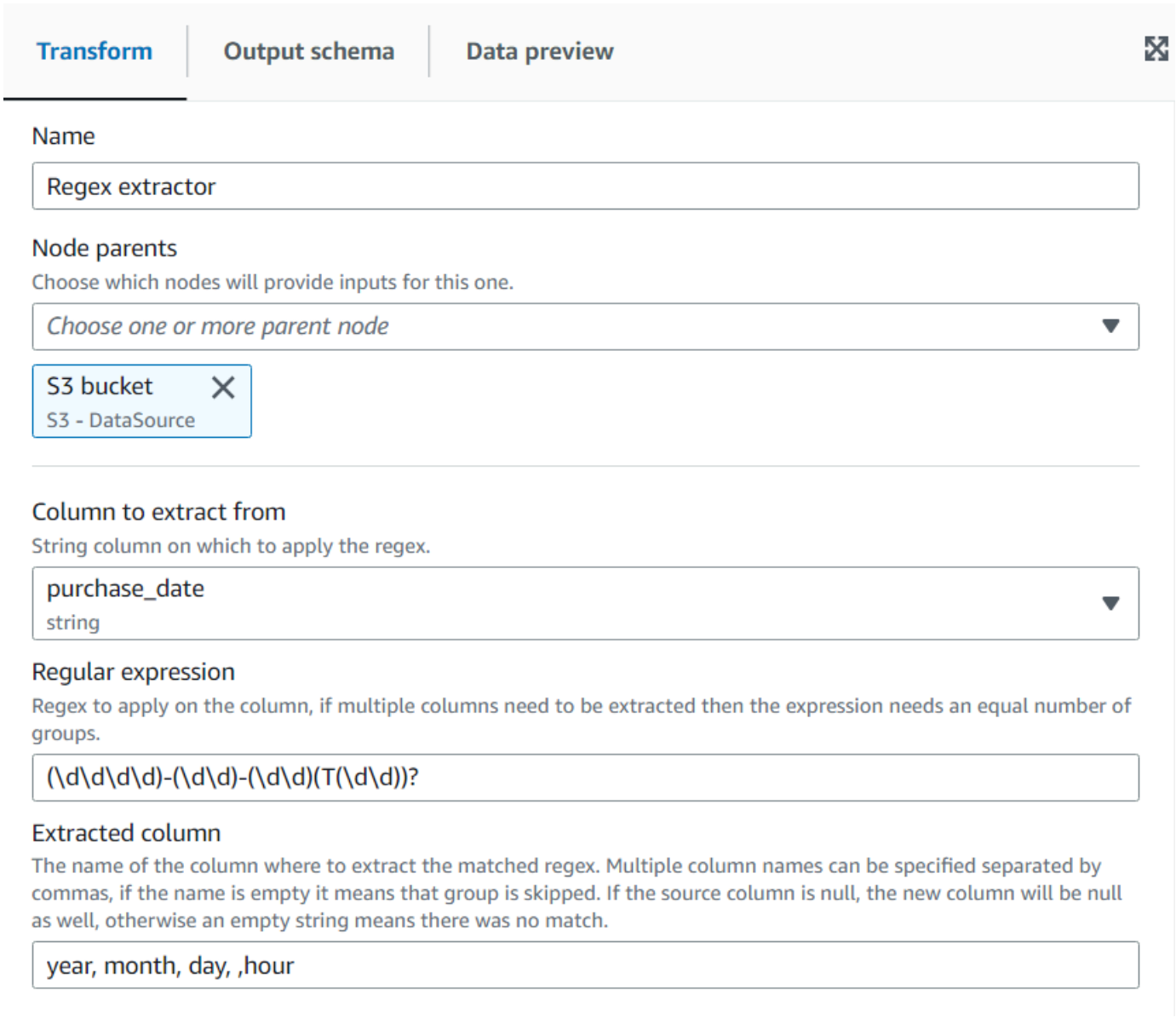
Pour ajouter un nœud de transformation Extracteur d'expressions régulières à votre diagramme de tâche

1. Ouvrez le panneau Ressources, puis choisissez Extracteur d'expressions régulières pour ajouter une nouvelle transformation à votre diagramme de tâches. Le nœud sélectionné au moment de l'ajout du nœud sera son parent.
2. Dans le panneau des propriétés du nœud, vous pouvez saisir un nom pour le nœud dans le diagramme de tâches. Si aucun parent de nœud n'est déjà sélectionné, choisissez un nœud dans la liste de Node parents (Parents de nœud) à utiliser comme source pour la transformation.

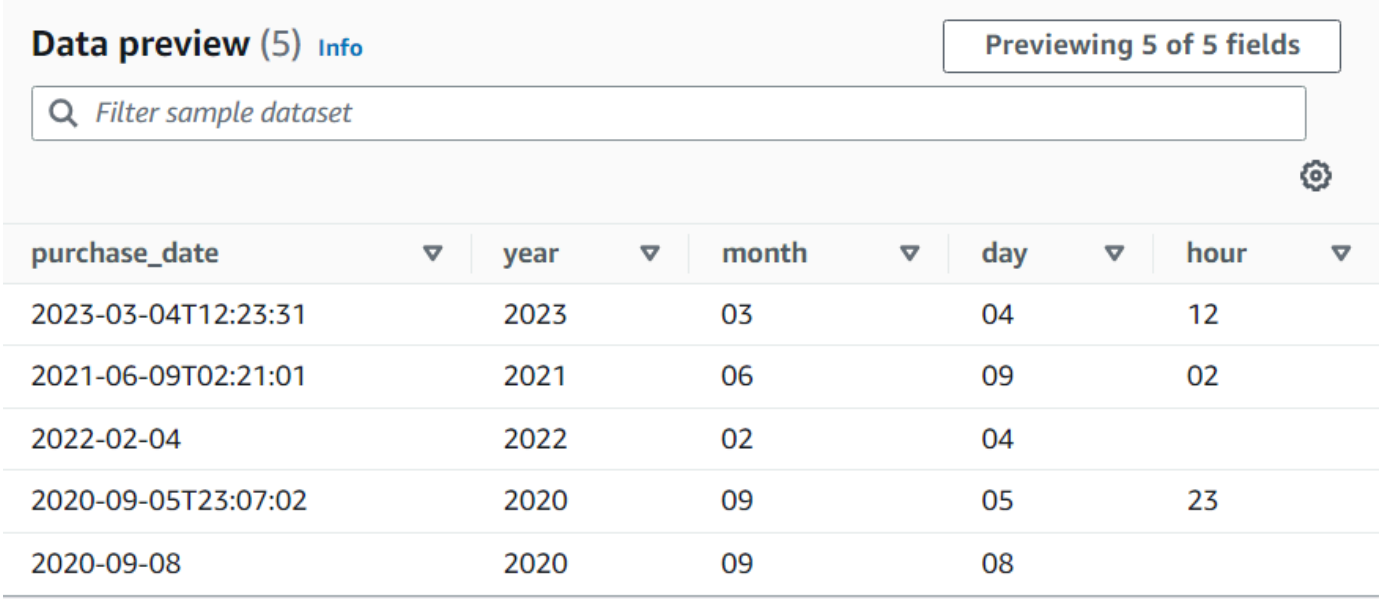
3. Dans l'onglet Transformer, saisissez l'expression régulière et la colonne sur laquelle elle doit être appliquée. Saisissez ensuite le nom de la nouvelle colonne dans laquelle vous souhaitez stocker la chaîne de caractères correspondante. La nouvelle colonne sera nulle uniquement si la colonne source est nulle, si l'expression régulière ne correspond pas, la colonne sera vide.

   Si l'expression régulière utilise des groupes, il doit y avoir un nom de colonne correspondant séparé par une virgule, mais vous pouvez ignorer les groupes en laissant le nom de colonne vide.

   Par exemple, si vous avez une colonne « purchase_date » avec une chaîne utilisant à la fois des formats de date ISO longs et courts, vous souhaitez extraire l'année, le mois, le jour et l'heure, lorsqu'ils sont disponibles. Remarquez que le groupe « heure » est facultatif, sinon dans les lignes où il n'y a pas de données disponibles, tous les groupes extraits seraient des chaînes vides (car l'expression régulière ne correspond pas). Dans ce cas, nous ne voulons pas que le groupe rende l'heure facultative, mais plutôt la partie interne. Nous laissons donc le nom vide et il ne sera pas extrait (ce groupe inclurait le caractère T).



L'aperçu des données s'affiche :



purchase_date	year	month	day	hour
2023-03-04T12:23:31	2023	03	04	12
2021-06-09T02:21:01	2021	06	09	02
2022-02-04	2022	02	04	
2020-09-05T23:07:02	2020	09	05	23
2020-09-08	2020	09	08	

Créer une transformation personnalisée

Si vous avez besoin d'effectuer des transformations plus compliquées sur vos données ou si vous souhaitez ajouter des clés de propriété de données au jeu de données, vous pouvez ajouter une transformation Custom code (Code personnalisé) à votre diagramme de tâches. Le nœud de code personnalisé vous permet de saisir un script qui effectue la transformation.

Lorsque vous utilisez du code personnalisé, vous devez utiliser un éditeur de schéma pour indiquer les modifications apportées au résultat au travers du code personnalisé. Lorsque vous modifiez le schéma, vous pouvez exécuter les actions suivantes :

- Ajouter ou supprimer des clés de propriété de données
- Modifier le type de données des clés de propriété de données
- Modifier le nom des clés de propriété de données
- Restructurer une clé de propriété imbriquée

Vous devez utiliser une SelectFromCollectiontransformation pour en choisir une `DynamicFrame` parmi le résultat de votre nœud de transformation personnalisé avant de pouvoir envoyer la sortie vers un emplacement cible.

Utilisez les tâches suivantes pour ajouter un nœud de transformation personnalisé à votre diagramme de tâches.

Ajout d'un nœud de transformation de code personnalisé au diagramme de tâche

Pour ajouter un nœud de transformation personnalisé à votre diagramme de tâche

1. (Facultatif) Ouvrez le panneau Ressources, puis choisissez Transformation personnalisée pour ajouter une nouvelle transformation à votre diagramme de tâches.
2. Sur la page Node properties (Propriétés de nœud) au cours de la tâche, saisissez un nom pour le nœud dans le diagramme de tâche. Si un parent de nœud n'est pas déjà sélectionné, ou si vous souhaitez plusieurs entrées pour la transformation personnalisée, choisissez un nœud dans la liste Node parents (Parents de nœud) à utiliser comme source pour la transformation.

Saisie du code pour le nœud de transformation personnalisé

Vous pouvez taper ou copier du code dans un champ de saisie. La tâche utilise ce code pour effectuer la transformation des données. Vous pouvez fournir un extrait de code Python ou Scala. Le code doit inclure un ou plusieurs `DynamicFrames` en entrée et retourne une collection de `DynamicFrames`.

Pour saisir le script d'un nœud de transformation personnalisé

1. Lorsque le nœud de transformation personnalisé est sélectionné dans le diagramme de tâche, choisissez l'onglet Transform (Transformation).
2. Dans le champ de saisie de texte sous l'en-tête Bloc de code, collez ou saisissez le code pour la transformation. Le code que vous utilisez doit correspondre au langage spécifié pour la tâche dans l'onglet Job details (Détails de la tâche).

   Lorsque vous faites référence aux nœuds d'entrée dans votre code, AWS Glue Studio nomme le `DynamicFrames` renvoyé par les nœuds du diagramme de tâche séquentiellement en fonction de l'ordre de création. Utilisez l'une des méthodes de dénomination suivantes dans votre code :

   - Génération de code classique – Utilisez des noms fonctionnels pour faire référence aux nœuds de votre diagramme de tâches.
     - Noeuds de source de données : `DataSource0`, `DataSource1`, `DataSource2`, etc.
     - Transform nodes : `Transform0`, `Transform1`, `Transform2`, etc.
   - Génération d'un nouveau code – Utilisez le nom spécifié dans l'onglet Node properties (Propriétés du nœud) d'un nœud, complété par « _node1 », « _node2 », etc.
     Par exemple `S3bucket_node1`, `ApplyMapping_node2`, `S3bucket_node2`, `MyCustomNodeName_node1`.

Pour en savoir plus sur le nouveau générateur de code, consultez Génération de code de script.

Les exemples suivants montrent le format du code à saisir dans la zone de code :

Python

L'exemple suivant prend le premier `DynamicFrame` reçu, le convertit en `DataFrame` pour appliquer la méthode de filtre native (ne conservant que les enregistrements qui ont plus de 1000 votes), puis la convertit en un `DynamicFrame` avant de le retourner.

```
def FilterHighVoteCounts (glueContext, dfc) -> DynamicFrameCollection:
    df = dfc.select(list(dfc.keys())[0]).toDF()
    df_filtered = df.filter(df["vote_count"] > 1000)
    dyf_filtered = DynamicFrame.fromDF(df_filtered, glueContext, "filter_votes")
    return(DynamicFrameCollection({"CustomTransform0": dyf_filtered}, glueContext))
```

Scala

L'exemple suivant prend le premier `DynamicFrame` reçu, le convertit en `DataFrame` pour appliquer la méthode de filtre native (ne conservant que les enregistrements qui ont plus de 1000 votes), puis la convertit en un `DynamicFrame` avant de le retourner.

```
object FilterHighVoteCounts {
  def execute(glueContext : GlueContext, input : Seq[DynamicFrame]) :
 Seq[DynamicFrame] = {
    val frame = input(0).toDF()
    val filtered = DynamicFrame(frame.filter(frame("vote_count") > 1000),
 glueContext)
    Seq(filtered)
  }
}
```

Modifier le schéma d'un nœud de transformation personnalisé

Lorsque vous utilisez un nœud de transformation personnalisé, AWS Glue Studio ne peut pas déduire automatiquement les schémas de sortie créés par la transformation. Vous utilisez l'éditeur de schéma pour décrire les modifications de schéma implémentées par le code de transformation personnalisé.

Un nœud de code personnalisé peut avoir n'importe quel nombre de nœuds parents, chacun fournissant un `DynamicFrame` comme source pour votre code personnalisé. Un nœud de code personnalisé renvoie un ensemble de `DynamicFrames`. Chaque `DynamicFrame` qui est utilisé comme entrée a un schéma associé. Vous devez ajouter un schéma qui décrit chaque `DynamicFrame` retourné par le nœud de code personnalisé.

> **Note**
>
> Lorsque vous définissez votre propre schéma sur une transformation personnalisée, AWS Glue Studio n'hérite pas de schémas des nœuds précédents. Pour mettre à jour le schéma, choisissez le nœud de transformation personnalisé, puis choisissez l'onglet d'aperçu des données. Une fois l'aperçu généré, choisissez « Use Preview Schema » (Utiliser le schéma d'aperçu). Le schéma sera ensuite remplacé par le schéma à l'aide de la prévisualisation des données.

Vous modifiez ici les schémas d'un nœud de transformation personnalisé

1. Lorsque le nœud de transformation personnalisé est sélectionné dans le diagramme de tâche, dans le volet de détails du nœud, choisissez l'onglet Output Schema (Schéma de sortie).
2. Choisissez Edit (Modifier) pour apporter des modifications au schéma.

   Si vous avez des clés de propriété de données imbriquées, telles qu'un tableau ou un objet, vous pouvez choisir l'icône Expand-Rows (Développez les lignes) ( ) en haut à droite de chaque volet de schéma pour développer la liste des clés de propriété de données enfant. Une fois que vous avez sélectionné cette icône, elle devient Collapse-Rows (Réduire les lignes) ( ), que vous pouvez choisir de réduire la liste des clés de propriété enfants.
3. Modifiez le schéma à l'aide des actions suivantes, dans la section à droite de la page :

   - Pour renommer une clé de propriété, placez le curseur dans la zone de texte Key (Clé) de la clé de propriété, puis saisissez le nouveau nom.
   - Pour modifier le type de données d'une clé de propriété, utilisez la liste pour choisir le nouveau type de données pour la clé de propriété.

- Pour ajouter une nouvelle colonne de niveau supérieur au schéma, choisissez l'icône Overflow (Surcharger)
  (⋯)
  à droite du bouton Cancel (Annuler), puis choisissez Add root key (Ajouter une clé racine).
- Pour ajouter une clé de propriété enfant au schéma, choisissez l'icône Add-Key (Ajouter une clé)

  associée à la clé parent. Saisissez un nom pour la clé enfant et choisissez le type de données.
- Pour supprimer une clé de propriété du schéma, choisissez l'icône Remove (Supprimer)
  ( )
  à l'extrême droite du nom de la clé.

4. Si votre code de transformation personnalisé utilise plusieurs `DynamicFrames`, vous pouvez ajouter d'autres schémas de sortie.

   - Pour ajouter un nouveau schéma vide, choisissez Overflow (Surcharger)
     (⋯),
     puis choisissez Add output schema (Ajouter un schéma de sortie).
   - Pour copier un schéma existant dans un nouveau schéma en sortie, assurez-vous que le schéma à copier est affiché dans le sélecteur de schéma. Choisissez l'icône Overflow (Surcharger)
     (⋯),
     puis choisissez Duplicate (Dupliquer).

   Si vous souhaitez supprimer un schéma en sortie, assurez-vous que le schéma que vous souhaitez copier est affiché dans le sélecteur de schéma. Choisissez l'icône Overflow (Surcharger)
   (⋯),
   puis choisissez Delete (Supprimer).

5. Ajoutez de nouvelles clés racine au nouveau schéma ou modifiez les clés dupliquées.
6. Lorsque vous modifiez les schémas en sortie, choisissez le bouton Apply (Appliquer) pour sauvegarder vos modifications et quitter l'éditeur de schéma.

   Si vous ne souhaitez pas enregistrer vos modifications, choisissez le bouton Cancel (Annuler).

Configurer la sortie de transformation personnalisée

Une transformation de code personnalisée renvoie une collection de `DynamicFrames`, même s'il n'y a qu'un seul `DynamicFrame` dans le jeu de résultats.

Pour traiter la sortie à partir d'un nœud de transformation personnalisé

1. Ajoutez un nœud de SelectFromCollectiontransformation dont le nœud de transformation personnalisé est le nœud parent. Mettez à jour cette transformation pour indiquer le jeu de données que vous souhaitez utiliser. Pour plus d'informations, consultez Utilisation SelectFromCollection pour choisir le jeu de données à conserver.
2. Ajoutez des SelectFromCollectiontransformations supplémentaires au diagramme de tâches si vous souhaitez utiliser des transformations supplémentaires `DynamicFrames` produites par le nœud de transformation personnalisé.

   Considérez un scénario dans lequel vous ajoutez un nœud de transformation personnalisé pour diviser un jeu de données de vol en plusieurs jeux de données, mais dupliquez certaines des clés de propriété d'identification dans chaque schéma en sortie, telles que la date de vol ou le numéro de vol. Vous ajoutez un nœud de SelectFromCollectiontransformation pour chaque schéma de sortie, le nœud de transformation personnalisé étant son parent.
3. (Facultatif) Vous pouvez ensuite utiliser chaque nœud de SelectFromCollectiontransformation comme entrée pour les autres nœuds de la tâche ou comme parent pour un nœud cible de données.

Transformation des données avec des transformations visuelles personnalisées

Les transformations visuelles personnalisées vous permettent de créer des transformations et de les utiliser dans des tâches AWS Glue Studio. Les transformations visuelles personnalisées permettent aux développeurs ETL, pour lesquels le codage n'est pas toujours un exercice familier, d'utiliser une bibliothèque de transformations et d'y effectuer des recherches à l'aide de l'interface AWS Glue Studio.

Vous pouvez créer une transformation visuelle personnalisée, puis la charger sur Amazon S3 afin de pouvoir l'utiliser dans des tâches via l'éditeur visuel de AWS Glue Studio.

Rubriques

- Premiers pas avec les transformations visuelles personnalisées

- Étape 1. Créer un fichier de configuration JSON
- Étape 2. Implémenter la logique de transformation
- Étape 3. Valider les transformations visuelles personnalisées et résoudre les problèmes liés à celles-ci dans AWS Glue Studio
- Étape 4 : Mettre à jour les transformations visuelles personnalisées selon les besoins
- Étape 5. Utiliser des transformations visuelles personnalisées dans AWS Glue Studio
- Exemples d'utilisation
- Exemples de scripts visuels personnalisés
- Vidéo

Premiers pas avec les transformations visuelles personnalisées

Pour créer une transformation visuelle personnalisée, suivez les étapes ci-dessous.

- Étape 1. Créer un fichier de configuration JSON
- Étape 2. Implémenter la logique de transformation
- Étape 3. Valider la transformation visuelle personnalisée
- Étape 4 : Mettre à jour la transformation visuelle personnalisée selon les besoins
- Étape 5. Utiliser la transformation visuelle personnalisée dans AWS Glue Studio

Commencez par configurer le compartiment Amazon S3, puis passez à la section Étape 1. Créer un fichier de configuration JSON.

Conditions préalables

Les transformations fournies par le client sont enregistrées dans un compte client AWS . Ce compte est propriétaire des transformations et dispose donc de toutes les autorisations nécessaires pour les consulter (recherche et utilisation), les modifier ou les supprimer.

Pour utiliser une transformation personnaliséeAWS Glue Studio, vous devez créer et télécharger deux fichiers dans le compartiment d'actifs Amazon S3 de ce AWS compte :

- Fichier Python : contient la fonction de transformation.
- Fichier JSON : décrit la transformation. Il s'agit également du fichier de configuration requis pour définir la transformation.

Pour associer les fichiers, utilisez le même nom pour les deux. Par exemple :

- maTransformation.json
- maTransformation.py

Vous pouvez éventuellement attribuer une icône personnalisée à votre transformation visuelle personnalisée en fournissant un fichier SVG contenant l'icône. Pour associer les fichiers, utilisez le même nom pour l'icône :

- myTransform.svg

AWS Glue Studio les associera automatiquement à partir de leur nom de fichier. Les noms de fichiers ne peuvent pas être les mêmes pour tous les modules existants.

Convention recommandée pour le nom du fichier de transformation

AWS Glue Studio importe votre fichier en tant que module (par exemple, `import myTransform`) dans votre script de travail. Par conséquent, le nom de votre fichier doit suivre les mêmes règles de dénomination que celles définies pour les noms (identifiants) de variables Python. Plus précisément, ils doivent commencer par une lettre ou un trait de soulignement, puis être entièrement composés de lettres, de chiffres et de traits de and/or soulignement.

> **Note**
>
> Assurez-vous que le nom de votre fichier de transformation n'est pas en conflit avec les modules Python existants déjà chargés (par exemple, `sys, array, copy`, etc.) pour éviter tout problème d'exécution inattendu.

Configuration du compartiment Amazon S3

Les transformations que vous créez sont stockées dans Amazon S3 et appartiennent à votre AWS compte. Pour créer de nouvelles transformations visuelles personnalisées, il vous suffit de charger des fichiers (json et py) dans le dossier des ressources Amazon S3 où sont actuellement stockés tous les scripts de travail (par exemple, `s3://aws-glue-assets-<accountid>-<region>/transforms`). Si vous utilisez une icône personnalisée, chargez-la également. Par défaut, AWS Glue Studio lit tous les fichiers .json à partir du dossier /transforms du même compartiment S3.

Étape 1. Créer un fichier de configuration JSON

Vous devez disposer d'un fichier de configuration JSON pour définir et décrire votre transformation visuelle personnalisée. Le schéma du fichier de configuration est le suivant.

Structure du fichier JSON

Champs

- `name: string` (obligatoire) : nom du système de transformation utilisé pour identifier les transformations. Respecte les mêmes règles de dénomination que celles définies pour les noms (identifiants) de variables Python. Plus précisément, ils doivent commencer par une lettre ou un trait de soulignement, puis être entièrement composés de lettres, de chiffres et de traits de and/or soulignement.
- `displayName: string` (facultatif) : nom de la transformation affiché dans l'éditeur de tâches visuel AWS Glue Studio. Si aucun `displayName` n'est spécifié, `name` est utilisé comme nom de la transformation dans AWS Glue Studio.
- `description: string` (facultatif) : description de la transformation affichée dans AWS Glue Studio ; celle-ci peut faire l'objet d'une recherche.
- `functionName: string` (obligatoire) : nom de la fonction Python utilisé pour identifier la fonction à appeler dans le script Python.
- `path: string` (facultatif) : chemin d'accès Amazon S3 au fichier source Python. S'il n'est pas spécifié, AWS Glue utilise la correspondance des noms de fichiers pour associer les fichiers .json et .py. Par exemple, le nom du fichier JSON, `myTransform.json`, est associé au fichier Python, `myTransform.py`, au même emplacement Amazon S3.
- `parameters: Array of TransformParameter object` (facultatif) : liste des paramètres à afficher lorsque vous les configurez dans l'éditeur visuel AWS Glue Studio.

TransformParameter champs

- `name: string` (obligatoire) : nom du paramètre transmis à la fonction Python en tant qu'argument nommé dans le script de travail. Respecte les mêmes règles de dénomination que celles définies pour les noms (identifiants) de variables Python. Plus précisément, ils doivent commencer par une lettre ou un trait de soulignement, puis être entièrement composés de lettres, de chiffres et de traits de and/or soulignement.

- `displayName: string` (facultatif) : nom de la transformation affiché dans l'éditeur de tâches visuel AWS Glue Studio. Si aucun `displayName` n'est spécifié, `name` est utilisé comme nom de la transformation dans AWS Glue Studio.
- `type: string` (obligatoire) : type de paramètre acceptant les types de données Python courants. Valeurs valides : 'str' | 'int' | 'float' | 'list' | 'bool'.
- `isOptional: boolean` : (facultatif) détermine si le paramètre est facultatif. Par défaut, tous les paramètres sont obligatoires.
- `description: string` (facultatif) : la description est affichée dans AWS Glue Studio pour aider l'utilisateur à configurer le paramètre de transformation.
- `validationType: string` (facultatif) : définit le mode de validation de ce paramètre. Actuellement, seules les expressions régulières sont prises en charge. Par défaut, le type de validation est défini sur `RegularExpression`.
- `validationRule: string` (facultatif) : expression régulière utilisée pour valider les entrées du formulaire avant l'envoi, lorsque `validationType` est défini sur `RegularExpression`. La syntaxe des expressions régulières doit être compatible avec les spécifications RegExp Ecmascript.
- `validationMessage: string` (facultatif) : message à afficher en cas d'échec de la validation.
- `listOptions: An array of TransformParameterListOption object` ou une `string` ou la valeur de chaîne « column » : (facultatif) options à afficher dans le contrôle de sélection unique ou multiple de l'interface utilisateur. Accepte une liste de valeurs séparées par des virgules ou un objet JSON de type `TransformParameterListOption`. Il peut également remplir dynamiquement la liste des colonnes à partir du schéma du nœud parent en spécifiant la valeur de chaîne « column ».
- `listType: string` (facultatif) : définit les types d'options de type = 'list'. Valeurs valides : 'str' | 'int' | 'float' | 'list' | 'bool'. Type de paramètre acceptant les types de données Python courants.

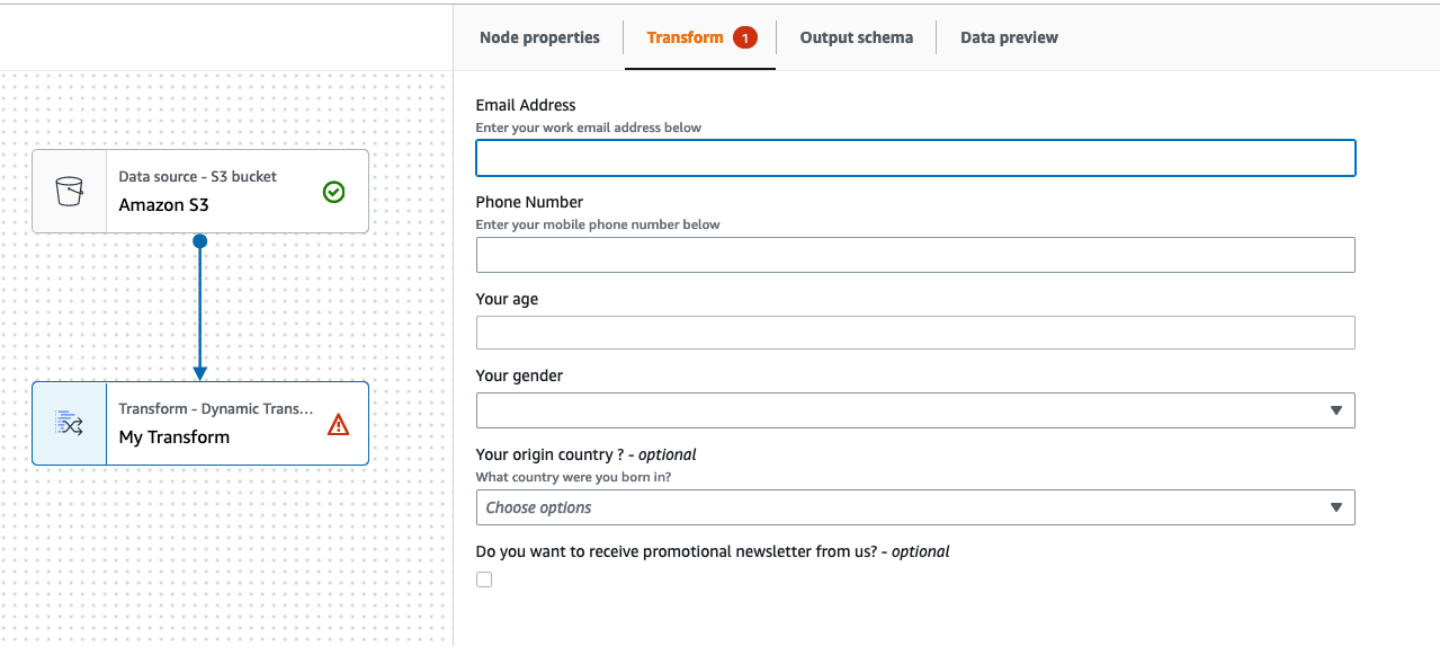
TransformParameterListOption champs

- `value: string | int | float | bool` (obligatoire) : valeur de l'option.
- `label: string` (facultatif) : étiquette d'option affichée dans la liste déroulante de sélection.

Paramètres des transformations dans AWS Glue Studio

Par défaut, les paramètres sont obligatoires, sauf s'ils sont accompagnés de la mention `isOptional` dans le fichier .json. Dans AWS Glue Studio, les paramètres sont affichés dans l'onglet Transform

(Transformation). L'exemple présente des paramètres définis par l'utilisateur, tels que Email Address (Adresse e-mail), Phone Number (Numéro de téléphone), Your age (Votre âge), Your gender (Votre sexe) et Your origin (Votre pays d'origine).

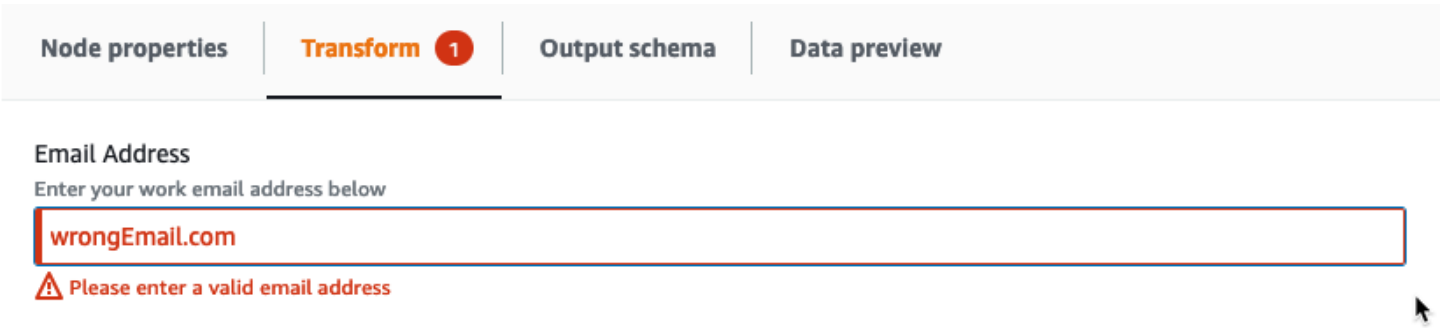


Vous pouvez appliquer certaines validations dans AWS Glue Studio en utilisant des expressions régulières dans le fichier json, en définissant le paramètre `validationRule` et en spécifiant un message de validation dans `validationMessage`.

```
    "validationRule": "^\\(?(\\d{3})\\)?[- ]?(\\d{3})[- ]?(\\d{4})$",
    "validationMessage": "Please enter a valid US number"
```

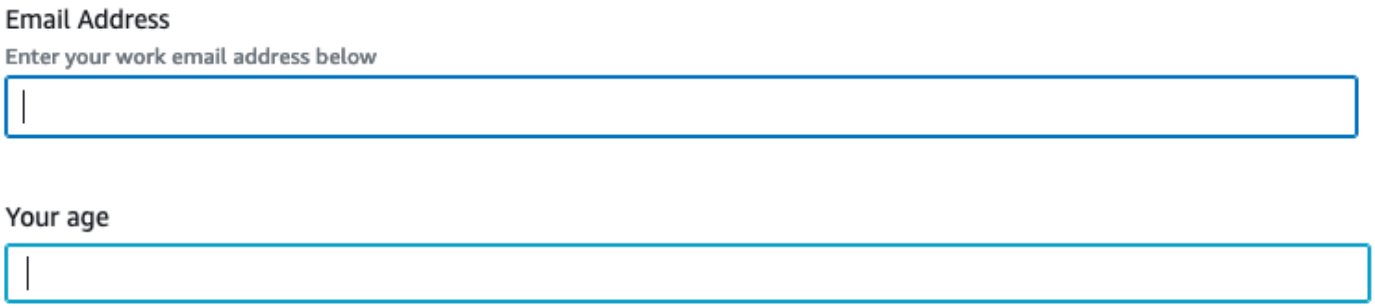
> **Note**
>
> La validation ayant lieu dans le navigateur, la syntaxe de vos expressions régulières doit être compatible avec les spécifications RegExp Ecmascript. La syntaxe Python n'est pas prise en charge pour ces expressions régulières.

L'ajout d'une validation empêchera l'utilisateur d'enregistrer la tâche avec une entrée incorrecte. AWS Glue Studio affiche le message de validation tel qu'il apparaît dans l'exemple :

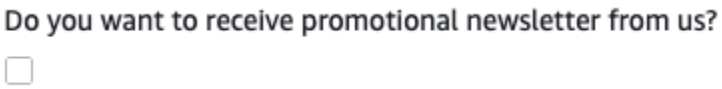


Les paramètres sont affichés dans AWS Glue Studio en fonction de la configuration des paramètres.

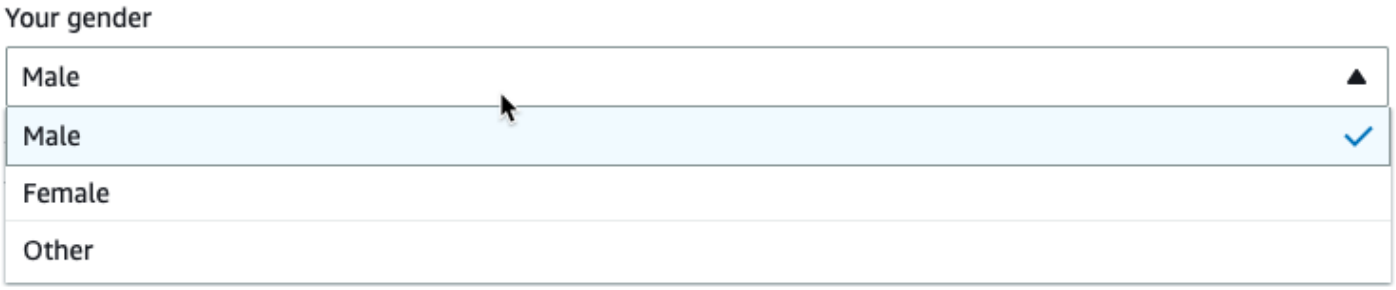
- Lorsque le `type` correspond à `str`, `int` ou `float`, un champ de saisie de texte s'affiche. Par exemple, la capture d'écran illustre les champs de saisie des paramètres « Adresse e-mail » et « Votre âge ».



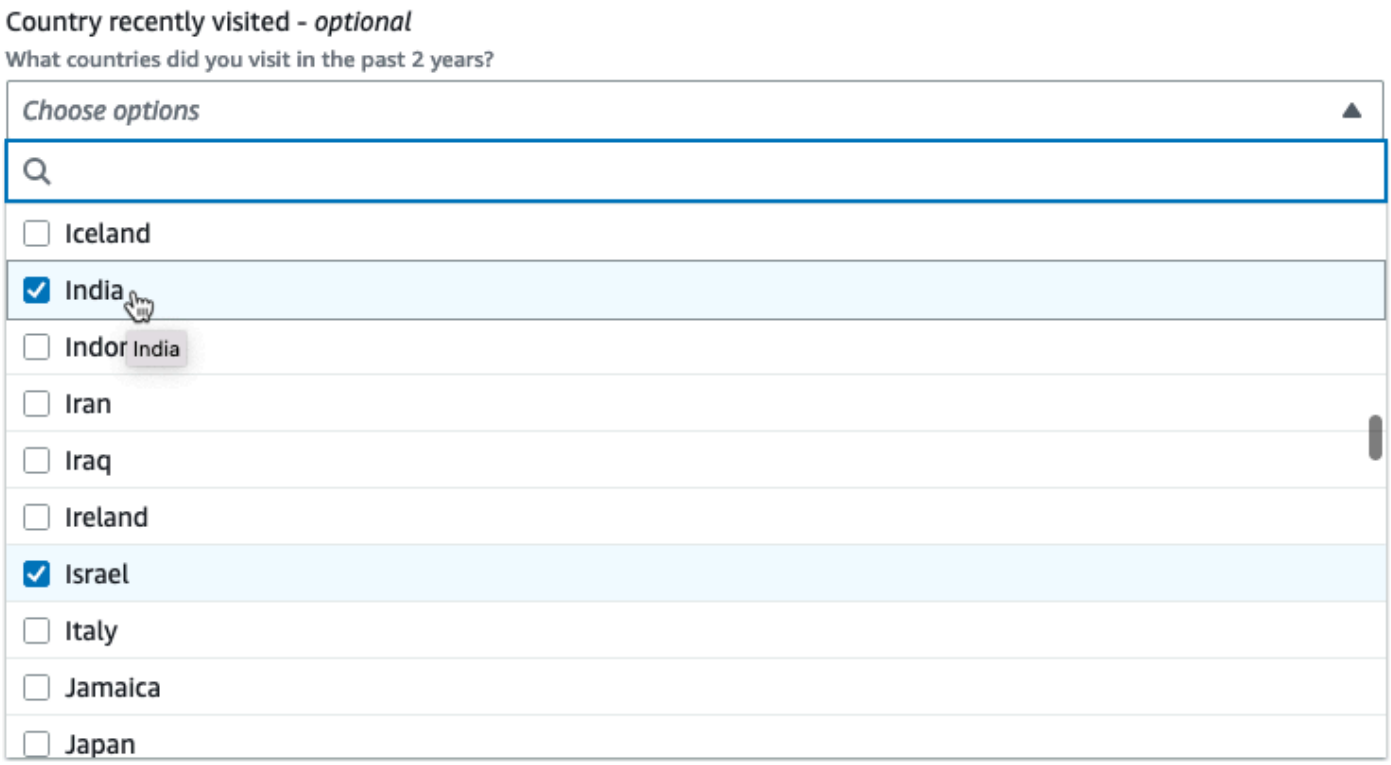
- Lorsque le `type` correspond à `bool`, une case à cocher s'affiche.



- Lorsque le `type` correspond à `str` et que `listOptions` est fourni, une liste de sélection unique s'affiche.



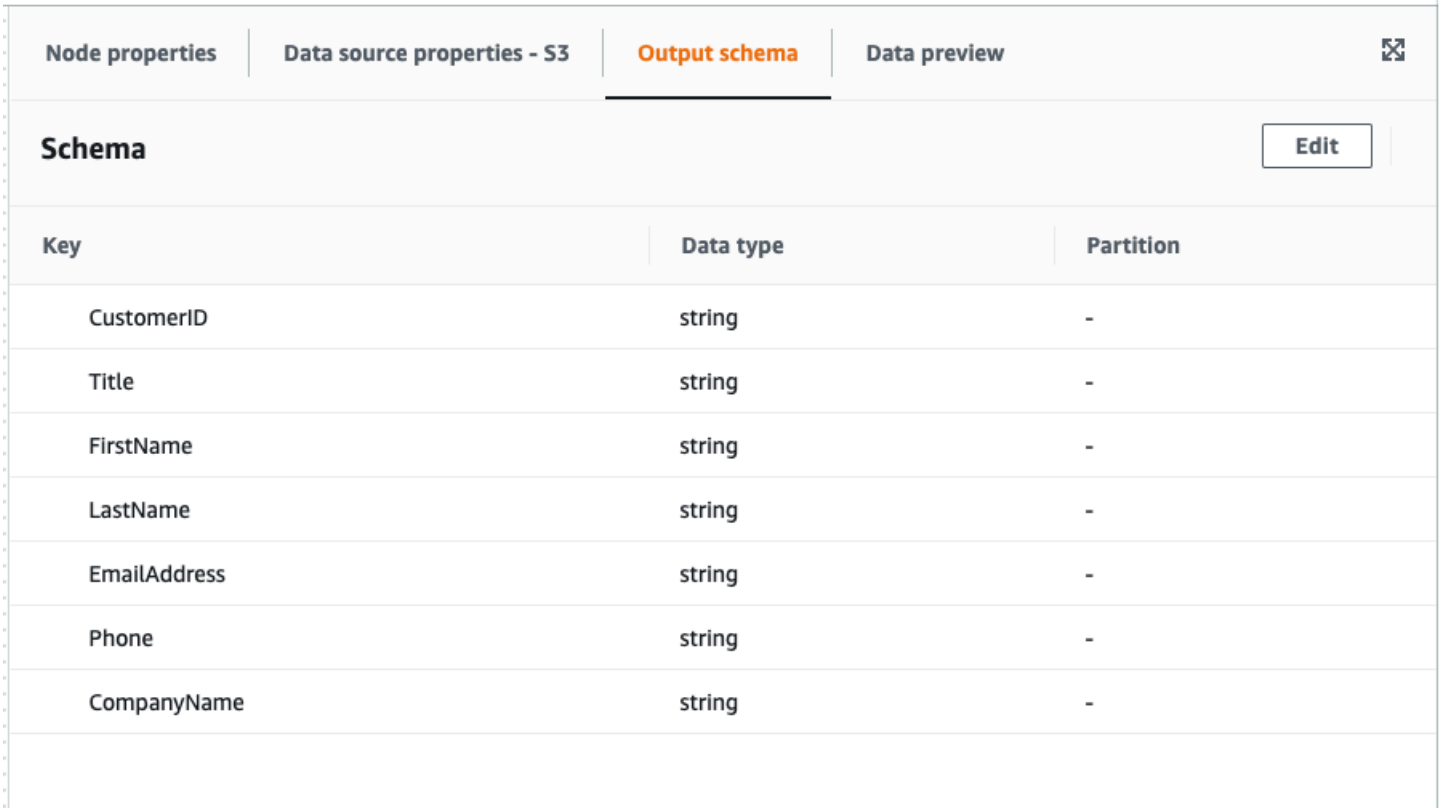
- Lorsque le `type` correspond à `list` et que `listOptions` et `listType` sont fournis, une liste de sélection multiple s'affiche.



Afficher un sélecteur de colonne en tant que paramètre

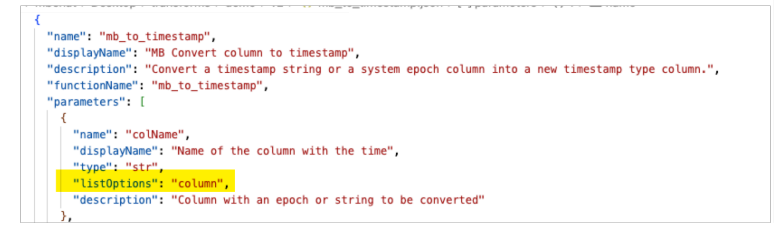
Si la configuration nécessite que l'utilisateur choisisse une colonne dans le schéma, vous pouvez afficher un sélecteur de colonne afin que l'utilisateur n'ait pas à saisir le nom de la colonne. En définissant le champ `listOptions` sur « column », AWS Glue Studio affiche dynamiquement un sélecteur de colonne basé sur le schéma de sortie du nœud parent. AWS Glue Studio peut afficher un sélecteur de colonne unique ou multiple.

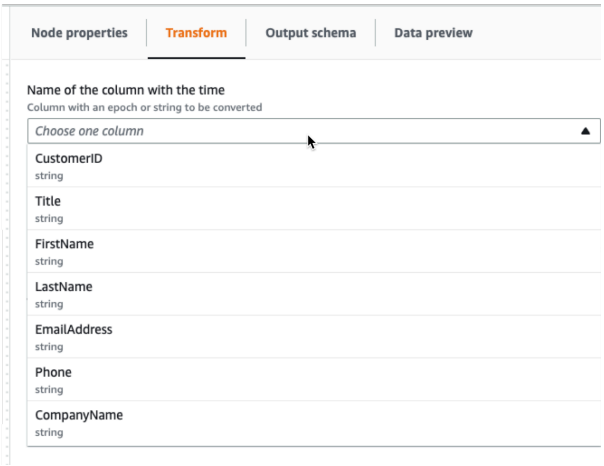
L'exemple suivant utilise le schéma :



Pour définir votre paramètre de transformation visuelle personnalisée afin d'afficher une seule colonne :

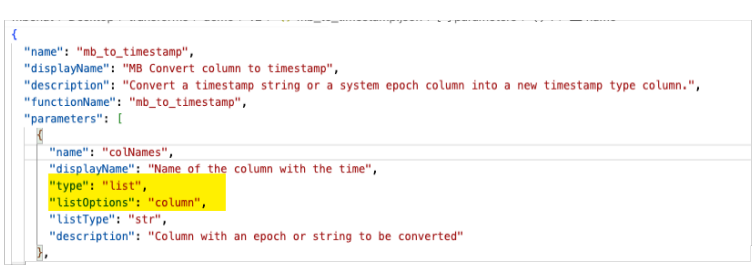
1. Dans votre fichier JSON, pour l'objet `parameters`, définissez la valeur `listOptions` sur « column ». Cela permet à un utilisateur de choisir une colonne à partir d'une liste de sélection dans AWS Glue Studio.

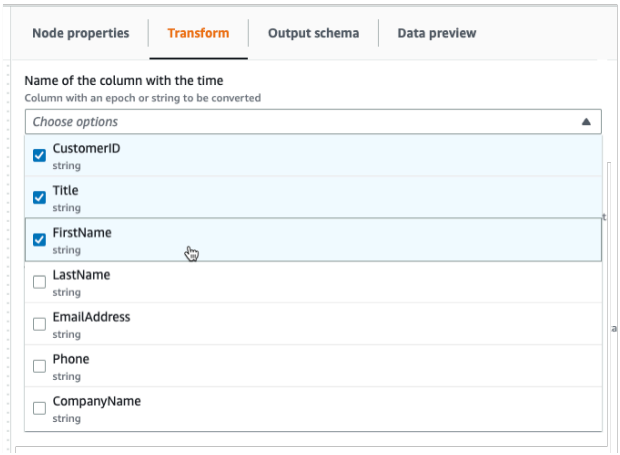




2. Vous pouvez également autoriser la sélection de plusieurs colonnes en définissant le paramètre comme suit :

- `listOptions: "column"`
- `type: "list"`





Étape 2. Implémenter la logique de transformation

> **Note**
>
> Les transformations visuelles personnalisées ne prennent en charge que les scripts Python. Scala n'est pas pris en charge.

Pour ajouter le code qui implémente la fonction définie par le fichier de configuration .json, il est recommandé de placer le fichier Python au même emplacement que le fichier .json, en lui attribuant le même nom mais avec l'extension « .py ». AWS Glue Studio associe automatiquement les fichiers .json et .py afin que vous n'ayez pas à spécifier le chemin du fichier Python dans le fichier de configuration.

Dans le fichier Python, ajoutez la fonction déclarée, avec les paramètres nommés configurés, et enregistrez-la pour qu'elle soit utilisée dans `DynamicFrame`. Voici un exemple de fichier Python :

```
from awsglue import DynamicFrame

# self refers to the DynamicFrame to transform,
# the parameter names must match the ones defined in the config
# if it's optional, need to provide a default value
def myTransform(self, email, phone, age=None, gender="",
```

```
                      country="", promotion=False):
   resulting_dynf = # do some transformation on self
   return resulting_dynf

DynamicFrame.myTransform = myTransform
```

Il est recommandé d'utiliser un bloc-notes AWS Glue, car il s'agit de l'outil le plus rapide pour développer et tester le code Python. Consultez Mise en route avec les blocs-notes dans AWS Glue Studio.

Pour illustrer l'implémentation de la logique de transformation, la transformation visuelle personnalisée présentée dans l'exemple ci-dessous permet de filtrer les données entrantes afin de ne conserver que les données relatives à un État américain en particulier. Le fichier .json contient le paramètre `custom_filter_state` pour `functionName` ainsi que deux arguments (« state » et « colName » avec le type « str »).

L'exemple de fichier de configuration .json est le suivant :

```
{
"name": "custom_filter_state",
"displayName": "Filter State",
"description": "A simple example to filter the data to keep only the state indicated.",
"functionName": "custom_filter_state",
"parameters": [
   {
    "name": "colName",
    "displayName": "Column name",
    "type": "str",
    "description": "Name of the column in the data that holds the state postal code"
   },
   {
    "name": "state",
    "displayName": "State postal code",
    "type": "str",
    "description": "The postal code of the state whole rows to keep"
   }
  ]
}
```

Pour implémenter le script compagnon en Python

1. Démarrez un bloc-notes AWS Glue et exécutez la cellule initiale fournie pour lancer la session. L'exécution de la cellule initiale crée les composants de base nécessaires.
2. Créez une fonction qui procède au filtrage, comme décrit dans l'exemple, et enregistrez-la sur `DynamicFrame`. Copiez le code ci-dessous et collez-le dans une cellule du bloc-notes AWS Glue.

```
from awsglue import DynamicFrame

def custom_filter_state(self, colName, state):
    return self.filter(lambda row: row[colName] == state)

DynamicFrame.custom_filter_state = custom_filter_state
```

3. Créez ou chargez des exemples de données pour tester le code dans la même cellule ou dans une nouvelle cellule. Si vous ajoutez les exemples de données dans une nouvelle cellule, n'oubliez pas d'exécuter la cellule. Par exemple :

```
# A few of rows of sample data to test
data_sample = [
    {"state": "CA", "count": 4},
    {"state": "NY", "count": 2},
    {"state": "WA", "count": 3}
]
df1 = glueContext.sparkSession.sparkContext.parallelize(data_sample).toDF()
dynf1 = DynamicFrame.fromDF(df1, glueContext, None)
```

4. Test pour valider « custom_filter_state » avec différents arguments :

```
[14]: dynf1.custom_filter_state("state", "NY").show()
      {"count": 2, "state": "NY"}
```

5. Après avoir effectué plusieurs tests, enregistrez le code avec l'extension .py et attribuez au fichier .py le même nom que le fichier .json. Les fichiers .py et .json doivent se trouver dans le même dossier de transformations.

Copiez le code suivant, collez-le dans un fichier et renommez celui-ci avec l'extension de fichier .py.

```
from awsglue import DynamicFrame

def custom_filter_state(self, colName, state):
    return self.filter(lambda row: row[colName] == state)

DynamicFrame.custom_filter_state = custom_filter_state
```

6. Dans AWS Glue Studio, ouvrez une tâche visuelle et ajoutez-y la transformation en la sélectionnant dans la liste des transformations disponibles.

   Pour réutiliser cette transformation dans le code d'un script Python, ajoutez le chemin d'accès Amazon S3 au fichier .py dans la tâche située sous « Referenced files path » (Chemin des fichiers référencés) et, dans le script, importez le nom du fichier Python (sans l'extension) en l'ajoutant en haut du fichier. Par exemple : `import` <nom du fichier (sans l'extension)>

Étape 3. Valider les transformations visuelles personnalisées et résoudre les problèmes liés à celles-ci dans AWS Glue Studio

AWS Glue Studio valide le fichier de configuration JSON avant le chargement des transformations visuelles personnalisées dans AWS Glue Studio. La validation consiste à rechercher ce qui suit :

- Présence de champs obligatoires
- Validation du format JSON
- Paramètres incorrects ou non valides
- Présence des fichiers .py et .json au même emplacement Amazon S3
- Correspondance des noms de fichiers .py et .json.

Si la validation aboutit, la transformation est répertoriée dans la liste des Actions disponibles de l'éditeur visuel. Si une icône personnalisée a été fournie, elle doit être visible à côté de l'action.

Si la validation échoue, AWS Glue Studio ne charge pas la transformation visuelle personnalisée.

Étape 4 : Mettre à jour les transformations visuelles personnalisées selon les besoins

Une fois créé et utilisé, le script de transformation peut être mis à jour tant que la transformation respecte la définition json correspondante :

- Le nom utilisé lors de l'attribution d'une DynamicFrame trop grande valeur correspond au json`functionName`.
- Les arguments de la fonction doivent être définis dans le fichier json comme décrit dans Étape 1. Créer un fichier de configuration JSON.
- Le chemin d'accès Amazon S3 au fichier Python ne peut pas être modifié, car les tâches en dépendent directement.

> **Note**
>
> Si des mises à jour sont nécessaires, veillez à ce que le script et le fichier .json soient mis à jour en conséquence et à ce que toutes les tâches visuelles soient correctement enregistrées avec la nouvelle transformation. Si les tâches visuelles ne sont pas enregistrées à l'issue des mises à jour, ces dernières ne sont ni appliquées ni validées. Si le fichier de script Python est renommé ou s'il n'est pas placé à côté du fichier .json, vous devez spécifier le chemin d'accès complet dans le fichier .json.

Icône personnalisée

Si vous déterminez que l'icône par défaut de votre action ne la distingue pas visuellement dans le cadre de vos flux de travail, vous pouvez fournir une icône personnalisée, comme décrit dans the section called “Premiers pas avec les transformations visuelles personnalisées”. Vous pouvez mettre à jour l'icône en mettant à jour le fichier SVG correspondant hébergé sur Amazon S3.

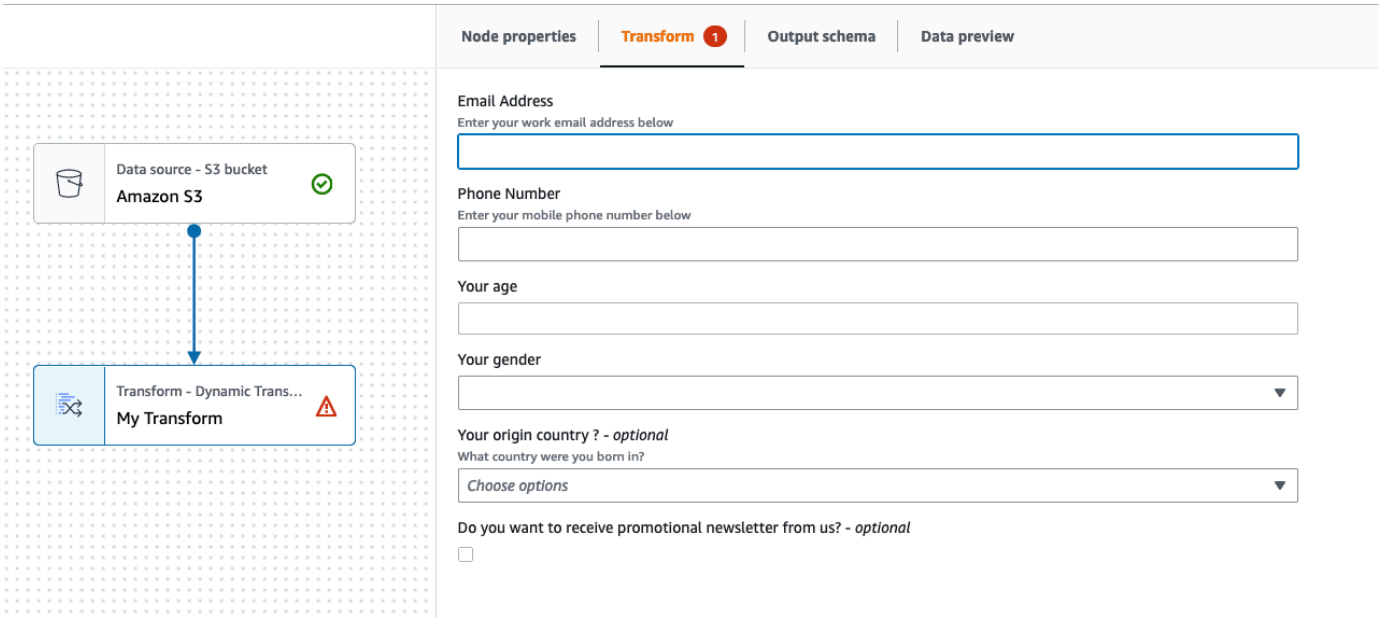
Pour de meilleurs résultats, concevez votre image pour qu'elle soit affichée en 32 × 32 pixels en suivant les directives de Cloudscape Design System. Pour plus d'informations sur les directives Cloudscape, consultez la documentation Cloudscape.

Étape 5. Utiliser des transformations visuelles personnalisées dans AWS Glue Studio

Pour utiliser une transformation visuelle personnalisée dans AWS Glue Studio, vous devez charger le fichier de configuration et le fichier source, puis sélectionner la transformation dans le menu Action.

Tous les paramètres nécessitant des valeurs ou des entrées sont disponibles dans l'onglet Transform (Transformation).

1. Chargez les deux fichiers (fichier source Python et fichier de configuration JSON) dans le dossier de ressources Amazon S3 où sont stockés les scripts de travail. Par défaut, AWS Glue extrait tous les fichiers JSON à partir du dossier /transform du même compartiment Amazon S3.
2. Dans le menu Action, sélectionnez la transformation visuelle personnalisée. Celle-ci porte le `displayName` ou le nom de la transformation que vous avez spécifié dans le fichier de configuration .json.
3. Entrez des valeurs pour tous les paramètres qui ont été définis dans le fichier de configuration.



Exemples d'utilisation

L'exemple suivant présente tous les paramètres possibles pour un fichier de configuration .json.

```
{
  "name": "MyTransform",
  "displayName": "My Transform",
  "description": "This transform description will be displayed in UI",
  "functionName": "myTransform",
  "parameters": [
      {
      "name": "email",
      "displayName": "Email Address",
      "type": "str",
```

```
      "description": "Enter your work email address below",
      "validationType": "RegularExpression",
      "validationRule": "^\\w+([\\.-]?\\w+)*@\\w+([\\.-]?\\w+)*(\\.\\w{2,3})+$",
      "validationMessage": "Please enter a valid email address"
    },
    {
      "name": "phone",
      "displayName": "Phone Number",
      "type": "str",
      "description": "Enter your mobile phone number below",
      "validationRule": "^\\(?(\\d{3})\\)?[- ]?(\\d{3})[- ]?(\\d{4})$",
      "validationMessage": "Please enter a valid US number"
    },
    {
      "name": "age",
      "displayName": "Your age",
      "type": "int",
      "isOptional": true
    },
    {
      "name": "gender",
      "displayName": "Your gender",
      "type": "str",
      "listOptions": [
            {"label": "Male", "value": "male"},
            {"label": "Female", "value": "female"},
            {"label": "Other", "value": "other"}
        ],
      "isOptional": true
    },
    {
      "name": "country",
      "displayName": "Your origin country ?",
      "type": "list",
      "listOptions": "Afghanistan,Albania,Algeria,American
Samoa,Andorra,Angola,Anguilla,Antarctica,Antigua and
Barbuda,Argentina,Armenia,Aruba,Australia,Austria,Azerbaijan,Bahamas,Bahrain,Bangladesh,Barbad
and Herzegovina,Botswana,Bouvet Island,Brazil,British Indian Ocean Territory,Brunei
Darussalam,Bulgaria,Burkina Faso,Burundi,Cambodia,Cameroon,Canada,Cape
Verde,Cayman Islands,Central African Republic,Chad,Chile,China,Christmas
Island,Cocos (Keeling Islands),Colombia,Comoros,Congo,Cook Islands,Costa
Rica,Cote D'Ivoire (Ivory Coast),Croatia (Hrvatska,Cuba,Cyprus,Czech
Republic,Denmark,Djibouti,Dominica,Dominican Republic,East Timor,Ecuador,Egypt,El
Salvador,Equatorial Guinea,Eritrea,Estonia,Ethiopia,Falkland Islands (Malvinas),Faroe
```

```
 Islands,Fiji,Finland,France,France,Metropolitan,French Guiana,French Polynesia,French
 Southern
 Territories,Gabon,Gambia,Georgia,Germany,Ghana,Gibraltar,Greece,Greenland,Grenada,Guadeloupe,G
Bissau,Guyana,Haiti,Heard and McDonald Islands,Honduras,Hong
 Kong,Hungary,Iceland,India,Indonesia,Iran,Iraq,Ireland,Israel,Italy,Jamaica,Japan,Jordan,Kazak
 (North),Korea
 (South),Kuwait,Kyrgyzstan,Laos,Latvia,Lebanon,Lesotho,Liberia,Libya,Liechtenstein,Lithuania,Lu
 Islands,Martinique,Mauritania,Mauritius,Mayotte,Mexico,Micronesia,Moldova,Monaco,Mongolia,Mont
 Antilles,New Caledonia,New Zealand,Nicaragua,Niger,Nigeria,Niue,Norfolk
 Island,Northern Mariana Islands,Norway,Oman,Pakistan,Palau,Panama,Papua
 New Guinea,Paraguay,Peru,Philippines,Pitcairn,Poland,Portugal,Puerto
 Rico,Qatar,Reunion,Romania,Russian Federation,Rwanda,Saint Kitts and Nevis,Saint
 Lucia,Saint Vincent and The Grenadines,Samoa,San Marino,Sao Tome and Principe,Saudi
 Arabia,Senegal,Seychelles,Sierra Leone,Singapore,Slovak Republic,Slovenia,Solomon
 Islands,Somalia,South Africa,S. Georgia and S. Sandwich Isls.,Spain,Sri
 Lanka,St. Helena,St. Pierre and Miquelon,Sudan,Suriname,Svalbard and Jan Mayen
 Islands,Swaziland,Sweden,Switzerland,Syria,Tajikistan,Tanzania,Thailand,Togo,Tokelau,Tonga,Tri
 and Tobago,Tunisia,Turkey,Turkmenistan,Turks and Caicos
 Islands,Tuvalu,Uganda,Ukraine,United Arab Emirates,United Kingdom
 (Britain / UK),United States of America (USA),US Minor Outlying
 Islands,Uruguay,Uzbekistan,Vanuatu,Vatican City State (Holy See),Venezuela,Viet
 Nam,Virgin Islands (British),Virgin Islands (US),Wallis and Futuna Islands,Western
 Sahara,Yemen,Yugoslavia,Zaire,Zambia,Zimbabwe",
      "description": "What country were you born in?",
      "listType": "str",
      "isOptional": true
    },
    {
      "name": "promotion",
      "displayName": "Do you want to receive promotional newsletter from us?",
      "type": "bool",
      "isOptional": true
    }
  ]
}
```

Exemples de scripts visuels personnalisés

Les exemples suivants effectuent des transformations équivalentes. Cela dit, le deuxième exemple (SparkSQL) est le plus propre et le plus efficace, suivi de l'exemple d'UDF Pandas et enfin du mappage de bas niveau du premier exemple. L'exemple suivant est une illustration complète de transformation simple permettant d'ajouter deux colonnes :

```
from awsglue import DynamicFrame

# You can have other auxiliary variables, functions or classes on this file, it won't
 affect the runtime
def record_sum(rec, col1, col2, resultCol):
    rec[resultCol] = rec[col1] + rec[col2]
    return rec


# The number and name of arguments must match the definition on json config file
# (expect self which is the current DynamicFrame to transform
# If an argument is optional, you need to define a default value here
#  (resultCol in this example is an optional argument)
def custom_add_columns(self, col1, col2, resultCol="result"):
    # The mapping will alter the columns order, which could be important
    fields = [field.name for field in self.schema()]
    if resultCol not in fields:
        # If it's a new column put it at the end
        fields.append(resultCol)
    return self.map(lambda record: record_sum(record, col1, col2,
 resultCol)).select_fields(paths=fields)


# The name we assign on DynamicFrame must match the configured "functionName"
DynamicFrame.custom_add_columns = custom_add_columns
```

L'exemple suivant illustre une transformation équivalente utilisant l'API SparkSQL.

```
from awsglue import DynamicFrame

# The number and name of arguments must match the definition on json config file
# (expect self which is the current DynamicFrame to transform
# If an argument is optional, you need to define a default value here
#  (resultCol in this example is an optional argument)
def custom_add_columns(self, col1, col2, resultCol="result"):
    df = self.toDF()
    return DynamicFrame.fromDF(
        df.withColumn(resultCol, df[col1] + df[col2]) # This is the conversion logic
        , self.glue_ctx, self.name)


# The name we assign on DynamicFrame must match the configured "functionName"
```

```
DynamicFrame.custom_add_columns = custom_add_columns
```

L'exemple suivant utilise les mêmes transformations mais en utilisant une UDF Pandas, plus efficace qu'une UDF simple. Pour plus d'informations sur l'écriture de pandas, UDFs consultez la documentation SQL d'Apache Spark.

```
from awsglue import DynamicFrame
import pandas as pd
from pyspark.sql.functions import pandas_udf

# The number and name of arguments must match the definition on json config file
# (expect self which is the current DynamicFrame to transform
# If an argument is optional, you need to define a default value here
#  (resultCol in this example is an optional argument)
def custom_add_columns(self, col1, col2, resultCol="result"):
    @pandas_udf("integer")  # We need to declare the type of the result column
    def add_columns(value1: pd.Series, value2: pd.Series) # pd.Series:
        return value1 + value2

    df = self.toDF()
    return DynamicFrame.fromDF(
        df.withColumn(resultCol, add_columns(col1, col2)) # This is the conversion
 logic
        , self.glue_ctx, self.name)

# The name we assign on DynamicFrame must match the configured "functionName"
DynamicFrame.custom_add_columns = custom_add_columns
```

Vidéo

La vidéo suivante contient une présentation des transformations visuelles personnalisées et de leur utilisation.

Utilisation d'infrastructures de lac de données avec AWS Glue Studio

Présentation de

Les infrastructures de lac de données open source simplifient le traitement incrémentiel des fichiers stockés dans les lacs de données créés sur Amazon S3. AWS Glue 3.0 et versions ultérieures prennent en charge les infrastructures de stockage de lacs de données open source suivantes :

- Apache Hudi
- Linux Foundation Delta Lake
- Apache Iceberg

À partir de AWS Glue 4.0, AWS Glue assure une prise en charge native de ces infrastructures, ce qui vous permet de lire et d'écrire les données que vous stockez dans Amazon S3 de manière cohérente sur le plan transactionnel. Il n'est pas nécessaire d'installer un connecteur distinct ou d'effectuer des étapes de configuration supplémentaires pour utiliser ces infrastructures dans les tâches AWS Glue.

Les cadres de lac de données peuvent être utilisés comme source ou cible dans AWS Glue Studio par l'intermédiaire des tâches de l'éditeur de script Spark. Pour plus d'informations sur l'utilisation d'Apache Hudi, d'Apache Iceberg et de Delta Lake, consultez : Using data lake frameworks with AWS Glue ETL jobs.

Création de formats de tableau ouverts à partir d'une source de AWS Glue streaming

AWS Glue les tâches ETL de streaming consomment en permanence des données provenant de sources de streaming, nettoient et transforment les données en vol, et les rendent disponibles pour analyse en quelques secondes.

AWS propose une large gamme de services pour répondre à vos besoins. Un service de réplication de base de AWS données tel que Database Migration Service peut répliquer les données de vos systèmes sources vers Amazon S3, qui héberge généralement la couche de stockage du lac de données. Bien qu'il soit simple d'appliquer des mises à jour à un système de gestion de base de données relationnelle (RDBMS) qui soutient une application source en ligne, il est difficile d'appliquer ce processus CDC à vos lacs de données. Les cadres de gestion de données open source simplifient le traitement incrémentiel des données et le développement de pipelines de données et constituent une bonne option pour résoudre ce problème.

Pour en savoir plus, consultez :

- Créez un lac de données near-real-time transactionnel basé sur Apache HUDI à l'aide du streaming AWS Glue
- Créer en temps réel un lac de données Apache Iceberg aligné sur le RGPD

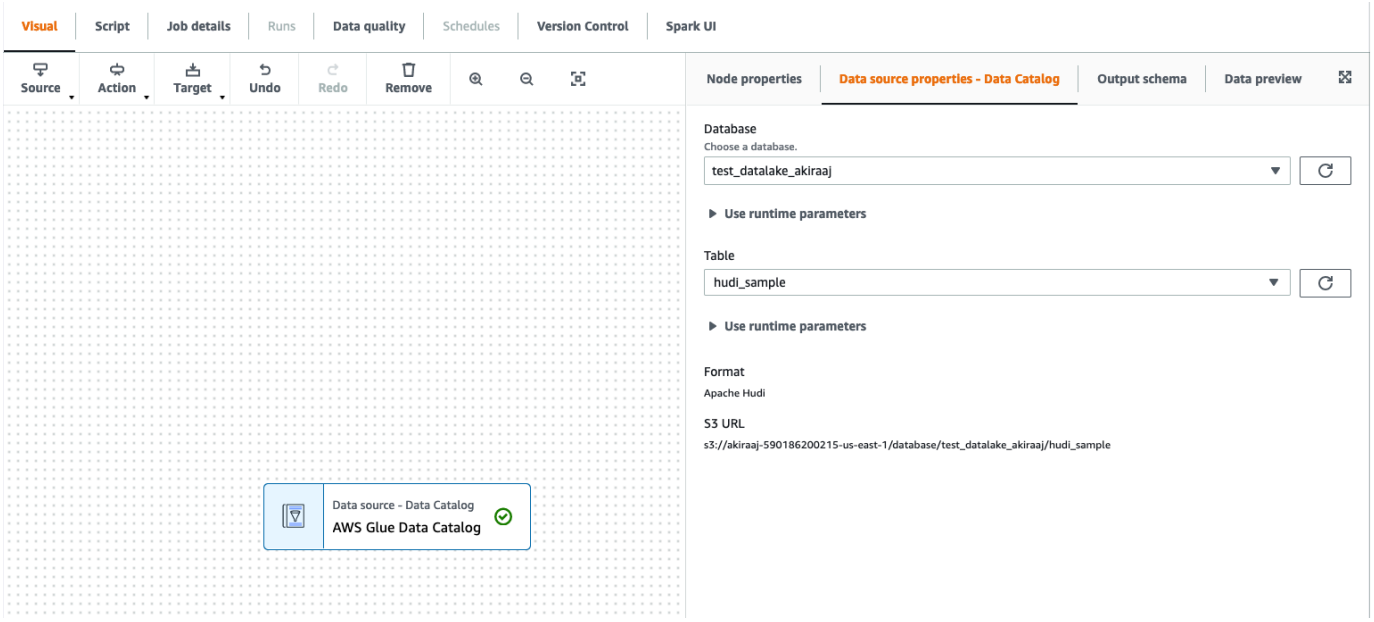
Utilisation du cadre Hudi dans AWS Glue Studio

Lorsque vous créez ou modifiez une tâche, AWS Glue Studio ajoute automatiquement les bibliothèques Hudi correspondantes en fonction de la version d'AWS Glue que vous utilisez. Pour plus d'informations, veuillez consulter la rubrique Utilisation du cadre Hudi dans AWS Glue.

Utilisation du cadre Apache Hudi dans les sources de données du catalogue de données

Pour ajouter un format de source de données Hudi à une tâche :

1. Dans le menu Source, choisissez Catalogue de données AWS Glue Studio.
2. Dans l'onglet Propriétés de source de données, choisissez une base de données et une table.
3. AWS Glue Studio affiche le type de format comme étant Apache Hudi et l'URL Amazon S3.

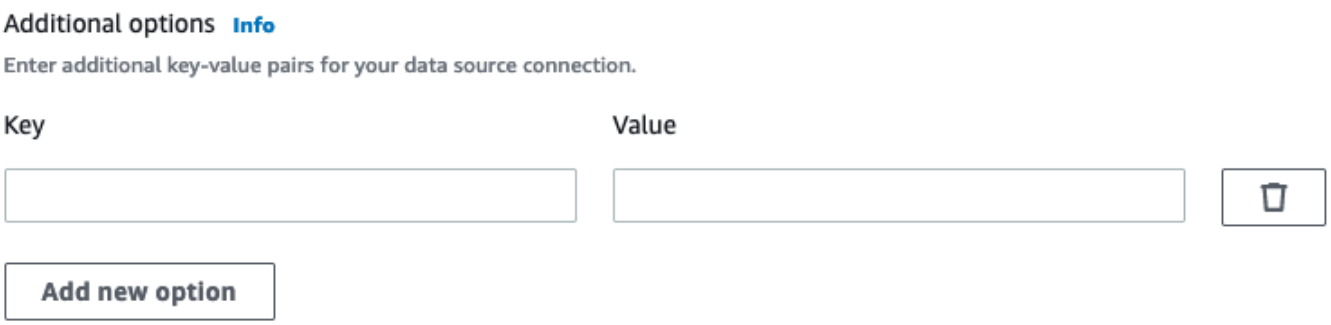


Utilisation du cadre Hudi dans les sources de données Amazon S3

1. Dans le menu Source, choisissez Amazon S3.
2. Si vous choisissez la table du catalogue de données comme type de source Amazon S3, choisissez une base de données et une table.
3. AWS Glue Studio affiche le format comme étant Apache Hudi et l'URL Amazon S3.
4. Si vous choisissez l'emplacement Amazon S3 comme Type de source Amazon S3, choisissez l'URL Amazon S3 en cliquant sur Parcourir Amazon S3.
5. Dans Format de données, sélectionnez Apache Hudi.

> **Note**
>
> Si AWS Glue Studio n'est pas en mesure de déduire le schéma à partir du dossier ou du fichier Amazon S3 que vous avez sélectionné, choisissez Options supplémentaires pour sélectionner un nouveau dossier ou fichier.
> Dans Options supplémentaires, choisissez l'une des options suivantes sous Inférence de schéma :
>
> - Laisser AWS Glue Studio choisir automatiquement un exemple de fichier : AWS Glue Studio choisit un exemple de fichier dans l'emplacement Amazon S3 afin que le schéma puisse être déduit. Dans le champ Fichier auto-échantillonné, vous pouvez afficher le fichier sélectionné automatiquement.
> - Choisir un exemple de fichier d'Amazon S3 : choisissez le fichier Amazon S3 à utiliser en cliquant sur Parcourir Amazon S3.

6. Cliquez sur Déduire un schéma. Vous pouvez ensuite consulter le schéma de sortie en cliquant sur l'onglet Schéma de sortie.
7. Choisissez Options supplémentaires pour saisir une paire clé-valeur.



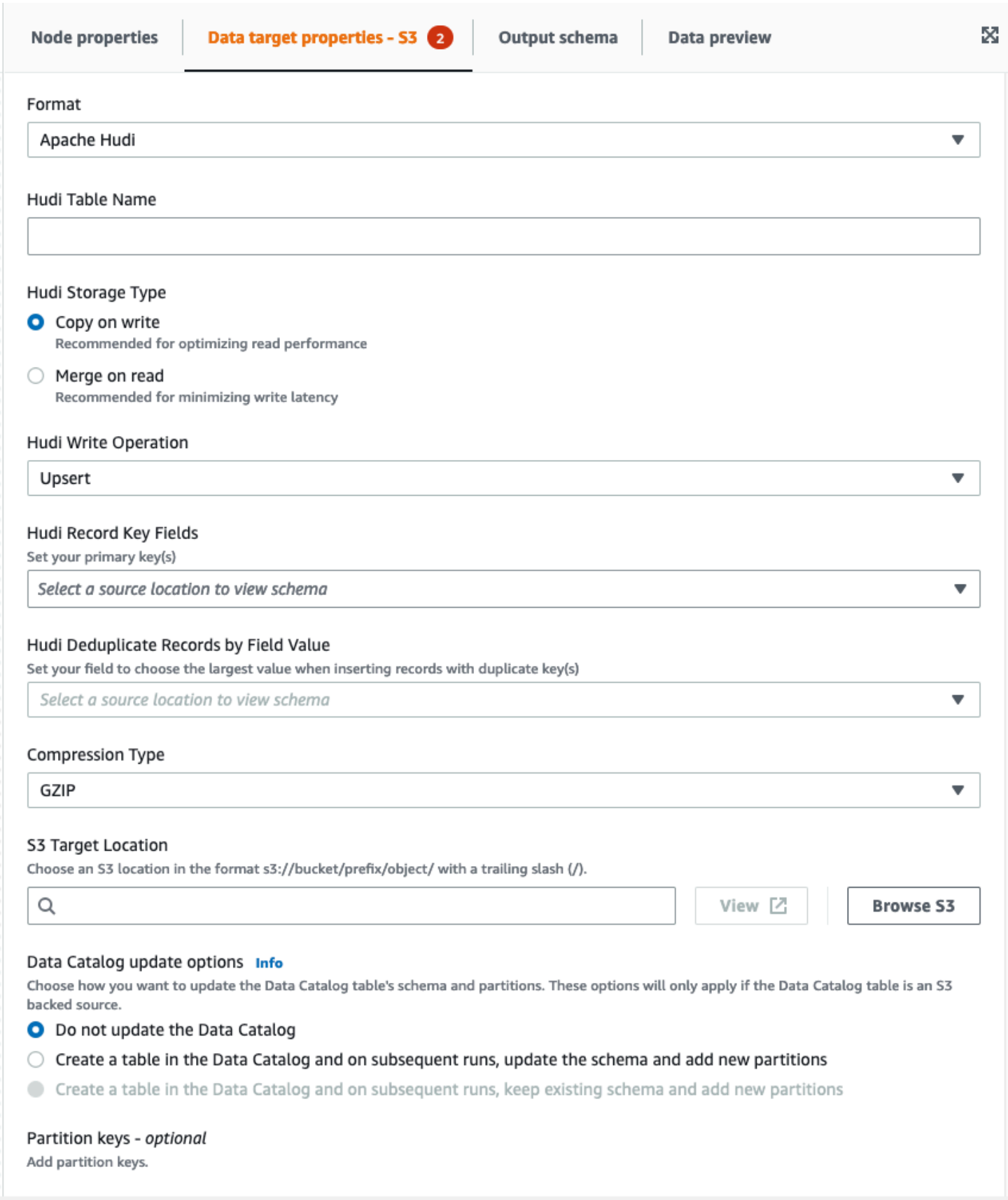
Utilisation du cadre Apache Hudi dans les cibles de données

Utilisation du cadre Apache Hudi dans les cibles de données du catalogue de données

1. Dans le menu Cible, choisissez Catalogue de données AWS Glue Studio.
2. Dans l'onglet Propriétés de source de données, choisissez une base de données et une table.
3. AWS Glue Studio affiche le type de format comme étant Apache Hudi et l'URL Amazon S3.

Utilisation du cadre Apache Hudi dans les cibles de données Amazon S3

Saisissez des valeurs ou sélectionnez-les parmi les options disponibles pour configurer le format Apache Hudi. Pour plus d'informations sur Apache Hudi, veuillez consulter la documentation Apache Hudi.


Node properties
Data target properties - S3 2
Output schema
Data preview
Format
Apache Hudi
Hudi Table Name
Hudi Storage Type
Copy on write
Recommended for optimizing read performance
Merge on read
Recommended for minimizing write latency
Hudi Write Operation
Upsert
Hudi Record Key Fields
Set your primary key(s)
Select a source location to view schema
Hudi Deduplicate Records by Field Value
Set your field to choose the largest value when inserting records with duplicate key(s)
Select a source location to view schema
Compression Type
GZIP
S3 Target Location
Choose an S3 location in the format s3://bucket/prefix/object/ with a trailing slash (/).
View
Browse S3
Data Catalog update options Info
Choose how you want to update the Data Catalog table's schema and partitions. These options will only apply if the Data Catalog table is an S3 backed source.
Do not update the Data Catalog
Create a table in the Data Catalog and on subsequent runs, update the schema and add new partitions
Create a table in the Data Catalog and on subsequent runs, keep existing schema and add new partitions
Partition keys - optional
Add partition keys.

- Nom de la table Hudi : il s'agit du nom de votre table Hudi.
- Type de stockage Hudi : choisissez parmi deux options :
  - Copie à l'écriture : recommandé pour optimiser les performances de lecture. Il s'agit du type de stockage Hudi par défaut. Chaque mise à jour crée une nouvelle version des fichiers lors d'une écriture.
  - Fusion en lecture : recommandée pour minimiser la latence d'écriture. Les mises à jour sont consignées dans des fichiers delta basés sur les lignes et sont compressées si nécessaire pour créer de nouvelles versions des fichiers en colonnes.
- Opération d'écriture Hudi : choisissez parmi les options suivantes :
  - Upsert (mise à jour/insertion) : il s'agit de l'opération par défaut lors de laquelle les enregistrements d'entrée sont d'abord balisés comme des insertions ou des mises à jour en recherchant l'index. Recommandé lorsque vous mettez à jour des données existantes.
  - Insertion : permet d'insérer des enregistrements mais ne vérifie pas s'ils existent déjà, ce qui peut entraîner des doublons.
  - Insertion en bloc : permet d'insérer des enregistrements et est recommandée pour les grandes quantités de données.
- Champs de clé d'enregistrement Hudi : utilisez la barre de recherche pour rechercher et choisir les clés d'enregistrement primaires. Les enregistrements dans Hudi sont identifiés par une clé primaire qui est une paire de clé d'enregistrement et de chemin de partition à laquelle appartient l'enregistrement.
- Champ de précombinaison Hudi : il s'agit du champ utilisé lors de la précombinaison avant l'écriture réelle. Lorsque deux enregistrements ont la même valeur de clé, AWS Glue Studio sélectionne celui avec la plus grande valeur pour le champ de précombinaison. Définissez un champ auquel appartient une valeur incrémentielle (par exemple updated_at).
- Type de compression : choisissez l'une des options de type de compression suivantes : Uncompressed, GZIP, LZO ou Snappy.
- Emplacement cible Amazon S3 : choisissez l'emplacement cible Amazon S3 en cliquant sur Parcourir S3.
- Options de mise à jour du catalogue de données : choisissez parmi les options suivantes :
  - Do not update the Data Catalog (Ne pas mettre à jour le catalogue de données) : (valeur par défaut) choisissez cette option si vous ne souhaitez pas que la tâche mette à jour le catalogue de données, même si le schéma change ou si de nouvelles partitions sont ajoutées.
  - Créer une table dans le catalogue de données et lors des exécutions suivantes, mettre à jour le schéma et ajouter de nouvelles partitions : si vous choisissez cette option, la tâche crée la table

dans le catalogue de données lors de sa première exécution. Lors des exécutions de tâches ultérieures, la tâche met à jour la table du catalogue de données si le schéma change ou si de nouvelles partitions sont ajoutées.

Vous devez également sélectionner une base de données dans le catalogue de données et entrer un nom de table.

- Create a table in the Data Catalog and on subsequent runs, keep existing schema and add new partitions (Créer une table dans le catalogue de données et lors des exécutions suivantes, conserver le schéma existant et ajouter de nouvelles partitions) : si vous choisissez cette option, la tâche crée la table dans le catalogue de données lors de sa première exécution. Lors des exécutions de tâches ultérieures, la tâche met à jour la table du catalogue de données uniquement pour ajouter de nouvelles partitions.

  Vous devez également sélectionner une base de données dans le catalogue de données et entrer un nom de table.

- Partition keys (Clé de partition) : choisissez les colonnes à utiliser comme clés de partitionnement dans la sortie. Pour ajouter d'autres clés de partition, choisissez Add a partition key (Ajouter une clé de partition).
- Options supplémentaires : saisissez une paire clé-valeur selon vos besoins.

Génération de code via AWS Glue Studio

Lorsque la tâche est enregistrée, les paramètres de tâche suivants sont ajoutés à la tâche si une source ou une cible Hudi est détectée :

- `--datalake-formats` : une liste distincte des formats de lacs de données détectés dans la tâche visuelle (soit directement en choisissant un « Format », soit indirectement en sélectionnant une table de catalogue soutenue par un lac de données).
- `--conf` : généré en fonction de la valeur des `--datalake-formats`. Par exemple, si la valeur des `--datalake-formats` est « hudi », AWS Glue génère une valeur de `spark.serializer=org.apache.spark.serializer.KryoSerializer –conf spark.sql.hive.convertMetastoreParquet=false` pour ce paramètre.

Remplacement des bibliothèques fournies par AWS Glue

Pour utiliser une version de Hudi non prise en charge par AWS Glue, vous pouvez spécifier vos propres fichiers JAR de bibliothèque Hudi. Pour utiliser votre propre fichier JAR :

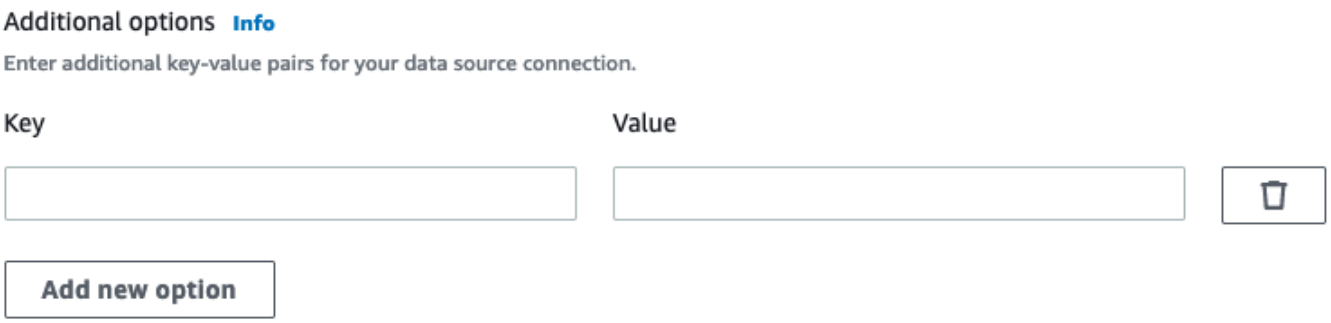
- utilisez le paramètre de tâche `--extra-jars`. Par exemple, `'--extra-jars': 's3pathtojarfile.jar'`. Pour plus d'informations, veuillez consulter la rubrique Paramètres des tâches AWS Glue.
- N'incluez pas `hudi` comme valeur du paramètre de tâche `--datalake-formats`. La saisie d'une chaîne vide en tant que valeur garantit qu'aucune bibliothèque de lacs de données ne vous est automatiquement fournie par AWS Glue. Pour plus d'informations, veuillez consulter la rubrique Utilisation du cadre Hudi dans AWS Glue.

Utilisation du cadre Delta Lake dans AWS Glue Studio

Utilisation du cadre Delta Lake dans des sources de données

Utilisation du cadre Delta Lake dans des sources de données Amazon S3

1. Dans le menu Source, choisissez Amazon S3.
2. Si vous choisissez la table du catalogue de données comme type de source Amazon S3, choisissez une base de données et une table.
3. AWS Glue Studio affiche le format comme étant Delta Lake et l'URL Amazon S3.
4. Choisissez Options supplémentaires pour saisir une paire clé-valeur. Par exemple, une paire clé-valeur peut être : clé : timestampAsOf et valeur : 24/02/2023 14:16:18.



5. Si vous choisissez l'emplacement Amazon S3 comme Type de source Amazon S3, choisissez l'URL Amazon S3 en cliquant sur Parcourir Amazon S3.
6. Dans Format de données, choisissez Delta Lake.

> **Note**
>
> Si AWS Glue Studio n'est pas en mesure de déduire le schéma à partir du dossier ou du fichier Amazon S3 que vous avez sélectionné, choisissez Options supplémentaires pour sélectionner un nouveau dossier ou fichier.
> Dans Options supplémentaires, choisissez l'une des options suivantes sous Inférence de schéma :
>
> - Laisser AWS Glue Studio choisir automatiquement un exemple de fichier : AWS Glue Studio choisit un exemple de fichier dans l'emplacement Amazon S3 afin que le schéma puisse être déduit. Dans le champ Fichier auto-échantillonné, vous pouvez afficher le fichier sélectionné automatiquement.
> - Choisir un exemple de fichier d'Amazon S3 : choisissez le fichier Amazon S3 à utiliser en cliquant sur Parcourir Amazon S3.

7. Cliquez sur Déduire un schéma. Vous pouvez ensuite consulter le schéma de sortie en cliquant sur l'onglet Schéma de sortie.

Utilisation du cadre Delta Lake dans des sources de données du catalogue de données

1. Dans le menu Source, choisissez Catalogue de données AWS Glue Studio.
2. Dans l'onglet Propriétés de source de données, choisissez une base de données et une table.
3. AWS Glue Studio affiche le type de format comme étant Delta Lake et l'URL Amazon S3.

> **Note**
>
> Si votre source Delta Lake n'est pas encore enregistrée comme table du catalogue de données AWS Glue, deux options s'offrent à vous :
>
> 1. Créer un crawler AWS Glue pour le magasin de données Delta Lake. Pour plus d'informations, veuillez consulter la rubrique Comment préciser les options de configuration pour un magasin de données Delta Lake.
> 2. Utilisez une source de données Amazon S3 pour sélectionner votre source de données Delta Lake. Consultez Utilisation du cadre Delta Lake dans des sources de données Amazon S3.

Utilisation des formats Delta Lake dans des cibles de données

Utilisation des formats Delta Lake dans des cibles de données du catalogue de données

1. Dans le menu Cible, choisissez Catalogue de données AWS Glue Studio.
2. Dans l'onglet Propriétés de source de données, choisissez une base de données et une table.
3. AWS Glue Studio affiche le type de format comme étant Delta Lake et l'URL Amazon S3.

Utilisation des formats Delta Lake dans des sources de données Amazon S3

Saisissez des valeurs ou sélectionnez-les parmi les options disponibles pour configurer le format Delta Lake.

- Type de compression : choisissez l'une des options de type de compression : Uncompressed ou Snappy.
- Emplacement cible Amazon S3 : choisissez l'emplacement cible Amazon S3 en cliquant sur Parcourir S3.
- Options de mise à jour du catalogue de données : la mise à jour du catalogue de données n'est pas prise en charge pour ce format dans l'éditeur visuel Glue Studio.
  - Do not update the Data Catalog (Ne pas mettre à jour le catalogue de données) : (valeur par défaut) choisissez cette option si vous ne souhaitez pas que la tâche mette à jour le catalogue de données, même si le schéma change ou si de nouvelles partitions sont ajoutées.
  - Pour mettre à jour le catalogue de données après l'exécution de la tâche AWS Glue, exécutez ou planifiez un crawler AWS Glue. Pour plus d'informations, veuillez consulter la rubrique Comment préciser les options de configuration pour un magasin de données Delta Lake.
- Clés de partition : choisissez les colonnes à utiliser comme clés de partitionnement dans la sortie. Pour ajouter d'autres clés de partition, choisissez Add a partition key (Ajouter une clé de partition).
- Choisissez éventuellement Options supplémentaires pour saisir une paire clé-valeur. Par exemple, une paire clé-valeur peut être : clé : timestampAsOf et valeur : 24/02/2023 14:16:18.

Utilisation du cadre Apache Iceberg dans AWS Glue Studio

Utilisation du cadre Apache Iceberg dans les cibles de données

Utilisation du cadre Apache Iceberg dans les cibles de données du catalogue de données

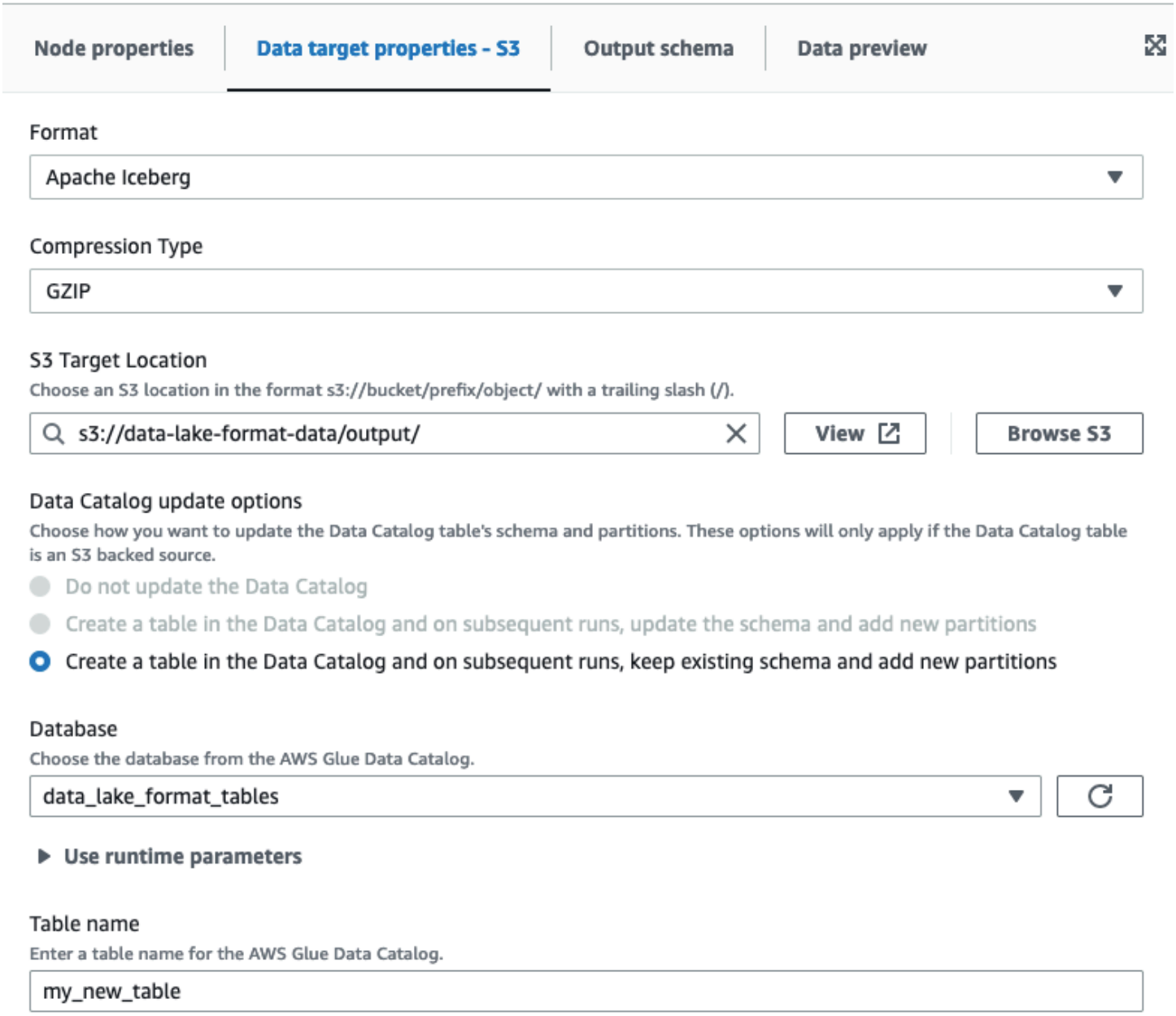
1. Dans le menu Cible, choisissez Catalogue de données AWS Glue Studio.

2. Dans l'onglet Propriétés de source de données, choisissez une base de données et une table.
3. AWS Glue Studio affiche le type de format comme étant Apache Iceberg et l'URL Amazon S3.

Utilisation du cadre Apache Iceberg dans les cibles de données Amazon S3

Saisissez des valeurs ou sélectionnez-les parmi les options disponibles pour configurer le format Apache Iceberg.

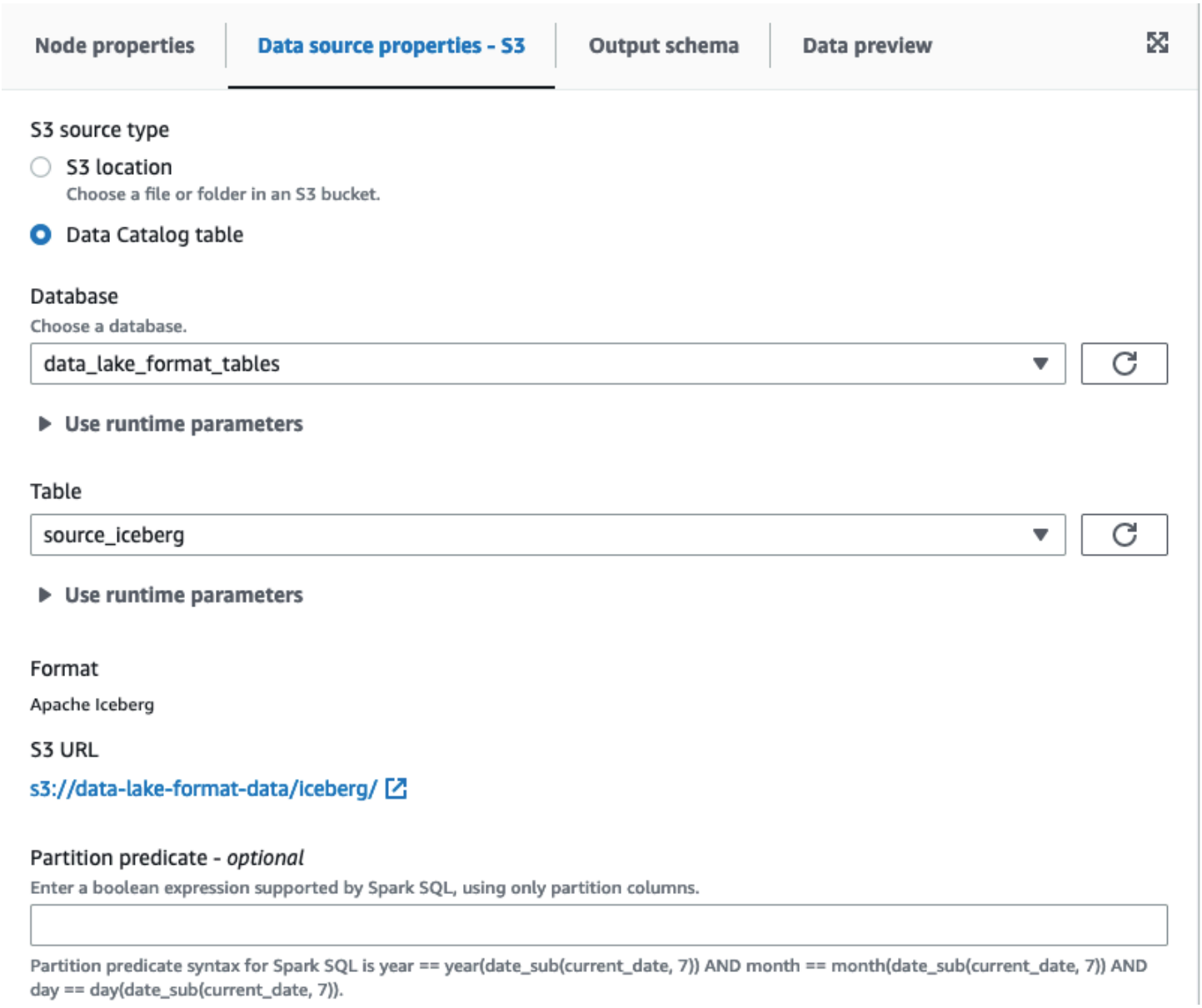
- Format : choisissez Apache Iceberg dans le menu déroulant.
- Emplacement cible Amazon S3 : choisissez l'emplacement cible Amazon S3 en cliquant sur Parcourir S3.
- Options de mise à jour du catalogue de données : créer une table dans le catalogue de données et lors des exécutions suivantes, conserver le schéma existant et ajouter de nouvelles partitions doivent être sélectionnés pour continuer. L'écriture d'une nouvelle table Iceberg à l'aide de AWS Glue nécessite que le Data Catalog soit configuré comme catalogue pour la table Iceberg. Pour mettre à jour une table Iceberg existante qui a été enregistrée dans le Data Catalog, choisissez Data Catalog comme cible.
  - Base de données : choisissez la base de données à partir du Data Catalog.
  - Nom de table : saisissez la valeur pour votre nom de table. Les noms des tables Apache Iceberg doivent être entièrement en minuscules. Utilisez des traits de soulignement si nécessaire, car les espaces ne sont pas autorisés. Par exemple « data_lake_format_tables ».



Utilisation du cadre Apache Iceberg dans les sources de données Amazon S3

Utilisation du cadre Apache Iceberg dans les sources de données du catalogue de données

1. Dans le menu Source, choisissez Catalogue de données AWS Glue Studio.
2. Dans l'onglet Propriétés de source de données, choisissez une base de données et une table.
3. AWS Glue Studio affiche le type de format comme étant Apache Iceberg et l'URL Amazon S3.



Utilisation du cadre Apache Iceberg dans les sources de données Amazon S3

Apache Iceberg n'est pas disponible en tant qu'option de données pour les nœuds source Amazon S3 dans AWS Glue Studio.

Se connecter aux sources de données à l'aide de tâches ETL visuelles

Lors de la création d'une nouvelle tâche, vous pouvez utiliser des connexions pour vous connecter aux données lors de la modification de tâches ETL visuelles dans AWS Glue. Pour ce faire, vous pouvez ajouter des nœuds source qui utilisent des connecteurs pour lire les données, et des nœuds cibles qui spécifient l'emplacement où les données sont écrites.

Rubriques

- Modifier les propriétés d'un nœud de source de données
- Utilisation des tables du catalogue de données pour la source de données
- Utilisation d'un connecteur pour la source de données
- Utilisation de fichiers dans Amazon S3 pour la source de données
- Utilisation d'une source de données en streaming
- Références

Modifier les propriétés d'un nœud de source de données

Pour spécifier les propriétés de la source de données, vous devez d'abord choisir un nœud de source de données dans le diagramme de tâche. Ensuite, sur le côté droit du volet de détails du nœud, vous configurez les propriétés du nœud.

Pour modifier les propriétés d'un nœud de source de données

1. Accédez à l'éditeur visuel pour une tâche nouvelle ou sauvegardée.
2. Choisissez un nœud de source de données dans le diagramme de tâche.
3. Choisissez l'onglet Node properties (Propriétés de nœud) dans le volet de détails du nœud, puis entrez les informations suivantes :

   - Name (Nom) : (facultatif) saisissez un nom à associer au nœud dans le diagramme des tâches. Ce nom doit être unique parmi tous les nœuds de cette tâche.
   - Node type (Type de nœud) : le type de nœud détermine l'action effectuée par le nœud. Dans la liste des options pour Node type (Type de nœud), choisissez l'une des valeurs répertoriées sous l'en-tête Data source (Source de données).
4. Configurer les informations de Data source properties (Propriétés de source de données). Pour plus d'informations, consultez les sections suivantes :

   - Utilisation des tables du catalogue de données pour la source de données
   - Utilisation d'un connecteur pour la source de données
   - Utilisation de fichiers dans Amazon S3 pour la source de données
   - Utilisation d'une source de données en streaming
5. (Facultatif) Après avoir configuré les propriétés du nœud et les propriétés de la source de données, vous pouvez afficher le schéma de votre source de données en sélectionnant l'onglet

Output Schema (Schéma de sortie) dans le volet de détails du nœud. La première fois que vous choisissez cet onglet pour un nœud de votre tâche, vous êtes invité à fournir un rôle IAM pour accéder aux données. Si vous n'avez pas spécifié de rôle IAM dans le Job détails (Détails de la tâche), vous y êtes invité à ce stade.

6. (Facultatif) Après avoir configuré les propriétés du nœud et les propriétés de la source de données, vous pouvez prévisualiser le jeu de données à partir de votre source de données en sélectionnant l'onglet Data preview (Prévisualisation des données) dans le volet de détails du nœud. La première fois que vous choisissez cet onglet pour un nœud de votre tâche, vous êtes invité à fournir un rôle IAM pour accéder aux données. Il y a un coût associé à l'utilisation de cette fonction, et la facturation commence dès que vous fournissez le rôle IAM.

Utilisation des tables du catalogue de données pour la source de données

Pour toutes les sources de données, à l'exception d'Amazon S3 et des connecteurs, une table doit exister dans AWS Glue Data Catalog pour le type de source que vous choisissez. AWS Glue ne crée pas la table de catalogue de données.

Pour configurer un nœud de source de données basé sur une table de catalogue de données

1. Accédez à l'éditeur visuel pour une tâche nouvelle ou sauvegardée.
2. Choisissez un nœud de source de données dans le diagramme de tâche.
3. Choisissez l'onglet Data source properties (Propriétés de source de données), puis saisissez les informations suivantes :

   - S3 source type (Type de source S3) : (pour les sources de données Amazon S3 uniquement) choisissez l'option Select a Catalog table (Sélectionner une table de catalogue) pour utiliser une table AWS Glue Data Catalog existante.
   - Database (Base de données) : choisissez la base de données dans le catalogue de données qui contient la table source que vous souhaitez utiliser pour cette tâche. Vous pouvez utiliser le champ de recherche pour rechercher une base de données par son nom.
   - Table : choisissez la table associée aux données source depuis la liste. Cette table doit déjà exister dans AWS Glue Data Catalog. Vous pouvez utiliser le champ de recherche pour rechercher une table par son nom.
   - Partition predicate (Prédicat de partition) : (pour les sources de données Amazon S3 uniquement) saisissez une expression booléenne basée sur Spark SQL qui inclut uniquement les colonnes de partitionnement. Par exemple : `"(year=='2020' and month=='04')"`

- Temporary directory (Répertoire temporaire) : (pour les sources de données Amazon Redshift uniquement) saisissez un chemin d'accès pour l'emplacement d'un répertoire de travail dans Amazon S3 où votre tâche ETL peut écrire des résultats intermédiaires temporaires.
- Role associated with the cluster (Rôle associé au cluster) : (pour les sources de données Amazon Redshift uniquement) saisissez un rôle que votre tâche ETL doit utiliser qui contient des autorisations pour les clusters Amazon Redshift . Pour de plus amples informations, veuillez consulter the section called “Autorisations de source et de cible de données”.

Utilisation d'un connecteur pour la source de données

Si vous sélectionnez un connecteur pour Node type (Type de nœud), suivez les instructions figurant dans Création de tâches avec des connecteurs personnalisés pour terminer la configuration des propriétés de la source de données.

Utilisation de fichiers dans Amazon S3 pour la source de données

Si vous choisissez Amazon S3 comme source de données, vous pouvez choisir parmis :

- Une base de données et une table de catalogue de données.
- Un compartiment, un dossier ou un fichier dans Amazon S3.

Si vous utilisez un compartiment Amazon S3 comme source de données, AWS Glue détecte le schéma des données à l'emplacement spécifié à partir de l'un des fichiers, ou en utilisant le fichier que vous spécifiez comme exemple de fichier. La détection de schéma se produit lorsque vous utilisez le bouton Infer schema (Déduire le schéma). Si vous modifiez l'emplacement Amazon S3 ou l'exemple de fichier, vous devez choisir Infer schema (Déduire le schéma) à nouveau pour effectuer la détection du schéma à l'aide des nouvelles informations.

Pour configurer un nœud de source de données qui lit directement à partir de fichiers dans Amazon S3

1. Accédez à l'éditeur visuel pour une tâche nouvelle ou sauvegardée.
2. Choisissez un nœud de source de données dans le diagramme de tâches pour une source Amazon S3.
3. Choisissez l'onglet Data source properties (Propriétés de source de données), puis saisissez les informations suivantes :

- S3 source type (Type de source S3) : (pour les sources de données Amazon S3 uniquement) choisissez l'option S3 location (Emplacement S3).
- S3 URL (URL S3) : saisissez le chemin d'accès au compartiment, dossier ou fichier Amazon S3 contenant les données de votre tâche. Vous pouvez choisir Browse S3 (Parcourir S3) pour sélectionner le chemin d'accès à partir des emplacements disponibles pour votre compte.
- Recursive (Récursif) : choisissez cette option si vous voulez que AWS Glue lise les données à partir de fichiers dans des dossiers enfants de l'emplacement S3.

  Si les dossiers enfants contiennent des données partitionnées, AWS Glue n'ajoute aucune information de partition spécifiée dans les noms de dossiers au catalogue de données. Prenons l'exemple des dossiers suivants dans Amazon S3 :

  ```
  S3://sales/year=2019/month=Jan/day=1
  S3://sales/year=2019/month=Jan/day=2
  ```

  Si vous choisissez Recursive (Récursif) et sélectionnez `sales` comme emplacement S3, AWS Glue lit les données dans tous les dossiers enfants, mais ne crée pas de partitions pour l'année, le mois ou le jour.
- Data format (Format de données) : choisissez le format dans lequel les données sont stockées. Vous pouvez choisir entre JSON, CSV ou Parquet. La valeur que vous sélectionnez indique à la tâche AWS Glue comment lire les données du fichier source.

  > **Note**
  >
  > Si vous ne sélectionnez pas le format correct pour vos données, AWS Glue pourrait déduire correctement le schéma, mais la tâche ne sera pas en mesure d'analyser correctement les données du fichier source.

  Vous pouvez saisir des options de configuration supplémentaires, selon le format que vous choisissez.

  - JSON (notation JavaScript d'objet)
    - JsonPath: entrez un chemin JSON qui pointe vers un objet utilisé pour définir un schéma de table. Les expressions de chemin JSON font toujours référence à une structure JSON de la même manière que les XPath expressions sont utilisées en combinaison avec un document XML. Le « root member object (objet membre racine) » dans le chemin JSON

est toujours appelé $, qu'il s'agisse d'un objet ou d'un tableau. Le chemin JSON peut être écrit en notation point ou en notation crochet.

Pour plus d'informations sur le chemin JSON, consultez JsonPathle GitHub site Web.

- Records in source files can span multiple lines (Les enregistrements dans les fichiers source peuvent s'étendre sur plusieurs lignes) : choisissez cette option si un même enregistrement peut couvrir plusieurs lignes dans le fichier CSV.

- CSV (valeurs séparées par des virgules)
  - Delimiter (Délimiteur) : saisissez un caractère pour indiquer ce qui sépare chaque entrée de colonne dans la ligne, par exemple, `;` ou `,`.
  - Escape character (Caractère d'échappement) : saisissez un caractère utilisé comme caractère d'échappement. Ce caractère indique que le caractère qui suit immédiatement le caractère d'échappement doit être pris littéralement et ne doit pas être interprété comme un délimiteur.
  - Quote character (Caractère de citation, ou guillemet) : saisissez le caractère utilisé pour regrouper des chaînes distinctes en une seule valeur. Par exemple, choisissez Double quote (") Guillemet double (") si vous avez des valeurs telles que `"This is a single value"` dans votre fichier CSV.
  - Records in source files can span multiple lines (Les enregistrements dans les fichiers source peuvent s'étendre sur plusieurs lignes) : choisissez cette option si un même enregistrement peut couvrir plusieurs lignes dans le fichier CSV.
  - First line of source file contains column headers (La première ligne du fichier source contient les en-têtes de colonne) : choisissez cette option si la première ligne du fichier CSV contient des en-têtes de colonne au lieu de données.
- Parquet (stockage en colonnes Apache Parquet)

  Il n'y a pas de paramètres supplémentaires à configurer pour les données stockées au format Parquet.
- Apache Hudi

  Il n'y a pas de paramètres supplémentaires à configurer pour les données stockées au format Apache Hudi.
- Lac Delta

  Il n'y a pas de paramètres supplémentaires à configurer pour les données stockées au format Delta Lake.

- Excel

  Il n'y a pas de paramètres supplémentaires à configurer pour les données stockées au format Excel.

- Partition predicate (Prédicat de partition) : pour partitionner les données lues à partir de la source de données, saisissez une expression booléenne basée sur Spark SQL qui inclut uniquement les colonnes de partitionnement. Par exemple : `"(year=='2020' and month=='04')"`
- Advanced options (Options avancées) : développez cette section si vous voulez que AWS Glue détecte le schéma de vos données en fonction d'un fichier spécifique.
  - Schema inference (Déduction de schéma) : choisissez l'option Choose a sample file from S3 (Choisir un exemple de fichier dans S3) si vous voulez utiliser un fichier spécifique au lieu de laisser AWS Glue choisir un fichier. L'inférence de schéma n'est pas disponible pour la source Excel.
  - Auto-sampled file (Fichier auto-échantillonné) : saisissez le chemin d'accès au fichier dans Amazon S3 à utiliser pour déduire le schéma.

  Si vous modifiez un nœud de source de données et modifiez l'exemple de fichier sélectionné, choisissez Reload schema (Recharger le schéma) pour détecter le schéma à l'aide du nouvel exemple de fichier.

4. Cliquez sur le bouton Infer schema (Déduire le schéma) pour détecter le schéma à partir des fichiers sources dans Amazon S3. Si vous modifiez l'emplacement Amazon S3 ou l'exemple de fichier, vous devez choisir Infer schema (Déduire le schéma) à nouveau pour déduire le schéma à l'aide des nouvelles informations.

Utilisation d'une source de données en streaming

Vous pouvez créer des tâches d'extraction, de transformation et de chargement en streaming (ETL) qui s'exécutent continu et consomment des données provenant de sources en streaming dans Amazon Kinesis Data Streams, Apache Kafka et Amazon Managed Streaming pour Apache Kafka (Amazon MSK).

Pour configurer les propriétés d'une source de données en streaming

1. Accédez à l'éditeur de graphes visuels pour une tâche nouvelle ou sauvegardée.
2. Choisissez un nœud de source de données dans le graphe pour Kafka ou Kinesis Data Streams.

3. Choisissez l'onglet Data source properties (Propriétés de source de données), puis saisissez les informations suivantes :

   Kinesis

   - Kinesis source type (Type de source Kinesis) : Choisissez l'option Stream details (Détails du flux) pour utiliser un accès direct à la source de streaming ou choisissez Data catalog table (Table du catalogue de données) pour utiliser les informations qui y sont stockées.

     Si vous choisissez Stream details (Détails du flux), spécifiez les informations supplémentaires suivantes.

     - Emplacement du flux de données : choisissez si le flux est associé à l'utilisateur actuel ou à un autre utilisateur.
     - Région : Choisissez l' Région AWS endroit où se trouve le flux. Ces informations sont utilisées pour construire l'ARN permettant d'accéder au flux de données.
     - Stream ARN : entrez l'Amazon Resource Name (ARN) pour le flux de données Kinesis. Si le flux est situé dans le compte courant, vous pouvez choisir le nom du flux dans la liste déroulante. Vous pouvez utiliser le champ de recherche pour rechercher un flux de données par son nom ou son ARN.
     - Data format (Format de données) : choisissez dans la liste le format utilisé par le flux de données.

       AWS Glue détecte automatiquement le schéma à partir des données en streaming.

     Si vous choisissez Data Catalog table (Table Catalogue de données), spécifiez les informations supplémentaires suivantes.

     - Database (Base de données) : (facultatif) dans le catalogue de données AWS Glue, choisissez la base de données qui contient la table associée à votre source de données en streaming. Vous pouvez utiliser le champ de recherche pour rechercher une base de données par son nom.
     - Table : (facultatif) choisissez la table associée aux données source depuis la liste. Cette table doit déjà exister dans le catalogue de données.AWS Glue. Vous pouvez utiliser le champ de recherche pour rechercher une table par son nom.
     - Detect schema (Détection de schéma) : choisissez cette option pour que AWS Glue détecte le schéma à partir des données en streaming, plutôt que d'utiliser les informations de schéma dans une table de catalogue de données. Cette option est activée automatiquement si vous choisissez l'option Stream details (Détails du flux).

- Starting position (Position de départ) : Par défaut, la tâche ETL utilise l'option Earliest, ce qui signifie qu'elle lit les données en commençant par le plus ancien enregistrement disponible dans le flux. Vous pouvez plutôt choisir Dernière, qui indique que la tâche ETL doit commencer à lire juste après l'enregistrement le plus récent du flux.
- Window size (Dimension de la fenêtre temporelle) : Par défaut, votre tâche ETL traite et écrit les données dans des fenêtres de 100 secondes. Cela permet de traiter les données de manière efficace et d'effectuer des agrégations sur les données qui arrivent plus tard que prévu. Vous pouvez modifier la taille de cette fenêtre temporelle pour augmenter la ponctualité ou la précision de l'agrégation.

  AWS Glue les tâches de streaming utilisent des points de contrôle plutôt que des signets de tâches pour suivre les données lues.
- Connection options (Options de connexion) : développez cette section pour ajouter des paires clé-valeur permettant de spécifier des options de connexion supplémentaires. Pour de plus amples informations sur les options que vous pouvez spécifier ici, veuillez consulter les rubriques « connectionType » : « kinesis » dans le Guide du développeur AWS Glue .

Kafka

- Apache Kafka source (Source Apache Kafka) : choisissez l'option Stream details (Détails du flux) pour utiliser l'accès direct à la source en streaming ou choisir Data Catalog table (Table du catalogue de données) pour utiliser les informations qui y sont stockées.

  Si vous choisissez Data Catalog table (Table Catalogue de données), spécifiez les informations supplémentaires suivantes.

  - Database (Base de données) : (facultatif) dans le catalogue de données AWS Glue, choisissez la base de données qui contient la table associée à votre source de données en streaming. Vous pouvez utiliser le champ de recherche pour rechercher une base de données par son nom.
  - Table : (facultatif) choisissez la table associée aux données source depuis la liste. Cette table doit déjà exister dans le catalogue de données.AWS Glue. Vous pouvez utiliser le champ de recherche pour rechercher une table par son nom.
  - Detect schema (Détection de schéma) : choisissez cette option pour que AWS Glue détecte le schéma à partir des données en streaming, plutôt que de stocker les informations de schéma dans une table de catalogue de données. Cette option est activée automatiquement si vous choisissez l'option Stream details (Détails du flux).

Si vous choisissez Stream details (Détails du flux), spécifiez les informations supplémentaires suivantes.

- Connection name (Nom de la connexion) : choisissez le AWS Glue qui contient les informations d'accès et d'authentification pour le flux de données Kafka. Vous devez utiliser une connexion avec les sources de données en streaming Kafka. Si aucune connexion n'existe, vous pouvez utiliser la console AWS Glue pour créer une connexion pour votre flux de données Kafka.
- Topic name (Nom de la rubrique) : entrez le nom de la rubrique à lire.
- Data format (Format de données) : choisissez le format à utiliser lors de la lecture de données à partir du flux d'événements Kafka.

- Starting position (Position de départ) : Par défaut, la tâche ETL utilise la commande Earliest (Au plus tôt), ce qui signifie qu'elle lit les données commençant par le plus ancien registre disponible dans le flux. Vous pouvez plutôt choisir Dernière, qui indique que la tâche ETL doit commencer à lire juste après l'enregistrement le plus récent du flux.
- Window size (Dimension de la fenêtre temporelle) : Par défaut, votre tâche ETL traite et écrit les données dans des fenêtres de 100 secondes. Cela permet de traiter les données de manière efficace et d'effectuer des agrégations sur les données qui arrivent plus tard que prévu. Vous pouvez modifier la taille de cette fenêtre temporelle pour augmenter la ponctualité ou la précision de l'agrégation.

  AWS Glue Les tâches en streaming utilisent des points de contrôle plutôt que des signets de tâche pour effectuer le suivit des données lues.
- Connection options (Options de connexion) : développez cette section pour ajouter des paires clé-valeur permettant de spécifier des options de connexion supplémentaires. Pour de plus amples informations sur les options que vous pouvez spécifier ici, veuillez consulter les rubriques « connectionType » : « kafka » dans le Guide du développeur AWS Glue .

> **Note**
>
> Les prévisualisations de données ne sont actuellement pas prises en charge pour les sources de données en streaming.

Références

Bonnes pratiques

- Créez un pipeline de services ETL pour charger les données de manière incrémentielle depuis Amazon S3 jusqu'à l'utilisation Amazon RedshiftAWS Glue

Programmation ETL

- Connection types and options for ETL in AWS Glue
- JDBC connectionType values
- Options avancées pour le transfert de données vers et depuis Amazon Redshift

Configuration des nœuds de données cible

La cible des données est l'endroit où la tâche écrit les données transformées.

Vue d'ensemble des options de cible de données

Votre cible de données (également appelée récepteur de données) peut avoir la valeur :

- S3 — La tâche écrit les données dans un fichier dans l'emplacement Amazon S3 que vous choisissez et dans le format que vous spécifiez.

  Si vous configurez des colonnes de partition pour la cible de données, la tâche écrit le jeu de données dans Amazon S3 dans des répertoires basés sur la clé de partition.
- AWS Glue Data Catalog — La tâche utilise les informations associées à la table dans le catalogue de données pour écrire les données en sortie dans un emplacement cible.

  Vous pouvez créer la table manuellement ou à l'aide de l'crawler. Vous pouvez également utiliser AWS CloudFormation pour créer des tables dans le catalogue de données.
- Un connecteur — Un connecteur est un morceau de code qui facilite la communication entre votre magasin de données et AWS Glue. La tâche utilise le connecteur et la connexion associée pour écrire les données en sortie vers un emplacement cible. Vous pouvez soit vous abonner à un connecteur proposé dans AWS Marketplace, soit créer votre propre connecteur personnalisé. Pour plus d'informations, veuillez consulter Ajouter des connecteurs àAWS Glue Studio

Vous pouvez choisir de mettre à jour le catalogue de données lorsque votre tâche écrit sur une cible de données Amazon S3. Au lieu de demander à un crawler de mettre à jour le catalogue de données lorsque le schéma ou les partitions changent, cette option facilite la mise à jour de vos tables. Cette option simplifie le processus de mise à disposition de vos données à des fins analytiques en ajoutant éventuellement de nouvelles tables au catalogue de données, en mettant à jour les partitions de table et en mettant à jour le schéma de vos tables directement à partir de la tâche.

Modification du nœud de données cible

La cible des données est l'endroit où la tâche écrit les données transformées.

Pour ajouter ou configurer un nœud de données cible dans votre diagramme de tâches

1. (Facultatif) Si vous devez ajouter un nœud cible, choisissez Target (Cible) dans la barre d'outils en haut de l'éditeur visuel, puis choisissez S3 ou Glue Data Catalog (Catalogue de données Glue).

   - Si vous choisissez S3 comme cible, la tâche écrit le jeu de données dans un ou plusieurs fichiers de l'emplacement Amazon S3 que vous spécifiez.
   - Si vous choisissez AWS Glue Data Catalog comme cible, la tâche écrit dans un emplacement décrit par la table sélectionnée dans le catalogue de données.
2. Choisir un nœud de source de données dans le diagramme de tâche. Lorsque vous choisissez un nœud, le volet de détails du nœud apparaît sur le côté droit de la page.
3. Choisissez l'onglet Node properties (Propriétés du nœud), puis saisissez les informations suivantes :

   - Name (Nom) : entrez un nom à associer au nœud dans le diagramme de tâche.
   - Node type (Type de nœud) : une valeur doit déjà être sélectionnée, mais vous pouvez la modifier si nécessaire.
   - Node parents (Parents de nœud) : le nœud parent est le nœud du diagramme de tâche qui fournit les données en sortie que vous souhaitez écrire à l'emplacement cible. Pour un diagramme de tâches prérempli, le nœud cible doit déjà avoir le nœud parent sélectionné. Si aucun nœud parent n'est affiché, choisissez un nœud parent dans la liste.

     Un nœud cible a un nœud parent unique.
4. Configurer les informations de Data target properties (Propriétés de cible de données). Pour plus d'informations, consultez les sections suivantes:

- Utilisation d'Amazon S3 pour la cible de données
- Utilisation des tables du catalogue de données pour la cible de données
- Utilisation d'un connecteur pour la cible de données

5. (Facultatif) Après avoir configuré les propriétés du nœud cible de données, vous pouvez afficher le schéma en sortie de vos données en sélectionnant l'onglet Output Schema (Schéma de sortie) dans le volet de détails du nœud. La première fois que vous choisissez cet onglet pour un nœud de votre tâche, vous êtes invité à fournir un rôle IAM pour accéder aux données. Si vous n'avez pas spécifié de rôle IAM dans le Job détails (Informations de la tâche), vous y êtes invité à ce stade.

Utilisation d'Amazon S3 pour la cible de données

Pour toutes les sources de données, à l'exception d'Amazon S3 et des connecteurs, une table doit exister dans AWS Glue Data Catalog pour le type de source que vous choisissez. AWS Glue Studio ne crée pas la table de catalogue de données.

Pour configurer un nœud cible de données qui écrit dans Amazon S3

1. Accédez à l'éditeur visuel pour une tâche nouvelle ou sauvegardée.
2. Choisissez un nœud de source de données dans le diagramme de tâche.
3. Choisissez l'onglet Data source properties (Propriétés de source de données), puis saisissez les informations suivantes :

   - Format : choisissez un format dans la liste. Les types de format disponibles pour les résultats de données sont les suivants :
     - JSON : notation JavaScript d'objets.
     - CSV : valeurs séparées par des virgules.
     - Avro : JSON binaire Apache Avro.
     - Parquet : un type de générateur Parquet personnalisé qui est optimisé pour `DynamicFrames` comme format de données. Au lieu de demander un schéma précalculé pour les données, il calcule et modifie le schéma de manière dynamique.
     - ORC : format Apache Optimized Row Columnar (ORC).
     - Apache Hudi : un cadre de stockage de lac de données open source qui simplifie le traitement incrémentiel des données et le développement de pipelines de données.

- Apache Iceberg : un format de table hautement performant qui fonctionne exactement comme une table SQL.
- Delta Lake : un cadre de stockage de lac de données open source qui vous permet d'effectuer des transactions ACID, de mettre à l'échelle la gestion des métadonnées et d'unifier le streaming et le traitement des données par lots.
- XML : Extensible Markup Language (XML).
- Tableau Hyper : technologie de moteur de données en mémoire de Tableau.

Pour en savoir plus sur ces options de format, veuillez consulter la rubrique Options de formatage pour les entrées et sorties ETL dans AWS Glue dans le Guide du développeur AWS Glue .

- Type de compression : vous pouvez choisir de compresser les données à l'aide des types de fichiers `CSV`, `JSON` ou `Parquet`. La valeur par défaut est sans compression, ou Aucun.

Type de fichier	Compressions
JSON/CSV/XML	GZIP, BROTLI BZIP2, DEFLATE, Snappy LZ4
Parquet	Snappy, LZO LZ4, BROTLI, GZIP
ORC	Snappy, ZLIB, non compressé, LZO
Avro	GZIP, BROTLI BZIP2, DEFLATE, Snappy LZ4
Delta Lake	GZIP, BROTLI BZIP2, DEFLATE, Snappy LZ4
Apache Hudi	GZIP, LZO, Snappy
Apache Iceberg	GZIP, LZO, Snappy
Tableau Hyper	Aucune

- S3 Target Location (Emplacement de cible S3) : compartiment Amazon S3 et emplacement pour la sortie de données. Vous pouvez choisir le bouton Browse S3 (Parcourir S3) pour

afficher les compartiments Amazon S3 auxquels vous avez accès et en choisir un comme cible.

- Options de mise à jour du catalogue de données
  - Do not update the Data Catalog (Ne pas mettre à jour le catalogue de données) : (valeur par défaut) choisissez cette option si vous ne souhaitez pas que la tâche mette à jour le catalogue de données, même si le schéma change ou si de nouvelles partitions sont ajoutées.
  - Create a table in the Data Catalog and on subsequent runs, update the schema and add new partitions (Créer une table dans le catalogue de données et lors des exécutions suivantes, mettre à jour le schéma et ajouter de nouvelles partitions) : si vous choisissez cette option, la tâche crée la table dans le catalogue de données lors de sa première exécution. Lors des exécutions de tâches ultérieures, la tâche met à jour la table du catalogue de données si le schéma change ou si de nouvelles partitions sont ajoutées.

    Vous devez également sélectionner une base de données dans le catalogue de données et entrer un nom de table.
  - Create a table in the Data Catalog and on subsequent runs, keep existing schema and add new partitions (Créer une table dans le catalogue de données et lors des exécutions suivantes, conserver le schéma existant et ajouter de nouvelles partitions) : si vous choisissez cette option, la tâche crée la table dans le catalogue de données lors de sa première exécution. Lors des exécutions de tâches ultérieures, la tâche met à jour la table du catalogue de données uniquement pour ajouter de nouvelles partitions.

    Vous devez également sélectionner une base de données dans le catalogue de données et entrer un nom de table.
- Partitionnement des fichiers : choisissez le type de partitionnement dans lequel vous souhaitez enregistrer la sortie.
  - Génération automatique de fichiers (recommandé) : il s'agit de la valeur par défaut du nombre de fichiers générés.
  - Sortie de fichiers multiples : spécifiez le nombre de sorties de fichiers que vous souhaitez. Pour des performances optimales, utilisez la valeur par défaut du nombre de fichiers générés automatiquement.
- Partition keys (Clé de partition) : choisissez les colonnes à utiliser comme clés de partitionnement dans la sortie. Pour ajouter d'autres clés de partition, choisissez Add a partition key (Ajouter une clé de partition).

Le partitionnement de fichiers n'est pas pris en charge pour Tableau Hyper en tant que format cible.

Utilisation des tables du catalogue de données pour la cible de données

Pour toutes les sources de données, à l'exception d'Amazon S3 et des connecteurs, une table doit exister dans AWS Glue Data Catalog pour le type de cible que vous choisissez. AWS Glue Studio ne crée pas la table de catalogue de données.

Pour configurer les propriétés de données d'une cible qui utilise une table du catalogue de données

1. Accédez à l'éditeur visuel pour une tâche nouvelle ou sauvegardée.
2. Choisir un nœud de source de données dans le diagramme de tâche.
3. choisissez l'onglet Data target properties (Propriétés de cible de données), puis saisissez les informations suivantes :

   - Database (Base de données) : depuis la liste, choisissez la base de données contenant la table que vous souhaitez utiliser comme cible. Cette base de données doit déjà exister dans le catalogue de données.
   - Table : choisissez la table qui définit le schéma de vos données en sortie dans la liste. Cette table doit déjà exister dans le catalogue de données.

     Une table dans le catalogue de données consiste en des noms de colonnes, des définitions de types de données, des informations de partition et d'autres métadonnées sur le jeu de données cible. Votre tâche écrit dans un emplacement décrit par cette table dans le catalogue de données.

     Pour plus d'informations sur la création de tables dans le catalogue de données, veuillez consulter la rubrique Définition de tables dans le catalogue de données dans le Guide du développeur AWS Glue .
   - Options de mise à jour du catalogue de données
     - Do not change table definition (Ne modifiez pas la définition de table) : (valeur par défaut) choisissez cette option si vous ne souhaitez pas que la tâche mette à jour le catalogue de données, même si le schéma change ou si de nouvelles partitions sont ajoutées.

- Update schema and add new partitions (Mettre à jour le schéma et ajouter de nouvelles partitions) : si vous choisissez cette option, la tâche met à jour la table Catalogue de données si le schéma change ou si de nouvelles partitions sont ajoutées.
- Keep existing schema and add new partitions (Conserver le schéma existant et ajouter de nouvelles partitions) : si vous choisissez cette option, la tâche met à jour la table Catalogue de données uniquement pour ajouter de nouvelles partitions.
- Partition keys (Clé de partition) : choisissez les colonnes à utiliser comme clés de partitionnement dans la sortie. Pour ajouter d'autres clés de partition, choisissez Add a partition key (Ajouter une clé de partition).

Utilisation d'un connecteur pour la cible de données

Si vous sélectionnez un connecteur comme Node type (Type de nœud), suivez les instructions figurant dans Création de tâches avec des connecteurs personnalisés pour terminer la configuration des propriétés de la cible de données.

Modification ou téléchargement d'un script de tâche

Utilisez l'éditeur visuel de AWS Glue Studio pour éditer le script de tâche ou charger votre propre script.

Vous ne pouvez utiliser l'éditeur visuel pour éditer des nœuds de tâche que si les tâches ont été créées avec AWS Glue Studio. Si la tâche a été créée à l'aide de la console AWS Glue, par des commandes d'API ou avec l'interface de ligne de commande (CLI), vous pouvez utiliser l'éditeur de script dans AWS Glue Studio pour modifier le script de tâche, les paramètres et la planification. Vous pouvez également modifier le script d'une tâche créée dans AWS Glue Studio en convertissant la tâche en mode script-only (script seul).

Pour modifier le script de tâche ou télécharger votre propre script

1. Si vous créez une tâche, sur la page Jobs (Tâches), choisissez l'option Spark script editor (Éditeur de script Spark) pour créer une tâche Spark ou l'option Python Shell script editor (Éditeur de script shell Python) pour créer une tâche shell Python. Vous pouvez soit écrire un nouveau script, soit en télécharger un existant. Si vous choisissez Spark script editor (Éditeur de script Spark), vous pouvez écrire ou télécharger un script Scala ou Python. Si vous choisissez Python Shell script editor (Éditeur de script shell Python), vous ne pouvez écrire ou télécharger qu'un script Python.

Après avoir choisi l'option de création d'une tâche, dans la section Options qui s'affiche, vous pouvez choisir soit de commencer avec un script de démarrage (Create a new script with boilerplate code (Créez un script avec le code standard)), soit de télécharger un fichier local à utiliser comme script de tâche.

Si vous avez choisi Spark script editor (Éditeur de script Spark), vous pouvez télécharger des fichiers de script Python ou Scala. Les scripts Scala doivent avoir l'extension de fichier `.scala`. Les scripts Python doivent être reconnus en tant que fichiers de type Python. Si vous choisissez Python Shell script editor (Éditeur de script shell Python), vous ne pouvez télécharger que des fichiers script Python.

Lorsque vous avez terminé vos choix, choisissez Create (Créer) pour créer la tâche et ouvrir l'éditeur visuel.

2. Accédez à l'éditeur de tâches visuel pour la nouvelle tâche ou la tâche enregistrée, puis choisissez l'onglet Script.
3. Si vous n'avez pas créé de tâche à l'aide de l'une des options d'éditeur de script et que vous n'avez jamais modifié le script d'une tâche existante, la propriété Script affiche l'en-têteScript (Locked — verrouillé). Cela signifie que l'éditeur de script est en mode lecture seule. Choisissez Edit Script (Modifier le script) pour déverrouiller le script à des fins d'édition.

   Pour rendre le script modifiable, AWS Glue Studio convertit le type de votre tâche de visuel à script-only. Si vous déverrouillez le script pour l'édition, vous ne pouvez plus utiliser l'éditeur visuel pour cette tâche après l'avoir sauvegardée.

   Dans la fenêtre de confirmation, choisissez Confirm (Confirmer) pour continuer ou Cancel (Annuler) pour garder la tâche disponible pour l'édition visuelle.

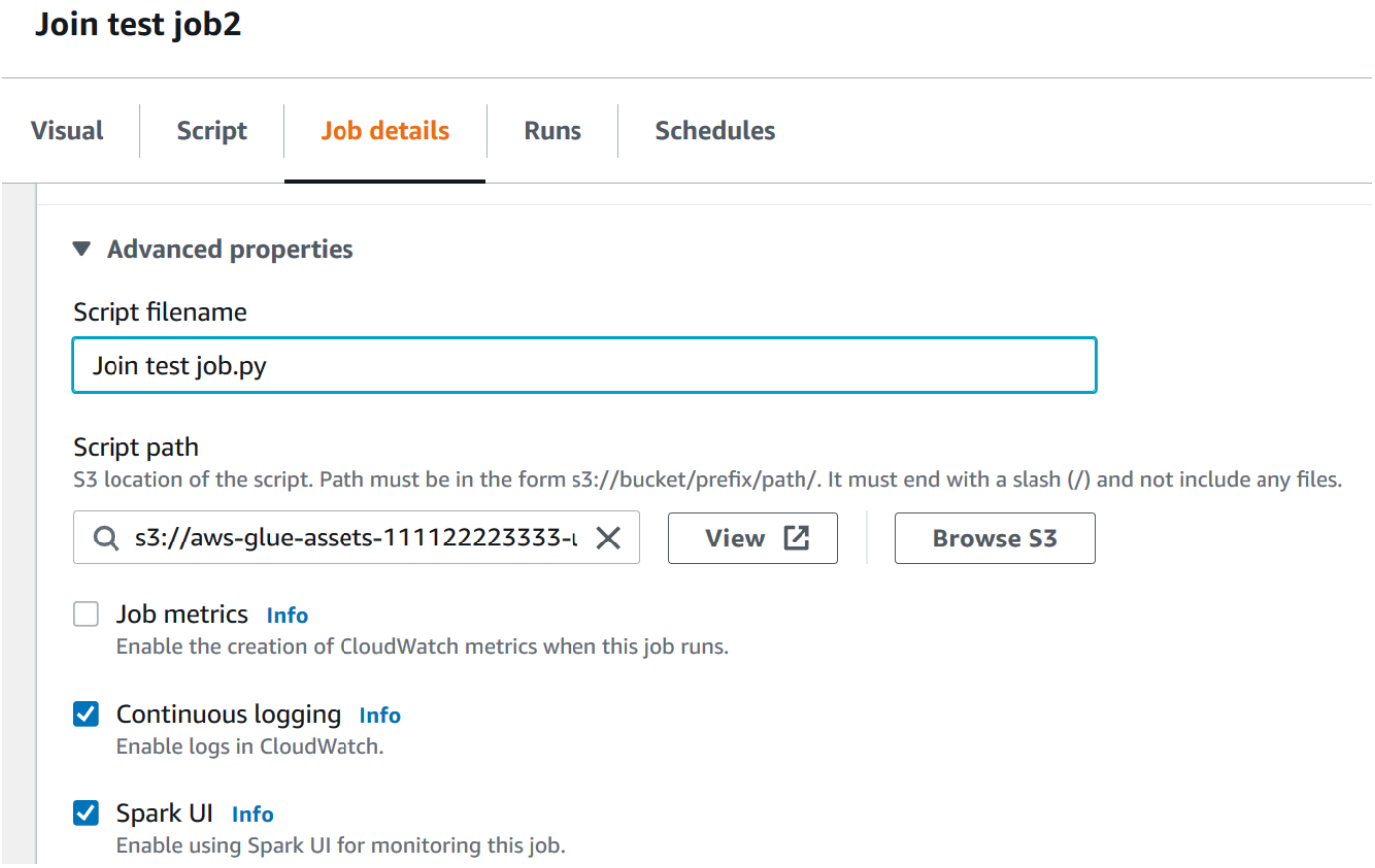
   Si vous choisissez Confirm (Confirmer), l'onglet Visual (Visuel) n'apparaît plus dans l'éditeur. Vous pouvez utiliser AWS Glue Studio pour modifier le script à l'aide de l'éditeur de script, modifier les détails de la tâche ou sa planification, ou afficher les exécutions de la tâche.

   > **Note**
   >
   > Tant que vous n'avez pas enregistré la tâche, la conversion en tâche script-only n'est pas permanente. Si vous actualisez la page Web de la console ou fermez la tâche avant de l'enregistrer et la rouvrez dans l'éditeur visuel, vous pourrez toujours modifier les nœuds individuels dans l'éditeur visuel.

4. Modifiez le script en fonction des besoins.

   Lorsque vous avez fini de modifier le script, choisissez Save (Enregistrer) pour sauvegarder la tâche et convertir définitivement la tâche de visuel à script-only.

5. (Facultatif) Vous pouvez télécharger le script à partir de la console AWS Glue Studio en choisissant le bouton Download (Télécharger) sur l'onglet Script. Lorsque vous cliquez sur ce bouton, une nouvelle fenêtre de navigateur s'ouvre et affiche le script à partir de son emplacement dans Amazon S3. Les paramètres Script filename (Nom du fichier de script) et Script path (Chemin du script) de la tâche dans l'onglet Job details (Détails de la tâche) déterminent le nom et l'emplacement du fichier de script dans Amazon S3.



   Lorsque vous enregistrez la tâche, AWS Glue enregistre le script de tâche à l'emplacement spécifié par ces champs. Si vous modifiez le fichier de script à cet emplacement dans Amazon S3, AWS Glue Studio chargera le script modifié la prochaine fois que vous éditez la tâche.

Création et modification de scripts Scala dans AWS Glue Studio

Lorsque vous choisissez l'éditeur de script pour créer une tâche, par défaut, le langage de programmation des tâches est défini sur `Python 3`. Si vous choisissez d'écrire un nouveau script au lieu de charger un script, AWS Glue Studio démarre un nouveau script avec du texte standard écrit en Python. Si vous voulez écrire un script Scala à la place, vous devez d'abord configurer l'éditeur de script pour utiliser Scala.

> **Note**
>
> Si vous choisissez Scala comme langage de programmation pour la tâche et que vous utilisez l'éditeur visuel pour concevoir votre tâche, le script de tâche généré est écrit en Scala, et aucune autre action n'est nécessaire.

Pour écrire un nouveau script Scala dans AWS Glue Studio

1. Créez une tâche en choisissant l'option Spark script editor (Éditeur de script Spark).
2. Sous Options, choisissez Create a new script with boilerplate code (Créez un script avec du code standard).
3. Choisissez l'onglet Job details (Détails de la tâche) et définissez Language (Langage) à `Scala` (au lieu de `Python 3`).

   > **Note**
   >
   > La propriété Type de la tâche est automatiquement définie à `Spark` lorsque vous choisissez l'option Spark script editor (Éditeur de script Spark) pour créer une tâche.

4. Choisissez l'onglet Script.
5. Supprimez le texte standard Python. Vous pouvez le remplacer par le texte standard Scala suivant.

```
import com.amazonaws.services.glue.{DynamicRecord, GlueContext}
import org.apache.spark.SparkContext
import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
```

```
object MyScript {
  def main(args: Array[String]): Unit = {
    val sc: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(sc)

    }
}
```

6. Écrivez votre script de tâche Scala dans l'éditeur. Ajoutez d'autres instructions `import` au besoin.

Création et modification de tâches shell Python dans AWS Glue Studio

Lorsque vous choisissez l'éditeur de script shell Python pour créer une tâche, vous pouvez télécharger un script Python existant ou en écrire un nouveau. Si vous choisissez d'écrire un nouveau script, le code standard est ajouté au nouveau script de tâche Python.

Pour créer une tâche shell Python

Reportez-vous aux instructions dans Démarrer une tâche dans AWS Glue Studio.

Les propriétés de tâche qui sont prises en charge pour les tâches shell Python ne sont pas les mêmes que celles prises en charge pour les tâches Spark. La liste suivante décrit les modifications apportées aux paramètres de tâche disponibles pour les tâches shell Python sur l'onglet Job details (Détails de la tâche).

- La propriété Type pour la tâche est automatiquement définie à `Python Shell` et ne peut pas être modifiée.
- Au lieu de Language (Langage), il y a une propriété Python version (Version de Python) pour la tâche. Actuellement, les tâches shell Python créées dans AWS Glue Studio utilisent Python 3.6.
- La propriété Glue version (Version de Glue) n'est pas disponible, car elle ne s'applique pas aux tâches shell Python.
- Au lieu de Worker type (Type d'employé) et Number of workers (Nombre d'employés), une propriété Data processing units (Unités de traitement des données) est affichée à la place. Cette propriété de tâche détermine le nombre d'unités de traitement de données (DPUs) consommées par le shell Python lors de l'exécution de la tâche.
- La propriété Job bookmark (Signet de tâche) n'est pas disponible, car elle n'est pas prise en charge pour les tâches shell Python.

- Sous Advanced properties (Propriétés avancées), les propriétés suivantes ne sont pas disponibles pour les tâches shell Python.
  - Métriques de tâche
  - Journalisation continue
  - Spark UI (Interface utilisateur Spark) et Spark UI logs path (Chemin d'accès à l'interface utilisateur Spark)
  - Dependent jars path (Chemin des fichiers .jar dépendants), sous l'en-tête Libraries (Bibliothèques)

Modification des noeuds parents d'un nœud dans le diagramme de tâche

Vous pouvez modifier les parents d'un nœud pour déplacer des nœuds dans le diagramme de tâche ou pour modifier une source de données pour un nœud.

Pour modifier le nœud parent

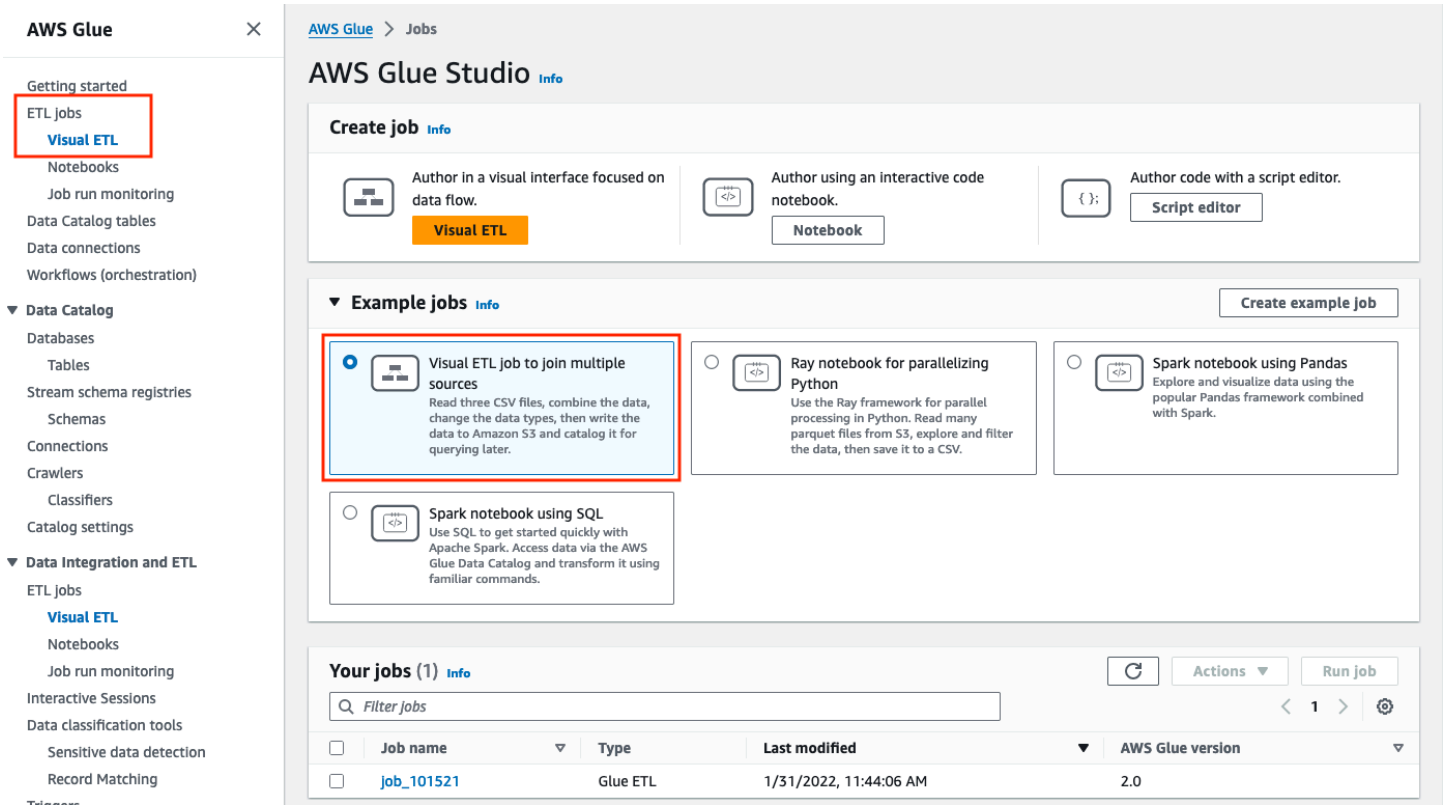
1. Choisissez le nœud dans le diagramme de la tâche que vous souhaitez modifier.
2. Dans le volet de détails du nœud, dans l'onglet Node properties (Propriétés de nœud), sous l'en-tête Node parents (Parents de nœud), supprimez le parent actuel du nœud.
3. Choisissez un nouveau nœud parent dans la liste.
4. Modifiez les autres propriétés du nœud si nécessaire pour correspondre au nœud parent nouvellement sélectionné.

Si vous avez modifié un nœud par erreur, vous pouvez utiliser Undo (Annuler) de la barre d'outils pour inverser l'action.

Suppression de noeuds du diagramme de tâches

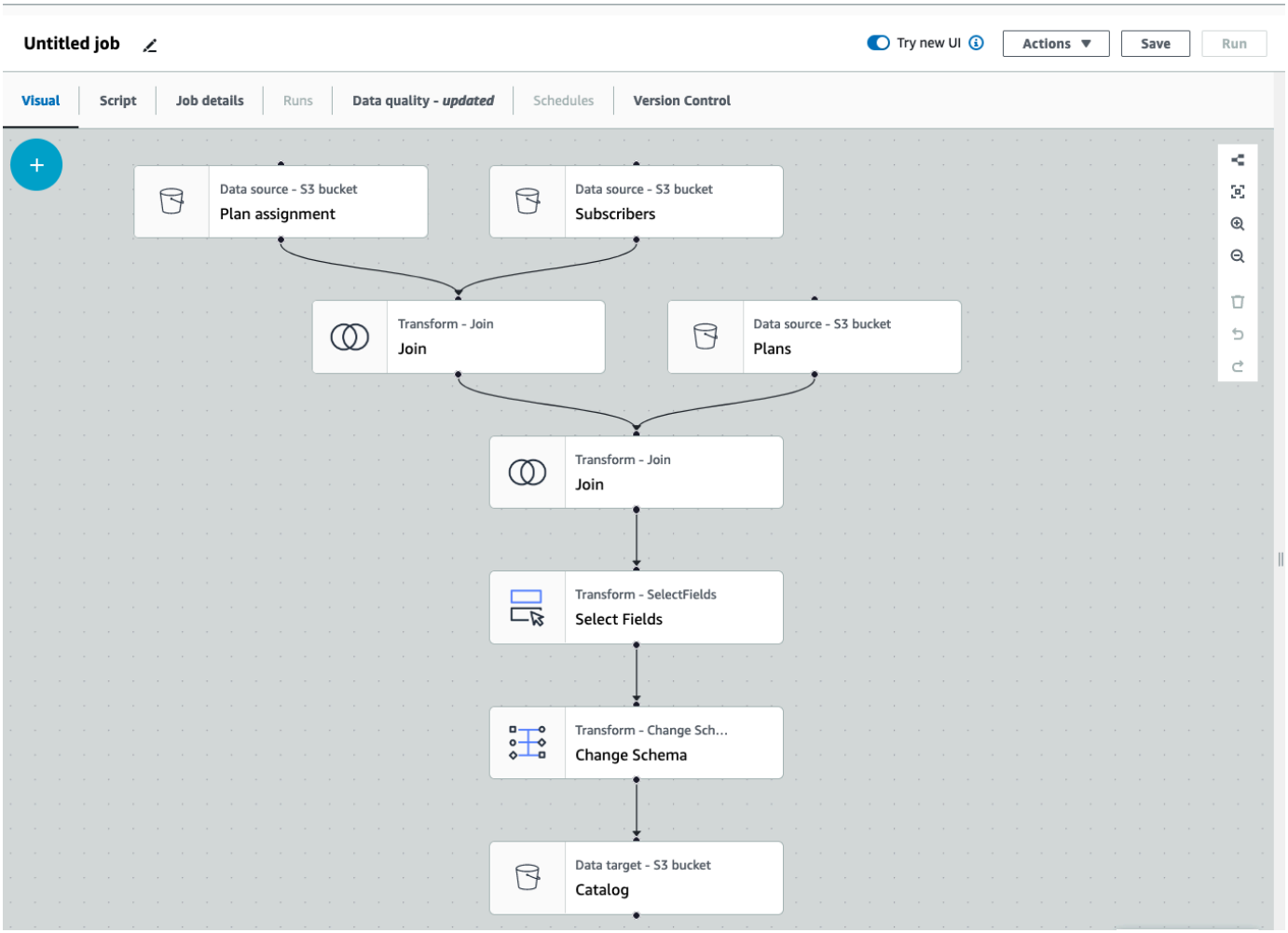
Lorsque vous travaillez avec des tâches ETL visuel, vous pouvez supprimer des nœuds du canevas sans avoir à ajouter à nouveau ou à restructurer les nœuds connectés au nœud supprimé.

Dans l'exemple ci-dessous, vous pouvez suivre en choisissant Tâches ETL > ETL visuel, puis dans Exemples de tâches, en choisissant Tâche ETL visuel pour joindre plusieurs sources. Choisissez Créer un exemple de tâche pour créer une tâche et suivez les étapes ci-dessous.



Pour supprimer un nœud du canevas

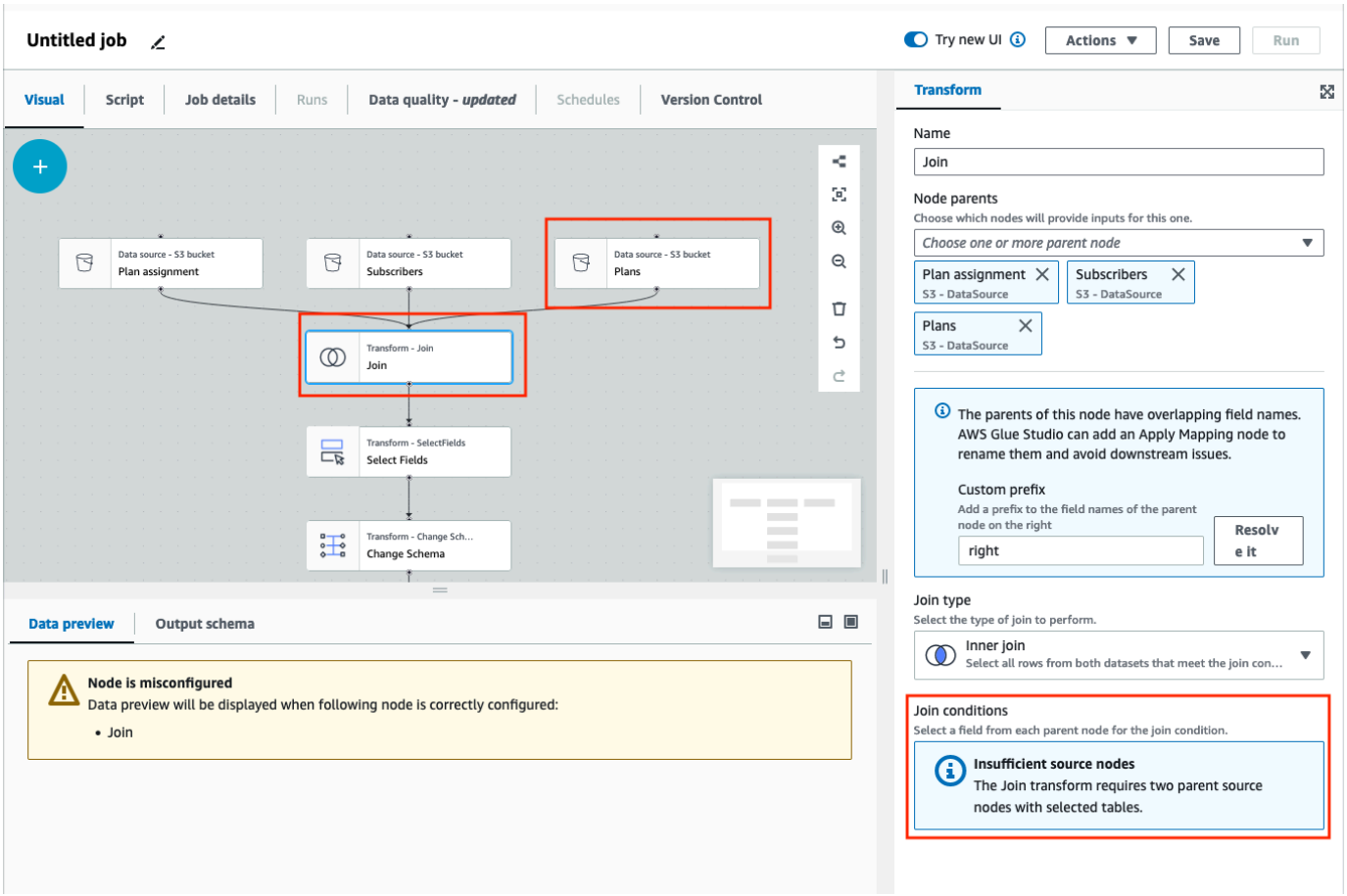
1. Dans la AWS Glue console, choisissez Visual ETL dans le menu de navigation et choisissez une tâche existante. Le canevas de la tâche affiche l'exemple de tâche, comme illustré ci-dessous.



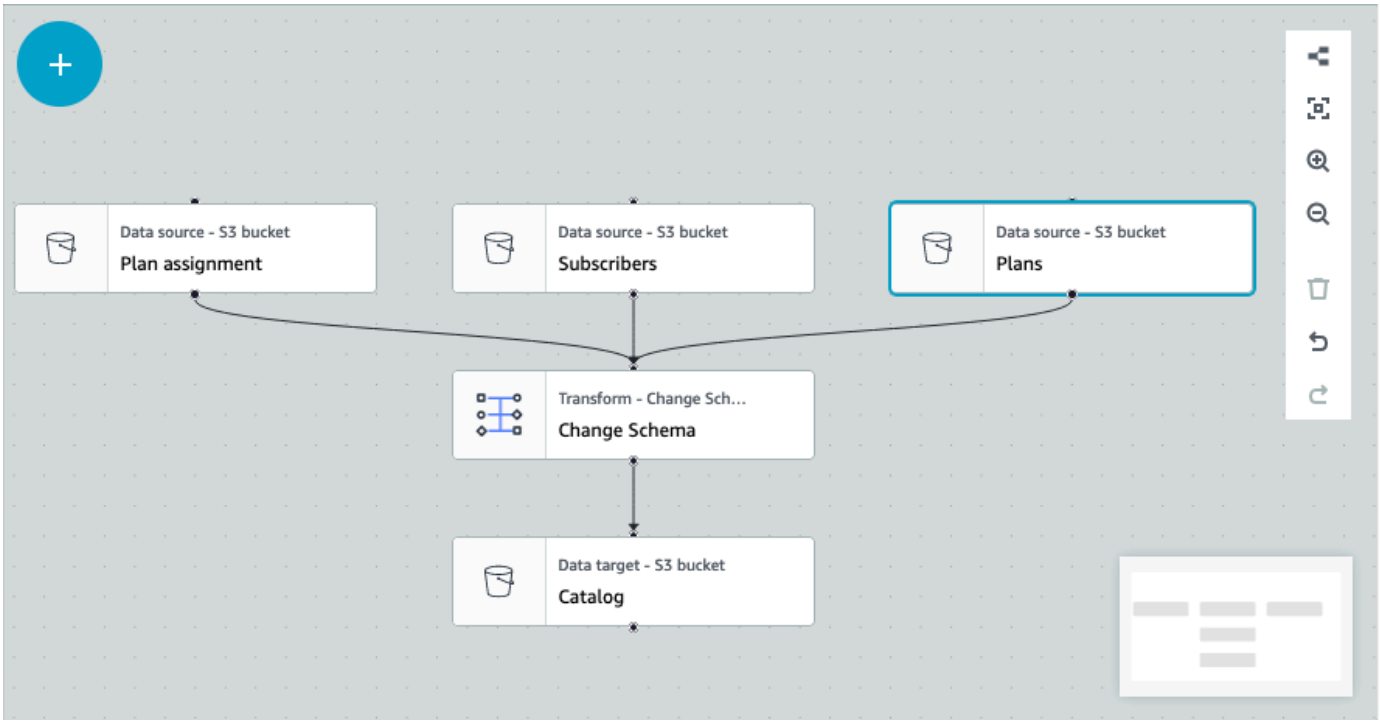
2. Choisissez le nœud que vous souhaitez supprimer. Le canevas zoomera sur le nœud. Dans la barre d'outils située à droite du canevas, cliquez sur l'icône de la corbeille. Cela supprimera le nœud et tout nœud connecté au nœud sera déplacé pour prendre sa place dans le flux de travail. Dans cet exemple, le premier nœud Join a été supprimé du canevas.

   Si vous supprimez un nœud dans le flux de travail, vous AWS Glue réorganiserez les nœuds de manière à ce qu'ils ne se traduisent pas par un flux de travail non valide. Il se peut que vous deviez toujours corriger la configuration d'un nœud.

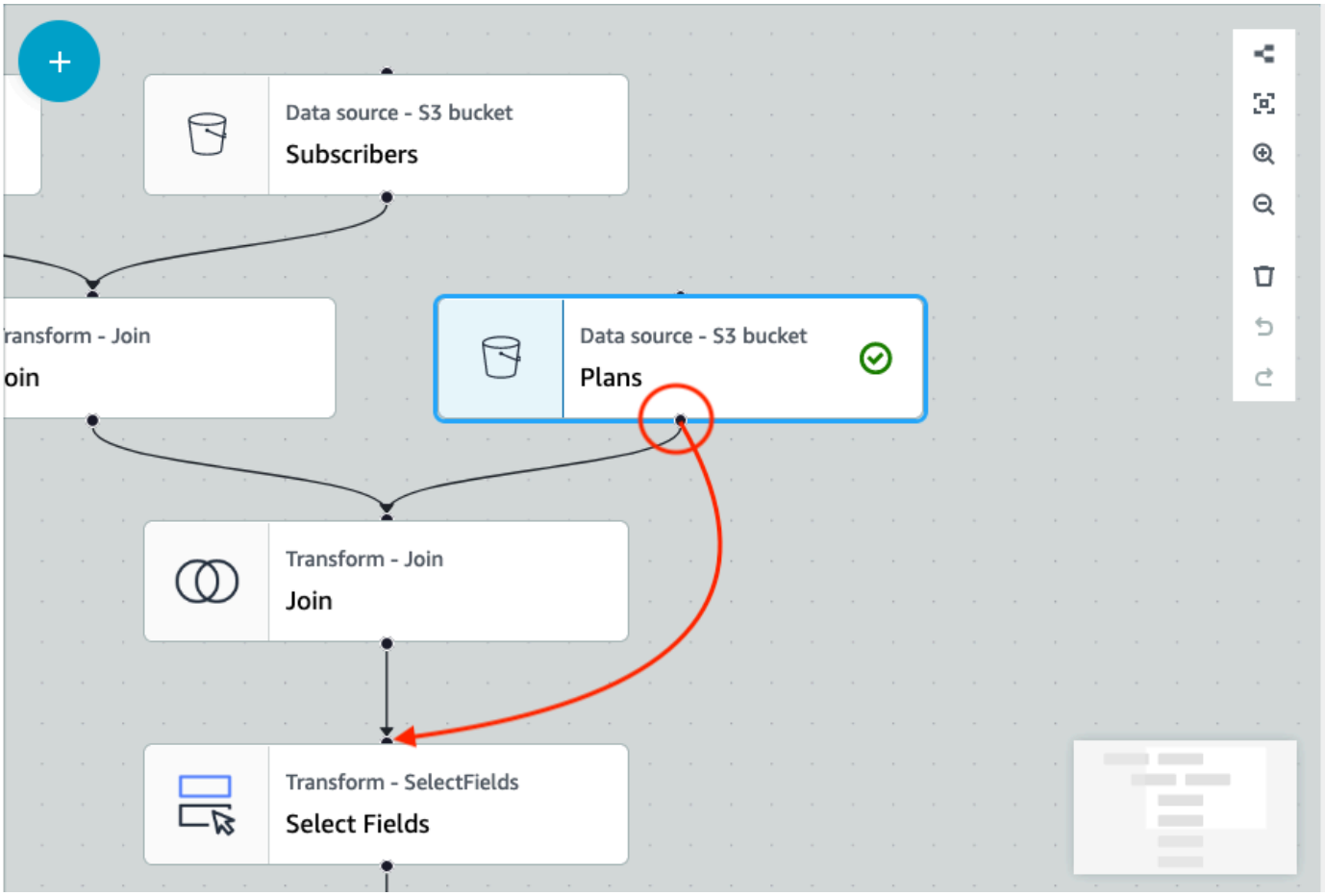
   Dans l'exemple, le nœud Join situé sous le nœud Subscribers a été supprimé. Par conséquent, le nœud source Plans a été déplacé au niveau supérieur et est toujours connecté au nœud Join enfant. Le nœud Join nécessite désormais une configuration supplémentaire, car Join nécessite deux nœuds source parent avec des tables sélectionnées. L'onglet Transformation situé à droite du canevas affiche l'exigence manquante sous Conditions de jointure.



3. Supprimez le deuxième nœud Join et le nœud Select Fields. Lorsque les nœuds ont été supprimés, le flux de travail ressemblera à l'exemple ci-dessous.



4. Pour modifier les connexions du nœud, cliquez sur la poignée du nœud et faites glisser la connexion vers un nouveau nœud. Cela vous permettra de supprimer des nœuds et de les réorganiser dans un flux logique. Dans l'exemple, une nouvelle connexion est établie en cliquant sur la poignée du nœud Plans et en faisant glisser la connexion vers le nœud Join, comme indiqué par la flèche rouge.



5. Si vous devez annuler une action, cliquez sur l'icône Annuler située juste en dessous de l'icône de la corbeille dans la barre d'outils située sur le côté droit du canevas.

Ajout de paramètres source et cible au AWS Glue nœud Data Catalog

AWS Glue Studio permet de paramétrer les tâches visuelles. Comme les noms des tables de catalogue peuvent être différents dans les environnements de production et de développement, vous pouvez définir et sélectionner des paramètres d'exécution pour les bases de données et les tables qui s'exécuteront lors de l'exécution de votre tâche.

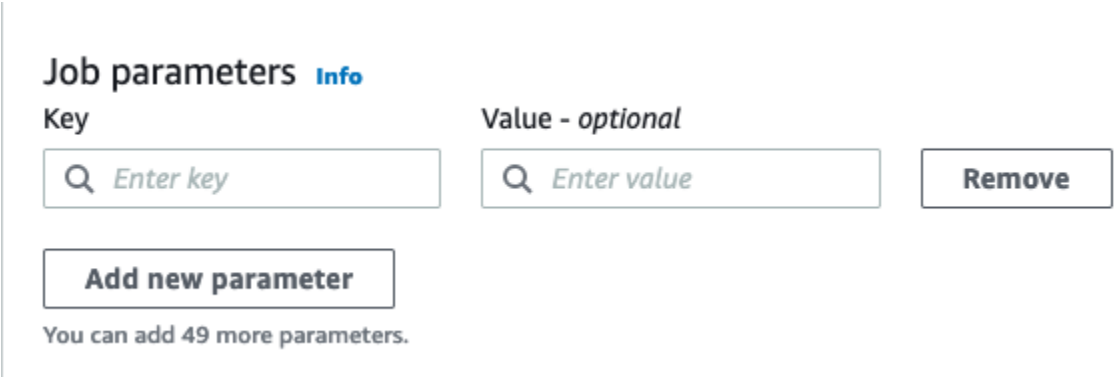
Le paramétrage de la tâche vous permet de configurer les sources et les cibles,et d'enregistrer ces paramètres dans la tâche lorsque vous utilisez le nœud Data Catalog AWS Glue. Lorsque vous spécifiez des sources et des cibles en tant que paramètres, vous activez la réutilisabilité des tâches, en particulier lorsque vous utilisez la même tâche dans plusieurs environnements. Cela s'avère utile lors de la promotion du code dans les environnements de déploiement, en économisant en temps et en énergie lors de la gestion de vos sources et cibles. Par ailleurs, les paramètres personnalisés que

vous définissez remplaceront tous les paramètres par défaut pour des exécutions spécifiques des tâches AWS Glue.

Ajouter des paramètres source et cible

Que vous utilisiez le nœud AWS Glue Data Catalog en tant que source ou cible, vous pouvez définir les paramètres d'exécution dans la section Advanced properties (Propriétés avancées) dans l'onglet Job details (Détails de la tâche).

1. Choisissez le nœud AWS Glue Data Catalog en tant que nœud source ou nœud cible.
2. Sélectionnez l'onglet Job details (Détails de la tâche).
3. Choisissez Advanced properties (Propriétés avancées).
4. Dans la section Job parameters (Paramètres de la tâche), saisissez une valeur clé. Par exemple, `--db.source` serait le paramètre d'une source de base de données. Vous pouvez saisir n'importe quel nom pour la clé, à condition que le nom de la clé soit suivi d'un « tiret ».

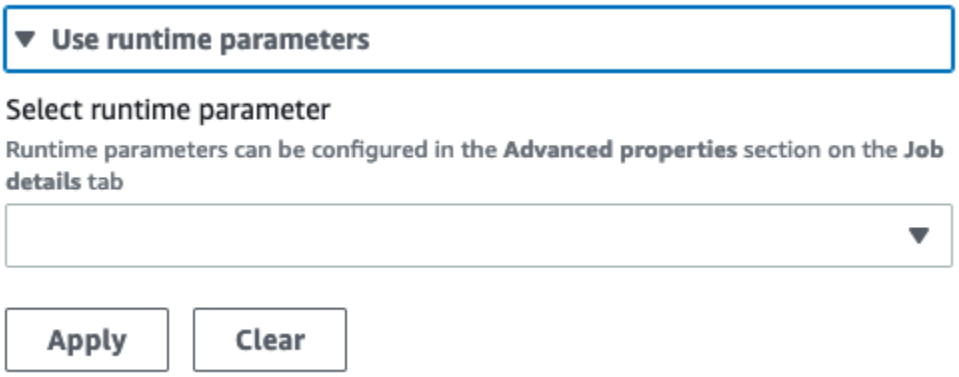


5. Saisissez la valeur. Par exemple, `databasename` serait la valeur d'une source de base de données paramétrée.
6. Choisissez Add new parameter (Ajouter de nouveaux paramètres) si vous souhaitez ajouter d'autres paramètres. Un maximum de 50 paramètres est autorisé. Une fois que la paire clé-valeur a été définie, vous pouvez utiliser le paramètre dans le nœud AWS Glue Data Catalog.

Sélectionner un paramètre d'exécution

> **Note**
>
> Le processus de sélection des paramètres d'exécution pour les bases de données et les tables est le même, que le nœud AWS Glue Data Catalog soit la source ou la cible.

1. Choisissez le nœud AWS Glue Data Catalog en tant que nœud source ou nœud cible.
2. Dans l'onglet Data source properties - Data Catalog (Propriétés de la source de données - Catalogue de données), sous Database (Base de données), choisissez Use runtime parameters (Utiliser les paramètres d'exécution).



3. Choisissez un paramètre dans le menu déroulant. Par exemple, lorsque vous sélectionnez un paramètre que vous avez défini pour une base de données source, la base de données est automatiquement renseignée dans le menu déroulant de la base de données lorsque vous choisissez Apply (Appliquer).
4. Dans la section Table, choisissez un paramètre que vous avez déjà défini comme table source. Lorsque vous choisissez Apply (Appliquer), la table est automatiquement renseignée en tant que table à utiliser.
5. Lorsque vous enregistrez et exécutez la tâche, AWS Glue Studio référencera les paramètres sélectionnés lors de l'exécution de la tâche.

Utiliser les systèmes de contrôle de version Git dans AWS Glue

> **Note**
>
> Les bloc-notes ne sont actuellement pas pris en charge pour le contrôle de version dans AWS Glue Studio. Cependant, le contrôle de version pour les scripts de tâche AWS Glue et les tâches ETL visuelles est pris en charge.

Si vous disposez de référentiels distants et que vous souhaitez gérer vos AWS Glue tâches à l'aide de ceux-ci, vous pouvez les utiliser AWS Glue Studio ou les synchroniser AWS CLI pour synchroniser les modifications apportées à vos référentiels et à vos tâches dans. AWS Glue Lorsque

vous synchronisez les changements de cette manière, vous transférez la tâche depuis AWS Glue Studio vers votre référentiel, ou l'extrayez depuis le référentiel vers AWS Glue Studio.

Avec l'intégration de Git dans AWS Glue Studio, vous pouvez :

- Intégrez les systèmes de contrôle de version Git AWS CodeCommit, tels que GitHub, GitLab, et Bitbucket
- modifier les tâches AWS Glue dans AWS Glue Studio si vous utilisez des tâches visuelles ou des tâches de script et que vous les synchronisez avec un référentiel ;
- paramétrer les sources et les cibles dans les tâches ;
- extraire des tâches d'un référentiel et lesmodifier dans AWS Glue Studio ;
- Testez des tâches en passant des succursales aux and/or succursales en utilisant des flux de travail multisuccursales dans AWS Glue Studio
- télécharger des fichiers à partir d'un référentiel et charger des tâches dans AWS Glue Studio pour la création de tâches entre comptes ;
- Utilisez l'outil d'automatisation de votre choix (par exemple, Jenkins AWS CodeDeploy, etc.)

Cette vidéo montre comment intégrer AWS Glue à Git et créer un pipeline de code continu et collaboratif.

Autorisations IAM

Assurez-vous que la tâche dispose de l'une des autorisations IAM suivantes. Pour plus d'informations sur la configuration des autorisations IAM, consultez Set up IAM permissions for AWS Glue Studio.

- `AWSGlueServiceRole`
- `AWSGlueConsoleFullAccess`

Au minimum, les actions suivantes sont nécessaires pour l'intégration de Git :

- `glue:UpdateJobFromSourceControl` : pour pouvoir mettre à jour AWS Glue avec une tâche présente dans un système de contrôle de version
- `glue:UpdateSourceControlFromJob` : pour pouvoir mettre à jour le système de contrôle de version avec une tâche stockée dans AWS Glue
- `s3:GetObject` : pour pouvoir récupérer le script de la tâche tout en le transférant vers le système de contrôle de version

- `s3:PutObject` : pour pouvoir mettre à jour le script lors de l'extraction d'une tâche depuis un système de contrôle de source

Conditions préalables

Pour transférer des tâches vers un référentiel de contrôle de source, vous aurez besoin :

- d'un référentiel qui a déjà été créé par votre administrateur ;
- d'une branche dans le référentiel ;
- d'un jeton d'accès personnel (pour Bitbucket, il s'agit du jeton d'accès au référentiel) ;
- du nom d'utilisateur du propriétaire du référentiel ;
- de définir des autorisations dans le référentiel pour autoriser AWS Glue Studio à lire et écrire dans le référentiel
  - GitLab— définit les portées des jetons sur api, read_repository et write_repository
  - Bitbucket : définissez les autorisations suivantes :
    - Abonnement à Workspace : read, write
    - Projets : write, admin read
    - Référentiels : read, write, admin, delete

> **Note**
>
> Lors de l'utilisation AWS CodeCommit, le jeton d'accès personnel et le propriétaire du référentiel ne sont pas nécessaires. Consultez Getting started with Git and AWS CodeCommit.

Utiliser des tâches de votre référentiel de contrôle de source dans AWS Glue Studio

Pour extraire une tâche de votre référentiel de contrôle de source qui ne se trouve pas dans AWS Glue Studio et pour l'utiliser dans AWS Glue Studio, les prérequis dépendront du type de tâche.

Pour les tâches visuelles :

- Vous avez besoin d'un dossier et d'un fichier JSON contenant la définition de la tâche correspondant au nom de la tâche.

Par exemple, consultez la définition de la tâche ci-dessous. La branche de votre référentiel doit contenir un chemin `my-visual-job/my-visual-job.json` où le dossier et le fichier JSON correspondent au nom de la tâche.

```
{
  "name" : "my-visual-job",
  "description" : "",
  "role" : "arn:aws:iam::aws_account_id:role/Rolename",
  "command" : {
    "name" : "glueetl",
    "scriptLocation" : "s3://foldername/scripts/my-visual-job.py",
    "pythonVersion" : "3"
  },
  "codeGenConfigurationNodes" : "{\"node-nodeID\":{\"S3CsvSource\":
{\"AdditionalOptions\":{\"EnableSamplePath\":false,\"SamplePath\":\"s3://notebook-
test-input/netflix_titles.csv\"},\"Escaper\":\"\",\"Exclusions\":[],\"Name\":\"Amazon
 S3\",\"OptimizePerformance\":false,\"OutputSchemas\":[{\"Columns\":[{\"Name\":
\"show_id\",\"Type\":\"string\"},{\"Name\":\"type\",\"Type\":\"string\"},{\"Name\":
\"title\",\"Type\":\"choice\"},{\"Name\":\"director\",\"Type\":\"string\"},{\"Name\":
\"cast\",\"Type\":\"string\"},{\"Name\":\"country\",\"Type\":\"string\"},{\"Name\":
\"date_added\",\"Type\":\"string\"},{\"Name\":\"release_year\",\"Type\":\"bigint\"},
{\"Name\":\"rating\",\"Type\":\"string\"},{\"Name\":\"duration\",\"Type\":\"string
\"},{\"Name\":\"listed_in\",\"Type\":\"string\"},{\"Name\":\"description\",\"Type
\":\"string\"}]}],\"Paths\":[\"s3://dalamgir-notebook-test-input/netflix_titles.csv
\"],\"QuoteChar\":\"quote\",\"Recurse\":true,\"Separator\":\"comma\",\"WithHeader
\":true}}}"
}
```

Pour les tâches de script :

- vous avez besoin d'un dossier, d'un fichier JSON contenant la définition de la tâche, et du script.
- le dossier et le fichier JSON doivent correspondre au nom de la tâche. Le nom du script doit correspondre à `scriptLocation` dans la définition de la tâche avec l'extension de fichier.

  Par exemple, dans la définition de la tâche ci-dessous, la branche de votre référentiel doit contenir un chemin `my-script-job/my-script-job.json` et `my-script-job/my-script-job.py`. Le nom du script doit correspondre au nom dans `scriptLocation` y compris l'extension du script.

```
{
  "name" : "my-script-job",
  "description" : "",
  "role" : "arn:aws:iam::aws_account_id:role/Rolename",
  "command" : {
    "name" : "glueetl",
    "scriptLocation" : "s3://foldername/scripts/my-script-job.py",
    "pythonVersion" : "3"
  }
}
```

Limitations

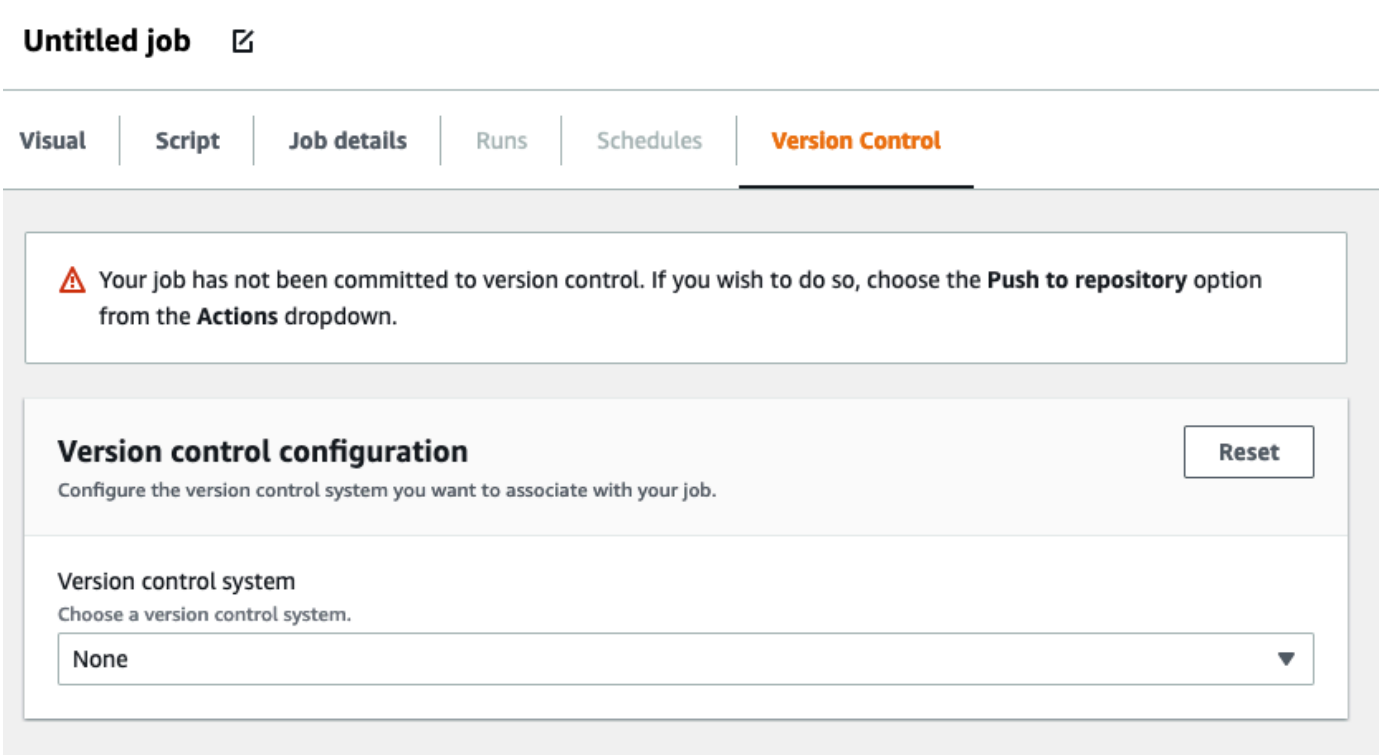
- AWS Gluen'est actuellement pas compatible pushing/pulling avec GitLab-Groups.

Connexion des référentiels de contrôle de version avec AWS Glue

Vous pouvez saisir les détails de votre référentiel de contrôle de version et les gérer dans l'onglet Version Control (Contrôle de version) dans l'éditeur de tâche AWS Glue Studio. Pour intégrer votre référentiel Git, vous devez vous connecter à votre référentiel Git chaque fois que vous vous connectez à AWS Glue Studio.

Pour connecter un système de contrôle de version Git :

1. Dans AWS Glue Studio, commencez une nouvelle tâche et choisissez l'onglet Version Control (Contrôle de version).



2. Dans Système de contrôle de version, choisissez Git Service parmi les options disponibles en cliquant sur le menu déroulant.

   - AWS CodeCommit
   - GitHub
   - GitLab
   - Bitbucket

3. Selon le système de contrôle de version Git que vous choisissez, vous aurez différents champs à remplir.

   Pour AWS CodeCommit :

   Terminez la configuration du référentiel en sélectionnant le référentiel et la branche correspondant à votre tâche :

   - Référentiel : si vous avez configuré des référentiels dans AWS CodeCommit, sélectionnez-les dans le menu déroulant. Vos référentiels apparaîtront automatiquement dans la liste
   - Branche : sélectionnez la branche dans le menu déroulant

- Dossier : facultatif – Saisissez le nom du dossier dans lequel vous souhaitez enregistrer votre tâche. S'il est laissé vide, un dossier est automatiquement créé. Le nom du dossier est par défaut le nom de la tâche.

Pour GitHub :

Complétez la GitHub configuration en remplissant les champs suivants :

- Jeton d'accès personnel : il s'agit du jeton fourni par le GitHub référentiel. Pour plus d'informations sur les jetons d'accès personnels, voir GitHub Docs
- Propriétaire du dépôt : il s'agit du propriétaire du GitHub dépôt.

Terminez la configuration du référentiel en sélectionnant le référentiel et la branche depuis GitHub.

- Référentiel : si vous avez configuré des référentiels dans GitHub, sélectionnez-les dans le menu déroulant. Vos référentiels apparaîtront automatiquement dans la liste
- Branche : sélectionnez la branche dans le menu déroulant
- Dossier : facultatif – Saisissez le nom du dossier dans lequel vous souhaitez enregistrer votre tâche. S'il est laissé vide, un dossier est automatiquement créé. Le nom du dossier est par défaut le nom de la tâche.

Pour GitLab :

> **Note**
>
> AWS Gluen'est actuellement pas compatible pushing/pulling avec GitLab-Groups.

- Jeton d'accès personnel : il s'agit du jeton fourni par le GitLab référentiel. Pour plus d'informations sur les jetons d'accès personnels, voir Jetons d'accès GitLab personnels
- Propriétaire du dépôt : il s'agit du propriétaire du GitLab dépôt.

Terminez la configuration du référentiel en sélectionnant le référentiel et la branche depuis GitLab.

- Référentiel : si vous avez configuré des référentiels dans GitLab, sélectionnez-les dans le menu déroulant. Vos référentiels apparaîtront automatiquement dans la liste
- Branche : sélectionnez la branche dans le menu déroulant
- Dossier : facultatif – Saisissez le nom du dossier dans lequel vous souhaitez enregistrer votre tâche. S'il est laissé vide, un dossier est automatiquement créé. Le nom du dossier est par défaut le nom de la tâche.

Pour Bitbucket :

- Mot de passe d'application : Bitbucket utilise des mots de passe d'application et non des jetons d'accès au référentiel. Pour plus d'informations sur les mots de passe des applications, consultez App passwords.
- Propriétaire du référentiel : il s'agit du propriétaire du référentiel Bitbucket. Dans Bitbucket, le propriétaire est le créateur du référentiel.

Terminez la configuration du référentiel en sélectionnant l'espace de travail, le référentiel, la branche et le dossier depuis Bitbucket.

- Espace de travail : si vous avez configuré des espaces de travail dans Bitbucket, sélectionnez l'espace de travail dans le menu déroulant. Vos espaces de travail sont automatiquement remplis.
- Référentiel : si vous avez configuré des référentiels dans Bitbucket, sélectionnez le référentiel dans le menu déroulant. Vos référentiels sont automatiquement remplis.
- Branche : sélectionnez la branche dans le menu déroulant. Vos branches sont automatiquement remplies.
- Dossier : facultatif – Saisissez le nom du dossier dans lequel vous souhaitez enregistrer votre tâche. Si elles sont laissées vides, un dossier est automatiquement créé avec le nom de la tâche.

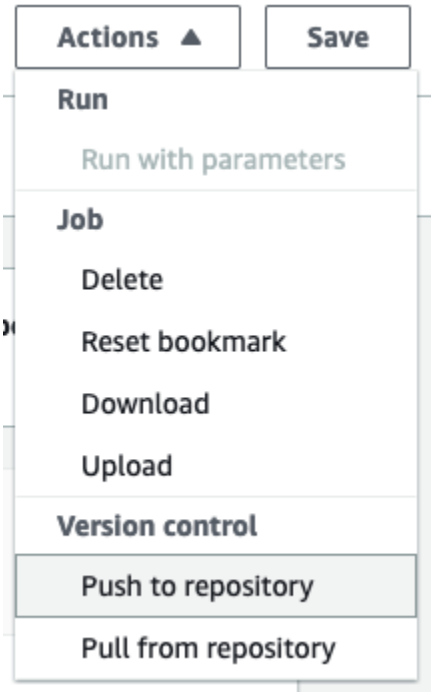
4. Choisissez Save (Enregistrer) en haut de la tâche AWS Glue Studio

Transfert de tâches AWS Glue vers le référentiel source

Une fois que vous avez saisi les détails de votre système de contrôle de version, vous pouvez modifier des tâches dans AWS Glue Studio et transférez des jobs vers votre référentiel source. Si vous n'êtes pas familier avec les concepts de Git tels que le transfert et l'extraction, consultez ce didacticiel sur Getting started with Git and AWS CodeCommit

Pour transférer votre tâche vers un référentiel, vous devez saisir les détails de votre système de contrôle de version et enregistrer votre tâche.

1. Dans la tâche AWS Glue Studio, choisissez Actions. Cette action ouvrira des options de menu supplémentaires.



2. Choisissez Push to repository.

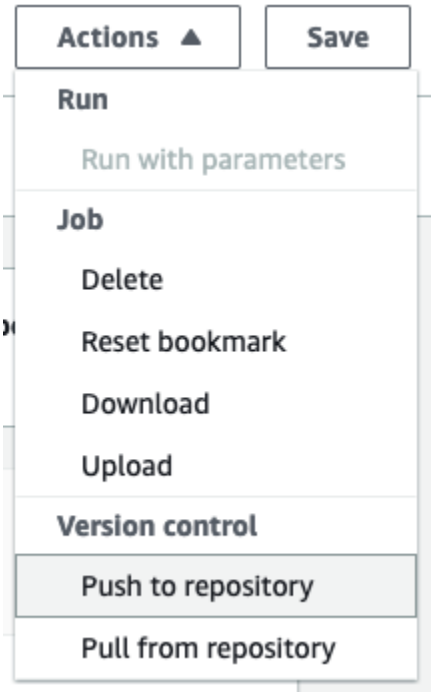
   Cette action permettra de sauvegarder la tâche. Lorsque vous transférez des tâches vers le référentiel, AWS Glue Studio transfert le dernier enregistré. Si la tâche dans le référentiel a été modifiée par vous ou par un autre utilisateur et n'est pas synchronisée avec la tâche dans AWS Glue Studio, la tâche dans le référentiel est remplacée par la tâche enregistrée dans AWS Glue Studio lorsque vous transférez la tâche depuis AWS Glue Studio.

3. Choisissez Confirm (Confirmer) pour terminer l'action. Cette action crée une validation dans le référentiel. Si vous l'utilisez AWS CodeCommit, un message de confirmation affichera un lien vers le dernier commit activé AWS CodeCommit.

Transfert de tâches AWS Glue depuis le référentiel source

Une fois que vous avez saisi les détails de votre référentiel Git dans l'onglet Version control (Contrôle de version), vous pouvez également extraire des tâches de votre référentiel et les modifier dans AWS Glue Studio.

1. Dans la tâche AWS Glue Studio, choisissez Actions. Cette action ouvrira des options de menu supplémentaires.



2. Choisissez Pull from repository (Extraire depuis le référentiel).
3. Choisissez Confirmer. Cette action récupère la dernière validation depuis le référentiel et met à jour votre tâche dans AWS Glue Studio.
4. Modifier votre tâche dans AWS Glue Studio. Si vous apportez des changements, vous pouvez synchroniser votre tâches avec votre référentiel en choisissant Push to repository (Transférer vers le référentiel) depuis le menu déroulant Actions.

Créer du code avec des blocs-notes AWS Glue Studio

Les ingénieurs de données peuvent créer des tâches AWS Glue plus rapidement et plus facilement qu'auparavant grâce à l'interface interactive de bloc-notes dans AWS Glue Studio ou aux séances interactives dans AWS Glue.

Limitations

- AWS Glue Studio les ordinateurs portables ne supportent pas Scala.

Rubriques

- Présentation de l'utilisation des blocs-notes
- Création d'une tâche ETL à l'aide de blocs-notes dans AWS Glue Studio
- Composants de l'éditeur de bloc-notes
- Enregistrement de votre bloc-notes et de votre script de tâche
- Gestion des séances de bloc-notes
- Utilisation d'Amazon Q Developer avec des blocs-notes AWS Glue Studio

Présentation de l'utilisation des blocs-notes

AWS Glue Studio vous autorise à créer des tâches de manière interactive dans une interface de bloc-notes basée sur les bloc-notes Jupyter. Grâce aux blocs-notes dans AWS Glue Studio, vous pouvez modifier des scripts de tâche et afficher le résultat sans avoir à exécuter une tâche complète. Vous pouvez également modifier le code d'intégration des données et afficher le résultat sans avoir à exécuter une tâche complète, et vous pouvez ajouter des Markdown et enregistrer des blocs-notes sous forme de fichiers .ipynb et de scripts de tâche. Vous pouvez démarrer un bloc-notes sans installation locale de logiciels ni gestion de serveurs. Lorsque vous êtes satisfait de votre code, AWS Glue Studio peut convertir votre bloc-notes en tâche Glue par simple clic sur un bouton.

Les avantages liés à l'utilisation des bloc-notes sont les suivants :

- Aucun cluster à allouer ou à gérer
- Aucun cluster inactif à payer
- Aucune configuration préalable n'est requise
- Aucune installation de bloc-notes Jupyter n'est requise
- runtime/platform Identique à AWS Glue ETL

Lorsque vous démarrez un bloc-notes via AWS Glue Studio, toutes les étapes de configuration sont effectuées pour vous afin que vous puissiez explorer vos données et commencer à développer votre script de tâche après quelques secondes seulement. AWS Glue Studio configure un bloc-

notes Jupyter avec le noyau pour Jupyter AWS Glue. Vous n'avez pas besoin de configurer VPCs de connexions réseau ou de terminaux de développement pour utiliser ce bloc-notes.

Pour créer des tâches à l'aide de l'interface du bloc-notes :

- configurez les autorisations IAM nécessaires.
- démarrez une séance de bloc-notes pour créer une tâche
- écrivez un code dans les cellules du bloc-notes
- exécutez et testez le code pour afficher le résultat
- sauvegardez la tâche

Une fois votre bloc-notes sauvegardé, il constitue une tâche AWS Glue complète. Vous pouvez gérer tous les aspects de la tâche, tels que la programmation de l'exécution des tâches, la définition des paramètres de tâche et l'affichage de l'historique des exécutions de tâches directement à côté de votre bloc-notes.

Création d'une tâche ETL à l'aide de blocs-notes dans AWS Glue Studio

Pour commencer à utiliser des blocs-notes dans la console AWS Glue Studio

1. Associez Gestion des identités et des accès AWS des politiques à l'AWS Glue Studioutilisateur et créez un rôle IAM pour votre tâche ETL et votre bloc-notes.
2. Configurez une sécurité IAM supplémentaire pour les blocs-notes, comme décrit dans Octroi d'autorisations pour le rôle IAM.
3. Ouvrez la console AWS Glue Studio à l'adresse https://console.aws.amazon.com/gluestudio/.

> **Note**
>
> Vérifiez que votre navigateur ne bloque pas les cookies tiers. Tout navigateur qui bloque les cookies tiers par défaut ou en tant que paramètre activé par l'utilisateur empêchera le lancement des bloc-notes. Pour en savoir plus sur la gestion des cookies, voir :

- Chrome
- Firefox
- Safari

4. Choisissez le lien Tâches dans le menu de navigation de gauche.
5. Choisissez Bloc-notes Jupyter, puis choisissez Créer pour démarrer une nouvelle séance de bloc-notes.
6. Dans la page Créer une tâche dans le bloc-notes Jupyter, indiquez le nom de la tâche, puis choisissez le rôle IAM à utiliser. Choisissez Créer une tâche.

   Après un court laps de temps, l'éditeur de bloc-notes apparaît.
7. Après avoir ajouté le code, vous devez exécuter la cellule pour démarrer une séance. Il existe plusieurs façons d'exécuter la cellule :

   - Appuyez sur le bouton Lecture.
   - Utilisez un raccourci clavier :
     - Sur macOS, Command (Commande) + Enter (Entrée) pour exécuter la cellule.
     - Sous Windows, Shift + Enter (Entrée) pour exécuter la cellule.

   Pour plus d'informations sur l'écriture de code à l'aide d'une interface de notebook Jupyter, voir la Documentation utilisateur de Jupyter Notebook.
8. Pour tester votre script, exécutez l'intégralité du script ou des cellules individuelles. Toute sortie de commande sera affichée dans la zone située sous la cellule.
9. Une fois que vous avez fini de développer votre bloc-notes, vous pouvez enregistrer la tâche, puis l'exécuter. Vous pouvez trouver ce script dans l'onglet Script. Toutes magics que vous avez ajoutées au bloc-notes seront supprimées et ne seront pas enregistrées comme faisant partie du script de la tâche générée AWS Glue. AWS Glue Studio ajoutera automatiquement un code `job.commit()` jusqu'à la fin de votre script généré à partir du contenu du bloc-notes.

   Pour de plus amples informations sur l'exécution d'une tâche, veuillez consulter Démarrer une exécution de tâche.

Composants de l'éditeur de bloc-notes

L'interface de l'éditeur de bloc-notes comporte les sections principales suivantes.

- Interface de bloc-notes (panneau principal) et barre d'outils
- Onglets d'édition de tâches

L'éditeur de bloc-notes

L'éditeur de bloc-notes AWS Glue Studio est basé sur l'application Jupyter Notebook. L'interface de bloc-notes AWS Glue Studio est similaire à celle fournie par Jupyter Notebooks, qui est décrite dans la section Interface utilisateur de bloc-notes. Le bloc-notes utilisé par les séances interactives est un bloc-notes Jupyter.

Bien que le bloc-notes AWS Glue Studio soit similaire à Jupyter Notebooks, il diffère sur quelques points essentiels :

- actuellement, le bloc-notes AWS Glue Studio ne peut pas installer d'extensions
- vous ne pouvez pas utiliser plusieurs onglets ; il existe une relation 1:1 entre une tâche et un bloc-notes
- le bloc-notes AWS Glue Studio ne possède pas le même menu de fichier supérieur que celui qui existe dans Jupyter Notebooks
- actuellement, le bloc-notes AWS Glue Studio fonctionne uniquement avec le noyau AWS Glue. Notez que vous ne pouvez pas mettre à jour le noyau par vous-même.

Onglets d'édition de tâches AWS Glue Studio

Les onglets que vous utilisez pour interagir avec la tâche ETL se trouvent en haut de la page du bloc-notes. Ils sont similaires aux onglets qui apparaissent dans l'éditeur visuel de tâches de AWS Glue Studio, et ils effectuent les mêmes actions.

- Bloc-notes – Utilisez cet onglet pour afficher le script de tâche à l'aide de l'interface bloc-notes.
- Détails de la tâche – configurez l'environnement et les propriétés des exécutions de travaux.
- Exécutions – Affichez des informations sur les exécutions précédentes de cette tâche.
- Planificateurs – Configurez un planificateurs d'exécution de votre tâche à des moments précis.

Enregistrement de votre bloc-notes et de votre script de tâche

Vous pouvez enregistrer votre bloc-notes et le script de tâche que vous créez à tout moment. Il suffit de choisir le bouton Enregistrer dans le coin supérieur droit, comme si vous utilisiez l'éditeur visuel ou de script.

Lorsque vous choisissez Save (Enregistrer), le fichier bloc-notes est enregistré dans les emplacements par défaut :

- Par défaut, le script de tâche est enregistré à l'emplacement Amazon S3 indiqué dans l'onglet Job Details (Détails de la tâche), sous Advanced properties (Propriétés avancées), dans la propriété Détails de la tâche Script path (Chemin du script). Les scripts de tâches sont enregistrés dans un sous-dossier nommé `Scripts`.
- Par défaut, le fichier du bloc-notes (`.ipynb`) est enregistré à l'emplacement Amazon S3 indiqué dans l'onglet Job Details (Détails de la tâche), sous Advanced properties (Propriétés avancées), dans les Détails de la tâche Script path (Chemin du script). Les fichiers de blocs-notes sont enregistrés dans un sous-dossier nommé `Notebooks`.

> **Note**
>
> Lorsque vous enregistrez la tâche, le script de tâche contient uniquement les cellules de code du bloc-notes. Les cellules Markdown et les magies ne sont pas incluses dans le script de tâche. Cependant, le fichier `.ipynb` contiendra tous les markdown et les magies.

Une fois la tâche enregistrée, vous pouvez ensuite l'exécuter à l'aide du script que vous avez créé dans le bloc-notes.

Gestion des séances de bloc-notes

Les blocs-notes dans AWS Glue Studio sont basés sur la fonction des séances interactives de AWS Glue. L'utilisation de séances interactives entraîne un coût. Pour gérer vos coûts, vous pouvez contrôler les séances créées pour votre compte et configurer les paramètres par défaut de toutes les séances.

Modifier le délai d'expiration par défaut pour toutes les séances de bloc-notes

Par défaut, le bloc-notes AWS Glue Studio provisionné expire au bout de 12 heures s'il a été lancé et si aucune cellule n'a été exécutée. Il n'y a aucun coût associé et le délai d'expiration n'est pas configurable.

Une fois que vous avez exécuté une cellule, une session interactive démarre. Cette session a un délai d'expiration par défaut de 48 heures. Ce délai d'expiration peut être configuré en entrant un champ magique `%idle_timeout` avant d'exécuter une cellule.

Pour modifier le délai d'expiration de séance par défaut des bloc-notes dans AWS Glue Studio

1. Dans le bloc-notes, entrez le champ magique `%idle_timeout` dans une cellule et spécifiez la valeur du délai d'attente en minutes.
2. Par exemple : `%idle_timeout 15` modifiera le délai d'expiration par défaut à 15 minutes. Si la séance n'est pas utilisée pendant 15 minutes, la séance est automatiquement arrêtée.

Installation de modules Python supplémentaires

Si vous souhaitez installer des modules supplémentaires à votre séance à l'aide de pip, vous pouvez le faire en utilisant `%additional_python_modules` pour les ajouter à votre séance :

```
%additional_python_modules awswrangler, s3://amzn-s3-demo-bucket/mymodule.whl
```

Tous les arguments de additional_python_modules sont transmis à `pip3 install -m <>`

Pour afficher la liste des modules Python disponibles, consultez Using Python libraries with AWS Glue.

Modification de la configuration AWS Glue

Vous pouvez utiliser magics pour contrôler les valeurs de configuration de la tâche AWS Glue. Si vous souhaitez modifier une valeur de configuration de tâche, vous devez utiliser magic appropriée dans le bloc-notes. Consultez Magics supported by AWS Glue interactive sessions for Jupyter.

> **Note**
>
> Le remplacement des propriétés d'une session en cours d'exécution n'est plus disponible. Pour modifier les configurations de la session, vous pouvez arrêter la session, définir les nouvelles configurations, puis démarrer une nouvelle session.

AWS Glue prend en charge différents types d'employés. Vous pouvez définir le type d'employé avec`%worker_type`. Par exemple : `%worker_type G.2X` . Les types de travailleurs disponibles incluent G.1X, G.2X, G.4X, G.8X, G.12X, G.16X, R.1X, R.2X, R.4X et R.8X. La valeur par défaut est G.1X.

Vous pouvez également spécifier le nombre d'employés avec `%number_of_workers`. Par exemple, pour spécifier 40 employés : `%number_of_workers 40`.

Pour de plus amples informations, veuillez consulter la rubrique Définir les propriétés d'une tâche.

Arrêt d'une séance de bloc-notes

Pour arrêter une séance de bloc-notes, utilisez la magie `%stop_session`.

Si vous quittez le bloc-notes dans la AWS console, vous recevrez un message d'avertissement dans lequel vous pourrez choisir d'arrêter la session.

Utilisation d'Amazon Q Developer avec des blocs-notes AWS Glue Studio

AWS Glue Studio vous autorise à créer des tâches de manière interactive dans une interface de bloc-notes basée sur les bloc-notes Jupyter. L'utilisation d'Amazon Q Developer améliore l'expérience de création dans les blocs-notes AWS Glue Studio.

L'extension Amazon Q Developer prend en charge l'écriture de code en générant des recommandations de code et en suggérant des améliorations liées aux problèmes de code. Amazon Q Developer prend en charge Python et Scala, les deux langages utilisés pour le codage de scripts ETL pour des tâches Spark dans des blocs-notes AWS Glue Studio.

Qu'est-ce qu'Amazon Q Developer ?

Amazon Q Developer est un service basé sur le machine learning qui permet d'améliorer la productivité des développeurs. Amazon Q Developer réalise cela en générant des recommandations de code basées sur les commentaires des développeurs en langage naturel et leur code dans l'IDE. Le service s'intègre à Amazon SageMaker AI Studio JupyterLab, aux instances de Amazon SageMaker AI notebook et à d'autres environnements de développement intégrés (IDEs).

Pour plus d'informations, consultez Using Amazon Q Developer with AWS Glue Studio.

États d'exécution des tâches AWS Glue dans la console

Vous pouvez afficher le statut d'une tâche Extract-transform-load (ETL) AWS Glue pendant son exécution ou après l'arrêt de celle-ci. Vous pouvez afficher l'état à l'aide de la console AWS Glue.

Accès au tableau de bord de surveillance de tâche

Vous accédez au tableau de bord de surveillance de tâche en choisissant le lien Surveillance de l'exécution des tâches dans le panneau de navigation AWS Glue, sous Tâches ETL.

Présentation du tableau de bord de surveillance de tâche

Le tableau de bord de surveillance de tâche fournit un récapitulatif global des exécutions de tâches, avec des totaux pour les tâches dont le statut est Running (En cours d'exécution), Canceled (Annulé), Success (Succès), ou Failed (Échec). Des vignettes supplémentaires fournissent le taux de réussite global de l'exécution des tâches, l'utilisation estimée du DPU pour les tâches, une ventilation des comptes de statut des tâches par type de tâche, type d'employé et par jour.

Les graphiques dans les vignettes sont interactifs. Vous pouvez choisir n'importe quel bloc d'un graphique pour exécuter un filtre qui affiche uniquement ces tâches dans le tableau Job runs (Exécutions de tâche) au bas de la page.

Vous pouvez modifier la plage de dates des informations affichées sur cette page à l'aide du sélecteur Date range (Plage de dates). Lorsque vous modifiez la plage de dates, les vignettes d'informations s'ajustent pour afficher les valeurs du nombre de jours spécifié avant la date actuelle. Vous pouvez également utiliser une plage de dates spécifique si vous sélectionnez Custom (Personnalisée) à partir du sélecteur de plage de dates.

Vue des exécutions de tâche

> **Note**
>
> L'historique d'exécution des tâches est accessible pendant 90 jours pour votre flux de travail et l'exécution des tâches.

La liste des ressources Job runs (Exécutions de tâche) affiche les tâches pour la plage de dates et les filtres spécifiés.

Vous pouvez filtrer les tâches en fonction de critères supplémentaires, tels que le statut, le type d'employé, le type de tâche et le nom de la tâche. Dans la zone de filtre située en haut du tableau, vous pouvez saisir le texte à utiliser comme filtre. Les résultats du tableau sont mis à jour avec des lignes qui contiennent du texte correspondant lorsque vous saisissez le texte.

Vous pouvez afficher un sous-ensemble des tâches en choisissant des éléments dans les graphiques du tableau de bord de surveillance de tâche. Par exemple, si vous choisissez le nombre de tâches en cours d'exécution dans la liste Job runs summary (Résumé des exécutions de tâches), la vignette Job runs (Exécutions de tâche) affiche uniquement les tâches dont le statut est actuellement `Running`. Si vous choisissez l'une des barres du diagramme à barres Worker type breakdown (Répartition par type d'employé), seules les exécutions de tâche dont le type et le statut d'employé correspondent sont affichées dans la liste Job runs (Exécutions de tâche).

La liste de ressources Job runs (Exécutions de tâche) affiche les détails des exécutions de tâche. Vous pouvez trier les lignes du tableau en choisissant un titre de colonne. Le tableau contient les informations suivantes :

Propriété	Description
Nom du job	Nom de la tâche .
Type	Le type d'environnement de la tâche : • Glue ETL : s'exécute dans un environnement Apache Spark géré par AWS Glue. • Glue Streaming : s'exécute dans un environnement Apache Spark et exécute ETL sur des flux de données. • Shell Python : exécute les scripts Python en tant que shell.
L'heure de début	Date et heure auxquelles cette exécution de tâche a démarré.
L'heure de fin	Date et heure auxquelles cette exécution de tâche s'est terminée.
Statut de l'exécution	État actuel de l'exécution de tâche. Les valeurs peuvent être : • `STARTING` • `RUNNING` • `STOPPING`

Propriété	Description
	• `STOPPED` • `SUCCEEDED` • `FAILED` • `TIMEOUT`
Durée de l'exécution	Le temps pendant lequel l'exécution de la tâche a consommé des ressources.
Capacity	Nombre d'unités de traitement de AWS Glue données (DPUs) allouées pour cette exécution de tâche. Pour plus d'informations sur la planification de la capacité, veuillez consulter Surveillance de la planification des capacités de DPU dans le Guide du développeur AWS Glue.

Propriété	Description
Type d'employé	Type d'employé prédéfini qui est alloué lorsqu'une tâche est exécutée. Les valeurs peuvent être `G.1X`, `G.2X`, `G.4X` ou `G.8X`. • **`G.1X`** – Lorsque vous choisissez ce type, vous devez également fournir une valeur pour Number of workers (Nombre d'employés). Chaque utilisateur est mappé sur 1 DPU (4 VCPUs, 16 Go de mémoire) avec un disque de 84 Go (environ 34 Go libres). Nous vous recommandons ce type d'employé pour les tâches utilisant beaucoup de mémoire. C'est la valeur Type d'employé par défaut pour les tâches AWS Glue de version 2.0 ou ultérieure. • **`G.2X`** – Lorsque vous choisissez ce type, vous devez également fournir une valeur pour Number of workers (Nombre d'employés). Chaque utilisateur mappe sur 2 processeurs (8 VCPUs, 32 Go de mémoire) avec un disque de 128 Go (environ 77 Go libres). Nous vous recommandons ce type d'employé pour des tâches qui requiert beaucoup de mémoire et des tâches qui effectuent des transformations Machine Learning. • **`G.4X`** – Lorsque vous choisissez ce type, vous devez également fournir une valeur pour Number of workers (Nombre d'employés). Chaque utilisateur mappe sur 4 processeurs (16 VCPUs, 64 Go de mémoire) avec un disque de 256 Go (environ 235 Go libres). Nous recommandons ce type de travailleur pour les tâches dont les charges de travail contiennent les transformations, les agrégatio

Propriété	Description
	ns, les jointures et les requêtes les plus exigeantes. Ce type de travailleur n'est disponible que pour les tâches ETL Spark des version 3.0 ou ultérieures d'AWS Glue dans les régions AWS suivantes : USA Est (Ohio), USA Est (Virginie du Nord), USA Ouest (Oregon), Asie-Pacifique (Singapour), Asie-Pacifique (Sydney), Asie-Pacifique (Tokyo), Canada (Centre), Europe (Francfort), Europe (Irlande) et Europe (Stockholm). • **`G.8X`** – Lorsque vous choisissez ce type, vous devez également fournir une valeur pour Number of workers (Nombre d'employés). Chaque utilisateur mappe 8 DPU (32 VCPUs, 128 Go de mémoire) sur un disque de 512 Go (environ 487 Go libres). Nous recommandons ce type de travailleur pour les tâches dont les charges de travail contiennent les transformations, les agrégations, les jointures et les requêtes les plus exigeantes. Ce type de travailleur n'est disponible que pour les tâches Spark ETL de AWS Glue version 3.0 ou ultérieure, dans les mêmes AWS régions que celles prises en charge pour le type de `G.4X` travailleur.
Heures DPU	Nombre estimé d'unités DPUs utilisées pour l'exécution de la tâche. Un DPU est une mesure relative de la puissance de traitement. DPUs sont utilisés pour déterminer le coût de fonctionnement de votre travail. Pour plus d'informations, consultez la page de tarification AWS Glue.

Vous pouvez sélectionner n'importe quelle exécution de tâche dans la liste et afficher des informations supplémentaires. Choisissez une exécution de tâche, puis procédez comme suit :

- Sélectionnez le menu Actions, puis l'option View job (Voir la tâche) pour afficher la tâche dans l'éditeur visuel.
- Sélectionnez le menu Actions, puis l'option Stop run (Arrêter l'exécution) pour arrêter l'exécution de la tâche en cours.
- Cliquez sur le bouton Afficher CloudWatch les journaux pour afficher les journaux d'exécution du travail correspondant à ce travail.
- Choisissez Afficher les détails pour afficher la page des détails de l'exécution de la tâche.

Afficher les journaux d'exécution de la tâche

Vous pouvez afficher les journaux des tâches de diverses manières :

- Sur la page Surveillance, dans le tableau Exécutions de tâches, choisissez une exécution de tâche, puis choisissez Afficher CloudWatch les journaux.
- Dans l'éditeur de tâches visuel, dans l'onglet Runs (Exécutions) d'une tâche, sélectionnez les liens hypertextes pour afficher les journaux :
  - Logs (Journaux) Liens vers les journaux de tâche Apache Spark écrits lorsque la journalisation continue est activée pour une exécution de tâche. Lorsque vous cliquez sur ce lien, vous accédez aux Amazon CloudWatch journaux du groupe de `/aws-glue/jobs/logs-v2` journaux. Par défaut, les journaux excluent les messages non utiles de Apache Hadoop YARN heartbeat et du pilote ou de l'exécuteur Apache Spark. Pour plus d'informations sur la journalisation continue, veuillez consulter la rubrique Journalisation continue pour les tâches AWS Glue dans le Guide du développeur AWS Glue.
  - Error logs (Journaux d'erreur) — Liens vers les journaux écrits dans `stderr` pour cette exécution de tâche. Lorsque vous sélectionnez ce lien, vous accédez aux journaux Amazon CloudWatch dans le groupe de journalisation `/aws-glue/jobs/error`. Vous pouvez utiliser ces journaux pour afficher les détails de toutes les erreurs rencontrées pendant l'exécution de la tâche.
  - Output logs (Journaux de sortie) — Liens vers les journaux écrits dans `stdout` pour cette exécution de tâche. Lorsque vous sélectionnez ce lien, vous accédez aux journaux Amazon CloudWatch dans le groupe de journalisation `/aws-glue/jobs/output`. De là, vous pouvez

afficher les détails sur les tables qui ont été créées dans AWS Glue Data Catalog et les erreurs qui ont été détectées.

Affichage des détails d'une exécution de tâche

Vous pouvez choisir une tâche dans la liste Job runs (Exécutions de tâche) sur la page Monitoring (Surveillance), puis choisir View run details (Afficher les détails de l'exécution) pour afficher des informations détaillées sur cette exécution de tâche.

Les informations affichées sur la page de détails de l'exécution de la tâche comprennent :

Propriété	Description
Nom du job	Nom de la tâche .
Statut de l'exécution	État actuel de l'exécution de tâche. Les valeurs peuvent être : • `STARTING` • `RUNNING` • `STOPPING` • `STOPPED` • `SUCCEEDED` • `FAILED` • `TIMEOUT`
Version Glue	La version AWS Glue utilisée par l'exécution de la tâche.
Tentative récente	Le nombre de tentatives automatiques de relance pour cette exécution de tâche.
L'heure de début	Date et heure auxquelles cette exécution de tâche a démarré.
L'heure de fin	Date et heure auxquelles cette exécution de tâche s'est terminée.

Propriété	Description
L'heure de début	Le temps consacré à la préparation de la tâche.
Durée d'exécution	Le temps consacré à l'exécution du script de tâche.
Nom du déclencheur	Le nom du déclencheur associé à la tâche.
L'heure de dernière modification	La date de dernière modification de la tâche.
Configuration de sécurité	La configuration de sécurité de la tâche, qui inclut le chiffrement Amazon S3, le chiffrement et les paramètres de CloudWatch chiffrement des signets de tâches.
Timeout	La valeur seuil du délai d'exécution de la tâche.
Capacité allouée	Nombre d'unités de traitement de AWS Glue données (DPUs) allouées pour cette exécution de tâche. Pour plus d'informations sur la planification de la capacité, veuillez consulter Surveillance de la planification des capacités de DPU dans le Guide du développeur AWS Glue.
Capacité max.	Capacité maximale disponible pour l'exécution de la tâche.
Nombre d'employés	Le nombre de travailleurs utilisés pour l'exécuti on de la tâche.

Propriété	Description
Type d'employé	Type d'employés prédéfinis alloués à l'exécuti on de la tâche. Les valeurs peuvent être `G.1X` ou `G.2X`. • **`G.1X`** – Lorsque vous choisissez ce type, vous devez également fournir une valeur pour Number of workers (Nombre d'employé s). Chaque travailleur est mappé à 1 DPU (4 VCPUs, 16 Go de mémoire, 64 Go de disque) et fournit 1 exécuteur par travaille ur. Nous vous recommandons ce type d'employé pour les tâches utilisant beaucoup de mémoire. C'est la valeur Type d'employé par défaut pour les tâches AWS Glue de version 2.0 ou ultérieure. • **`G.2X`** – Lorsque vous choisissez ce type, vous devez également fournir une valeur pour Number of workers (Nombre d'employé s). Chaque worker correspond à 2 DPUs (8 vCPUs, 32 Go de mémoire, 128 Go de disque) et fournit 1 exécuteur par worker. Nous vous recommandons ce type d'employé pour des tâches qui requiert beaucoup de mémoire et des tâches qui effectuent des transformations Machine Learning.
Journaux	Un lien vers les journaux de tâches pour la journalisation continue (`/aws-glue/jobs/logs-v2`).
Journaux de sortie	Un lien vers les fichiers journaux de sortie de la tâche (`/aws-glue/jobs/output`).

Propriété	Description
Journaux des erreurs	Un lien vers les fichiers de journalisation des erreurs de la tâche (`/aws-glue/jobs/error`).

Vous pouvez également afficher les éléments supplémentaires suivants, qui sont disponibles lorsque vous affichez les informations relatives aux exécutions de tâches récentes. Pour de plus amples informations, veuillez consulter the section called “Afficher les informations sur les exécutions de tâche récentes”.

- Arguments d'entrée
- Journaux continus
- Métriques : vous pouvez consulter des vues des métriques de base. Pour plus d'informations sur les métriques fournies, consultez the section called “Afficher Amazon CloudWatch les statistiques relatives à l'exécution d'une tâche Spark”.
- Interface utilisateur Spark : vous pouvez visualiser les journaux Spark relatifs à votre tâche dans l'interface utilisateur Spark. Pour plus d'informations sur l'utilisation de l'interface utilisateur web Spark, consultez the section called “Surveillance à l'aide de l'interface utilisateur Spark”. Activez cette fonctionnalité en suivant la procédure présentée dans the section called “Activation de l'interface utilisateur Spark pour des tâches”.

Afficher Amazon CloudWatch les statistiques relatives à l'exécution d'une tâche Spark

Sur la page de détails d'une exécution de tâche, sous la section Détails de l'exécution, vous pouvez consulter les statistiques de la tâche. AWS Glue Studioenvoie des métriques de tâche Amazon CloudWatch pour chaque exécution de tâche.

AWS Glue rapporte les métriques Amazon CloudWatch toutes les 30 secondes. Les métriques AWS Glue représentent des valeurs delta des valeurs précédemment rapportées. Le cas échéant, les tableaux de bord de métriques regroupent (additionnent) les valeurs de plages de 30 secondes pour obtenir une valeur pour la totalité de la dernière minute. Toutefois, les métriques d'Apache AWS Glue Spark transmises à Amazon CloudWatch sont généralement des valeurs absolues qui représentent l'état actuel au moment où elles sont signalées.

> **Note**
>
> Vous devez configurer votre compte pour accéder à Amazon CloudWatch,.

Les métriques fournissent des informations sur votre exécution de tâche, telles que :

- ETL Data Movement (Mouvement de données ETL) — Nombre d'octets lus ou écrits dans Amazon S3.
- Memory Profile: Heap used (Profil de la mémoire : tas utilisé) — Le nombre d'octets de mémoire utilisés par le tas de la machine virtuelle Java (JVM).
- Memory Profile: heap usage (Profil mémoire : utilisation du tas) — La fraction de la mémoire (échelle : 0-1), indiquée en pourcentage, utilisée par le tas de la JVM.
- CPU Load (Charge CPU) — La fraction de la charge du système CPU utilisée (échelle : 0-1), indiquée en pourcentage.

Afficher Amazon CloudWatch les statistiques relatives à l'exécution d'une tâche Ray

Sur la page de détails d'une exécution de tâche, sous la section Détails de l'exécution, vous pouvez consulter les statistiques de la tâche. AWS Glue Studioenvoie des métriques de tâche Amazon CloudWatch pour chaque exécution de tâche.

AWS Glue rapporte les métriques Amazon CloudWatch toutes les 30 secondes. Les métriques AWS Glue représentent des valeurs delta des valeurs précédemment rapportées. Le cas échéant, les tableaux de bord de métriques regroupent (additionnent) les valeurs de plages de 30 secondes pour obtenir une valeur pour la totalité de la dernière minute. Toutefois, les métriques d'Apache AWS Glue Spark transmises à Amazon CloudWatch sont généralement des valeurs absolues qui représentent l'état actuel au moment où elles sont signalées.

> **Note**
>
> Vous devez configurer votre compte pour y accéder Amazon CloudWatch, comme décrit dans.

Dans tâches Ray, vous pouvez afficher les graphiques de métriques agrégés suivants. Vous pouvez ainsi créer un profil de votre cluster et de vos tâches, ainsi qu'accéder à des informations détaillées sur chaque nœud. Les séries chronologiques à l'appui de ces graphiques sont disponibles CloudWatch pour une analyse plus approfondie.

Profil de tâche : statut de la tâche

Indique le nombre de tâches Ray dans le système. Chaque cycle de vie d'une tâche se voit attribuer sa propre série temporelle.

Profil de tâche : nom de la tâche

Indique le nombre de tâches Ray dans le système. Seules les tâches en attente et actives sont affichées. Chaque type de tâche (par son nom) se voit attribuer sa propre série temporelle.

Profil du cluster : CPUs en cours d'utilisation

Indique le nombre de cœurs de processeur utilisés. Chaque nœud se voit attribuer sa propre série temporelle. Les nœuds sont identifiés par des adresses IP, qui sont éphémères et ne servent qu'à l'identification.

Profil du cluster : utilisation de la mémoire du magasin d'objets

Indique l'utilisation de la mémoire par le cache d'objets Ray. Chaque emplacement de mémoire (mémoire physique, mémoire mise en cache sur disque et déversée dans Amazon S3) reçoit sa propre série temporelle. Le magasin d'objets gère le stockage de données sur tous les nœuds du cluster. Pour en savoir plus, consultez Objects dans la documentation Ray.

Profil du cluster : nombre de nœuds

Affiche le nombre de nœuds provisionnés pour le cluster.

Détail du nœud : utilisation du processeur

Affiche l'utilisation du processeur sur chaque nœud sous forme de pourcentage. Chaque série affiche un pourcentage agrégé de l'utilisation du processeur sur tous les cœurs du nœud.

Détail du nœud : utilisation de la mémoire

Affiche l'utilisation de la mémoire sur chaque nœud en Go. Chaque série montre la mémoire agrégée entre tous les processus du nœud, y compris les tâches Ray et le processus de stockage Plasma. Cela ne reflète pas les objets stockés sur le disque ou déversés sur Amazon S3.

Détail du nœud : utilisation du disque

Affiche l'utilisation du disque sur chaque nœud en Go.

Détail du nœud : I/O vitesse du disque

Affiche le I/O disque de chaque nœud en Ko/s.

Détail du nœud : I/O débit du réseau

Affiche I/O le réseau de chaque nœud en Ko/s.

Détail du nœud : utilisation du processeur par le composant Ray

Affiche l'utilisation du processeur dans fractions d'un cœur. Chaque composant Ray sur chaque nœud se voit attribuer sa propre série temporelle.

Détail du nœud : utilisation de la mémoire par le composant Ray

Affiche l'utilisation de la mémoire en Gio. Chaque composant Ray sur chaque nœud se voit attribuer sa propre série temporelle.

Détecter et traiter les données sensibles

La transformation Detect PII identifie les informations personnelles identifiables (PII) dans votre source de données. Vous choisissez l'entité PII à identifier, la manière dont vous voulez que les données soient analysées et ce qui doit être fait avec l'entité PII qui a été identifiée par la transformation Detect PII.

La transformation Détecter les PII permet de détecter, masquer ou supprimer des entités que vous définissez ou qui sont prédéfinies par AWS. Cela vous permet d'accroître la conformité et de réduire la responsabilité. Par exemple, vous voudrez peut-être vous assurer qu'aucune information personnelle ne peut être lue dans vos données et masquer les numéros de sécurité sociale avec une chaîne fixe (telle que xxx-xx-xxxx), des numéros de téléphone ou des adresses.

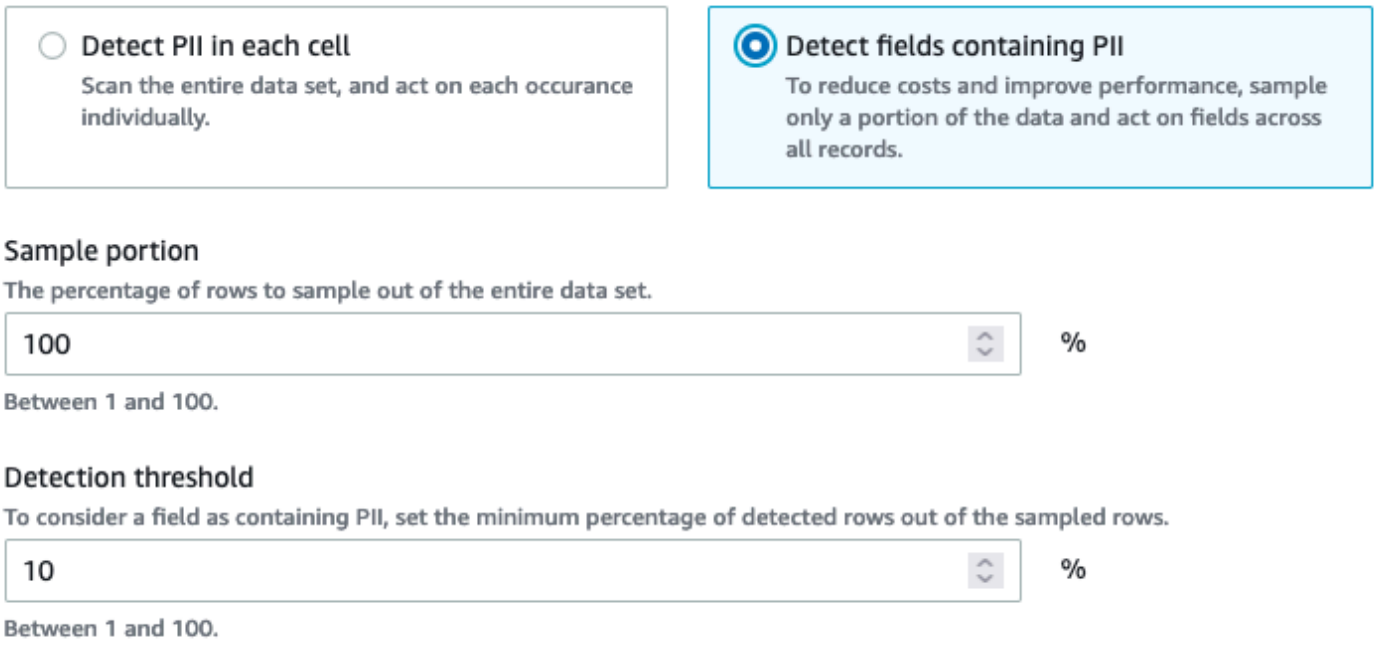
Pour travailler avec des données sensibles en dehors de AWS Glue Studio, consultez Utilisation de la détection des données sensibles en dehors de AWS Glue Studio

Rubriques

- Choix du mode d'analyse des données
- Choix des entités PII à détecter
- Spécification du niveau de sensibilité de détection
- Choix de ce qu'il faut faire avec les données PII identifiées
- Ajout des remplacements précis des actions

Choix du mode d'analyse des données

Lorsque vous analysez votre jeu de données à la recherche de données sensibles telles que des données d'identification personnelle (PII), vous pouvez choisir de détecter les données d'identification personnelle dans chaque ligne ou de détecter les colonnes contenant des données d'identification personnelle.



Lorsque vous choisissez Detect PII in each cell (Détectez les PII dans chaque cellule), vous choisissez d'analyser toutes les lignes de la source de données. Il s'agit d'une analyse complète qui permet de s'assurer que les entités PII sont identifiées.

Lorsque vous choisissez Detect fields containing PII (Détecter les champs contenant des informations personnelles identifiables), vous choisissez d'analyser un échantillon de lignes à la recherche d'entités PII. Il s'agit d'un moyen de réduire les coûts et les ressources tout en identifiant les champs dans lesquels les entités PII sont trouvées.

Lorsque vous choisissez de détecter des champs contenant des informations personnelles, vous pouvez réduire les coûts et améliorer les performances en échantillonnant une partie de lignes. Le choix de cette option vous permettra de spécifier des options supplémentaires :

- Portion d'échantillon : vous permet de spécifier le pourcentage de lignes à échantillonner. Par exemple, si vous saisissez « 50 », vous spécifiez que vous voulez que 50 % des lignes soient analysées pour l'entité PII.

- Seuil de détection : vous permet de préciser le pourcentage de lignes contenant l'entité PII afin que la colonne entière soit identifiée comme ayant l'entité PII. Par exemple, si vous saisissez le nombre « 10 », vous spécifiez que le numéro de l'entité PII, US Phone, dans les lignes analysées doit être égal ou supérieur à 10 % pour que le champ soit identifié comme ayant l'entité PII, US Phone. Si le pourcentage de lignes contenant l'entité PII est inférieur à 10 %, ce champ ne sera pas étiqueté comme comportant l'entité PII, US Phone, dans celui-ci.

Choix des entités PII à détecter

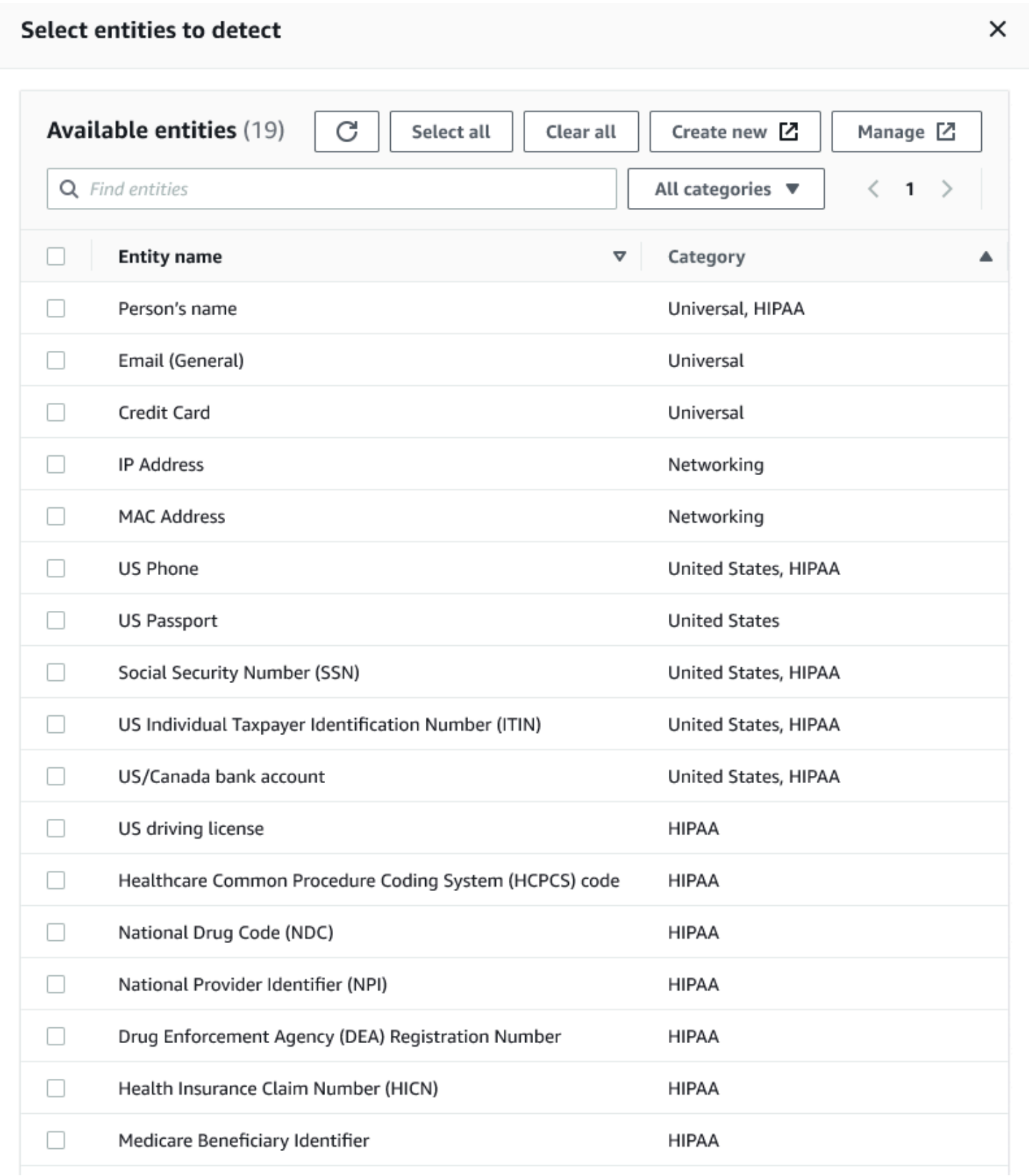
Si vous avez choisi Detect PII in each cell (Détectez les PII dans chaque cellule), vous pouvez choisir parmi trois options :

- Tous les modèles d'informations personnelles disponibles, y compris AWS les entités.
- Sélectionnez des catégories : les modèles PII incluent automatiquement des modèles dans les catégories que vous sélectionnez.
- Sélectionnez des modèles spécifiques : seuls les modèles que vous sélectionnez seront détectés.

Pour afficher la liste complète des types de données sensibles gérés, consultez Managed data types.

Choisir parmi tous les modèles PII disponibles

Si vous sélectionnez Tous les modèles d'informations personnelles disponibles, sélectionnez les entités prédéfinies par. AWS Vous pouvez sélectionner une, plusieurs ou toutes les entités.


Select entities to detect
Available entities (19)
Select all
Clear all
Create new
Manage
Find entities
All categories
1
Entity name
Category
Person's name
Universal, HIPAA
Email (General)
Universal
Credit Card
Universal
IP Address
Networking
MAC Address
Networking
US Phone
United States, HIPAA
US Passport
United States
Social Security Number (SSN)
United States, HIPAA
US Individual Taxpayer Identification Number (ITIN)
United States, HIPAA
US/Canada bank account
United States, HIPAA
US driving license
HIPAA
Healthcare Common Procedure Coding System (HCPCS) code
HIPAA
National Drug Code (NDC)
HIPAA
National Provider Identifier (NPI)
HIPAA
Drug Enforcement Agency (DEA) Registration Number
HIPAA
Health Insurance Claim Number (HICN)
HIPAA
Medicare Beneficiary Identifier
HIPAA

Sélectionner des catégories

Si vous avez choisi Select categories (Sélectionner des catégories) comme les modèles PII à détecter, vous pouvez sélectionner les options du menu déroulant. Notez que certaines entités peuvent appartenir à plusieurs catégories. Par exemple, Nom de la personne est une entité appartenant aux catégories Universel et HIPAA.

- Universel (exemples : e-mail, carte de crédit)
- HIPAA (exemples : permis de conduire américain, code du Healthcare Common Procedure Coding System (HCPCS))
- Réseaux (exemples : adresse IP, adresse MAC)
- Argentine
- Australie
- Autriche
- Belgique
- Bosnie
- Bulgarie
- Canada
- Chili
- Colombie
- Croatie
- Chypre
- Tchéquie
- Danemark
- Estonie
- Finlande
- France
- Allemagne
- Grèce
- Hongrie
- Irlande
- Corée

- Japon
- Mexique
- Pays-Bas
- Nouvelle-Zélande
- Norvège
- Portugal
- Roumanie
- Singapour
- Slovaquie
- Slovénie
- Espagne
- Suède
- Suisse
- Turquie
- Ukraine
- États-Unis
- Royaume-Uni
- Venezuela

Sélectionner des modèles spécifiques

Si vous choisissez Select specific patterns (Sélectionner des modèles spécifiques) en tant que modèles PII à détecter, vous pouvez rechercher ou parcourir une liste de modèles que vous avez déjà créés, ou créer un modèle d'entité de détection.

Les étapes ci-dessous décrivent comment créer un nouveau modèle personnalisé pour détecter des données sensibles. Vous allez créer le modèle personnalisé en saisissant un nom pour le modèle personnalisé, en ajoutant une expression régulière et, éventuellement, en définissant des mots contextuels.

1. Pour créer un modèle, cliquez sur le bouton Créer un nouveau

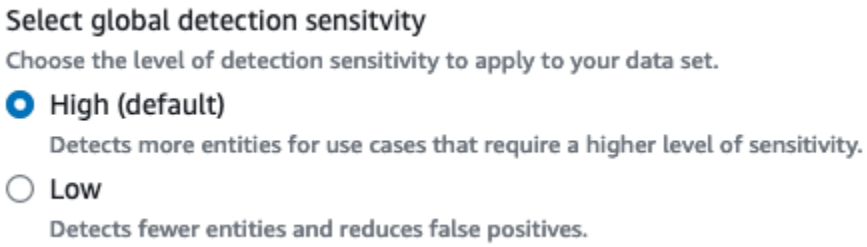
2. Dans la page Créer une entité de détection, saisissez le nom de l'entité et une expression régulière. L'expression régulière (Regex) est ce que AWS Glue utilisera pour faire correspondre des entités.
3. Cliquez sur Valider. Si la validation est réussie, un message de confirmation s'affiche indiquant que la chaîne est une expression régulière valide. Si la validation échoue, vous verrez un message indiquant que la chaîne n'est pas conforme au formatage approprié et aux littéraux de caractères, aux opérateurs ou aux constructions acceptés.
4. Vous pouvez choisir d'ajouter des mots contextuels en plus de l'expression régulière. Les mots contextuels peuvent augmenter la probabilité d'une correspondance. Ils peuvent être utiles dans les cas où les noms des champs ne sont pas descriptifs de l'entité. Par exemple, les numéros de sécurité sociale peuvent être nommés « SSN » ou « SS ». L'ajout de ces mots contextuels peut aider à faire correspondre l'entité.
5. Cliquez sur Créer pour créer l'entité de détection. Toutes les entités créées sont visibles dans la console AWS Glue Studio. Cliquez sur Detection entities (Entités de détection) dans le menu de navigation de gauche.

   Vous pouvez modifier, supprimer ou créer des entités de détection depuis la page Entités de détection. Vous pouvez également rechercher un modèle à l'aide du champ de recherche.

Spécification du niveau de sensibilité de détection

Vous pouvez définir le niveau de sensibilité lorsque vous utilisez la détection de données sensibles.

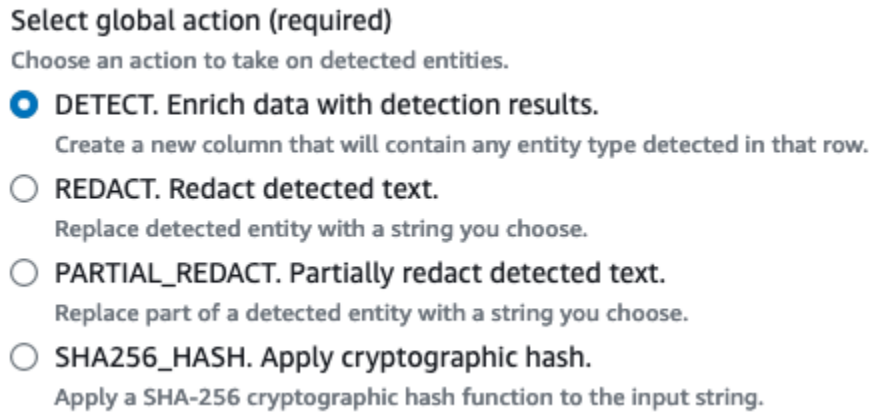
- Élevé : (par défaut) détecte un plus grand nombre d'entités pour les cas d'utilisation nécessitant un niveau de sensibilité plus élevé. Toutes les tâches AWS Glue créés après novembre 2023 sont automatiquement activés pour ce paramètre.
- Faible : détecte moins d'entités et réduit le nombre de faux positifs.



Choix de ce qu'il faut faire avec les données PII identifiées

Si vous avez choisi de détecter les PII dans l'ensemble de la source de données, vous pouvez sélectionner une action globale à appliquer :

- Enrichir les données avec des résultats de détection : Si vous avez choisi de détecter les PII dans chaque cellule, vous pouvez stocker les entités détectées dans une nouvelle colonne.
- Remplacer le texte détecté : Vous pouvez remplacer la valeur de PII détectée par une chaîne que vous spécifiez dans le champ de saisie Remplacement de texte facultatif. Si aucune chaîne n'est spécifiée, l'entité PII détectée est remplacée par '*******'.
- Censurer partiellement le texte détecté : vous pouvez remplacer une partie de la valeur PII détectée par une chaîne de votre choix. Deux options sont possibles : soit laisser les extrémités non masquées, soit les masquer en fournissant un modèle régulier explicite. Cette fonction n'est pas disponible dans AWS Glue 2.0.
- Appliquer un hachage de chiffrement : vous pouvez transmettre la valeur PII détectée à une fonction de hachage de chiffrement SHA-256 et remplacer la valeur par la sortie de la fonction.



Différences entre les versions AWS Glue 2.0 et 3.0 et ultérieures

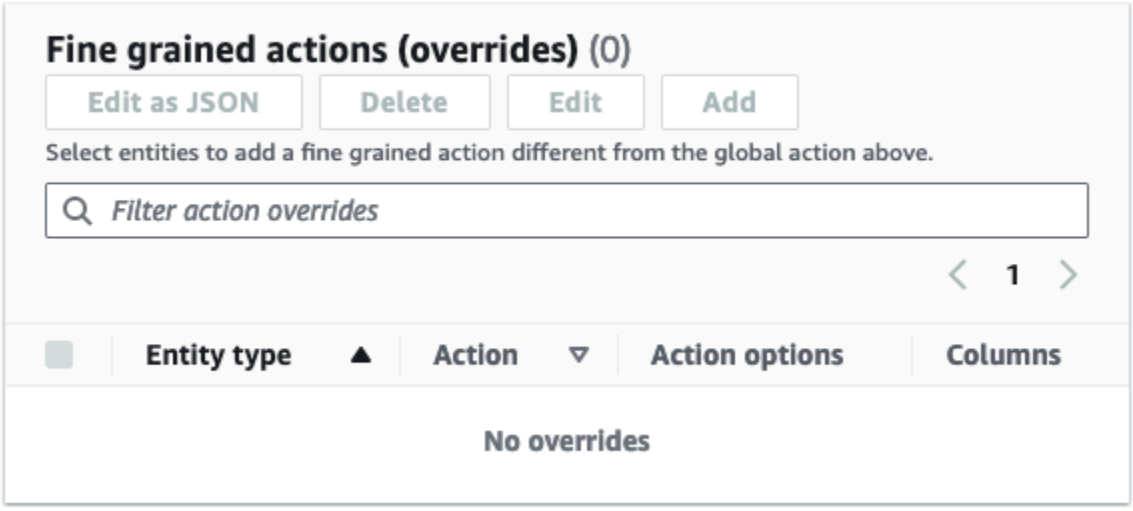
AWS GlueLes tâches 2.0 renverront une nouvelle DataFrame avec les informations PII détectées pour chaque colonne dans une colonne supplémentaire. Tout travail de censure ou de hachage est visible dans le script AWS Glue dans l'onglet visuel.

AWS GlueLes tâches 3.0 et 4.0 renverront une nouvelle DataFrame avec cette même colonne supplémentaire. Une nouvelle clé pour « actionUsed » est présente et peut être l'une des valeurs `DETECT`, `REDACT`, `PARTIAL_REDACT` ou `SHA256_HASH`. Si une action de masquage est sélectionnée, les données DataFrame seront renvoyées avec des données sensibles masquées.

Ajout des remplacements précis des actions

Des paramètres de détection et d'action supplémentaires peuvent être ajoutés à la table des remplacements précis des actions. Cela vous permet de :

- Inclure ou exclure certaines colonnes de la détection : un schéma déduit sur la source de données remplira la table avec les colonnes disponibles.
- Spécifier des paramètres spécifiques plus précis qu'en utilisant des actions globales : par exemple, vous pouvez spécifier différents paramètres de texte de censure pour différents types d'entités.
- Spécifier une action différente de l'action globale : si une action différente doit être appliquée à un autre type de données sensibles, vous pouvez le faire ici. Notez que deux edit-in-place actions différentes (rédaction et hachage) ne peuvent pas être utilisées sur la même colonne, mais que la détection peut toujours être utilisée.



Gestion des tâches ETL avec AWS Glue Studio

Vous pouvez utiliser l'interface graphique simple dans AWS Glue Studio pour gérer vos tâches ETL. Dans le menu de navigation, choisissez Jobs (Tâches)pour afficher la page Jobs (Tâches). Sur cette page, vous pouvez voir toutes les tâches que vous avez créées soit avec AWS Glue Studio, soit avec la console AWS Glue. Vous pouvez afficher, gérer et exécuter vos tâches sur cette page.

Cette page vous permet également d'effectuer les actions suivantes :

- Démarrer une exécution de tâche
- Planifier des exécutions de tâche
- Gestion des planifications de tâche
- Arrêter les exécutions de tâche
- Voir vos tâches
- Afficher les informations sur les exécutions de tâche récentes
- Afficher le script de tâche
- Modifier les propriétés de tâche
- Sauvegarder la tâche
- Cloner une tâche
- Supprimer une tâche

Démarrer une exécution de tâche

Dans AWS Glue Studio, vous pouvez exécuter vos tâches à la demande. Une tâche peut s'exécuter plusieurs fois, et chaque fois que vous l'exécutez, AWS Glue recueille des informations sur les activités et les performances de celle-ci. Cette information est appelée job run (exécution de tâches) et est identifiée par un ID d'exécution de tâche.

Vous pouvez lancer une exécution de tâche de la manière suivante dans AWS Glue Studio :

- Sur la page Jobs (Tâches), choisissez la tâche à démarrer, puis choisissez le bouton Run job (Exécuter une tâche).
- Si vous affichez une tâche dans l'éditeur visuel et que la tâche a été sauvegardée, vous pouvez choisir le bouton Run (Exécuter) pour démarrer une exécution de tâche.

Pour plus d'informations sur les exécutions de tâche, consultezUtilisation des tâches sur la console AWS Glue dans le Guide du développeur AWS Glue .

Planifier des exécutions de tâche

Dans AWS Glue Studio, vous pouvez créer un planning pour que vos tâches soient exécutées à des moments précis. Vous pouvez spécifier des contraintes, telles que le nombre d'exécutions des tâches, les jours de la semaine où elles sont exécutées et l'heure à laquelle elles le sont. Ces contraintes sont basées sur `cron` et possèdent les mêmes limitations que `cron`. Par exemple, si vous choisissez d'exécuter votre tâche le 31 de chaque mois, n'oubliez pas que certains mois ne comportent pas 31 jours. Pour plus d'informations sur `cron`, veuillez consulter la rubrique Cron expressions dans le Guide du développeur AWS Glue .

Pour exécuter des tâches selon une planification

1. Créez une planification de tâche à l'aide d'une des méthodes suivantes :
   - Sur la page Jobs (Tâches), choisissez la tâche pour laquelle vous voulez créer une planification, choisissez Actions, puis choisissez Schedule job (Planification de tâches).
   - Si vous visualisez une tâche dans l'éditeur visuel et que cette tâche a été enregistrée, choisissez l'onglet Schedules (Planifications). Puis choisissez Create Schedule (Créer une planification).
2. Sur la pageSchedule job run (Planification des exécutions de tâches), saisissez les informations suivantes :
   - Name (Nom) : saisissez un nom pour votre planification de tâche.
   - Frequency (Fréquence) : entrez la fréquence de la planification de tâche. Vous pouvez choisir parmi les options suivantes :
     - Hourly (Horaire) : la tâche s'exécutera toutes les heures, à partir d'une minute spécifique. Vous pouvez spécifier la Minute de l'heure à laquelle la tâche doit être exécutée. Par défaut, lorsque vous choisissez horaire, la tâche s'exécute au début de l'heure (minute 0).
     - Daily (Journalier) : La tâche s'exécutera tous les jours, en commençant à un moment donné. Vous pouvez spécifier les valeurs de Minute et Start hour (Heure de début) pour définir l'heure à laquelle la tâche s'exécutera. Les heures sont spécifiées à l'aide d'une horloge de 23 heures, où vous utilisez les chiffres 13 à 23 pour les heures de l'après-midi. Les valeurs par défaut pour les minutes et les heures sont 0, ce qui signifie que si vous sélectionnez Daily (Journalier), la tâche s'exécutera par défaut à minuit.

- Weekly (Hebdomadaire) : la tâche s'exécutera chaque semaine, un ou plusieurs jours. En plus des mêmes paramètres décrits précédemment pour Daily, vous pouvez choisir les jours de la semaine où la tâche sera exécutée. Vous pouvez choisir un ou plusieurs jours.
- Monthly (Mensuel) : la tâche s'exécutera tous les mois à un jour précis. En plus des mêmes paramètres que ceux décrits précédemment pour Daily, vous pouvez choisir le jour du mois où la tâche sera exécutée. Spécifiez le jour sous la forme d'une valeur numérique comprise entre 1 et 31. Si vous sélectionnez un jour qui n'existe pas au cours d'un mois, par exemple le 30e jour de février, la tâche ne sera pas exécutée ce mois-là.
- Custom (Personnalisée) : entrez une expression pour votre planification de tâche à l'aide de la syntaxe `cron`. Les expressions Cron vous permettent de créer des planifications plus complexes, telles que le dernier jour du mois (au lieu d'un jour spécifique du mois) ou tous les trois mois, les 7e et 21e jours du mois.

  Veuillez consulter la rubrique Expressions cron dans le Guide du développeur AWS Glue

- Description : vous pouvez éventuellement saisir une description pour votre planification de tâche. Si vous envisagez d'utiliser la même planification pour plusieurs tâches, la description vous permet de déterminer plus facilement l'action de la planification.

3. Choisissez Create schedule (Créer une planification) pour enregistrer la planification de la tâche.
4. Une fois que vous avez créé la planification, un message de réussite s'affiche en haut de la page de la console. Vous pouvez choisir Job details (Détails de la tâche) dans cette bannière pour afficher les détails de la tâche. Cela ouvre la page de l'éditeur de tâches visuel, avec l'onglet Schedules (Planifications) sélectionné.

Gestion des planifications de tâche

Une fois que vous avez créé des planifications pour une tâche, vous pouvez l'ouvrir dans l'éditeur visuel et choisir l'onglet Schedules (Planifications) pour gérer les planifications.

Sur la page Schedules (Planifications) de l'éditeur visuel, vous pouvez effectuer les tâches suivantes :

- Créer une planification.

  Choisissez Create schedule (Création de planification), puis entrez les informations de votre planification comme décrit dans the section called "Planifier des exécutions de tâche".

- Modifier une planification existante.

Choisissez la planification à modifier, puis choisissez Action suivi de Edit schedule (Modifier une planification). Lorsque vous choisissez de modifier une planification existante, la Frequency (Fréquence) s'affiche comme Custom (Personnalisée), et la planification s'affiche comme une expression `cron`. Vous pouvez modifier l'expression `cron`, ou spécifier une nouvelle planification à l'aide du bouton Frequency (Fréquence). Lorsque vous avez terminé vos modifications, choisissez Update schedule (Mettre à jour la planification).

- Suspendre une planification active.

  Choisissez une planification active, puis choisissez Action suivi de Pause schedule (Suspendre la planification). La planification est désactivée instantanément. Choisissez le bouton d'actualisation (rechargement) pour consulter l'état actualisé de la planification de tâche.

- Reprendre une planification suspendue.

  Choisissez une planification inactive, puis choisissez Action suivi de Resume schedule (Reprendre la planification). La planification est activée instantanément. Choisissez le bouton d'actualisation (rechargement) pour consulter l'état actualisé de la planification de tâche.

- Supprimer une planification.

  Choisissez la planification à supprimer, puis choisissez Action suivi de Delete schedule (Supprimer une planification). La planification est supprimée instantanément. Choisissez le bouton d'actualisation (rechargement) pour consulter la liste actualisée de planification de tâche. La planification affichera un état Deleting (Suppression) jusqu'à ce qu'elle ait été complètement supprimée.

Arrêter les exécutions de tâche

Vous pouvez arrêter une tâche avant qu'elle n'ait terminé son exécution. Vous pouvez choisir cette option si vous savez que la tâche n'est pas configurée correctement ou si la tâche prend trop de temps à s'exécuter.

Sur la page Monitoring (Surveillance), dans la liste Job runs (Exécutions de tâche), choisissez la tâche que vous souhaitez arrêter, choisissez Actions, puis choisissez Stop run (Arrêter l'exécution).

Voir vos tâches

Vous pouvez afficher toutes vos tâches sur la page Jobs (Tâches). Vous pouvez accéder à cette page en choisissant Jobs (Tâches) dans le panneau de navigation.

Sur la page Jobs (Tâches), vous pouvez voir toutes les tâches qui ont été créées dans votre compte. La liste Your jobs (Vos tâches) affiche le nom de la tâche, son type, le statut de la dernière exécution de cette tâche, ainsi que les dates de création et de dernière modification de la tâche. Vous pouvez choisir le nom d'une tâche pour afficher des informations détaillées sur celle-ci.

Vous pouvez également utiliser le tableau de bord Surveillance pour afficher toutes vos tâches. Vous pouvez accéder au tableau de bord en sélectionnant Monitoring (Surveillance) dans le panneau de navigation.

Personnalisation de l'affichage des tâches

Vous pouvez personnaliser la façon dont les tâches sont affichées dans la section Your jobs (Vos tâches) de la page Jobs (Tâches). En outre, vous pouvez saisir du texte dans le champ de recherche pour afficher uniquement les tâches dont le nom contient ce texte.

Si vous choisissez l'icône de paramètres

dans la section Your jobs (Vos tâches), vous pouvez personnaliser la façon dont AWS Glue Studio affiche les informations contenues dans le tableau. Vous pouvez choisir de regrouper les lignes de texte dans l'affichage, de modifier le nombre de tâches affichées sur la page et de spécifier les colonnes à afficher.

Afficher les informations sur les exécutions de tâche récentes

Une tâche peut être exécutée plusieurs fois à mesure que de nouvelles données sont ajoutées à l'emplacement source. Chaque fois qu'une tâche s'exécute, un ID unique lui est attribué et des informations sur cette exécution sont collectées. Vous pouvez consulter ces informations en utilisant les méthodes suivantes :

- Choisissez l'onglet Runs (Exécutions) de l'éditeur visuel pour afficher les informations d'exécution de la tâche actuellement affichée.

  Sur l'onglet Runs (Exécutions) (la page Recent job runs (Exécutions de tâche récentes)), il y a une carte pour chaque exécution de tâche. Les informations affichées sur l'onglet Runs (Exécutions)comprennent :

  - ID d'exécution de la tâche
  - Nombre de tentatives d'exécution de cette tâche
  - Statut de l'exécution de la tâche

  - Heure de début et de fin de l'exécution de la tâche
  - Le moteur d'exécution de la tâche
  - Un lien vers les fichiers du journal de tâche
  - Un lien vers les fichiers de journalisation des erreurs de tâche
  - L'erreur renvoyée pour les tâches ayant échoué
- Vous pouvez sélectionner une tâche pour afficher des informations supplémentaires sur celle-ci, notamment :
  - Arguments d'entrée
  - Journaux continus
  - Métriques : vous pouvez consulter des vues des métriques de base. Pour plus d'informations sur les métriques fournies, consultez the section called "Afficher Amazon CloudWatch les statistiques relatives à l'exécution d'une tâche Spark".
  - Interface utilisateur Spark : vous pouvez visualiser les journaux Spark relatifs à votre tâche dans l'interface utilisateur Spark. Pour plus d'informations sur l'utilisation de l'interface utilisateur web Spark, consultez the section called "Surveillance à l'aide de l'interface utilisateur Spark". Activez cette fonctionnalité en suivant la procédure présentée dans the section called "Activation de l'interface utilisateur Spark pour des tâches".

Vous pouvez sélectionner Afficher les détails pour afficher des informations similaires sur la page des détails de l'exécution de la tâche. Vous pouvez également accéder à la page des détails de l'exécution de la tâche via la page Surveillance. Dans le panneau de navigation, choisissez Surveillance. Faites défiler jusqu'à la liste Job runs (Exécutions de tâche). Choisissez la tâche, puis choisissez View run details (Afficher les détails de l'exécution). Le contenu est décrit dans Affichage des détails d'une exécution de tâche.

Pour plus d'informations sur les valeurs de tâche, veuillez consulter Afficher les journaux d'exécution de la tâche.

Afficher le script de tâche

Après avoir fourni les informations pour tous les nœuds de la tâche, AWS Glue Studio génère un script qui est utilisé par la tâche pour lire les données de la source, transformer les données et écrire les données dans l'emplacement cible. Si vous sauvegardez la tâche, vous pouvez afficher ce script à tout moment.

Pour afficher le script généré pour votre tâche

1. Dans le panneau de navigation, sélectionnez Jobs (Tâches).
2. Sur la page Jobs (Tâches), dans la liste our jobs (Vos tâches), choisissez le nom de la tâche que vous souhaitez vérifier. Vous pouvez également sélectionner une tâche dans la liste, choisir le menu Actions, puis choisir Edit job (Modifier une tâche).
3. Sur la page de l'éditeur visuel, choisissez l'onglet Script en haut pour afficher le script de tâche.

   Si vous souhaitez modifier le script de tâche, veuillez consulter AWS Glue guide de programmation.

Modifier les propriétés de tâche

Les nœuds du diagramme de tâche définissent les actions effectuées par la tâche, mais il existe également plusieurs propriétés que vous pouvez configurer pour la tâche. Ces propriétés déterminent l'environnement dans lequel la tâche s'exécute, les ressources qu'elle utilise, les paramètres de seuil, les paramètres de sécurité, etc.

Pour personnaliser l'environnement d'exécution de la tâche

1. Dans le panneau de navigation, sélectionnez Jobs (Tâches).
2. Sur la page Jobs (Tâches), dans la liste our jobs (Vos tâches), choisissez le nom de la tâche que vous souhaitez vérifier.
3. Sur la page de l'éditeur visuel, choisissez l'onglet Job details (Détails de la tâche) en haut du volet de modification des tâches.
4. Modifiez les propriétés de la tâche, si nécessaire.

   Pour plus d'informations sur les propriétés de tâche, veuillez consulter Définition des propriétés d'une tâche dans le Guide du développeur AWS Glue .
5. Développez la section Advanced properties (Propriétés avancées) si vous devez spécifier les propriétés de tâche supplémentaires suivantes :

   - Script filename (Nom du fichier de script) — Le nom du fichier qui stocke le script de tâche dans Amazon S3.
   - Script path (Chemin du script) — L'emplacement Amazon S3 dans lequel le script de tâche est stocké.

- Métriques de tâche — (Non disponible pour les tâches shell Python) Active la création de Amazon CloudWatch métriques lors de l'exécution de cette tâche.
- Journalisation continue — (Non disponible pour les tâches du shell Python) Active la journalisation continue afin que les journaux puissent être consultés avant la fin de la tâche. CloudWatch
- Spark UI (Interface utilisateur Spark) et Spark UI logs path (Chemin des journaux de l'interface utilisateur Spark) — (Non disponible pour les tâches shell Python) Active l'utilisation de l'interface utilisateur Spark pour la surveillance de ce travail et spécifie l'emplacement des journaux de l'interface utilisateur Spark.
- Maximum concurrency (Concurrence maximale) — Définit le nombre maximal d'exécutions simultanées autorisées pour cette tâche.
- Temporary path (Chemin temporaire) — Chemin temporaire — L'emplacement d'un répertoire de travail dans Amazon S3 où les résultats intermédiaires temporaires sont écrits lorsque AWS Glue exécute le script de tâche.
- Delay notification threshold (minutes) (Seuil de notification de délai, en minutes) — Spécifie un seuil de temporisation pour la tâche. Si la tâche s'exécute pendant une durée supérieure à celle spécifiée par le seuil, AWS Glue envoie une notification de retard pour la tâche à CloudWatch.
- Security configuration (Configuration de la sécurité) et Server-side encryption (Chiffrement côté serveur) — Utilisez ces champs pour choisir les options de chiffrement de la tâche.
- Use Glue Data Catalog as the Hive metastore (Utiliser le catalogue de données Glue en tant que metastore Hive) — Choisissez cette option si vous souhaitez utiliser AWS Glue Data Catalog comme alternative à Apache Hive Metastore.
- Additional network connection (Connexion réseau supplémentaire) — Pour une source de données dans un VPC, vous pouvez spécifier une connexion de type `Network` afin de vous assurer que votre tâche accède à vos données via le VPC.
- Python library path (Chemin de la bibliothèque Python), Dependent jars path (Chemin des fichiers .jar dépendants) (Non disponible pour les tâches shell Python), ou Referenced files path (Chemin de fichiers référencés) — Utilisez ces champs pour spécifier l'emplacement des fichiers supplémentaires utilisés par la tâche lors de l'exécution du script.
- Job Parameters (Paramètres de tâche) — Vous pouvez ajouter un ensemble de paires clé-valeur qui sont transmises comme paramètres nommés au script de tâche. Dans les appels Python à AWS Glue API s, il est préférable de transmettre les paramètres explicitement par leur nom. Pour de plus amples informations sur l'utilisation des paramètres dans un script

de tâche, veuillez consulter la rubrique Transmission de paramètres Python et accès à ces paramètres dans AWS Glue dans le Guide du développeur AWS Glue .

- Tags (Balises) — Vous pouvez ajouter des balises à la tâche pour mieux les organiser et les identifier.

6. Après avoir modifié les propriétés de la tâche, sauvegardez la tâche.

Stocker les fichiers Spark shuffle sur Amazon S3

Certaines tâches ETL nécessitent la lecture et la combinaison d'informations provenant de plusieurs partitions, par exemple lors de l'utilisation d'une transformation de jointure. Cette opération est appelée shuffling (brassage). Lors d'un brassage, les données sont écrites sur le disque et transférées sur le réseau. Avec AWS Glue version 3.0, vous pouvez configurer Amazon S3 comme emplacement de stockage pour ces fichiers. AWS Glue fournit un gestionnaire de brassage qui écrit et lit les fichiers de brassage vers et depuis Amazon S3. L'écriture et la lecture de fichiers shuffle depuis Amazon S3 sont plus lentes (de 5 % à 20 %) par rapport au disque local (ou à Amazon EBS, qui est fortement optimisé pour Amazon). EC2 Toutefois, Amazon S3 offre une capacité de stockage illimitée, de sorte que vous n'avez pas à vous soucier des erreurs « `No space left on device` » lors de l'exécution de votre tâche.

Pour configurer votre tâche de manière à utiliser Amazon S3 pour les fichiers de brassage

1. Sur la page Jobs (Tâches), dans la liste our jobs (Vos tâches), choisissez le nom de la tâche que vous souhaitez modifier.
2. Sur la page de l'éditeur visuel, choisissez l'onglet Job details (Détails de la tâche) en haut du volet de modification des tâches.

   Faites défiler jusqu'à la section Job parameters (Paramètres de la tâche).
3. Spécifiez les paires clé-valeur suivantes.

   - `--write-shuffle-files-to-s3` — `true`

     Il s'agit du paramètre principal qui configure le gestionnaire de brassage dans AWS Glue pour utiliser les compartiments Amazon S3 pour l'écriture et la lecture des données de brassage. Par défaut, ce paramètre a une valeur de `false`.
   - (Facultatif) `--write-shuffle-spills-to-s3` — `true`

Ce paramètre vous permet de décharger les fichiers de déversement vers des compartiments Amazon S3, ce qui fournit une résilience supplémentaire à votre tâche Spark dans AWS Glue. Ceci n'est requis que pour les charges de travail volumineuses qui déversent beaucoup de données sur le disque. Par défaut, ce paramètre a une valeur de `false`.

- (Facultatif) `--conf spark.shuffle.glue.s3ShuffleBucket` — `S3://<shuffle-bucket>`

  Ce paramètre spécifie le compartiment Amazon S3 à utiliser lors de l'écriture des fichiers de brassage. Si vous ne définissez pas ce paramètre, l'emplacement est le dossier `shuffle-data` dans l'emplacement spécifié pour Temporary path (Chemin temporaire) (`--TempDir`).

  > **Note**
  >
  > Assurez-vous que l'emplacement du compartiment shuffle se trouve dans le même emplacement que celui Région AWS dans lequel la tâche s'exécute.
  > En outre, le service de brassage ne nettoie pas les fichiers une fois la tâche terminée. Vous devez donc configurer les stratégies de cycle de vie de stockage Amazon S3 sur l'emplacement du compartiment de brassage. Pour plus d'informations, veuillez consulter Gestion du cycle de vie des objets dans le Guide de l'utilisateur Amazon S3.

Sauvegarder la tâche

Un rappel rouge Job has not been saved (La tâche n'a pas été sauvegardée) s'affiche à gauche du bouton Save (Enregistrer) jusqu'à ce que vous sauvegardiez la tâche.

Pour sauvegarder votre tâche

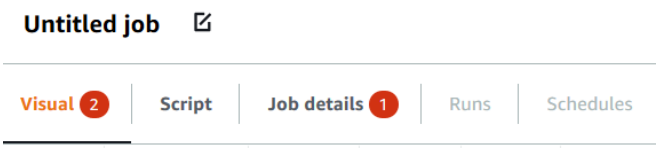
1. Fournissez toutes les informations requises dans les onglets Visual (Visuel) et Job details (Détails de la tâche).
2. Choisissez le bouton Enregistrer.

   Après avoir enregistré la tâche, le rappel « not saved (non sauvegardée) » change pour afficher l'heure et la date de la dernière sauvegarde de la tâche.

Si vous quittez AWS Glue Studio avant d'enregistrer votre tâche, la prochaine fois que vous vous connectez à AWS Glue Studio, une notification s'affiche. La notification indique qu'il existe une tâche non enregistrée et vous demande si vous souhaitez la restaurer. Si vous choisissez de restaurer la tâche, vous pouvez continuer à la modifier.

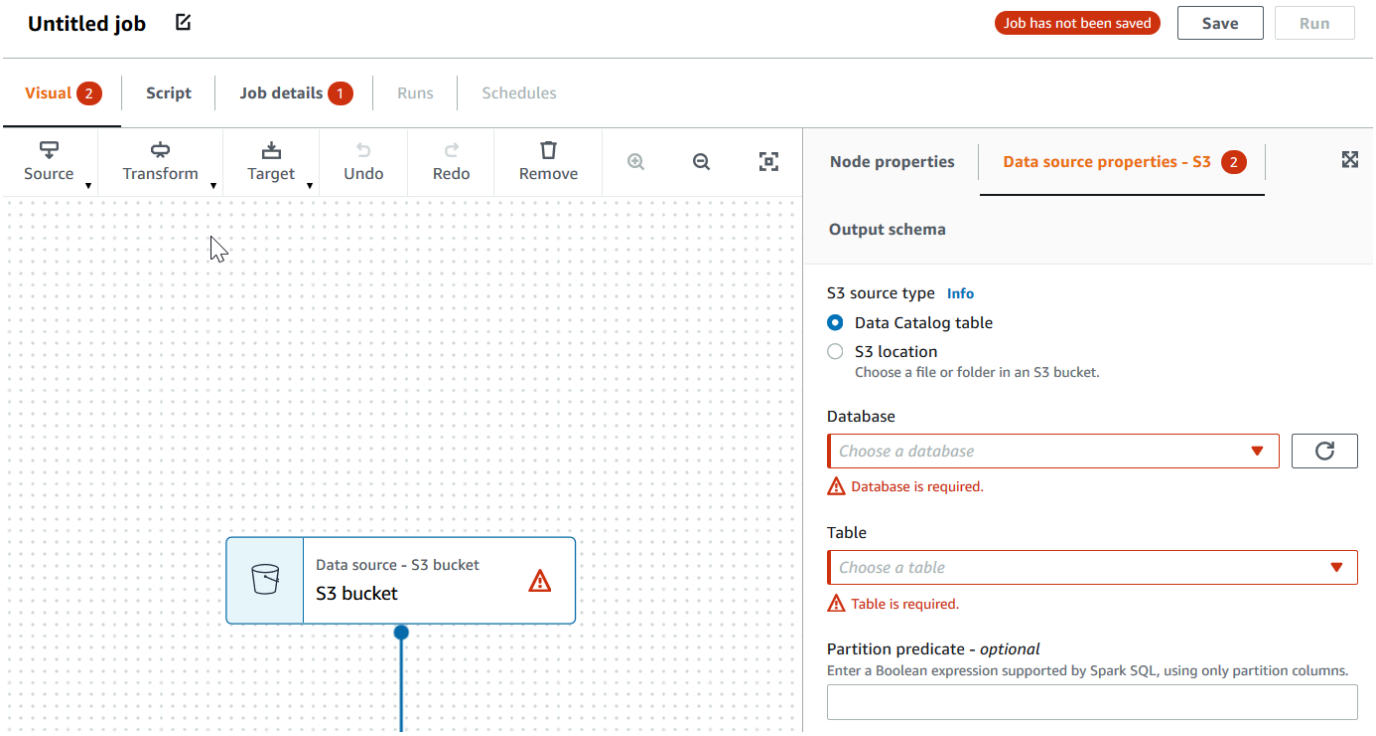
Résolution des erreurs lors de l'enregistrement d'une tâche

Si vous choisissez le bouton Save (Enregistrer), mais qu'il manque certaines informations requises dans votre tâche, un rappel rouge apparaît sur l'onglet où les informations sont manquantes. Le numéro dans le rappel indique le nombre de champs manquants détectés.

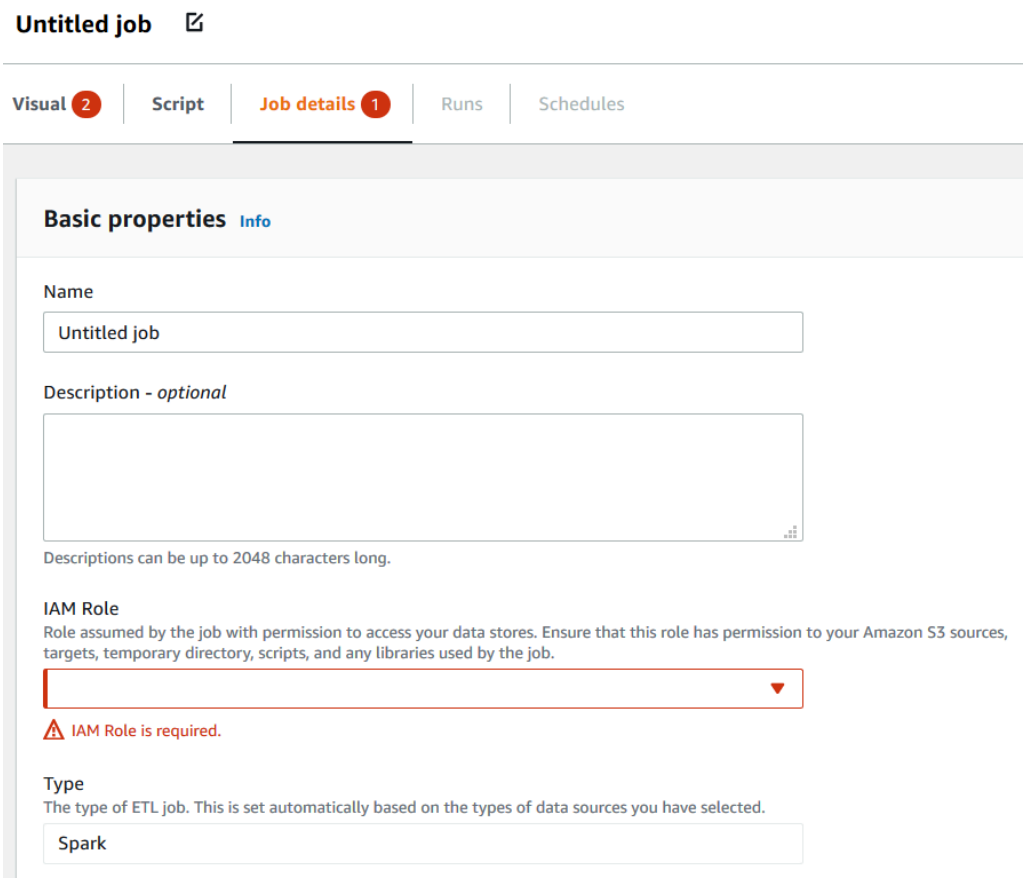


- Si un nœud de l'éditeur visuel n'est pas configuré correctement, l'onglet Visual (Visuel) affiche un rappel rouge et le nœud erroné affiche un symbole d'avertissement ⚠ .

  1. Choisissez le nœud. Dans le panneau de détails du nœud, un rappel rouge apparaît sur l'onglet où se trouvent les informations manquantes ou incorrectes.
  2. Choisissez l'onglet dans le panneau de détails du nœud qui affiche un rappel rouge, puis recherchez les champs problématiques qui sont mis en évidence. Un message d'erreur sous les champs fournit des informations supplémentaires sur le problème.



- S'il y a un problème avec les propriétés de la tâche, l'onglet Job details (Détails de la tâche) affiche un rappel rouge. Choisissez cet onglet et localisez les champs problématiques, qui sont mis en évidence. Les messages d'erreur situés sous les champs fournissent des informations supplémentaires sur le problème.



Cloner une tâche

Vous pouvez utiliser l'action Clone job (Cloner la tâche) pour copier une tâche existante dans une nouvelle tâche.

Pour créer une tâche en copiant une tâche existante

1. Sur la page Jobs (Tâches), dans la liste our jobs (Vos tâches), choisissez la tâche que vous souhaitez dupliquer.
2. Dans le menu Actions, choisissez Clone job (Cloner la tâche).
3. Entrez le nom de la nouvelle tâche. Vous pouvez ensuite enregistrer ou modifier la tâche.

Supprimer une tâche

Vous pouvez supprimer des tâches dont vous n'avez plus besoin. Vous pouvez supprimer une ou plusieurs tâches en une seule opération.

Pour supprimer des tâches de AWS Glue Studio

1. Sur la page Jobs (Tâches), dans la liste our jobs (Vos tâches), choisissez les tâches que vous souhaitez supprimer.
2. Dans le menu Actions, choisissez Delete job (Supprimer la tâche).
3. Vérifiez que vous souhaitez supprimer la tâche en saisissant **`delete`**.

Vous pouvez également supprimer une tâche enregistrée lorsque vous consultez l'onglet Job details (Détails de la tâche) dans l'éditeur visuel.

Utilisation des tâches dans AWS Glue

Les sections suivantes renseignent sur les tâches ETL et Ray dans AWS Glue.

Rubriques

- AWS Glue versions
- Utilisation des jobs Spark dans AWS Glue
- Travailler avec Ray Jobs dans AWS Glue
- Configuration des propriétés pour les tâches shell Python dans AWS Glue
- Surveillance AWS Glue

AWS Glue versions

Vous pouvez configurer le paramètre de AWS Glue version lorsque vous ajoutez ou mettez à jour une tâche. La AWS Glue version détermine les versions d'Apache Spark et de Python prises AWS Glue en charge. La version de Python indique la version qui est prise en charge pour les tâches de type Spark. Le tableau suivant répertorie les versions d' AWS Glue Glue disponibles, les versions Spark et Python correspondantes, ainsi que les autres modifications de fonctionnalité.

Vous pouvez utiliser les mises à niveau de Generative AI pour Apache Spark afin de mettre à niveau vos jobs Glue ETL des anciennes versions de Glue (≥ 2.0) vers la dernière version de Glue.

AWS Glue versions

AWS Glue version	Versions d'environnement d'exécution prises en charge	Version Java prise en charge	Changements de fonctionnalité
AWS Glue 5,1	• Spark 3.5.6 • Python 3.11 • Scala 2.12.18	Java 17	Outre la mise à niveau du moteur Spark, des optimisations et des mises à niveau sont intégrées à cette AWS Glue version, telles que :

AWS Glue version	Versions d'environnement d'exécution prises en charge	Version Java prise en charge	Changements de fonctionnalité
			• Formats de table ouverts (OTF) mis à jour vers Hudi 1.0.2, Iceberg 1.10.0 et Delta Lake 3.3.2 • Version 3.0 du format Iceberg • Accès complet à la table Hudi (FTA) avec lecture et écriture • DDL/DML Opérations de contrôle d'accès à grain fin (FGAC) natif de Spark sur les tables Hive, Iceberg et Delta Lake enregistrées dans Lake Formation • Contexte d'audit pour les appels d'API Glue et Lake Formation dans le CloudTrail journal • Support pour Iceberg Materialized Views

AWS Glue version	Versions d'environnement d'exécution prises en charge	Version Java prise en charge	Changements de fonctionnalité
AWS Glue 5,0	• Spark 3.5.4 • Python 3.11 • Scala 2.12.18	Java 17	En plus des mises à jour du cadre, des optimisations et des mises à niveau sont intégrées à cette version d' AWS Glue , telles que : • Assistance avec Amazon SageMaker Unified Studio • Assistance Amazon SageMaker Lakehouse • Open Table Formats (OTF) mis à jour vers Hudi 0.15.0, Iceberg 1.7.1 et Delta Lake 3.3.0 • Contrôle précis des accès natif Spark utilisant Lake Formation. • Prise en charge d'autorisations d'accès Amazon S3 • Prise en charge de `requirements.txt` pour installer des

AWS Glue version	Versions d'environnement d'exécution prises en charge	Version Java prise en charge	Changements de fonctionnalité
			bibliothèques Python supplémentaires • Support du lignage des données sur Amazon DataZone • Prise en charge des compartiments de tableaux Amazon S3 • AWS Glue Prise en charge de l'affichage multidialecte du catalogue de données Limitations Les limites de la AWS Glue version 5.0 sont les suivantes : • Le contrôle d'accès au niveau de la table `GlueContext` basé sur Glue Dynamic Frame/ avec AWS Lake Formation des autorisations prises en charge dans Glue 4.0 ou

AWS Glue version	Versions d'environnement d'exécution prises en charge	Version Java prise en charge	Changements de fonctionnalité
			antérieur n'est pas pris en charge dans Glue 5.0. Utilisez le nouveau contrôle précis des accès (FGAC) natif Spark de Glue 5.0. Pour plus d'informations sur la migration vers la AWS Glue version 5.0, consultez Migration AWS Glue des tâches Spark vers la AWS Glue version 5.0.

AWS Glue version	Versions d'environnement d'exécution prises en charge	Version Java prise en charge	Changements de fonctionnalité

AWS Glue 4,0	Versions de l'environnement Spark
	• Spark 3.3.0
	• Python 3.10
		Java 8	AWS Glue La version 4.0 intègre un certain nombre d'optimisations et de mises à niveau, telles que : AWS Glue
			• De nombreuses mises à niveau des fonctionnalités Spark de Spark 3.1 vers Spark 3.3 :
			  • Plusieurs améliorations de fonctionnalités lorsqu'il est associé à Pandas. Pour plus d'informations, consultez Nouveautés de Spark 3.3.
			  • Optimisations supplémentaires développées sur Amazon EMR.
			  • Mise à niveau vers le système de fichiers EMR (EMRFS) 2.53.

AWS Glue version	Versions d'environnement d'exécution prises en charge	Version Java prise en charge	Changements de fonctionnalité
			• Migration de Log4j 2 à partir de Log4j 1.x • Plusieurs mises à jour de modules Python depuis AWS Glue version 3.0, comme une version de Boto mise à niveau. • Mise à niveau de plusieurs connecteurs, notamment le connecteur Amazon Redshift par défaut. Consultez Annexe C : Mises à niveau des connecteurs. • Mise à niveau de plusieurs pilotes JDBC. Consultez Annexe B : Mises à niveau du pilote JDBC. • Utilisation d'un nouveau connecteur Amazon Redshift et d'un nouveau pilote JDBC. • Prise en charge native des infrastructures de lac de

AWS Glue version	Versions d'environnement d'exécution prises en charge	Version Java prise en charge	Changements de fonctionnalité
			données ouverts avec Apache Hudi, Delta Lake et Apache Iceberg. • Prise en charge native du plug-in Cloud Shuffle Storage basé sur Amazon S3 (un plug-in Apache Spark) permettant d'utiliser Amazon S3 pour la réorganisation et la capacité de stockage élastique. Limitations Voici les limites de AWS Glue 4.0 : • AWS Glue l'apprentissage automatique et les transformations des informations personnelles identifiables (PII) ne sont pas encore disponibles dans la AWS Glue version 4.0.

AWS Glue version	Versions d'environnement d'exécution prises en charge	Version Java prise en charge	Changements de fonctionnalité
			Pour plus d'informations sur la migration vers AWS Glue version 4.0, consultez Migration AWS Glue des tâches Spark vers la AWS Glue version 4.0.

AWS Glue version	Versions d'environnement d'exécution prises en charge	Version Java prise en charge	Changements de fonctionnalité
	Versions de l'environnement Ray • Ray 2.4.0 Python 3.9	N/A	Créez et exécutez des applications Python distribuées avec AWS Glue for Ray. • Prend en charge la distribution de données Ray-2.4.0 (`ray[data]`) avec Python 3.9. Pour plus d'informations sur cette version de Ray, consultez Ray-2.4.0 dans le référentiel Ray. GitHub • Prend en charge l'installation de bibliothèques Python supplémentaires dans l'environnement d'exécution `Ray2.4`. Pour de plus amples informations, veuillez consulter the section called "Modules Python supplémentaires pour les tâches Ray".

AWS Glue version	Versions d'environnement d'exécution prises en charge	Version Java prise en charge	Changements de fonctionnalité
			• Intègre les journaux et les statistiques de Ray Jobs avec Amazon CloudWatch. Pour plus d'informations, consultez the section called "Dépannage des erreurs Ray" et the section called "Métriques des tâches Ray". • Regroupe et visualise les métriques des tâches Ray dans AWS Glue Studio, sur la page d'exécution de chaque tâche. • Prend en charge la distribution des fichiers dans chaque répertoire de travail de votre cluster, le déversement d'objets du magasin d'objets Ray vers Amazon S3 et le contrôle du nombre

AWS Glue version	Versions d'environnement d'exécution prises en charge	Version Java prise en charge	Changements de fonctionnalité
			minimum de composants master alloués à votre tâche Ray. Pour de plus amples informations, veuillez consulter the section called "Paramètres de tâche Ray". Limitations des tâches Ray dans la version 4.0 de AWS Glue • AWS Glue les sessions interactives pour Ray sont toujours disponibles en avant-première pour cette version. • AWS Glue pour Ray, l'intégration avec Amazon VPC n'est pas disponible actuellement. Les ressources d'un VPC in ne AWS seront pas accessibles sans itinéraire public. Pour plus d'informa

AWS Glue version	Versions d'environnement d'exécution prises en charge	Version Java prise en charge	Changements de fonctionnalité
			tions sur l'utilisation AWS Glue avec Amazon VPC, consultez. the section called "Configuration de points de terminaison d'un VPC d'interface (AWS PrivateLink) pour AWS Glue " • AWS Glue for Ray est disponible dans l'est des États-Unis (Virginie du Nord), dans l'est des États-Unis (Ohio), dans l'ouest des États-Unis (Oregon), en Asie-Pacifique (Tokyo) et en Europe (Irlande).

AWS Glue version	Versions d'environnement d'exécution prises en charge	Version Java prise en charge	Changements de fonctionnalité
AWS Glue 3,0	• Spark 3.1.1 • Python 3.7	Java 8	En plus de la mise à niveau du moteur Spark vers 3.0, des optimisations et des mises à niveau sont intégrées à cette version de AWS Glue , telles que : • Construit la bibliothèque AWS Glue ETL par rapport à Spark 3.0, qui est une version majeure de Spark. • Les tâches de streaming sont prises en charge sur AWS Glue 3.0. • Inclut de nouvelles optimisations d'exécution de AWS Glue Spark en termes de performances et de fiabilité : • Traitement plus rapide des colonnes en mémoire basé sur Apache Arrow pour la lecture

AWS Glue version	Versions d'environnement d'exécution prises en charge	Version Java prise en charge	Changements de fonctionnalité
			des données CSV. • Exécution basée sur SIMD pour les lectures vectorisées avec des données CSV. • La mise à niveau Spark inclut également des optimisations supplémentaires développées sur Amazon EMR. • EMRFS mis à niveau de la version 2.38 à 2.46 offrant de nouvelles fonctionnalités et des corrections de bogues pour l'accès à Amazon S3. • Plusieurs dépendances requises pour la nouvelle version de Spark ont été mises à niveau.

AWS Glue version	Versions d'environnement d'exécution prises en charge	Version Java prise en charge	Changements de fonctionnalité
			• Pilotes JDBC mis à niveau pour nos sources de données prises en charge en mode natif. Limitations Voici les limites de AWS Glue 3.0 : • AWS Glue les transformations du machine learning ne sont pas encore disponibles dans la AWS Glue version 3.0. • Certains connecteurs Spark personnalisés ne fonctionnent pas avec AWS Glue 3.0 s'ils dépendent de Spark 2.4 et ne sont pas compatibles avec Spark 3.1.

AWS Glue version	Versions d'environnement d'exécution prises en charge	Version Java prise en charge	Changements de fonctionnalité
AWS Glue 2.0 (fin de vie le 1er avril 2026)	• Spark 2.4.3 • Python 3.7	N/A	Outre les fonctionnalités fournies dans la AWS Glue version 1.0, AWS Glue la version 2.0 fournit également : • Une infrastructure améliorée pour exécuter des tâches ETL Apache Spark AWS Glue avec des temps de démarrage réduits. • Désormais, la journalisation par défaut est en temps réel, avec des flux séparés pour les pilotes et les exécuteurs, ainsi que des sorties et des erreurs. • Prise en charge de la spécification de modules Python supplémentaires ou de versions différentes au niveau de la tâche.

AWS Glue version	Versions d'environnement d'exécution prises en charge	Version Java prise en charge	Changements de fonctionnalité
			Note AWS Glue la version 2.0 diffère de AWS Glue la version 1.0 pour certaines dépendances et versions en raison de modifications architecturales sous-jacentes. Validez vos tâches AWS Glue avant de migrer vers les versions AWS Glue majeures.

AWS Glue version	Versions d'environnement d'exécution prises en charge	Version Java prise en charge	Changements de fonctionnalité
AWS Glue 1,0 (fin de vie le 1er avril 2026)	• Spark 2.4.3 • Python 2.7 • Python 3.6	N/A	Vous pouvez gérer les signets de tâche pour les formats Parquet et ORC dans les tâches ETL AWS Glue (avec AWS Glue version 1.0). Auparavant, vous ne pouviez mettre en signet que les formats source courants d'Amazon S3 tels que JSON, CSV, Apache Avro et XML dans les tâches AWS Glue ETL. Lorsque vous définissez les options de format pour les entrées et sorties ETL, vous pouvez spécifier d'utiliser le reader/writer format Apache Avro 1.8 pour prendre en charge la lecture et l'écriture du type logique Avro (à l'aide de la AWS Glue version 1.0). Auparavant, seul le reader/writer format Avro version 1.7 était pris en charge.

AWS Glue version	Versions d'environnement d'exécution prises en charge	Version Java prise en charge	Changements de fonctionnalité
			Le type de connexion DynamoDB prend en charge une option d'écriture (à l'aide de AWS Glue version 1.0). Limitations Voici les limites de AWS Glue 1.0 : • AWS Glue les versions 0.9 et 1.0 ne seront pas disponibles en Asie-Pacifique (Jakarta) (`ap-southeast-3`), au Moyen-Orient (Émirats arabes unis) (`me-central-1`) ou dans d'autres nouvelles régions à l'avenir.

AWS Glue version	Versions d'environnement d'exécution prises en charge	Version Java prise en charge	Changements de fonctionnalité
AWS Glue 0,9 (fin de vie le 1er avril 2026)	• Spark 2.2.1 • Python 2.7	N/A	Tâches créées sans spécifier de AWS Glue version par défaut, AWS Glue 0,9. Limitations Voici les limites de AWS Glue 0.9 : • AWS Glue les versions 0.9 et 1.0 ne seront pas disponibles en Asie-Pacifique (Jakarta) (`ap-southeast-3`), au Moyen-Orient (Émirats arabes unis) (`me-central-1`) ou dans d'autres nouvelles régions à l'avenir.

> **Note**
>
> Les versions de Glue suivantes sont compatibles avec les versions suivantes de PythonShell :
>
> • PythonShell La version 3.6 est prise en charge dans la version 1.0 de Glue.
> • PythonShell La version 3.9 est prise en charge dans la version 3.0 de Glue.

De plus, les points de terminaison de développement ne sont pris en charge que dans les versions 1.0 et 0.9 de Glue.

Politique de prise en charge des versions AWS Glue

AWS Glue est un service d'intégration de données sans serveur qui facilite la découverte, la préparation et la combinaison de données pour l'analytique, le machine learning et le développement d'applications. Une tâche AWS Glue contient la logique métier qui effectue le travail d'intégration des données dans AWS Glue. Il existe trois types de tâches dans AWS Glue : Spark (lot et streaming) et Ray et shell Python. Lorsque vous définissez votre tâche, vous spécifiez la version de AWS Glue qui configure les versions dans l'environnement d'exécution sous-jacent Spark, Ray ou Python. Par exemple : une tâche Spark AWS Glue version 5.0 prend en charge Spark 3.5.4 et Python 3.11.

Politique de prise en charge

AWS Glue les versions sont construites autour d'une combinaison de systèmes d'exploitation, de langages de programmation et de bibliothèques logicielles soumises à des mises à jour de maintenance et de sécurité. AWS Glue la politique de support des versions consiste à mettre fin au support d'une version lorsqu'un composant majeur de la version atteint la fin du support communautaire à long terme (LTS) et que les mises à jour de sécurité ne sont plus disponibles. AWS Glue la politique de support des versions inclut les statuts suivants :

Fin du support (EOS) - Lorsqu'une AWS Glue version atteint EOS :

- AWS Glue n'appliquera plus de correctifs de sécurité ou d'autres mises à jour aux versions d'EOS.
- AWS Glue les offres d'emploi sur les versions EOS ne sont pas éligibles au support technique.
- AWS Glue peut ne pas être honoré SLAs lorsque les tâches sont exécutées sur des versions EOS.

Fin de vie (EOL) - Lorsqu'une AWS Glue version atteint la fin de vie :

- Vous ne pouvez plus créer de nouvelles AWS Glue tâches ni de sessions interactives sur les versions EOL.
- Vous ne pouvez plus démarrer l'exécution de tâches sur ces AWS Glue versions.
- AWS Glue arrêtera les exécutions de tâches existantes et les sessions interactives sur les versions EOL.

- Les versions EOL seront supprimées de AWS Glue SDKs et. APIs

Les versions suivantes d'AWS Glue ont atteint la fin du support et ne seront plus disponibles après la fin de vie. Les modifications du statut de support d'une version commencent à minuit (fuseau horaire du Pacifique) à la date spécifiée.

Type	Version Glue	Fin de l'assistance	Fin de vie
Spark	Glue version 0.9 (Spark 2.2, Scala 2, Python 2)	01/06/2022	01/04/2026
Spark	Glue version 1.0 (Spark 2.4, Python 2)	01/06/2022	01/04/2026
Spark	Glue version 1.0 (Spark 2.4, Scala 2, Python 3)	30/09/2022	01/04/2026
Spark	Glue version 2.0 (Spark 2.4, Python 3)	31/01/2024	01/04/2026
Type	Version Python	Fin de l'assistance	Fin de vie
(shell Python)	Python 2 (AWS Glue Version 1.0)	01/06/2022	01/04/2026
(shell Python)	PythonShell 3.6 (Glue version 1.0)	31/03/2026	NA
Type	Version Notebook	Fin de l'assistance	Fin de vie
Point de terminaison de développement	Zeppelin notebook	30/09/2022	NA

> **Note**
>
> La création de nouvelles tâches AWS Glue Python Shell 3.6 ne sera pas autorisée une fois le support terminé, le 31 mars 2026, mais vous pouvez continuer à mettre à jour et à exécuter les tâches existantes. Toutefois, les tâches exécutées sur des versions abandonnées ne sont pas éligibles au support technique. AWS Glue n'appliquera pas de correctifs de sécurité ou d'autres mises à jour aux versions abandonnées. AWS Glue ne sera pas non plus honoré SLAs lorsque les tâches sont exécutées sur des versions abandonnées.

AWS vous recommande fortement de migrer vos jobs vers des versions prises en charge.

Pour plus d'informations sur la migration de vos tâches Spark vers la dernière AWS Glue version, consultez la section Migration des AWS Glue tâches vers la AWS Glue version 5.1.

Pour migrer vos tâches de shell Python vers la dernière version de AWS Glue :

- Dans la console, choisissez `Python 3 (Glue Version 4.0)`.
- Dans l'UpdateJobAPI CreateJob/, définissez le `GlueVersion` paramètre sur `2.0` et sur `PythonVersion` `3` sous le `Command` paramètre. La configuration `GlueVersion` n'affecte pas le comportement des tâches shell Python. L'incrémentation de `GlueVersion` ne présente donc aucun avantage.
- Vous devez rendre votre script de tâche compatible avec Python 3.

Migration AWS Glue des tâches Spark vers la AWS Glue version 5.1

Cette rubrique décrit les modifications apportées entre AWS Glue les versions 0.9, 1.0, 2.0, 3.0, 4.0 et 5.0 pour vous permettre de migrer vos applications Spark et vos tâches ETL vers la AWS Glue version 5.1. Il décrit également les fonctionnalités de la AWS Glue version 5.1 et les avantages de son utilisation.

Pour utiliser cette fonctionnalité avec vos tâches AWS Glue ETL, **`5.1`** choisissez-la `Glue version` lors de la création de vos tâches.

Rubriques

- Nouvelles fonctionnalités
- Actions pour migrer vers la AWS Glue version 5.1

- Liste de contrôle de migration
- Migration de la AWS Glue version 5.0 vers AWS Glue la version 5.1
- Migration des anciennes AWS Glue versions vers AWS Glue la version 5.1
- Migration du connecteur et du pilote JDBC pour 5.1 AWS Glue

Nouvelles fonctionnalités

Cette section décrit les nouvelles fonctionnalités et les avantages de la AWS Glue version 5.1.

- Mise à jour d'Apache Spark de la version 3.5.4 en AWS Glue 5.0 à la version 3.5.6 en AWS Glue 5.1.
- Open Table Formats (OTF) mis à jour vers Hudi 1.0.2, Iceberg 1.10.0 et Delta Lake 3.3.2
- Vues matérialisées d'icebergs - Créez et gérez des vues matérialisées d'icebergs (MV). Pour plus d'informations, consultez le billet de blog
- Format Iceberg version 3.0 - Étend les types de données et les structures de métadonnées existantes pour ajouter de nouvelles fonctionnalités. Pour plus d'informations, consultez les spécifications de la table Iceberg.
- Accès complet aux tables Hudi - Contrôle de l'accès complet aux tables (FTA) pour Apache Hudi dans Apache Spark en fonction de vos politiques définies dans. AWS Lake Formation Cette fonctionnalité permet d'effectuer des opérations de lecture et d'écriture à partir de vos tâches AWS Glue ETL sur des tables AWS Lake Formation enregistrées lorsque le rôle de tâche dispose d'un accès complet aux tables.
- Supporte le contrôle d'accès fin natif (FGAC) à l'aide AWS Lake Formation d' DDL/DML opérations (telles que CREATE, ALTER, DELETE, DROP) avec un contrôle d'accès fin pour les tables Apache Hive, Apache Iceberg et Delta Lake enregistrées dans. AWS Lake Formation
- Contexte d'audit pour les tâches Spark - Le contexte d'audit pour les tâches AWS Glue ETL sera disponible AWS Glue et les appels AWS Lake Formation d'API seront disponibles dans les AWS CloudTrail journaux.

Limites et problèmes connus

Notez les problèmes et limitations connus suivants :

- Support limité pour la clause View SQL pour la création de vues matérialisées, la réécriture de requêtes et l'actualisation incrémentielle. Vous trouverez plus de détails sur la page de documentation de la fonctionnalité Iceberg Materialized Views.
- Les écritures Hudi FTA doivent être utilisées HoodieCredentialedHadoopStorage pour la vente d'informations d'identification lors de l'exécution du travail. Définissez la configuration suivante lors de l'exécution des tâches Hudi :

  `hoodie.storage.class=org.apache.spark.sql.hudi.storage.HoodieCredentialedHadoop`
- Le support d'écriture Hudi FTA fonctionne uniquement avec les configurations Hudi par défaut. Les paramètres Hudi personnalisés ou autres que ceux par défaut peuvent ne pas être entièrement pris en charge et peuvent entraîner un comportement inattendu. Le clustering pour les tables Hudi Merge-On-Read (MOR) n'est pas non plus pris en charge en mode d'écriture FTA.

Évolutions

Prenez note des modifications majeures suivantes :

- Le système de fichiers S3A a remplacé EMRFS en tant que connecteur S3 par défaut. Pour plus d'informations sur la façon de migrer, consultezMigration de la AWS Glue version 5.0 vers AWS Glue la version 5.1.

Actions pour migrer vers la AWS Glue version 5.1

Pour les tâches existantes, modifiez la `Glue version` depuis la version précédente vers `Glue 5.1` dans la configuration de la tâche.

- Dans AWS Glue Studio, choisissez « `Glue 5.1 - Supports Spark 3.5.6, Scala 2, Python 3` in `Glue version` ».
- Dans l'API, choisissez **`5.1`** dans le paramètre `GlueVersion` de l'opération d'API `UpdateJob`.

Pour les nouvelles tâches, choisissez `Glue 5.1` lorsque vous créez une tâche.

- Dans la console, choisissez `Spark 3.5.6, Python 3 (Glue Version 5.1) or Spark 3.5.6, Scala 2 (Glue Version 5.1)` dans `Glue version`.
- Dans AWS Glue Studio, choisissez « `Glue 5.1 - Supports Spark 3.5.6, Scala 2, Python 3` in `Glue version` ».

- Dans l'API, choisissez **`5.1`** dans le paramètre `GlueVersion` de l'opération d'API `CreateJob`.

Pour consulter les journaux d'événements Spark AWS Glue 5.1 issus de la AWS Glue version 2.0 ou d'une version antérieure, lancez un serveur d'historique Spark mis à niveau pour AWS Glue 5.1 à l'aide CloudFormation de Docker.

Liste de contrôle de migration

Consultez cette liste de contrôle pour la migration :

- [Python] Mettez à jour les références de démarrage de la version 1.34 à la version 1.40.

Migration de la AWS Glue version 5.0 vers AWS Glue la version 5.1

Tous les paramètres de travail existants et les fonctionnalités principales qui existent dans la AWS Glue version 5.0 existeront dans la AWS Glue version 5.1. Notez les modifications suivantes lors de la migration :

- Dans la AWS Glue version 5.1, le système de fichiers S3A a remplacé EMRFS comme connecteur S3 par défaut. Si `spark.hadoop.fs.s3a.endpoint` les deux ne `spark.hadoop.fs.s3a.endpoint.region` sont pas définis, la région par défaut utilisée par S3A est `us-east-2`. Cela peut entraîner des problèmes, tels que des erreurs de délai de chargement dans S3, en particulier pour les tâches VPC. Pour atténuer les problèmes causés par cette modification, définissez la configuration `spark.hadoop.fs.s3a.endpoint.region` Spark lorsque vous utilisez le système de fichiers S3A dans la AWS Glue version 5.1.
- Pour continuer à utiliser EMRFS au lieu de S3A, définissez les configurations Spark suivantes :

```
    --conf spark.hadoop.fs.s3.impl=com.amazon.ws.emr.hadoop.fs.EmrFileSystem
    --conf spark.hadoop.fs.s3n.impl=com.amazon.ws.emr.hadoop.fs.EmrFileSystem
    --conf
spark.hadoop.fs.AbstractFileSystem.s3.impl=org.apache.hadoop.fs.s3.EMRFSDelegate
```

Reportez-vous à la documentation sur la migration de Spark :

- Guide de migration : Spark Core
- Guide de migration : SQL, ensembles de données et DataFrame

- Guide de migration : streaming structuré
- Upgrading PySpark

Migration des anciennes AWS Glue versions vers AWS Glue la version 5.1

- Pour les étapes de migration liées à la AWS Glue version 4.0 vers la AWS Glue version 5.0, consultez la section Migration de la AWS Glue version 4.0 vers la AWS Glue version 5.0.
- Pour les étapes de migration liées à la AWS Glue version 3.0 vers la AWS Glue version 5.0, consultez la section Migration de la AWS Glue version 3.0 vers la AWS Glue version 5.0.
- Pour les étapes de migration liées à la version AWS Glue 2.0 vers la version AWS Glue 5.0 et la liste des différences de migration entre les AWS Glue versions 2.0 et 4.0, consultez la section Migration de la version AWS Glue 2.0 vers la version AWS Glue 5.0.

Migration du connecteur et du pilote JDBC pour 5.1 AWS Glue

Pour en savoir plus sur les versions des connecteurs JDBC et de lac de données qui ont été mises à niveau, consultez :

- Annexe B : Mises à niveau du pilote JDBC
- Annexe C : Mises à niveau des connecteurs
- Annexe D : Mises à niveau du format de table ouverte

Les modifications suivantes s'appliquent aux mises à niveau de la version OTF identifiées dans Annexe D : Mises à niveau du format de table ouverte la version AWS Glue 5.1.

Apache Hudi

Notez les modifications suivantes :

- Support de l'accès en lecture et en écriture au FTA sur les tables enregistrées de Lake Formation.

Apache Iceberg

Notez les modifications suivantes :

- Support du format Iceberg version 3. Les fonctionnalités suivantes sont prises en charge :
  - Transformations à arguments multiples pour le partitionnement et le tri.

  - Suivi du lignage des lignes.
  - Vecteurs de suppression. Pour en savoir plus, consultez un article de blog
  - Clés de chiffrement des tables.
  - Support des valeurs par défaut pour les colonnes.
- Support les écritures FGAC natives de Spark sur AWS Lake Formation les tables enregistrées.
- Compatibilité avec Athena SQL - Impossible de lire les tables Iceberg V3 créées par EMR Spark en raison d'une erreur : `GENERIC_INTERNAL_ERROR: Cannot read unsupported version 3`

Delta Lake

Notez les modifications suivantes :

- Support de l'accès en lecture et en écriture au FTA sur les tables enregistrées de Lake Formation.

Annexe A : Mises à niveau notables des dépendances

Voici les mises à niveau des dépendances :

Dépendance	Version en AWS Glue 5.1	Version en AWS Glue 5.0	Version en AWS Glue 4.0	Version en AWS Glue 3.0	Version en AWS Glue 2.0	Version en AWS Glue 1.0
Java	17	17	8	8	8	8
Spark	3.5.6	3.5.4	3.3.0-amzn-1	3.1.1-amzn-0	2.4.3	2.4.3
Hadoop	3.4.1	3.4.1	3.3.3-amzn-0	3.2.1-amzn-3	2.8.5-amzn-5	2.8.5-amzn-1
Scala	2,1,18	2,1,18	2,12	2,12	2.11	2.11
Jackson	2.15.2	2.15.2	2,12	2,12	2.11	2.11
Hive	2.3.9-amzn-4.	2.3.9-amzn-4.	2.3.9-amzn-2	2.3.7-amzn-4	1.2	1.2
EMRFS	2,73,0	2,69,0	2,54,0	2,46,0	2.38.0	2.30.0

Dépendance	Version en AWS Glue 5.1	Version en AWS Glue 5.0	Version en AWS Glue 4.0	Version en AWS Glue 3.0	Version en AWS Glue 2.0	Version en AWS Glue 1.0
Json4s	3.7.0-M11	3.7.0-M11	3.7.0-M11	3.6.6	3.5.x	3.5.x
Flèche	12,0.1	12,0.1	7.0.0	2.0.0	0.10.0	0.10.0
AWS Glue Client de catalogue de données	4.9.0	4.5.0	3.7.0	3.0.0	1.10.0	N/A
AWS SDK pour Java	2,35,5	2,29,52	1.12	1.12		
Python	3,11	3,11	3,10	3.7	2.7 et 3.6	2.7 et 3.6
Boto	1,40,61	1,34,131	1,26	1,18	1.12	N/A
Connecteur EMR DynamoDB	5.7.0	5.6.0	4,16,0			

Annexe B : Mises à niveau du pilote JDBC

Voici les mises à niveau du pilote JDBC :

Pilote	Version du pilote JDBC en 5.1 AWS Glue	Version du pilote JDBC en 5.0 AWS Glue	Version du pilote JDBC en 4.0 AWS Glue	Version du pilote JDBC en 3.0 AWS Glue	Version du pilote JDBC dans les versions précédentes AWS Glue
MySQL	8,0,33	8,0,33	8.0.23	8.0.23	5.1

Pilote	Version du pilote JDBC en 5.1 AWS Glue	Version du pilote JDBC en 5.0 AWS Glue	Version du pilote JDBC en 4.0 AWS Glue	Version du pilote JDBC en 3.0 AWS Glue	Version du pilote JDBC dans les versions précédentes AWS Glue
Microsoft SQL S	10.2.0	10.2.0	9.4.0	7.0.0	6.1.0
Oracle Databases	23,3.0,23,09	23,3.0,23,09	21,7	21,1	11.2
PostgreSQL	42.7.3	42.7.3	42,3.6	42,2,18	42,10
Amazon Redshi	redshift-jdbc42-2.1.0.29	redshift-jdbc42-2.1.0.29	redshift-jdbc42-2.1.0.16	redshift-jdbc41-1.2.12.1017	redshift-jdbc41-1.2.12.1017
SAP Hana	2,20,17	2,20,17	2,17,12		
Teradata	20,00.00.33	20,00.00.33	20,00.00.06		

Annexe C : Mises à niveau des connecteurs

Les mises à niveau des connecteurs sont les suivantes :

Pilote	Version du connecteur en AWS Glue 5.1	Version du connecteur en AWS Glue 5.0	Version du connecteur en AWS Glue 4.0	Version du connecteur en AWS Glue 3.0
Connecteur EMR DynamoDB	5.7.0	5.6.0	4,16,0	
Amazon Redshift	6.4.2	6.4.0	6.1.3	
OpenSearch	1.2.0	1.2.0	1.0.1	
MongoDB	10.3.0	10.3.0	10,0.4	3.0.0

Pilote	Version du connecteur en AWS Glue 5.1	Version du connecteur en AWS Glue 5.0	Version du connecteur en AWS Glue 4.0	Version du connecteur en AWS Glue 3.0
Snowflake	3.1.1	3.0.0	2.12.0	
Google BigQuery	0,32,2	0,32,2	0,32,2	
AzureCosmos	4,33,0	4,33,0	4,22,0	
AzureSQL	1.3.0	1.3.0	1.3.0	
Vertica	3.3.5	3.3.5	3.3.5	

Annexe D : Mises à niveau du format de table ouverte

Les mises à niveau du format de table ouverte sont les suivantes :

OTF	Version du connecteur en AWS Glue 5.1	Version du connecteur en AWS Glue 5.0	Version du connecteur en AWS Glue 4.0	Version du connecteur en AWS Glue 3.0
Hudi	1.0.2	0,15,0	0.12.1	0,1,1
Delta Lake	3.3.2	3.3.0	2.1.0	1.0.0
Iceberg	1.10.0	1.7.1	1.0.0	0.13.1

Migration AWS Glue des tâches Spark vers la AWS Glue version 5.0

Cette rubrique décrit les modifications apportées entre AWS Glue les versions 0.9, 1.0, 2.0, 3.0 et 4.0 pour vous permettre de migrer vos applications Spark et vos tâches ETL vers la AWS Glue version 5.0. Il décrit également les fonctionnalités de la AWS Glue version 5.0 et les avantages de son utilisation.

Pour utiliser cette fonctionnalité avec vos tâches AWS Glue ETL, **`5.0`** choisissez-la `Glue version` lors de la création de vos tâches.

Rubriques

- Nouvelles fonctionnalités
- Actions pour migrer vers la AWS Glue version 5.0
- Liste de contrôle de migration
- AWS Glue Caractéristiques de la version 5.0
- Migration de la AWS Glue version 4.0 vers AWS Glue la version 5.0
- Migration de la AWS Glue version 3.0 à AWS Glue la version 5.0
- Migration de la AWS Glue version 2.0 vers AWS Glue la version 5.0
- Changements de comportement de journalisation dans la AWS Glue version 5.0
- Migration du connecteur et du pilote JDBC pour la version 5.0 AWS Glue

Nouvelles fonctionnalités

Cette section décrit les nouvelles fonctionnalités et les avantages de la AWS Glue version 5.0.

- Mise à jour d'Apache Spark de la version 3.3.0 dans la AWS Glue version 4.0 à la version 3.5.4 dans AWS Glue la version 5.0. Consultez Améliorations majeures de Spark 3.3.0 à Spark 3.5.4.
- Contrôle précis des accès (FGAC) natif Spark utilisant Lake Formation. Cela inclut FGAC pour les tables Iceberg, Delta et Hudi. Pour plus d'informations, consultez la section Utilisation AWS Glue avec AWS Lake Formation pour un contrôle d'accès précis.

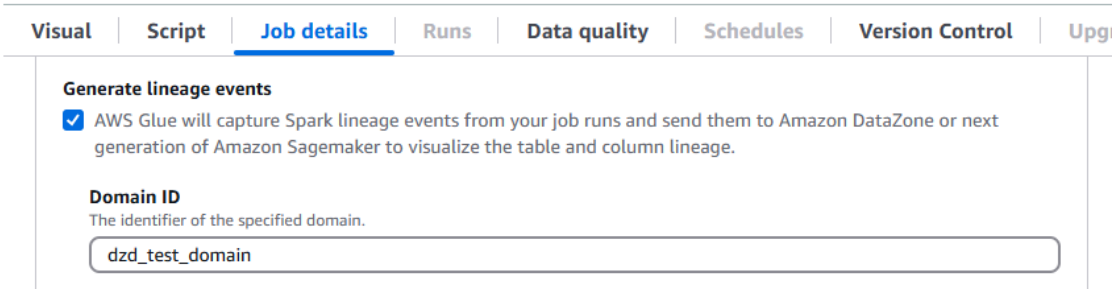
  Prenez note des considérations ou limites suivantes pour FGAC natif Spark :

  - Actuellement, les écritures de données ne sont pas prises en charge.
  - Pour écrire dans Iceberg via `GlueContext` en utilisant Lake Formation, vous devez plutôt utiliser le contrôle d'accès IAM.

  Pour une liste complète des restrictions et des considérations relatives à l'utilisation de FGAC natif Spark, consultez the section called "Considérations".
- Support des subventions d'accès Amazon S3 en tant que solution de contrôle d'accès évolutive à vos données Amazon S3 depuis AWS Glue. Pour de plus amples informations, veuillez consulter Utilisation d'Amazon S3 Access Grants avec AWS Glue.
- Open Table Formats (OTF) mis à jour vers Hudi 0.15.0, Iceberg 1.7.1 et Delta Lake 3.3.0
- Support d'Amazon SageMaker Unified Studio.

- Amazon SageMaker Lakehouse et intégration de l'abstraction de données. Pour de plus amples informations, veuillez consulter Interrogation de catalogues de données de métastore à partir de l'ETL AWS Glue.
- Prise en charge pour installer des bibliothèques Python supplémentaires à l'aide de `requirements.txt`. Pour de plus amples informations, veuillez consulter Installation de bibliothèques Python supplémentaires dans AWS Glue 5.0 ou version ultérieure à l'aide de requirements.txt.
- AWS Glue La version 5.0 prend en charge le lignage des données sur Amazon DataZone. Vous pouvez configurer AWS Glue pour collecter automatiquement des informations de lignage lors de l'exécution des tâches Spark et envoyer les événements de lignage à visualiser sur Amazon. DataZone Pour plus d'informations, consultez la section Lignage des données sur Amazon DataZone.

  Pour le configurer sur la AWS Glue console, activez Generate lineage events et saisissez votre identifiant de DataZone domaine Amazon dans l'onglet Job details.



  Vous pouvez également fournir le paramètre de tâche suivant (indiquez votre ID de DataZone domaine) :

  - Clé : `--conf`
  - Valeur :

```
extraListeners=io.openlineage.spark.agent.OpenLineageSparkListener
—conf spark.openlineage.transport.type=amazon_datazone_api
-conf spark.openlineage.transport.domainId=<your-domain-ID>
```

- Mises à jour du connecteur et du pilote JDBC. Pour plus d'informations, consultez Annexe B : Mises à niveau du pilote JDBC et Annexe C : Mises à niveau des connecteurs.
- Mise à jour de Java de la version 8 vers la version 17.
- Stockage accru pour les `G.2X` employés AWS Glue `G.1X` et les employés, l'espace disque passant respectivement à 94 Go et 138 Go. De plus, de nouveaux types de `G.12X` travailleurs et des

outils optimisés pour la mémoire `R.1X` `R.8X` sont disponibles dans les AWS Glue versions 4.0 et ultérieures. `G.16X` `R.2X` `R.4X` Pour de plus amples informations, consultez Tâches.

- Support pour les tâches AWS SDK for Java, versions 2 AWS Glue à 5.0, peuvent utiliser les versions 1.12.569 ou 2.28.8 pour Java si la tâche prend en charge la version v2. Le AWS SDK pour Java 2.x est une réécriture majeure de la base de code de la version 1.x. Il repose sur Java 8+ et ajoute plusieurs fonctionnalités fréquemment demandées. Cela inclut la prise en charge d'E/S non bloquantes et la possibilité de brancher une implémentation HTTP différente au moment de l'exécution. Pour plus d'informations, notamment un Guide de migration du kit SDK pour la version 1 de Java vers la version 2, consultez le guide du Kit SDK AWS pour Java, version 2.

Évolutions

Prenez note des modifications majeures suivantes :

- Dans la AWS Glue version 5.0, lorsque vous utilisez le système de fichiers S3A et si `fs.s3a.endpoint` et `fs.s3a.endpoint.region` ne sont pas définis, la région par défaut utilisée par S3A est `us-east-2`. Cela peut entraîner des problèmes, tels que des erreurs de délai de chargement dans S3, en particulier pour les tâches VPC. Pour atténuer les problèmes causés par cette modification, définissez la configuration Spark `fs.s3a.endpoint.region` lorsque vous utilisez le système de fichiers S3A dans la version 5.0. AWS Glue
- Contrôle précis des accès (FGAC) de Lake Formation
  - AWS Glue La version 5.0 prend uniquement en charge le nouveau FGAC natif de Spark utilisant Spark. DataFrames Il ne prend pas en charge l'utilisation AWS Glue DynamicFrames du FGAC.
    - L'utilisation de FGAC dans la version 5.0 nécessite une migration depuis Spark AWS Glue DynamicFrames DataFrames
    - Si vous n'avez pas besoin du FGAC, il n'est pas nécessaire de migrer vers Spark DataFrame et les GlueContext fonctionnalités, telles que les signets de tâches et les prédicats push down, continueront de fonctionner.
  - Les tâches avec FGAC natif Spark nécessitent un minimum de quatre travailleurs : un pilote utilisateur, un pilote système, un exécuteur système et un exécuteur utilisateur de secours.
  - Pour plus d'informations, consultez la section Utilisation AWS Glue avec AWS Lake Formation pour un contrôle d'accès précis.
- Accès complet aux tables (FTA) Lake Formation
  - AWS Glue 5.0 prend en charge le FTA avec Spark native DataFrames (nouveau) et GlueContext DynamicFrames (ancien, avec des limitations)

  - FTA natif Spark
    - Si le script 4.0 l'utilise GlueContext, migrez vers Native Spark.
    - Cette fonctionnalité est limitée aux tables Hive et Iceberg.
    - Pour plus d'informations sur la configuration d'une tâche 5.0 pour utiliser FTA natif Spark, consultez
  - GlueContext DynamicFrame ALE
    - Aucune modification de code requise
    - Cette fonctionnalité est limitée aux tables non OTF. Elle ne fonctionnera pas avec Iceberg, Delta Lake et Hudi.
- Le lecteur CSV SIMD vectorisé n'est pas pris en charge.
- La journalisation continue dans le groupe de journaux de sortie n'est pas prise en charge. Utilisez plutôt un groupe de journaux `error`.
- Le AWS Glue Job Run Insights `job-insights-rule-driver` est devenu obsolète. Le flux de journaux `job-insights-rca-driver` se trouve désormais dans le groupe des journaux d'erreurs.
- Les custom/marketplace connecteurs basés sur Athena ne sont pas pris en charge.
- Les connecteurs Adobe Marketo Engage, Facebook Ads, Google Ads, Google Analytics 4, Google Sheets, Hubspot, Instagram Ads, Intercom, Jira Cloud, Oracle, Salesforce, Salesforce Marketing Cloud NetSuite, Salesforce Marketing Cloud Account Engagement, SAP, Slack, Snapchat Ads OData ServiceNow, Stripe, Zendesk et Zoho CRM ne sont pas pris en charge.
- Les propriétés log4j personnalisées ne sont pas prises en charge dans AWS Glue la version 5.0.

Améliorations majeures de Spark 3.3.0 à Spark 3.5.4

Notez les améliorations suivantes :

- Client Python pour Spark Connect (SPARK-39375).
- Mise en œuvre de la prise en charge des valeurs DEFAULT pour les colonnes dans les tables (SPARK-38334).
- Prise en charge de « Références d'alias de colonne latérale » (SPARK-27561).
- Renforcement de l'utilisation de SQLSTATE pour les classes d'erreur (SPARK-41994).
- Activation des jointures du filtre Bloom par défaut (SPARK-38841).

- Amélioration de la capacité de mise à l'échelle de l'interface utilisateur Spark et de la stabilité des pilotes pour les applications de grande envergure (SPARK-41053).
- Suivi de la progression asynchrone dans le streaming structuré (SPARK-39591).
- Traitement avec état dynamique arbitraire Python dans le streaming structuré (SPARK-40434).
- Améliorations de la couverture de l'API Pandas (SPARK-42882) et de la prise en charge des NumPy entrées dans (SPARK-39405). PySpark
- Fournissez un profileur de mémoire pour les fonctions PySpark définies par l'utilisateur (SPARK-40281).
- PyTorch Distributeur d'outils (SPARK-41589).
- Publication des artefacts SBOM (SPARK-41893).
- Support IPv6 uniquement pour l'environnement (SPARK-39457).
- Planificateur K8s personnalisé (Apache YuniKorn et Volcano) GA (SPARK-42802).
- Prise en charge de client Scala et Go dans Spark Connect (SPARK-42554) et (SPARK-43351).
- PyTorchSupport ML distribué basé sur Spark Connect (SPARK-42471).
- Prise en charge du streaming structuré pour Spark Connect dans Python et Scala (SPARK-42938).
- Prise en charge de l'API Pandas pour le client Python Spark Connect (SPARK-42497).
- Présentez Arrow Python UDFs (SPARK-40307).
- Prise en charge des fonctions de table définies par l'utilisateur dans Python (SPARK-43798).
- Migrez PySpark les erreurs vers les classes d'erreurs (SPARK-42986).
- PySpark cadre de test (SPARK-44042).
- Ajout du support pour Datasketches HllSketch (SPARK-16484).
- Amélioration de la fonction SQL intégrée (SPARK-41231).
- Clause IDENTIFIER (SPARK-43205).
- Ajout de fonctions SQL dans Scala, Python et R API (SPARK-43907).
- Ajout de la prise en charge des arguments nommés pour les fonctions SQL (SPARK-43922).
- Éviter de réexécuter une tâche inutile sur un exécuteur hors service en cas de migration des données réorganisées (SPARK-41469).
- ML distribué <> spark connect (SPARK-42471).
- DeepSpeed distributeur (SPARK-44264).

- Mise en œuvre du point de contrôle du journal des modifications pour le magasin d'état RocksDB (SPARK-43421).
- Introduction de la propagation des filigranes parmi les opérateurs (SPARK-42376).
- Présentez dropDuplicatesWithin Watermark (SPARK-42931).
- Améliorations apportées à la gestion de la mémoire du fournisseur de magasin d'état RocksDB (SPARK-43311).

Actions pour migrer vers la AWS Glue version 5.0

Pour les tâches existantes, modifiez la `Glue version` depuis la version précédente vers `Glue 5.0` dans la configuration de la tâche.

- Dans AWS Glue Studio, choisissez « `Glue 5.0 - Supports Spark 3.5.4, Scala 2, Python 3` in `Glue version` ».
- Dans l'API, choisissez **`5.0`** dans le paramètre `GlueVersion` de l'opération d'API `UpdateJob`.

Pour les nouvelles tâches, choisissez `Glue 5.0` lorsque vous créez une tâche.

- Dans la console, choisissez `Spark 3.5.4, Python 3 (Glue Version 5.0) or Spark 3.5.4, Scala 2 (Glue Version 5.0)` dans `Glue version`.
- Dans AWS Glue Studio, choisissez « `Glue 5.0 - Supports Spark 3.5.4, Scala 2, Python 3` in `Glue version` ».
- Dans l'API, choisissez **`5.0`** dans le paramètre `GlueVersion` de l'opération d'API `CreateJob`.

Pour consulter les journaux d'événements Spark de la AWS Glue version 5.0 à partir de la version AWS Glue 2.0 ou d'une version antérieure, lancez un serveur d'historique Spark mis à niveau pour la AWS Glue version 5.0 à l'aide CloudFormation de Docker.

Liste de contrôle de migration

Consultez cette liste de contrôle pour la migration :

- Mises à jour de Java 17
- [Scala] Mettre à niveau les appels du AWS SDK de la v1 à la v2
- Migration de Python 3.10 vers Python 3.11
- [Python] Mise à jour des références boto de la version 1.26 vers la version 1.34

AWS Glue Caractéristiques de la version 5.0

Cette section décrit les AWS Glue fonctionnalités de manière plus détaillée.

Interrogation de catalogues de données de métastore à partir de l'ETL AWS Glue

Vous pouvez enregistrer votre AWS Glue travail pour accéder au AWS Glue Data Catalog, qui met les tables et autres ressources de métastore à la disposition de clients disparates. Le catalogue de données prend en charge une hiérarchie de catalogues multiples, qui unifie toutes vos données dans les lacs de données Amazon S3. Il fournit également une API de métastore Hive et une API Apache Iceberg open source pour accéder aux données. Ces fonctionnalités sont disponibles pour AWS Glue d'autres services axés sur les données tels qu'Amazon EMR, Amazon Athena et Amazon Redshift.

Lorsque vous créez des ressources dans le catalogue de données, vous pouvez y accéder à partir de n'importe quel moteur SQL prenant en charge l'API REST Apache Iceberg. AWS Lake Formation gère les autorisations. Après la configuration, vous pouvez tirer parti des capacités AWS Glue de l'application pour interroger des données disparates en interrogeant ces ressources de métastore avec des applications familières. Il s’agit notamment d’Apache Spark et de Trino.

Organisation des ressources de métadonnées

Les données sont organisées selon une hiérarchie logique de catalogues, de bases de données et de tables, à l' AWS Glue Data Catalog aide des éléments suivants :

- Catalogue : conteneur logique qui contient des objets provenant d’un entrepôt de données, tels que des schémas ou des tables.
- Base de données : organise les objets de données tels que les tables et les vues dans un catalogue.
- Tables et vues : objets de données d’une base de données qui fournissent une couche d’abstraction avec un schéma compréhensible. Ils facilitent l’accès aux données sous-jacentes, qui peuvent être dans différents formats et à différents emplacements.

Migration de la AWS Glue version 4.0 vers AWS Glue la version 5.0

Tous les paramètres de travail existants et les principales fonctionnalités qui existent dans la AWS Glue version 4.0 existeront dans la AWS Glue version 5.0, à l'exception des transformations liées à l'apprentissage automatique.

Les nouveaux paramètres suivants ont été ajoutés :

- `--enable-lakeformation-fine-grained-access`: Active la fonctionnalité de contrôle d'accès détaillé (FGAC) dans les tables de AWS Lake Formation.

Reportez-vous à la documentation sur la migration de Spark :

- Guide de migration : Spark Core
- Guide de migration : SQL, ensembles de données et DataFrame
- Guide de migration : streaming structuré
- Upgrading PySpark

Migration de la AWS Glue version 3.0 à AWS Glue la version 5.0

> **Note**
>
> Pour les étapes de migration liées à la AWS Glue version 4.0, voirMigration de la AWS Glue version 3.0 à AWS Glue la version 4.0.

Tous les paramètres de travail existants et les fonctionnalités principales qui existent dans la AWS Glue version 3.0 existeront dans la AWS Glue version 5.0, à l'exception des transformations de machine learning.

Migration de la AWS Glue version 2.0 vers AWS Glue la version 5.0

> **Note**
>
> Pour les étapes de migration liées à la version AWS Glue 4.0 et la liste des différences de migration entre les AWS Glue versions 3.0 et 4.0, voirMigration de la AWS Glue version 3.0 à AWS Glue la version 4.0.

Notez également les différences de migration suivantes entre AWS Glue les versions 3.0 et 2.0 :

- Tous les paramètres de travail existants et les principales fonctionnalités qui existent dans la AWS Glue version 2.0 existeront dans la AWS Glue version 5.0, à l'exception des transformations de machine learning.

- Plusieurs modifications Spark peuvent à elles seules nécessiter une révision de vos scripts pour s'assurer que les fonctions supprimées ne sont pas référencées. Par exemple, Spark 3.1.1 et versions ultérieures n'activent pas Scala-Untyped, UDFs mais Spark 2.4 le permet.
- Python 2.7 n'est pas pris en charge.
- Tout fichier jar supplémentaire fourni dans les tâches AWS Glue 2.0 existantes peut entraîner des dépendances conflictuelles, car plusieurs dépendances ont été mises à niveau. Vous pouvez éviter les conflits de chemin de classe avec le paramètre de tâche `--user-jars-first`.
- Modifications du comportement des fichiers loading/saving de from/to parquet horodatés. Pour plus de détails, veuillez consulter Mise à niveau de Spark SQL 3.0 vers 3.1.
- Parallélisme différent des tâches Spark pour driver/executor la configuration. Vous pouvez ajuster le parallélisme des tâches en transmettant l'argument de la tâche `--executor-cores`.

Changements de comportement de journalisation dans la AWS Glue version 5.0

Les modifications apportées au comportement de journalisation dans la AWS Glue version 5.0 sont les suivantes. Pour plus d'informations, consultez la section Journalisation des AWS Glue tâches.

- Tous les journaux (journaux du système, journaux de démon Spark, journaux des utilisateurs et journaux Glue Logger) sont désormais écrits dans le groupe de journaux `/aws-glue/jobs/error` par défaut.
- Le groupe de journaux `/aws-glue/jobs/logs-v2` utilisé pour la journalisation continue dans les versions précédentes n'est plus utilisé.
- Vous ne pouvez plus renommer ou personnaliser les noms des groupes de journaux ou des flux de journaux à l'aide des arguments de journalisation continue supprimés. Consultez plutôt les nouveaux arguments de tâche dans la AWS Glue version 5.0.

Deux nouveaux arguments de travail sont introduits dans la AWS Glue version 5.0

- `--custom-logGroup-prefix` : vous permet de spécifier un préfixe personnalisé pour les groupes de journaux `/aws-glue/jobs/error` et `/aws-glue/jobs/output`.
- `--custom-logStream-prefix` : vous permet de spécifier un préfixe personnalisé pour les noms des flux de journaux au sein des groupes de journaux.

  Les règles de validation et les restrictions relatives aux préfixes personnalisés sont les suivantes :

  - Le nom complet du flux de journaux doit comporter entre 1 et 512 caractères.

- Le préfixe personnalisé pour les noms des flux de journaux est limité à 400 caractères.
- Les caractères autorisés dans les préfixes incluent les caractères alphanumériques, les traits de soulignement (« _ »), les traits d'union (« - ») et les barres obliques (« / »).

Arguments de journalisation continue obsolètes dans la version 5.0 AWS Glue

Les arguments de tâche suivants pour la journalisation continue ont été déconseillés dans la version 5.0 AWS Glue

- `--enable-continuous-cloudwatch-log`
- `--continuous-log-logGroup`
- `--continuous-log-logStreamPrefix`
- `--continuous-log-conversionPattern`
- `--enable-continuous-log-filter`

Migration du connecteur et du pilote JDBC pour la version 5.0 AWS Glue

Pour en savoir plus sur les versions des connecteurs JDBC et de lac de données qui ont été mises à niveau, consultez :

- Annexe B : Mises à niveau du pilote JDBC
- Annexe C : Mises à niveau des connecteurs
- Annexe D : Mises à niveau du format de table ouverte

Les modifications suivantes s'appliquent aux versions du connecteur ou du pilote identifiées dans les annexes de Glue 5.0.

Amazon Redshift

Notez les modifications suivantes :

- Ajoute la prise en charge des noms de table en trois parties pour permettre au connecteur d'interroger les tables de partage de données Redshift.
- Corrige le mappage de Spark `ShortType` pour utiliser Redshift `SMALLINT` plutôt que `INTEGER`, afin de mieux correspondre à la taille de données attendue.

- Ajout de la prise en charge des noms de cluster personnalisés (CNAME) pour Amazon Redshift sans serveur.

Apache Hudi

Notez les modifications suivantes :

- Prise en charge des indices au niveau des enregistrements.
- Prise en charge de la génération automatique de clés d'enregistrement. Vous n'avez plus besoin de spécifier le champ de clé d'enregistrement.

Apache Iceberg

Notez les modifications suivantes :

- Support d'un contrôle d'accès précis avec. AWS Lake Formation
- Prise en charge du branchement et du balisage, qui sont des références nommées à des instantanés dotés de leur propre cycle de vie indépendant.
- Ajout d'une procédure d'affichage du journal des modifications qui génère une vue contenant les modifications apportées à une table au cours d'une période spécifiée ou entre des instantanés spécifiques.

Delta Lake

Notez les modifications suivantes :

- Support du format Delta Universal (UniForm) qui permet un accès fluide via Apache Iceberg et Apache Hudi.
- Support des vecteurs de suppression qui implémentent un Merge-on-Read paradigme.

AzureCosmos

Notez les modifications suivantes :

- Ajout de la prise en charge des clés de partition hiérarchiques.
- Ajout d'une option permettant d'utiliser un schéma personnalisé avec StringType (json brut) pour une propriété imbriquée.

- Ajout d'une option `spark.cosmos.auth.aad.clientCertPemBase64` de configuration pour permettre d'utiliser l'authentification SPN (ServicePrincipal nom) avec un certificat au lieu du secret client.

Pour plus d'informations, consultez Azure Cosmos DB Spark connector change log.

Microsoft SQL Server

Notez les modifications suivantes :

- Le chiffrement TLS est activé par défaut.
- Lorsque encrypt = false, mais que le serveur exige le chiffrement, le certificat est validé en fonction du paramètre de connexion `trustServerCertificate`.
- `aadSecurePrincipalId` et `aadSecurePrincipalSecret` sont obsolètes.
- API `getAADSecretPrincipalId` supprimée.
- Résolution CNAME ajoutée lorsque le domaine est spécifié.

MongoDB

Notez les modifications suivantes :

- Prise en charge du mode microlots avec Spark Structured Streaming.
- Prise en charge des types de données BSON.
- Ajout de la prise en charge de la lecture de plusieurs collections lors de l'utilisation des modes microlots ou de streaming continu.
  - Si le nom d'une collection utilisée dans votre option de configuration `collection` contient une virgule, le connecteur Spark la traite comme deux collections différentes. Pour éviter ceci, échappez la virgule en la faisant précéder d'une barre oblique inverse (\).
  - Si le nom d'une collection utilisée dans votre option de configuration `collection` est « * », le connecteur Spark l'interprète comme une spécification pour analyser toutes les collections. Pour éviter ceci, échappez l'astérisque en le faisant précéder d'une barre oblique inverse (\).
  - Si le nom d'une collection utilisée dans votre option de configuration `collection` contient une barre oblique inverse (\), le connecteur Spark traite la barre oblique inverse comme un caractère d'échappement, ce qui peut modifier la façon dont il interprète la valeur. Pour éviter cela, vous devez échapper la barre oblique inverse en la faisant précéder d'une autre barre oblique inverse.

Pour plus d'informations, consultez MongoDB connector for Spark release notes.

Snowflake

Notez les modifications suivantes :

- Ajout d'un nouveau paramètre `trim_space` que vous pouvez utiliser pour réduire automatiquement les valeurs des colonnes `StringType` lors de l'enregistrement dans une table Snowflake. Valeur par défaut : `false`.
- Le paramètre `abort_detached_query` a été désactivé par défaut au niveau de la session.
- Suppression de l'exigence du paramètre `SFUSER` lors de l'utilisation d'OAUTH.
- Suppression de la fonctionnalité Advanced Query Pushdown. Des alternatives à cette fonctionnalité sont disponibles. Par exemple, au lieu de charger des données à partir de tables Snowflake, les utilisateurs peuvent charger directement des données à partir de requêtes SQL Snowflake.

Pour plus d'informations, consultez Snowflake Connector for Spark release notes.

Annexe A : Mises à niveau notables des dépendances

Voici les mises à niveau des dépendances :

Dépendance	Version en AWS Glue 5.0	Version en AWS Glue 4.0	Version en AWS Glue 3.0	Version en AWS Glue 2.0	Version en AWS Glue 1.0
Java	17	8	8	8	8
Spark	3.5.4	3.3.0-amzn-1	3.1.1-amzn-0	2.4.3	2.4.3
Hadoop	3.4.1	3.3.3-amzn-0	3.2.1-amzn-3	2.8.5-amzn-5	2.8.5-amzn-1
Scala	2,1,18	2,12	2,12	2.11	2.11
Jackson	2.15.2	2,12	2,12	2.11	2.11
Hive	2.3.9-amzn-4.	2.3.9-amzn-2	2.3.7-amzn-4	1.2	1.2
EMRFS	2,69,0	2,54,0	2,46,0	2.38.0	2.30.0
Json4s	3.7.0-M11	3.7.0-M11	3.6.6	3.5.x	3.5.x

Dépendance	Version en AWS Glue 5.0	Version en AWS Glue 4.0	Version en AWS Glue 3.0	Version en AWS Glue 2.0	Version en AWS Glue 1.0
Flèche	12,0.1	7.0.0	2.0.0	0.10.0	0.10.0
AWS Glue Client de catalogue de données	4.5.0	3.7.0	3.0.0	1.10.0	N/A
AWS SDK pour Java	2,29,52	1.12	1.12		
Python	3,11	3,10	3.7	2.7 et 3.6	2.7 et 3.6
Boto	1,34,131	1,26	1,18	1.12	N/A
Connecteur EMR DynamoDB	5.6.0	4,16,0			

Annexe B : Mises à niveau du pilote JDBC

Voici les mises à niveau du pilote JDBC :

Pilote	Version du pilote JDBC en 5.0 AWS Glue	Version du pilote JDBC en 4.0 AWS Glue	Version du pilote JDBC en 3.0 AWS Glue	Version du pilote JDBC dans les versions précédentes AWS Glue
MySQL	8,0,33	8.0.23	8.0.23	5.1
Microsoft SQL Serv	10.2.0	9.4.0	7.0.0	6.1.0
Oracle Databases	23,3.0,23,09	21,7	21,1	11.2

Pilote	Version du pilote JDBC en 5.0 AWS Glue	Version du pilote JDBC en 4.0 AWS Glue	Version du pilote JDBC en 3.0 AWS Glue	Version du pilote JDBC dans les versions précédentes AWS Glue
PostgreSQL	42.7.3	42,3.6	42,2,18	42,10
Amazon Redshift	redshift-jdbc42-2.1.0.29	redshift-jdbc42-2.1.0.16	redshift-jdbc41-1.2.12.1017	redshift-jdbc41-1.2.12.1017
SAP Hana	2,20,17	2,17,12		
Teradata	20,00.00.33	20,00.00.06		

Annexe C : Mises à niveau des connecteurs

Les mises à niveau des connecteurs sont les suivantes :

Pilote	Version du connecteur en AWS Glue 5.0	Version du connecteur en AWS Glue 4.0	Version du connecteur en AWS Glue 3.0
Connecteur EMR DynamoDB	5.6.0	4,16,0	
Amazon Redshift	6.4.0	6.1.3	
OpenSearch	1.2.0	1.0.1	
MongoDB	10.3.0	10,0.4	3.0.0
Snowflake	3.0.0	2.12.0	
Google BigQuery	0,32,2	0,32,2	
AzureCosmos	4,33,0	4,22,0	
AzureSQL	1.3.0	1.3.0	

Pilote	Version du connecteur en AWS Glue 5.0	Version du connecteur en AWS Glue 4.0	Version du connecteur en AWS Glue 3.0
Vertica	3.3.5	3.3.5	

Annexe D : Mises à niveau du format de table ouverte

Les mises à niveau du format de table ouverte sont les suivantes :

OTF	Version du connecteur en AWS Glue 5.0	Version du connecteur en AWS Glue 4.0	Version du connecteur en AWS Glue 3.0
Hudi	0,15,0	0.12.1	0,1,1
Delta Lake	3.3.0	2.1.0	1.0.0
Iceberg	1.7.1	1.0.0	0.13.1

Migration AWS Glue des tâches Spark vers la AWS Glue version 4.0

Cette rubrique décrit les modifications apportées entre AWS Glue les versions 0.9, 1.0, 2.0 et 3.0 pour vous permettre de migrer vos applications Spark et vos tâches ETL vers la AWS Glue version 4.0. Il décrit également les fonctionnalités de la AWS Glue version 4.0 et les avantages de son utilisation.

Pour utiliser cette fonctionnalité avec vos tâches AWS Glue ETL, **`4.0`** choisissez-la `Glue version` lors de la création de vos tâches.

Rubriques

- Nouvelles fonctionnalités prises en charge
- Actions pour migrer vers la AWS Glue version 4.0
- Liste de contrôle de migration
- Migration de la AWS Glue version 3.0 à AWS Glue la version 4.0
- Migration de la AWS Glue version 2.0 vers AWS Glue la version 4.0
- Migration de la AWS Glue version 1.0 à AWS Glue la version 4.0
- Migration de la version AWS Glue 0.9 à AWS Glue la version 4.0

- Migration du connecteur et du pilote JDBC pour la version 4.0 AWS Glue
- Annexe A : Mises à niveau notables des dépendances
- Annexe B : Mises à niveau du pilote JDBC
- Annexe C : Mises à niveau des connecteurs

Nouvelles fonctionnalités prises en charge

Cette section décrit les nouvelles fonctionnalités et les avantages de la AWS Glue version 4.0.

- Elle est basée sur Apache Spark 3.3.0, mais propose des optimisations dans AWS Glue et Amazon EMR, telles que les exécutions adaptatives de requêtes, les lecteurs vectorisés, les remaniements optimisées et la fusion de partitions.
- Pilotes JDBC mis à niveau pour toutes les sources AWS Glue natives, notamment MySQL, Microsoft SQL Server, Oracle, PostgreSQL, MongoDB, ainsi que les bibliothèques et dépendances Spark mises à niveau introduites par Spark 3.3.0.
- Utilisation d'un nouveau connecteur Amazon Redshift et d'un nouveau pilote JDBC.
- Accès optimisé à Amazon S3 avec EMRFS (système de fichiers EMR) mis à niveau et validateurs de sortie optimisés Amazon S3 activés par défaut.
- Accès optimisé au catalogue de données avec des index de partition, le déploiement des prédicats, une liste de partitions et un client de métastore Hive mis à niveau.
- Intégration à Lake Formation pour les tables de catalogue régies avec filtrage au niveau des cellules et transactions de lac de données.
- Réduction de la latence au démarrage afin d'améliorer les temps globaux de réalisation de tâche et l'interactivité.
- Les tâches Spark sont facturées par incréments d'une seconde avec une durée de facturation minimale 10 fois inférieure, allant d'un minimum de 10 minutes à un minimum de 1 minute.
- Prise en charge native des infrastructures de lacs de données ouverts avec Apache Hudi, Delta Lake et Apache Iceberg.
- Prise en charge native du plug-in Cloud Shuffle Storage basé sur Amazon S3 (un plug-in Apache Spark) permettant d'utiliser Amazon S3 pour la réorganisation et la capacité de stockage élastique.

Améliorations majeures de Spark 3.1.1 à Spark 3.3.0

Notez les améliorations suivantes :

- Filtrage d'exécution au niveau des lignes (SPARK-32268).
- Améliorations ANSI (SPARK-38860).
- Améliorations des messages d'erreur (SPARK-38781).
- Prise en charge des types complexes pour le lecteur vectorisé Parquet (SPARK-34863).
- Prise en charge des métadonnées de fichiers masqués pour Spark SQL (SPARK-37273).
- Fournissez un profileur pour Python/Pandas UDFs (SPARK-37443).
- Présentez Trigger. AvailableNow pour exécuter des requêtes de streaming telles que Trigger.Once en plusieurs lots (SPARK-36533).
- Fonctionnalités pushdown de Datasource V2 plus complètes (SPARK-38788).
- Migration de log4j 1 vers log4j 2 (SPARK-37814).

Autres changements notables

Notez les modifications suivantes :

- Évolutions
  - Supprimez les références au support de Python 3.6 dans les documents et Python/docs (SPARK-36977).
  - Suppression du tuple hack nommé en remplaçant le pickle intégré par cloudpickle (SPARK-32079).
  - Passage de la version minimale de Pandas à 1.0.5 (SPARK-37465).

Actions pour migrer vers la AWS Glue version 4.0

Pour les tâches existantes, modifiez la `Glue version` depuis la version précédente vers `Glue 4.0` dans la configuration de la tâche.

- Dans AWS Glue Studio, choisissez « `Glue 4.0 - Supports Spark 3.3, Scala 2, Python 3` in `Glue version` ».
- Dans l'API, choisissez **`4.0`** dans le paramètre `GlueVersion` de l'opération d'API `UpdateJob`.

Pour les nouvelles tâches, choisissez `Glue 4.0` lorsque vous créez une tâche.

- Dans la console, choisissez `Spark 3.3, Python 3 (Glue Version 4.0) or Spark 3.3, Scala 2 (Glue Version 3.0)` dans `Glue version`.

- Dans AWS Glue Studio, choisissez « `Glue 4.0 - Supports Spark 3.3, Scala 2, Python 3` in `Glue version` ».
- Dans l'API, choisissez **`4.0`** dans le paramètre `GlueVersion` de l'opération d'API `CreateJob`.

Pour consulter les journaux d'événements Spark AWS Glue 4.0 issus de la version AWS Glue 2.0 ou d'une version antérieure, lancez un serveur d'historique Spark mis à niveau pour AWS Glue 4.0 à l'aide CloudFormation de Docker.

Liste de contrôle de migration

- Les bibliothèques Python externes de votre tâche dépendent-elles de Python 2.7/3.6 ?
  - Mettez à jour les bibliothèques dépendantes de Python 2.7/3.6 vers Python 3.10, car Spark 3.3.0 a supprimé la prise en charge de Python 2.7 et 3.6.

Migration de la AWS Glue version 3.0 à AWS Glue la version 4.0

Notez les modifications suivantes lors de la migration :

- Tous les paramètres de travail existants et les fonctionnalités principales qui existent dans la AWS Glue version 3.0 existeront dans la AWS Glue version 4.0.
- AWS Glue La version 3.0 utilise Spark 3.1.1 optimisé pour Amazon EMR, et la version 4.0 AWS Glue utilise Spark 3.3.0 optimisé pour Amazon EMR.

  Plusieurs modifications Spark pourraient à elles seules nécessiter une révision de vos scripts pour s'assurer que les fonctions supprimées ne sont pas référencées.
- AWS Glue La version 4.0 inclut également une mise à jour d'EMRFS et de Hadoop. Pour en savoir plus sur cette version spécifique, consultez Annexe A : Mises à niveau notables des dépendances.
- Le AWS SDK fourni dans les tâches ETL est désormais mis à niveau de la version 1.11 à la version 1.12.
- Toutes les tâches Python utiliseront la version 3.10 de Python. Auparavant, Python 3.7 était utilisé dans la AWS Glue version 3.0.

  Par conséquent, certains pymodules proposés AWS Glue sont mis out-of-the-box à niveau.
- Log4j a été mis à niveau vers Log4j2.
  - Pour plus d'informations sur le chemin de migration Log4j2, consultez la documentation de Log4j.

  - Vous devez plutôt renommer tout fichier log4j.properties personnalisé comme fichier log4j2.properties, avec les propriétés log4j2 appropriées.
- Pour la migration de certains connecteurs, consultez Migration du connecteur et du pilote JDBC pour la version 4.0 AWS Glue.
- Le SDK de AWS chiffrement est mis à niveau de la version 1.x à la version 2.x. AWS Glue les tâches utilisant des configurations AWS Glue de sécurité et les tâches dépendantes de la dépendance du SDK de AWS chiffrement fournie lors de l'exécution sont affectées. Consultez les instructions relatives à la migration des AWS Glue tâches.

  Vous pouvez mettre à niveau en toute sécurité une tâche AWS Glue 2.0/3.0 vers une tâche AWS Glue 4.0, car la version AWS Glue 2.0/3.0 contient déjà la version pont du SDK de AWS chiffrement.

Reportez-vous à la documentation sur la migration de Spark :

- Upgrading from Spark SQL 3.1 to 3.2
- Upgrading from Spark SQL 3.2 to 3.3

Migration de la AWS Glue version 2.0 vers AWS Glue la version 4.0

Notez les modifications suivantes lors de la migration :

> **Note**
>
> Pour les étapes de migration liées à la AWS Glue version 3.0, voirMigration de la AWS Glue version 3.0 à AWS Glue la version 4.0.

- Tous les paramètres de travail existants et les principales fonctionnalités qui existent dans la AWS Glue version 2.0 existeront dans la AWS Glue version 4.0.
- Le validateur EMRFS S3 optimisé pour écrire des données Parquet dans Amazon S3 est activé par défaut depuis la version 3.0. AWS Glue Cependant, vous pouvez toujours le désactiver en définissant `--enable-s3-parquet-optimized-committer` sur `false`.
- AWS Glue La version 2.0 utilise le logiciel libre Spark 2.4 et la AWS Glue version 4.0 utilise Spark 3.3.0 optimisé pour Amazon EMR.

  - Plusieurs modifications Spark peuvent à elles seules nécessiter une révision de vos scripts pour s'assurer que les fonctions supprimées ne sont pas référencées.
  - Par exemple, Spark 3.3.0 n'active pas Scala-Untyped UDFs, mais Spark 2.4 le permet.
- Le AWS SDK fourni dans les tâches ETL est désormais mis à niveau de la version 1.11 à la version 1.12.
- AWS Glue La version 4.0 inclut également une mise à jour d'EMRFS, des pilotes JDBC mis à jour et l'inclusion d'optimisations supplémentaires sur Spark lui-même fournies par. AWS Glue
- Scala est mis à jour depuis la version 2.11 vers la version 2.12, et Scala 2.12 n'est pas rétrocompatible avec Scala 2.11.
- Python 3.10 est la version utilisée par défaut pour les scripts Python, comme AWS Glue 2.0 utilisait uniquement Python 3.7 et 2.7.
  - Python 2.7 n'est pas pris en charge avec Spark 3.3.0. Toute tâche demandant Python 2 dans la configuration de la tâche échouera avec un IllegalArgumentException.
  - Un nouveau mécanisme d'installation de modules Python supplémentaires est disponible depuis la AWS Glue version 2.0.
- Plusieurs mises à jour de dépendance, mises en évidence dans Annexe A : Mises à niveau notables des dépendances.
- Tout fichier JAR supplémentaire fourni dans les tâches AWS Glue 2.0 existantes peut entraîner des dépendances conflictuelles car plusieurs dépendances ont été mises à niveau dans la version 4.0 par rapport à la version 2.0. Vous pouvez éviter les conflits de chemin de classe dans la AWS Glue version 4.0 avec le paramètre `--user-jars-first` AWS Glue job.
- AWS Glue La version 4.0 utilise Spark 3.3. À partir de Spark 3.1, il y a eu un changement dans le comportement loading/saving des fichiers from/to parquet horodatés. Pour plus de détails, veuillez consulter Mise à niveau de Spark SQL 3.0 vers 3.1.

  Nous vous recommandons de définir les paramètres suivants lorsque vous parcourez des reading/writing données contenant des colonnes d'horodatage. La définition de ces paramètres peut résoudre le problème d'incompatibilité du calendrier qui se produit lors de la mise à niveau de Spark 2 vers Spark 3, tant pour le AWS Glue Dynamic Frame que pour le Spark Data Frame. Utilisez l'option CORRECTED pour lire la valeur datetime telle quelle, et l'option LEGACY pour rebaser les valeurs datetime par rapport à la différence de calendrier pendant la lecture.

```
- Key: --conf
```

```
- Value: spark.sql.legacy.parquet.int96RebaseModeInRead=[CORRECTED|LEGACY] --
conf spark.sql.legacy.parquet.int96RebaseModeInWrite=[CORRECTED|LEGACY] --conf
 spark.sql.legacy.parquet.datetimeRebaseModeInRead=[CORRECTED|LEGACY]
```

- Pour la migration de certains connecteurs, consultez Migration du connecteur et du pilote JDBC pour la version 4.0 AWS Glue.
- Le SDK de AWS chiffrement est mis à niveau de la version 1.x à la version 2.x. AWS Glue les tâches utilisant des configurations AWS Glue de sécurité et les tâches dépendantes de la dépendance du SDK de AWS chiffrement fournie lors de l'exécution sont affectées. Consultez les instructions suivantes pour la migration des AWS Glue tâches :
  - Vous pouvez mettre à niveau une tâche AWS Glue 2.0 vers une tâche AWS Glue 4.0 en toute sécurité, car la version AWS Glue 2.0 contient déjà la version pont du SDK de AWS chiffrement.

Reportez-vous à la documentation sur la migration de Spark :

- Upgrading from Spark SQL 2.4 to 3.0
- Upgrading from Spark SQL 3.1 to 3.2
- Upgrading from Spark SQL 3.2 to 3.3
- Changes in Datetime behavior to be expected since Spark 3.0.

Migration de la AWS Glue version 1.0 à AWS Glue la version 4.0

Notez les modifications suivantes lors de la migration :

- AWS Glue La version 1.0 utilise le logiciel libre Spark 2.4 et la AWS Glue version 4.0 utilise Spark 3.3.0 optimisé pour Amazon EMR.
  - Plusieurs modifications Spark peuvent à elles seules nécessiter une révision de vos scripts pour s'assurer que les fonctions supprimées ne sont pas référencées.
  - Par exemple, Spark 3.3.0 n'active pas Scala-Untyped UDFs, mais Spark 2.4 le permet.
- Toutes les tâches de la AWS Glue version 4.0 seront exécutées avec des temps de démarrage nettement améliorés. Les tâches Spark seront facturées par incréments d'une seconde avec une durée de facturation minimale 10 fois plus faible, puisque la latence de démarrage passera de 10 minutes maximum à 1 minute maximum.
- Le comportement de journalisation a changé de manière significative dans la AWS Glue version 4.0, la configuration minimale requise pour Spark 3.3.0 est Log4j2.
- Plusieurs mises à jour de dépendance, mises en évidence dans l'annexe.

- Scala est également mis à jour depuis la version 2.11 vers la version 2.12, et Scala 2.12 n'est pas rétrocompatible avec Scala 2.11.
- Python 3.10 est également la version utilisée par défaut pour les scripts Python, comme AWS Glue 0.9 utilisait uniquement Python 2.

  Python 2.7 n'est pas pris en charge avec Spark 3.3.0. Toute tâche demandant Python 2 dans la configuration de la tâche échouera avec un IllegalArgumentException.
- Un nouveau mécanisme d'installation de modules Python supplémentaires via pip est disponible depuis la AWS Glue version 2.0. Pour plus d'informations, consultez Installing additional Python modules with pip in AWS Glue 2.0+.
- AWS Glue La version 4.0 ne s'exécute pas sur Apache YARN, les paramètres YARN ne s'appliquent donc pas.
- AWS Glue La version 4.0 ne possède pas de système de fichiers distribué Hadoop (HDFS).
- Tout fichier JAR supplémentaire fourni dans les tâches AWS Glue 1.0 existantes peut entraîner des dépendances conflictuelles car plusieurs dépendances ont été mises à niveau dans la version 4.0 par rapport à la version 1.0. Pour éviter ce problème, nous activons la AWS Glue version 4.0 avec le paramètre `--user-jars-first` AWS Glue job par défaut.
- AWS Glue La version 4.0 prend en charge la mise à l'échelle automatique. Par conséquent, la ExecutorAllocationManager métrique sera disponible lorsque la mise à l'échelle automatique sera activée.
- Dans les tâches de la AWS Glue version 4.0, vous spécifiez le nombre de travailleurs et le type de travailleur, mais vous ne spécifiez pas de`maxCapacity`.
- AWS Glue La version 4.0 ne prend pas encore en charge les transformations du machine learning.
- Pour la migration de certains connecteurs, consultez Migration du connecteur et du pilote JDBC pour la version 4.0 AWS Glue.
- Le SDK de AWS chiffrement est mis à niveau de la version 1.x à la version 2.x. AWS Glue les tâches utilisant des configurations AWS Glue de sécurité et les tâches dépendantes de la dépendance du SDK de AWS chiffrement fournie lors de l'exécution sont affectées. Consultez ces instructions pour la migration des AWS Glue tâches.
  - Vous ne pouvez pas migrer AWS Glue directement une tâche 0.9/1.0 vers une tâche AWS Glue 4.0. En effet, lors de la mise à niveau directe vers la version 2.x ou ultérieure et de l'activation immédiate de toutes les nouvelles fonctionnalités, le SDK de AWS chiffrement ne sera pas en mesure de déchiffrer le texte chiffré dans les versions antérieures du SDK de chiffrement. AWS

- Pour effectuer une mise à niveau en toute sécurité, nous vous recommandons d'abord de migrer vers une tâche AWS Glue 2.0/3.0 contenant la version pont du SDK AWS Encryption. Exécutez la tâche une fois pour utiliser la version pont du SDK de AWS chiffrement.
- Une fois terminé, vous pouvez migrer en toute sécurité la tâche AWS Glue 2.0/3.0 vers AWS Glue la version 4.0.

Reportez-vous à la documentation sur la migration de Spark :

- Upgrading from Spark SQL 2.4 to 3.0
- Upgrading from Spark SQL 3.0 to 3.1
- Upgrading from Spark SQL 3.1 to 3.2
- Upgrading from Spark SQL 3.2 to 3.3
- Changes in Datetime behavior to be expected since Spark 3.0.

Migration de la version AWS Glue 0.9 à AWS Glue la version 4.0

Notez les modifications suivantes lors de la migration :

- AWS Glue La version 0.9 utilise le logiciel libre Spark 2.2.1 et la AWS Glue version 4.0 utilise Spark 3.3.0 optimisé pour Amazon EMR.
  - Plusieurs modifications Spark pourraient à elles seules nécessiter une révision de vos scripts pour s'assurer que les fonctions supprimées ne sont pas référencées.
  - Par exemple, Spark 3.3.0 n'active pas Scala-Untyped UDFs, mais Spark 2.2 le permet.
- Toutes les tâches de la AWS Glue version 4.0 seront exécutées avec des temps de démarrage nettement améliorés. Les tâches Spark seront facturées par incréments d'une seconde avec une durée de facturation minimale 10 fois plus faible, car la latence de démarrage passera de 10 minutes maximum à 1 minute maximum.
- Le comportement de journalisation a changé de manière significative depuis la AWS Glue version 4.0, Spark 3.3.0 requiert au minimum Log4j2, comme indiqué ici (https://spark.apache.org/docs/latest/core-migration-guide.html# -32-to-33). upgrading-from-core
- Plusieurs mises à jour de dépendance, mises en évidence dans l'annexe.
- Scala est également mis à jour depuis la version 2.11 vers la version 2.12, et Scala 2.12 n'est pas rétrocompatible avec Scala 2.11.

- Python 3.10 est également la version utilisée par défaut pour les scripts Python, comme AWS Glue 0.9 utilisait uniquement Python 2.
  - Python 2.7 n'est pas pris en charge avec Spark 3.3.0. Toute tâche demandant Python 2 dans la configuration de la tâche échouera avec un IllegalArgumentException.
  - Un nouveau mécanisme d'installation de modules Python supplémentaires via pip est disponible.
- AWS Glue La version 4.0 ne s'exécute pas sur Apache YARN, les paramètres YARN ne s'appliquent donc pas.
- AWS Glue La version 4.0 ne possède pas de système de fichiers distribué Hadoop (HDFS).
- Tout fichier JAR supplémentaire fourni dans les tâches AWS Glue 0.9 existantes peut entraîner des dépendances conflictuelles car plusieurs dépendances ont été mises à niveau dans la version 3.0 par rapport à la version 0.9. Vous pouvez éviter les conflits de chemin de classe dans la AWS Glue version 3.0 avec le paramètre `--user-jars-first` AWS Glue job.
- AWS Glue La version 4.0 prend en charge la mise à l'échelle automatique. Par conséquent, la ExecutorAllocationManager métrique sera disponible lorsque la mise à l'échelle automatique sera activée.
- Dans les tâches de la AWS Glue version 4.0, vous spécifiez le nombre de travailleurs et le type de travailleur, mais vous ne spécifiez pas de`maxCapacity`.
- AWS Glue La version 4.0 ne prend pas encore en charge les transformations du machine learning.
- Pour la migration de certains connecteurs, consultez Migration du connecteur et du pilote JDBC pour la version 4.0 AWS Glue.
- Le SDK de AWS chiffrement est mis à niveau de la version 1.x à la version 2.x. AWS Glue les tâches utilisant des configurations AWS Glue de sécurité et les tâches dépendantes de la dépendance du SDK de AWS chiffrement fournie lors de l'exécution sont affectées. Consultez ces instructions pour la migration des AWS Glue tâches.
  - Vous ne pouvez pas migrer AWS Glue directement une tâche 0.9/1.0 vers une tâche AWS Glue 4.0. En effet, lors de la mise à niveau directe vers la version 2.x ou ultérieure et de l'activation immédiate de toutes les nouvelles fonctionnalités, le SDK de AWS chiffrement ne sera pas en mesure de déchiffrer le texte chiffré dans les versions antérieures du SDK de chiffrement. AWS
  - Pour effectuer une mise à niveau en toute sécurité, nous vous recommandons d'abord de migrer vers une tâche AWS Glue 2.0/3.0 contenant la version pont du SDK AWS Encryption. Exécutez la tâche une fois pour utiliser la version pont du SDK de AWS chiffrement.
  - Une fois terminé, vous pouvez migrer en toute sécurité la tâche AWS Glue 2.0/3.0 vers AWS Glue la version 4.0.

Reportez-vous à la documentation sur la migration de Spark :

- Upgrading from Spark SQL 2.2 to 2.3
- Upgrading from Spark SQL 2.3 to 2.4
- Upgrading from Spark SQL 2.4 to 3.0
- Upgrading from Spark SQL 3.0 to 3.1
- Upgrading from Spark SQL 3.1 to 3.2
- Upgrading from Spark SQL 3.2 to 3.3
- Changes in Datetime behavior to be expected since Spark 3.0.

Migration du connecteur et du pilote JDBC pour la version 4.0 AWS Glue

Pour en savoir plus sur les versions des connecteurs JDBC et de lac de données qui ont été mises à niveau, consultez :

- Annexe B : Mises à niveau du pilote JDBC
- Annexe C : Mises à niveau des connecteurs

Hudi

- Améliorations de la prise en charge de Spark SQL :
  - Cette commande `Call Procedure` permet de renforcer la prise en charge pour la mise à niveau, la rétrogradation, le démarrage, le nettoyage et la réparation. La syntaxe `Create/Drop/Show/Refresh Index` est autorisée dans Spark SQL.
  - Un écart de performance a été comblé entre les utilisations via Spark DataSource et Spark SQL. Dans le passé, les écritures Datasource étaient plus rapides qu'avec SQL.
  - Tous les générateurs de clés intégrés implémentent des opérations d'API spécifiques à Spark plus performantes.
  - Remplacement de la transformation UDF dans le cadre de l'`insert`opération en masse par des transformations RDD afin de réduire les coûts d'utilisation. SerDe
  - Spark SQL avec Hudi nécessite une `primaryKey` devant être spécifiée par `tblproperites` ou des options dans l'instruction SQL. Pour les opérations de mise à jour et de suppression, `preCombineField` est également obligatoire.

- Toute table Hudi créée avant la version 0.10.0 sans avoir besoin de recréer un `primaryKey` avec un champ `primaryKey` depuis la version 0.10.0.

PostgreSQL

- Plusieurs vulnérabilités (CVEs) ont été corrigées.
- Java 8 est pris en charge en mode natif.
- Si la tâche utilise des tableaux de tableaux, à l'exception des tableaux d'octets, ce scénario peut être traité sous forme de tableaux multidimensionnels.

MongoDB

- Le connecteur MongoDB actuel prend en charge Spark version 3.1 ou ultérieure et MongoDB version 4.0 ou ultérieure.
- En raison de la mise à niveau du connecteur, quelques noms de propriétés ont été modifiés. Par exemple, le nom de la propriété URI a été remplacé par `connection.uri`. Pour plus d'informations sur les options actuelles, consultez le blog MongoDB Spark Connector.
- L'utilisation de MongoDB 4.0 hébergé par Amazon DocumentDB présente certaines différences fonctionnelles. Pour plus d'nformations, consultez les rubriques suivantes :
  - Différences fonctionnelles : Amazon DocumentDB et MongoDB
  - MongoDB APIs, opérations et types de données pris en charge.
- L'option « partitionneur » est limitée à `ShardedPartitioner`, `PaginateIntoPartitionsPartitioner`, et `SinglePartitionPartitioner`. Elle ne peut pas utiliser les valeurs `SamplePartitioner` et `PaginateBySizePartitioner` par défaut pour Amazon DocumentDB, car l'opérateur de l'étape ne prend pas en charge l'API MongoDB. Pour plus d'informations, consultez MongoDB APIs, Opérations et Types de données pris en charge.

Delta Lake

- Désormais, Delta Lake prend en charge les requêtes Time Travel dans SQL pour interroger facilement les données plus anciennes. Avec cette mise à jour, le voyage dans le temps est désormais disponible à la fois dans Spark SQL et via l' DataFrame API. La prise en charge de la version actuelle de TIMESTAMP dans SQL a été ajoutée.

- Spark 3.3 présente Trigger. AvailableNowpour exécuter des requêtes de streaming comme équivalent à `Trigger.Once` des requêtes par lots. Cette prise en charge est également disponible en cas d'utilisation des tables Delta comme source de streaming.
- Prise en charge de la fonction SHOW COLUMNS pour renvoyer la liste des colonnes d'une table.
- Support de DESCRIBE DETAIL dans les DeltaTable API Scala et Python. Il récupère des informations détaillées sur une table Delta à l'aide de l' DeltaTable API ou de Spark SQL.
- Prise en charge du renvoi de métriques d'opération à partir des commandes SQL Delete, Merge et Update. Auparavant, ces commandes SQL renvoyaient une valeur vide DataFrame, maintenant elles renvoient un DataFrame contenant des mesures utiles concernant l'opération effectuée.
- Pour optimiser les améliorations de performances :
  - Définissez l'option de configuration `spark.databricks.delta.optimize.repartition.enabled=true` de manière à utiliser `repartition(1)` plutôt que `coalesce(1)` dans la commande Optimize afin de bénéficier de meilleures performances lors du compactage de nombreux petits fichiers.
  - Performances améliorées grâce à une approche basée sur la file d'attente afin d'exécuter en parallèle les tâches de compactage.
- Autres changements notables :
  - Prise en charge de l'utilisation de variables dans les commandes SQL VACUUM et OPTIMIZE.
  - Améliorations apportées à CONVERT TO DELTA avec des tables de catalogue, notamment :
    - Remplissez automatiquement le schéma de partition à partir du catalogue s'il n'est pas fourni.
    - Utilisez les informations de partition du catalogue pour rechercher les fichiers de données à valider au lieu d'effectuer une analyse complète du répertoire. Plutôt que de valider tous les fichiers de données du répertoire des tables, seuls les fichiers de données figurant dans les répertoires des partitions actives seront validés.
  - Prise en charge de la lecture par lots du flux CDF sur les tables permettant le mappage de colonnes lorsque DROP COLUMN et RENAME COLUMN n'ont pas été utilisés. Pour plus d'informations, consultez la documentation de Delta Lake.
  - Améliorez les performances de la commande Update en activant le nettoyage du schéma dès le premier passage.

Apache Iceberg

- Ajout de plusieurs améliorations des performances pour la planification des scans et les requêtes Spark.

- Ajout d'un client de catalogue REST commun qui utilise des validations basées sur les modifications pour résoudre les conflits de validation côté service.
- La syntaxe `AS OF` des requêtes SQL Time Travel est prise en charge.
- Ajout merge-on-read du support pour les requêtes MERGE et UPDATE.
- Ajout de la prise en charge de la réécriture des partitions à l'aide de Z-order.
- Ajout d'une spécification et d'une implémentation pour Puffin, un format adapté aux statistiques volumineuses et aux blobs d'index, tels que les esquisses Theta ou les filtres Bloom.
- Ajout de nouvelles interfaces permettant de consommer les données par incréments (analyses des ajouts et du journal des modifications).
- Ajout de la prise en charge des opérations en bloc et des lectures par plage sur les interfaces FileIO.
- Ajout de tables de métadonnées supplémentaires pour afficher les fichiers supprimés dans l'arborescence des métadonnées.
- Le comportement de suppression de la table a changé. Dans Iceberg 0.13.1, l'exécution de `DROP TABLE` supprime la table du catalogue ainsi que le contenu de la table. Dans Iceberg 1.0.0, `DROP TABLE` supprime uniquement la table du catalogue. Pour supprimer le contenu de la table, utilisez `DROP TABLE PURGE`.
- Les lectures vectorisés Parquet sont activées par défaut dans Iceberg 1.0.0. Pour désactiver les lectures vectorisées, définissez `read.parquet.vectorization.enabled` sur `false`.

Oracle

Les modifications sont mineures.

MySQL

Les modifications sont mineures.

Amazon Redshift

AWS Glue La version 4.0 intègre un nouveau connecteur Amazon Redshift avec un nouveau pilote JDBC. Pour plus d'informations sur les améliorations et sur la façon de migrer depuis AWS Glue les versions précédentes, consultezthe section called "Connexions Redshift".

Annexe A : Mises à niveau notables des dépendances

Voici les mises à niveau des dépendances :

Dépendance	Version en AWS Glue 4.0	Version en AWS Glue 3.0	Version en AWS Glue 2.0	Version en AWS Glue 1.0
Spark	3.3.0-amzn-1	3.1.1-amzn-0	2.4.3	2.4.3
Hadoop	3.3.3-amzn-0	3.2.1-amzn-3	2.8.5-amzn-5	2.8.5-amzn-1
Scala	2,12	2,12	2.11	2.11
Jackson	2.13.3	2.10.x	2.7.x	2.7.x
Hive	2.3.9-amzn-2	2.3.7-amzn-4	1.2	1.2
EMRFS	2,54,0	2,46,0	2.38.0	2.30.0
Json4s	3.7.0-M11	3.6.6	3.5.x	3.5.x
Flèche	7.0.0	2.0.0	0.10.0	0.10.0
AWS Glue Client de catalogue de données	3.7.0	3.0.0	1.10.0	N/A
Python	3,10	3.7	2.7 et 3.6	2.7 et 3.6
Boto	1,26	1,18	1.12	N/A

Annexe B : Mises à niveau du pilote JDBC

Voici les mises à niveau du pilote JDBC :

Pilote	Version du pilote JDBC dans les versions précédentes AWS Glue	Version du pilote JDBC en 3.0 AWS Glue	Version du pilote JDBC en 4.0 AWS Glue
MySQL	5.1	8.0.23	8.0.23
Microsoft SQL Server	6,10	7.0.0	9.4.0

Pilote	Version du pilote JDBC dans les versions précédentes AWS Glue	Version du pilote JDBC en 3.0 AWS Glue	Version du pilote JDBC en 4.0 AWS Glue
Oracle Databases	11.2	21,1	21,7
PostgreSQL	42,10	42,2,18	42,3.6
MongoDB	2.0.0	4.0.0	4.7.2
Amazon Redshift	redshift-jdbc41-1.2.12.1017	redshift-jdbc41-1.2.12.1017	redshift-jdbc42-2.1.0.16

Annexe C : Mises à niveau des connecteurs

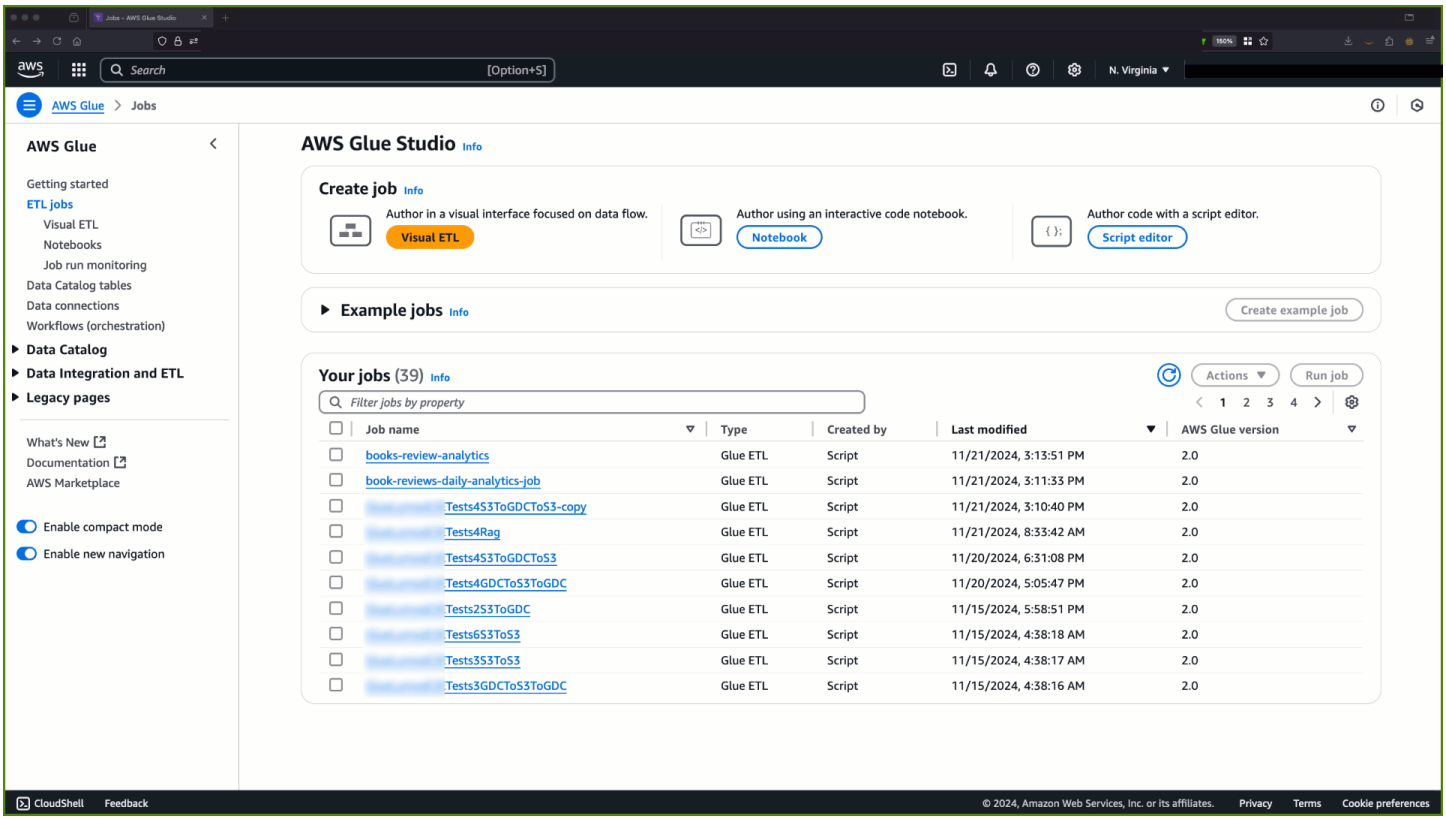
Les mises à niveau des connecteurs sont les suivantes :

Pilote	Version du connecteur en AWS Glue 3.0	Version du connecteur en AWS Glue 4.0
MongoDB	3.0.0	10,0.4
Hudi	0,1,1	0.12.1
Delta Lake	1.0.0	2.1.0
Iceberg	0.13.1	1.0.0
DynamoDB	1.11	1.12

Améliorations génératives de l'IA pour Apache Spark in AWS Glue

Spark Upgrades in AWS Glue permet aux ingénieurs de données et aux développeurs de mettre à niveau et de migrer leurs tâches AWS Glue Spark existantes vers les dernières versions de Spark à l'aide de l'IA générative. Les ingénieurs de données peuvent l'utiliser pour scanner leurs tâches AWS Glue Spark, générer des plans de mise à niveau, exécuter des plans et valider les résultats. Cela permet de réduire le temps et le coût des mises à niveau de Spark en automatisant le travail

indifférencié d'identification et de mise à jour des scripts, des configurations, des dépendances, des méthodes et des fonctionnalités de Spark.



Comment ça marche

Lorsque vous utilisez l'analyse des mises à niveau, AWS Glue identifie les différences entre les versions et les configurations dans le code de votre tâche afin de générer un plan de mise à niveau. Le plan de mise à niveau détaille toutes les modifications du code et les étapes de migration requises. Ensuite, AWS Glue crée et exécute l'application mise à niveau dans un environnement pour valider les modifications et génère une liste de modifications de code afin que vous puissiez migrer votre travail. Vous pouvez consulter le script mis à jour, ainsi que le résumé détaillant les modifications proposées. Après avoir effectué vos propres tests, acceptez les modifications et le job AWS Glue sera automatiquement mis à jour vers la dernière version avec le nouveau script.

Le processus d'analyse de la mise à niveau peut prendre un certain temps, selon la complexité de la tâche et la charge de travail. Les résultats de l'analyse de la mise à niveau seront stockés dans le chemin Amazon S3 spécifié, qui pourra être consulté pour comprendre la mise à niveau et les éventuels problèmes de compatibilité. Après avoir examiné les résultats de l'analyse de la mise à niveau, vous pouvez décider de poursuivre la mise à niveau proprement dite ou d'apporter les modifications nécessaires à la tâche avant sa mise à niveau.

Conditions préalables

Les prérequis suivants sont requis pour utiliser l'IA générative afin de mettre à niveau les jobs dans AWS Glue :

- AWS Tâches Glue 2 PySpark : seules les tâches AWS Glue 2 peuvent être mises à niveau vers AWS Glue 5.
- Les autorisations IAM sont nécessaires pour démarrer l'analyse, examiner les résultats et améliorer votre tâche. Pour plus d'informations, consultez les exemples dans la section Permissions ci-dessous.
- Si vous l'utilisez AWS KMS pour chiffrer des artefacts d'analyse, des AWS AWS KMS autorisations supplémentaires sont nécessaires. Pour plus d'informations, consultez les exemples dans la section AWS KMS politique ci-dessous.

Permissions

Pour démarrer une nouvelle analyse de la mise à niveau, vous devez disposer des autorisations suivantes :

1. Mettez à jour la politique IAM de l'appelant avec l'autorisation suivante :

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:StartJobUpgradeAnalysis",
                "glue:StartJobRun",
                "glue:GetJobRun",
                "glue:GetJob",
                "glue:BatchStopJobRun"
            ],
            "Resource": [
                "arn:aws:glue:us-east-1:111122223333:job/jobName"
            ]
        },
        {
            "Effect": "Allow",
```

```
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
              "arn:aws:s3:::amzn-s3-demo-bucket/script-location/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObject"
            ],
            "Resource": [
              "arn:aws:s3:::amzn-s3-demo-bucket/results/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt",
                "kms:GenerateDataKey"
            ],
            "Resource": "arn:aws:kms:us-east-1:111122223333:key/key-id"
        }
    ]
}
```

2. Mettez à jour le rôle d'exécution de la tâche que vous mettez à niveau pour inclure la politique en ligne suivante :

```
{
  "Effect": "Allow",
  "Action": ["s3:GetObject"],
  "Resource": [
    "ARN of the Amazon S3 path provided on API",
    "ARN of the Amazon S3 path provided on API/*"
  ]
}
```

Par exemple, si vous utilisez le chemin d'accès à Amazon S3 `s3://amzn-s3-demo-bucket/upgraded-result`, la politique sera la suivante :

```
{
      "Effect": "Allow",
      "Action": ["s3:GetObject"],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/upgraded-result/",
        "arn:aws:s3:::amzn-s3-demo-bucket/upgraded-result/*"
      ]
    }
```

Pour récupérer les détails d'une analyse, vous devez disposer des autorisations suivantes :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:GetJobUpgradeAnalysis"
      ],
      "Resource": [
        "arn:aws:glue:us-east-1:111122223333:job/jobName"
      ]
    }
  ]
}
```

Pour arrêter une analyse en cours, vous devez disposer des autorisations suivantes :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
```

```
        "glue:StopJobUpgradeAnalysis",
        "glue:BatchStopJobRun"
      ],
      "Resource": [
        "arn:aws:glue:us-east-1:111122223333:job/jobName"
      ]
    }
  ]
}
```

Pour répertorier toutes les analyses soumises dans le cadre d'une tâche spécifique, vous devez disposer des autorisations suivantes :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListJobUpgradeAnalyses"
      ],
      "Resource": [
        "arn:aws:glue:us-east-1:111122223333:job/jobName"
      ]
    }
  ]
}
```

Pour accepter les modifications d'une analyse et mettre à niveau une tâche, vous devez disposer des autorisations suivantes :

AWS KMS politique

Pour transmettre votre propre AWS KMS clé personnalisée lors du démarrage d'une analyse, reportez-vous à la section suivante pour configurer les autorisations appropriées sur les AWS KMS clés.

Configuration du chiffrement des artefacts obtenus à l'aide d'une AWS KMS clé :

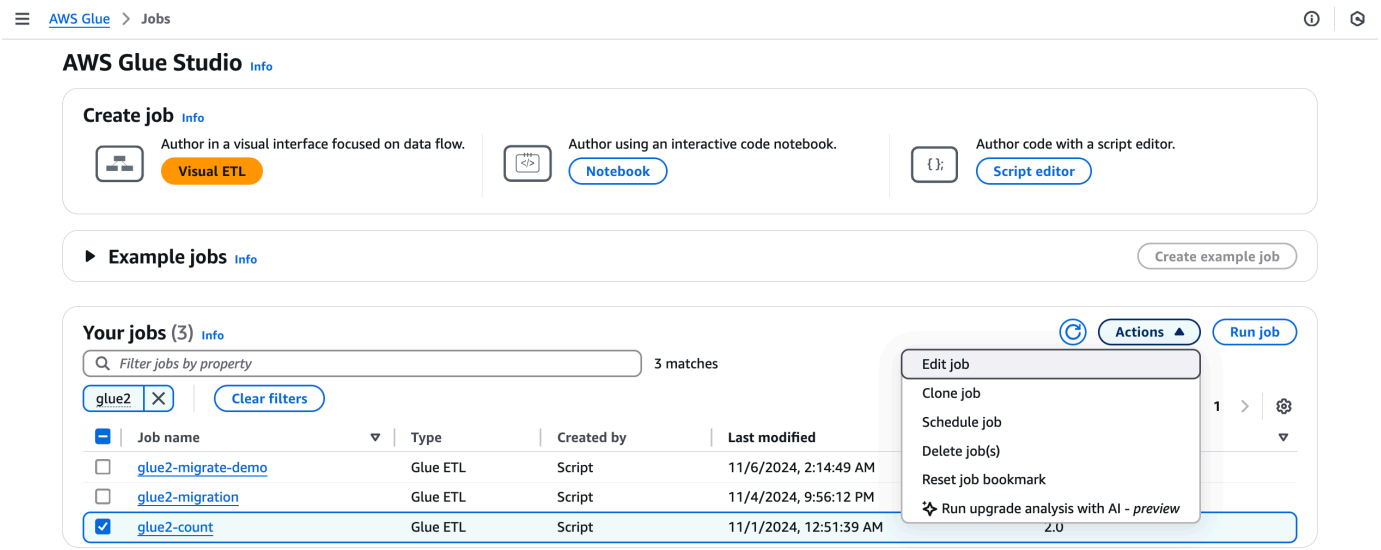
Cette politique garantit que vous disposez à la fois des autorisations de chiffrement et de déchiffrement sur la AWS KMS clé.

```
{
    "Effect": "Allow",
    "Principal":{
        "AWS": "<IAM Customer caller ARN>"
    },
    "Action": [
      "kms:Decrypt",
      "kms:GenerateDataKey",
    ],
    "Resource": "<key-arn-passed-on-start-api>"
}
```

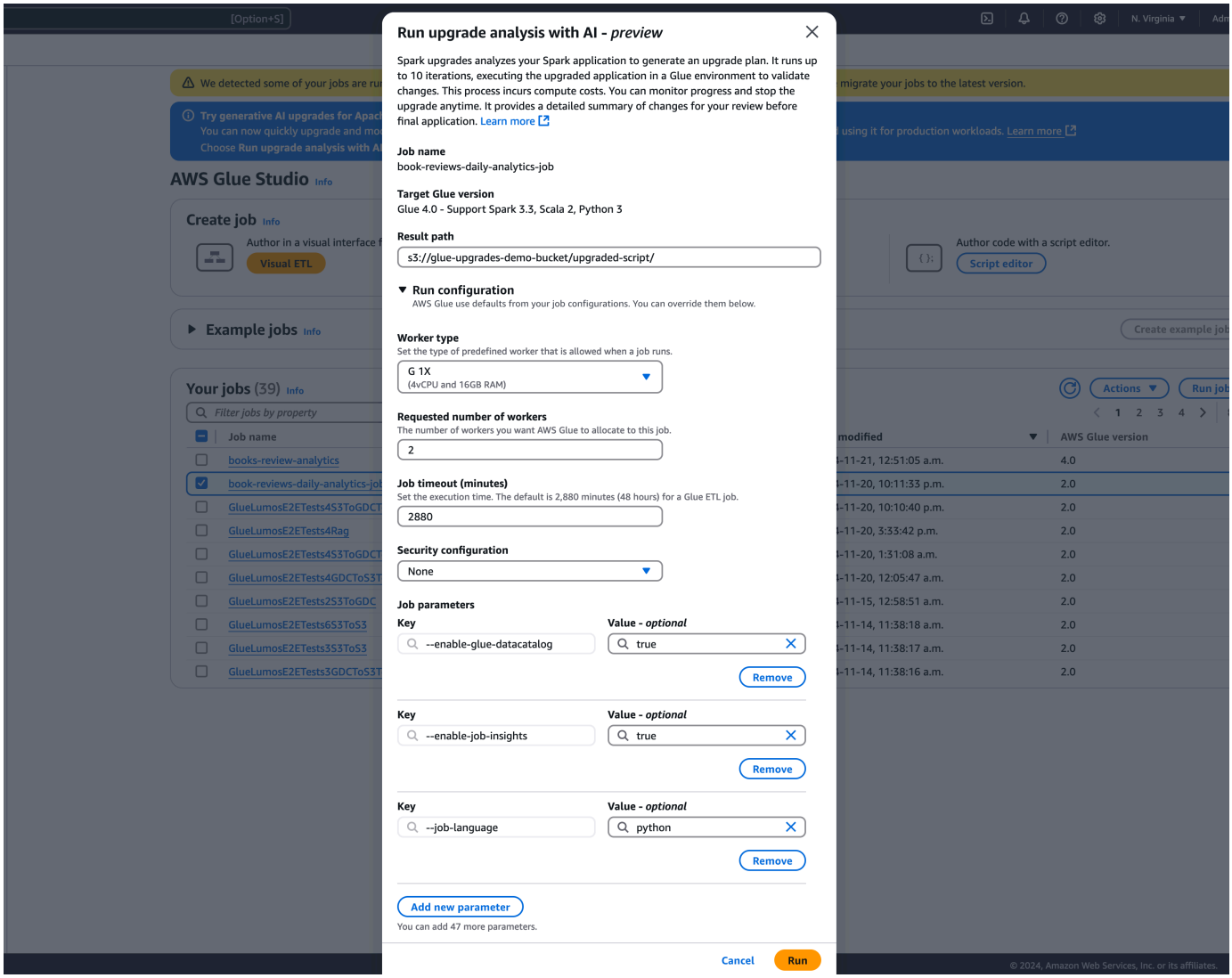
Exécution d'une analyse de la mise à niveau et application du script de mise à niveau

Vous pouvez exécuter une analyse de la mise à niveau, qui générera un plan de mise à niveau pour une tâche que vous sélectionnez dans la vue Tâches.

1. Dans Jobs, sélectionnez une tâche AWS Glue 2.0, puis choisissez Run upgrade analysis dans le menu Actions.



2. Dans la fenêtre modale, sélectionnez un chemin d'accès pour stocker votre plan de mise à niveau généré dans le chemin d'accès aux résultats. Il doit s'agir d'un compartiment Amazon S3 auquel vous pouvez accéder et sur lequel vous pouvez écrire.



3. Configurez les options supplémentaires au besoin :

- Configuration d'exécution (facultative) : la configuration d'exécution est un paramètre facultatif qui vous permet de personnaliser divers aspects des cycles de validation effectués lors de l'analyse de la mise à niveau. Cette configuration est utilisée pour exécuter le script mis à niveau et vous permet de sélectionner les propriétés de l'environnement de calcul (type de travailleur, nombre de travailleurs, etc.). Notez que vous devez utiliser vos comptes de développeur hors production pour exécuter les validations sur des exemples de jeux de données avant de vérifier, d'accepter les modifications et de les appliquer aux environnements de production. La configuration d'exécution comprend les paramètres personnalisables suivants :

- Type de travailleur : vous pouvez spécifier le type de travailleur à utiliser pour les cycles de validation, ce qui vous permet de choisir les ressources de calcul appropriées en fonction de vos besoins.
- Nombre de travailleurs : vous pouvez définir le nombre de travailleurs à affecter pour les cycles de validation, ce qui vous permet de mettre à l'échelle les ressources en fonction de vos besoins en matière de charge de travail.
- Délai d'expiration des tâches (en minutes) : ce paramètre vous permet de définir une limite de temps pour les cycles de validation, en veillant à ce que les tâches se terminent après une durée spécifiée afin d'éviter une consommation excessive de ressources.
- Configuration de la sécurité : vous pouvez configurer les paramètres de sécurité, tels que le chiffrement et le contrôle d'accès, pour garantir la protection de vos données et de vos ressources pendant les cycles de validation.
- Paramètres de tâche supplémentaires : si nécessaire, vous pouvez ajouter de nouveaux paramètres de tâche afin de personnaliser davantage l'environnement d'exécution pour les cycles de validation.

En tirant parti de la configuration d'exécution, vous pouvez adapter les cycles de validation à vos besoins spécifiques. Par exemple, vous pouvez configurer les cycles de validation pour qu'ils utilisent un jeu de données plus petit, ce qui permet d'effectuer l'analyse plus rapidement et d'optimiser les coûts. Cette approche garantit que l'analyse de la mise à niveau est effectuée efficacement tout en minimisant l'utilisation des ressources et les coûts associés pendant la phase de validation.

- Configuration du chiffrement (facultatif) :
  - Activer le chiffrement des artefacts de mise à niveau : activez le chiffrement au repos lorsque vous écrivez des données dans le chemin des résultats. Si vous ne souhaitez pas chiffrer vos artefacts de mise à niveau, laissez cette option désactivée.

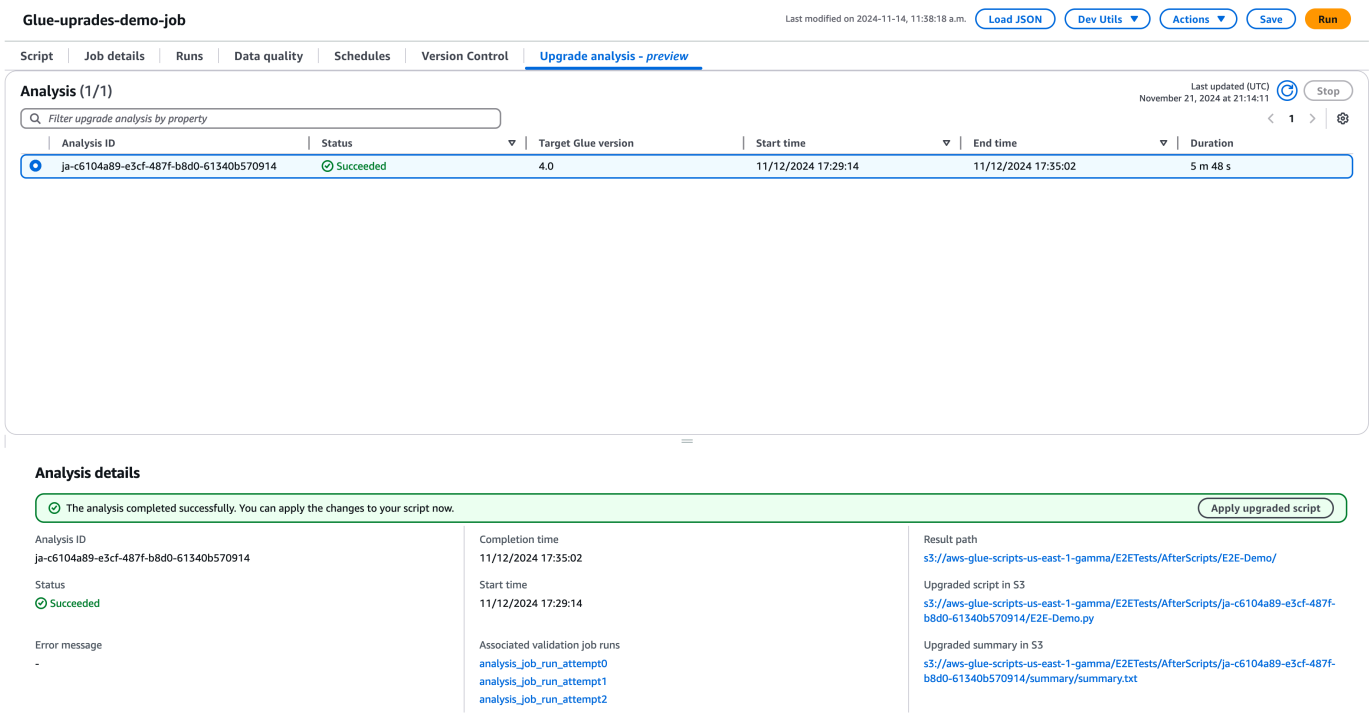
4. Choisissez Exécuter pour démarrer l'analyse de la mise à niveau. Pendant que l'analyse est en cours, vous pouvez consulter les résultats dans l'onglet Analyse de la mise à niveau. La fenêtre des détails de l'analyse affiche des informations sur l'analyse, ainsi que des liens vers les artefacts de mise à niveau.

   - Chemin d'accès au résultat : c'est à cet endroit que le résumé des résultats et le script de mise à niveau sont stockés.
   - Script mis à niveau dans Amazon S3 : emplacement du script de mise à niveau dans Amazon S3. Vous pouvez afficher le script avant d'appliquer la mise à niveau.

- Résumé de la mise à niveau dans Amazon S3 : l'emplacement du résumé de la mise à niveau dans Amazon S3. Vous pouvez consulter le résumé de la mise à niveau avant d'appliquer cette dernière.

5. Lorsque l'analyse de la mise à niveau se termine correctement, vous pouvez appliquer le script de mise à niveau pour mettre automatiquement à niveau votre tâche en choisissant Appliquer le script mis à niveau.

   Une fois appliquée, la version AWS Glue sera mise à jour vers la version 4.0. Vous pouvez consulter le nouveau script dans l'onglet Script.



Comprendre le résumé de votre mise à niveau

Cet exemple illustre le processus de mise à niveau d'une tâche AWS Glue de la version 2.0 à la version 4.0. L'exemple de tâche lit les données du produit depuis un compartiment Amazon S3, applique plusieurs transformations aux données à l'aide de Spark SQL, puis enregistre les résultats transformés dans un compartiment Amazon S3.

Code original (AWS Glue 2.0) - avant la mise à niveau

```
from awsglue.transforms import *
from pyspark.context import SparkContext
```

```
from awsglue.context import GlueContext
from pyspark.sql.types import *
from pyspark.sql.functions import *
from awsglue.job import Job
import json
from pyspark.sql.types import StructType

sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
job = Job(glueContext)

gdc_database = "s3://aws-glue-scripts-us-east-1-gamma/demo-database/"
schema_location = (
    "s3://aws-glue-scripts-us-east-1-gamma/DataFiles/"
)

products_schema_string = spark.read.text(
    f"{schema_location}schemas/products_schema"
).first()[0]

product_schema = StructType.fromJson(json.loads(products_schema_string))

products_source_df = (
    spark.read.option("header", "true")
    .schema(product_schema)
    .option(
        "path",
        f"{gdc_database}products/",
    )
    .csv(f"{gdc_database}products/")
)

products_source_df.show()
products_temp_view_name = "spark_upgrade_demo_product_view"
products_source_df.createOrReplaceTempView(products_temp_view_name)

query = f"select {products_temp_view_name}.*, format_string('%0$s-%0$s', category,
 subcategory) as unique_category from {products_temp_view_name}"
products_with_combination_df = spark.sql(query)
products_with_combination_df.show()

products_with_combination_df.createOrReplaceTempView(products_temp_view_name)
product_df_attribution = spark.sql(
```

```
    f"""
SELECT *,
unbase64(split(product_name, ' ')[0]) as product_name_decoded,
unbase64(split(unique_category, '-')[1]) as subcategory_decoded
FROM {products_temp_view_name}
"""
)
product_df_attribution.show()


product_df_attribution.write.mode("overwrite").option("header", "true").option(
    "path", f"{gdc_database}spark_upgrade_demo_product_agg/"
).saveAsTable("spark_upgrade_demo_product_agg", external=True)

spark_upgrade_demo_product_agg_table_df = spark.sql(
    f"SHOW TABLE EXTENDED in default like 'spark_upgrade_demo_product_agg'"
)
spark_upgrade_demo_product_agg_table_df.show()
job.commit()
```

Nouveau code (Glue 4.0) : après la mise à niveau

```
from awsglue.transforms import *
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from pyspark.sql.types import *
from pyspark.sql.functions import *
from awsglue.job import Job
import json
from pyspark.sql.types import StructType

sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
# change 1
spark.conf.set("spark.sql.adaptive.enabled", "false")
# change 2
spark.conf.set("spark.sql.legacy.pathOptionBehavior.enabled", "true")
job = Job(glueContext)

gdc_database = "s3://aws-glue-scripts-us-east-1-gamma/demo-database/"
schema_location = (
    "s3://aws-glue-scripts-us-east-1-gamma/DataFiles/"
```

```
)

products_schema_string = spark.read.text(
    f"{schema_location}schemas/products_schema"
).first()[0]

product_schema = StructType.fromJson(json.loads(products_schema_string))

products_source_df = (
    spark.read.option("header", "true")
    .schema(product_schema)
    .option(
        "path",
        f"{gdc_database}products/",
    )
    .csv(f"{gdc_database}products/")
)

products_source_df.show()
products_temp_view_name = "spark_upgrade_demo_product_view"
products_source_df.createOrReplaceTempView(products_temp_view_name)

# change 3
query = f"select {products_temp_view_name}.*, format_string('%1$s-%1$s', category,
 subcategory) as unique_category from {products_temp_view_name}"
products_with_combination_df = spark.sql(query)
products_with_combination_df.show()

products_with_combination_df.createOrReplaceTempView(products_temp_view_name)
# change 4
product_df_attribution = spark.sql(
    f"""
SELECT *,
try_to_binary(split(product_name, ' ')[0], 'base64') as product_name_decoded,
try_to_binary(split(unique_category, '-')[1], 'base64') as subcategory_decoded
FROM {products_temp_view_name}
"""
)
product_df_attribution.show()


product_df_attribution.write.mode("overwrite").option("header", "true").option(
    "path", f"{gdc_database}spark_upgrade_demo_product_agg/"
).saveAsTable("spark_upgrade_demo_product_agg", external=True)
```

```
spark_upgrade_demo_product_agg_table_df = spark.sql(
    f"SHOW TABLE EXTENDED in default like 'spark_upgrade_demo_product_agg'"
)
spark_upgrade_demo_product_agg_table_df.show()
job.commit()
```

Explication du résumé de l'analyse

```
In Spark 3.2, spark.sql.adaptive.enabled is enabled by default. To restore the behavior before
Spark 3.2,
you can set spark.sql.adaptive.enabled to false.


In Spark 3.1, path option cannot coexist when the following methods are called with path
parameter(s): DataFrameReader.load(), DataFrameWriter.save(), DataStreamReader.load(), or
DataStreamWriter.start(). In addition, paths option cannot coexist for DataFrameReader.load().
For example, spark.read.format(csv).option(path, /tmp).load(/tmp2) or spark.read.option(path, /
tmp).csv(/tmp2) will throw org.apache.spark.sql.AnalysisException. In Spark version 3.0 and
below, path option is overwritten if one path parameter is passed to above methods; path option
is added to the overall paths if multiple path parameters are passed to DataFrameReader.load().
To restore the behavior before Spark 3.1, you can set
spark.sql.legacy.pathOptionBehavior.enabled to true.


Since Spark 3.3, the `strfmt` in `format_string(strfmt, obj, ...)` and `printf(strfmt,
obj, ...)` will no longer support to use 0$ to specify the first argument, the first argument
should always reference by 1$ when use argument index to indicating the position of the argument
in the argument list.


Since Spark 3.3, the unbase64 function throws error for a malformed str input. Use
try_to_binary(<str>, 'base64') to tolerate malformed input and return NULL instead. In Spark 3.2
and earlier, the unbase64 function returns a best-efforts result for a malformed str input.
```

Sur la base du résumé, quatre modifications ont été proposées par AWS Glue afin de réussir la mise à niveau du script de AWS Glue 2.0 vers AWS Glue 4.0 :

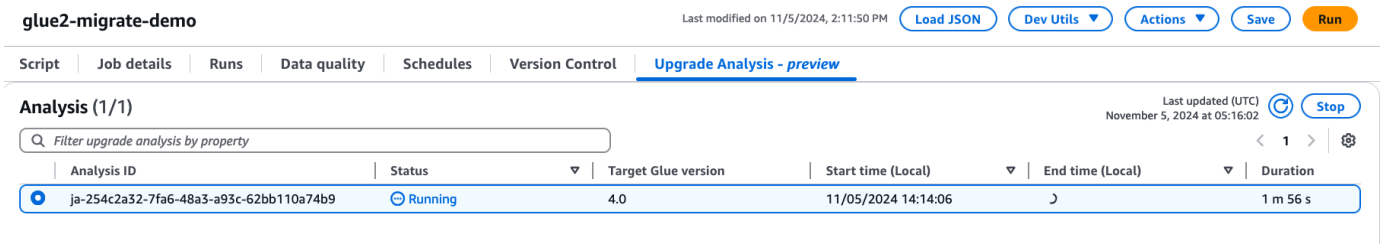
1. Configuration de Spark SQL (spark.sql.adaptive.enabled) : cette modification vise à restaurer le comportement de l'application, car une nouvelle fonctionnalité d'exécution de requêtes adaptatives Spark SQL est introduite à partir de Spark 3.2. Vous pouvez inspecter cette modification de configuration et l'activer ou la désactiver selon les préférences.
2. DataFrame Modification de l'API : l'option path ne peut pas coexister avec d'autres DataFrameReader opérations telles que`load()`. Pour conserver le comportement précédent, AWS Glue a mis à jour le script pour ajouter une nouvelle configuration SQL (spark.sql.legacy). pathOptionBehavior.activé).

3. Modification de l'API Spark SQL : le comportement de `strfmt` dans `format_string(strfmt, obj, ...)` a été mis à jour pour interdire `0$` comme premier argument. Pour garantir la compatibilité, AWS Glue a modifié le script pour l'utiliser `1$` comme premier argument à la place.
4. Modification de l'API Spark SQL : la fonction `unbase64` n'autorise pas les entrées de chaîne mal formées. Pour conserver le comportement précédent, AWS Glue a mis à jour le script afin d'utiliser la `try_to_binary` fonction.

Arrêt d'une analyse de la mise à niveau en cours

Vous pouvez annuler une analyse de la mise à niveau en cours ou simplement arrêter l'analyse.

1. Choisissez l'onglet Analyse de la mise à niveau.
2. Sélectionnez la tâche en cours d'exécution, puis choisissez Arrêter. Cela arrêtera l'analyse. Vous pouvez ensuite exécuter une autre analyse de la mise à niveau sur la même tâche.



Considérations

Lorsque vous commencez à utiliser Spark Upgrades, vous devez prendre en compte plusieurs aspects importants pour une utilisation optimale du service.

- Étendue et limites du service : La version actuelle se concentre sur les mises à niveau du PySpark code depuis les versions 2.0 de AWS Glue vers la version 5.0. À l'heure actuelle, le service gère le PySpark code qui ne repose pas sur des dépendances de bibliothèque supplémentaires. Vous pouvez exécuter des mises à niveau automatisées pour un maximum de 10 tâches simultanément dans un AWS compte, ce qui vous permet de mettre à niveau efficacement plusieurs tâches tout en préservant la stabilité du système.
  - Seuls les PySpark emplois sont pris en charge.
  - L'analyse de la mise à niveau expirera au bout de 24 heures.

  - Une seule analyse de la mise à niveau active peut être exécutée à la fois pour une tâche. Au niveau du compte, vous pouvez exécuter jusqu’à 10 analyses de la mise à niveau en même temps.
- Optimisation des coûts pendant le processus de mise à niveau : étant donné que Spark Upgrades utilise l'IA générative pour valider le plan de mise à niveau par le biais de plusieurs itérations, chaque itération étant exécutée sous forme de tâche AWS Glue dans votre compte, il est essentiel d'optimiser les configurations d'exécution des tâches de validation pour des raisons de rentabilité. Pour ce faire, nous vous recommandons de spécifier une configuration d’exécution lors du lancement d’une analyse de la mise à niveau, comme suit :
  - Utilisez des comptes de développeur hors production et sélectionnez des exemples de jeux de données fictifs qui représentent vos données de production, mais dont la taille est réduite pour la validation avec Spark Upgrades.
  - Utilisez des ressources de calcul correctement dimensionnées, telles que des travailleurs G1X, et sélectionnez un nombre approprié de travailleurs pour traiter vos échantillons de données.
  - Activation de l'auto-scaling des tâches AWS Glue, le cas échéant, pour ajuster automatiquement les ressources en fonction de la charge de travail.

  Par exemple, si votre tâche de production traite des téraoctets de données avec 20 travailleurs G.2X, vous pouvez configurer la tâche de mise à niveau pour traiter quelques gigaoctets de données représentatives avec 2 travailleurs G.2X et l’autoscaling activé pour la validation.
- Bonnes pratiques : nous vous recommandons vivement de commencer votre processus de mise à niveau par des tâches non liées à la production. Cette approche vous permet de vous familiariser avec le flux de travail de mise à niveau et de comprendre comment le service gère les différents types de modèles de code Spark.
- Alarmes et notifications : lorsque vous utilisez la fonction de mise à niveau de l'IA générative sur une tâche, assurez-vous qu' alarms/notifications elle est désactivée en cas d'échec des tâches. Au cours du processus de mise à niveau, il se peut que jusqu’à 10 tâches aient échoué dans votre compte avant que les artefacts mis à niveau ne soient fournis.
- Règles de détection des anomalies : désactivez également les règles de détection des anomalies sur la tâche en cours de mise à niveau, car les données écrites dans les dossiers de sortie lors des exécutions de tâches intermédiaires risquent de ne pas être au format attendu pendant la validation de la mise à niveau.
- Utiliser l’analyse de la mise à niveau avec les tâches idempotentes : utilisez l’analyse de la mise à niveau avec les tâches idempotentes pour vous assurer que chaque tentative d’exécution de tâche de validation suivante est similaire à la précédente et ne présente aucun problème. Les tâches

idempotentes sont des tâches qui peuvent être exécutées plusieurs fois avec les mêmes données d'entrée et qui produiront à chaque fois le même résultat. Lorsque vous utilisez les mises à niveau de Generative AI pour Apache Spark in AWS Glue, le service exécutera plusieurs itérations de votre tâche dans le cadre du processus de validation. Au cours de chaque itération, il apportera des modifications à votre code et à vos configurations Spark afin de valider le plan de mise à niveau. Si votre tâche Spark n'est pas idempotente, l'exécuter plusieurs fois avec les mêmes données d'entrée peut entraîner des problèmes.

Régions prises en charge

Les mises à niveau génératives de l'IA pour Apache Spark sont disponibles dans les régions suivantes :

- Asie-Pacifique : Tokyo (ap-northeast-1), Séoul (ap-northeast-2), Mumbai (ap-southeast-1), Singapour (ap-southeast-1) et Sydney (ap-southeast-2)
- Amérique du Nord : Canada (ca-central-1)
- Europe : Francfort (eu-central-1), Stockholm (eu-nord-1), Irlande (eu-west-1), Londres (eu-west-2) et Paris (eu-west-3)
- Amérique du Sud : São Paulo (sa-east-1)
- États-Unis : Virginie du Nord (us-east-1), Ohio (us-east-2) et Oregon (us-west-2)

Inférence entre régions dans Spark Upgrades

Spark Upgrades est alimenté par Amazon Bedrock et exploite l'inférence interrégionale (CRIS). Grâce à l'inférence entre régions, Spark Upgrades sélectionne automatiquement la région optimale au sein de votre zone géographique (comme décrit plus en détail ici) pour traiter votre demande d'inférence, optimisant ainsi les ressources de calcul disponibles et la disponibilité des modèles, et offrant la meilleure expérience client. L'utilisation de l'inférence entre régions n'entraîne aucun coût supplémentaire.

Les demandes d'inférence interrégionales sont conservées dans les AWS régions qui font partie de la zone géographique dans laquelle les données se trouvent à l'origine. Par exemple, une demande faite aux États-Unis est conservée dans les AWS régions des États-Unis. Bien que les données restent stockées uniquement dans la région principale, lorsque vous utilisez l'inférence entre régions, vos invites de saisie et les résultats de sortie peuvent être déplacés en dehors de votre région principale. Toutes les données seront transmises chiffrées sur le réseau sécurisé d'Amazon.

Utilisation des jobs Spark dans AWS Glue

Fournit des informations sur AWS Glue les tâches ETL de Spark.

Rubriques

- Utilisation des paramètres des tâches dans les tâches AWS Glue
- AWS Glue Spark et PySpark jobs
- AWS Glue types de travailleurs
- Tâches ETL en streaming dans AWS Glue
- Record correspondant à AWS Lake Formation FindMatches
- Migrer les programmes Apache Spark vers AWS Glue

Utilisation des paramètres des tâches dans les tâches AWS Glue

Lorsque vous créez une tâche AWS Glue, vous définissez des champs standard, tels que `Role` et`WorkerType`. Vous pouvez fournir des informations de configuration supplémentaires par le biais des champs `Argument` (Paramètres de la tâche dans la console). Dans ces champs, vous pouvez fournir aux tâches AWS Glue les arguments (paramètres) répertoriés dans cette rubrique.

Pour plus d'informations sur l'API AWS Glue Job, consultezthe section called "Tâches".

> **Note**
>
> La taille maximale des arguments de tâche est de 260 Ko. Une vérification de validation génère une erreur si la taille de l'argument est supérieure à 260 Ko.

Configuration des paramètres de tâche

Vous pouvez configurer une tâche via la console, dans les Détails de la tâche, sous l'en-tête Paramètres des tâches. Vous pouvez également configurer une tâche via le paramètre « AWS CLI en configurant » `DefaultArguments` ou `NonOverridableArguments` sur une tâche, ou en configurant `Arguments` l'exécution d'une tâche. Les arguments définis pour la tâche seront transmis à chaque exécution de la tâche, tandis que les arguments définis lors de l'exécution de la tâche ne seront transmis que pour cette exécution individuelle.

Par exemple, ce qui suit est la syntaxe pour exécuter une tâche en utilisant `--arguments` pour définir un paramètre de tâche.

```
$ aws glue start-job-run --job-name "CSV to CSV" --arguments='--scriptLocation="s3://
my_glue/libraries/test_lib.py"'
```

Accès aux paramètres des tâches

Lorsque vous écrivez des scripts AWS Glue, vous souhaiterez peut-être accéder aux valeurs des paramètres des tâches afin de modifier le comportement de votre propre code. Nous fournissons des méthodes d'aide pour ce faire dans nos bibliothèques. Ces méthodes résolvent les valeurs des paramètres d'exécution des tâches qui remplacent les valeurs des paramètres des tâches. Lors de la résolution de paramètres définis à plusieurs emplacements, la tâche `NonOverridableArguments` remplacera les `Arguments` de l'exécution de la tâche, qui remplacera les `DefaultArguments` de la tâche.

Dans Python :

Dans les tâches Python, nous fournissons une fonction nommée `getResolvedParameters`. Pour de plus amples informations, veuillez consulter the section called “getResolvedOptions”. Les paramètres de la tâche sont disponibles dans la variable `sys.argv`.

Dans Scala :

Dans les tâches Scala, nous fournissons un objet nommé `GlueArgParser`. Pour de plus amples informations, veuillez consulter the section called “GlueArgParser”. Les paramètres de la tâche sont disponibles dans la variable `sysArgs`.

Référence des paramètres de la tâche

AWS Glue reconnaît les noms d'arguments suivants que vous pouvez utiliser pour configurer l'environnement de script pour vos tâches et leurs exécutions :

`--additional-python-modules`

une liste délimitée par des virgules représentant un ensemble de packages Python à installer. Vous pouvez installer des packages depuis PyPI ou fournir une distribution personnalisée. Une entrée de package PyPI sera au format *`package`*==*`version`*, avec le nom PyPI et la version de votre package cible. Une entrée de distribution personnalisée est le chemin S3 vers la distribution.

Les entrées utilisent la correspondance entre versions de Python pour faire correspondre le package et la version. Cela signifie que vous devrez utiliser deux signes d'égalité, comme ==. Il existe d'autres opérateurs de correspondance de version. Pour plus d'informations, voir PEP 440.

Pour transmettre les options d'installation du module à `pip3`, utilisez le paramètre --python-modules-installer-option.

`--auto-scale-within-microbatch`

La valeur par défaut est True. Ce paramètre ne peut être utilisé que pour les tâches de streaming AWS Glue, qui traitent les données de streaming en une série de microlots, et le dimensionnement automatique doit être activé. Lorsque cette valeur est définie sur false, il calcule la moyenne mobile exponentielle de la durée du lot pour les microlots terminés et compare cette valeur à la taille de la fenêtre pour déterminer s'il convient d'augmenter ou de réduire le nombre d'exécuteurs. La mise à l'échelle ne se produit que lorsqu'un microlot est terminé. Lorsque cette valeur est définie sur true, lors d'un microlot, elle augmente lorsque le nombre de tâches Spark reste le même pendant 30 secondes ou que le traitement par lots en cours est supérieur à la taille de la fenêtre. Le nombre d'exécuteurs diminuera si un exécuteur est inactif depuis plus de 60 secondes ou si la moyenne mobile exponentielle de la durée du lot est faible.

`--class`

La classe Scala qui sert de point d'entrée à votre script Scala. Cela s'applique uniquement si votre `--job-language` est défini sur `scala`.

`--continuous-log-conversionPattern`

Spécifie un modèle de journal de conversion personnalisé pour une tâche activée pour la journalisation en continu. Le modèle de conversion s'applique uniquement aux journaux des pilotes et des programmes d'exécution. Cela n'affecte pas la barre de progression de AWS Glue.

`--continuous-log-logGroup`

Spécifie un nom de groupe de CloudWatch journaux Amazon personnalisé pour une tâche activée pour la journalisation continue.

`--continuous-log-logStreamPrefix`

Spécifie un préfixe de flux de CloudWatch journal personnalisé pour une tâche activée pour la journalisation continue.

`--customer-driver-env-vars` et **`--customer-executor-env-vars`**

Ces paramètres définissent des variables d'environnement sur le système d'exploitation respectivement pour chaque travailleur (pilote ou exécuteur). Vous pouvez utiliser ces paramètres lorsque vous créez des plateformes et des frameworks personnalisés sur AWS Glue, afin de permettre à vos utilisateurs d'écrire des jobs dessus. L'activation de ces deux indicateurs vous permettra de définir des variables d'environnement différentes sur le pilote et sur l'exécuteur, respectivement, sans avoir à injecter la même logique dans le script de tâche lui-même.

Exemple d'utilisation

Voici un exemple de l'utilisation de ces paramètres :

```
"–customer-driver-env-vars", "CUSTOMER_KEY1=VAL1,CUSTOMER_KEY2=\"val2,val2 val2\"",
"–customer-executor-env-vars", "CUSTOMER_KEY3=VAL3,KEY4=VAL4"
```

Les définir dans l'argument d'exécution de tâche revient à exécuter les commandes suivantes :

Dans le pilote :

- exporter CUSTOMER_ = KEY1 VAL1
- export CUSTOMER_ KEY2 ="val2, val2 val2"

Dans l'exécuteur :

- exporter CUSTOMER_ = KEY3 VAL3

Ensuite, dans le script de tâche lui-même, vous pouvez récupérer les variables d'environnement à l'aide de `os.environ.get("CUSTOMER_KEY1")` ou de `System.getenv("CUSTOMER_KEY1")`.

Syntaxe appliquée

Respectez les normes suivantes lors de la définition des variables d'environnement :

- Chaque clé doit avoir le `CUSTOMER_ prefix`.

  Par exemple : pour `"CUSTOMER_KEY3=VAL3,KEY4=VAL4"`, `KEY4=VAL4` sera ignoré et ne sera pas défini.
- Chaque paire clé/valeur doit être délimitée par une simple virgule.

  Par exemple : `"CUSTOMER_KEY3=VAL3,CUSTOMER_KEY4=VAL4"`
- Si la « valeur » comporte des espaces ou des virgules, elle doit être définie entre guillemets.

Par exemple : `CUSTOMER_KEY2=\"val2,val2 val2\"`

Cette syntaxe modélise étroitement les normes de définition des variables d'environnement bash.

`--datalake-formats`

Pris en charge dans AWS Glue 3.0 et versions ultérieures.

Spécifie la structure du lac de données à utiliser. AWS Glue ajoute les fichiers JAR requis pour les frameworks que vous spécifiez dans le`classpath`. Pour de plus amples informations, veuillez consulter Utilisation de frameworks de lacs de données avec des tâches AWS Glue ETL.

Vous pouvez spécifier une ou plusieurs des valeurs suivantes, séparées par une virgule :

- `hudi`
- `delta`
- `iceberg`

Par exemple, transmettez l'argument suivant pour spécifier les trois infrastructures.

```
'--datalake-formats': 'hudi,delta,iceberg'
```

`--disable-proxy-v2`

Désactivez le proxy de AWS service pour autoriser les appels de service vers Amazon S3 et AWS Glue provenant de votre script via votre VPC. CloudWatch Pour plus d'informations, consultez Configurer AWS les appels pour qu'ils passent par votre VPC. Pour désactiver le proxy de service, définissez la valeur de ce paramètre sur `true`.

`--enable-auto-scaling`

Lorsque cette valeur est paramétrée sur `true`, active la mise à l'échelle automatique ainsi que la facturation par employé.

`--enable-continuous-cloudwatch-log`

Permet la journalisation continue en temps réel des tâches AWS Glue. Vous pouvez afficher les journaux de tâches Apache Spark en temps réel dans CloudWatch.

`--enable-continuous-log-filter`

Spécifie un filtre standard (`true`) ou aucun filtre (`false`) lorsque vous créez ou modifiez une tâche activée pour la journalisation continue. Le choix du filtre standard élimine les messages

inutiles du journal des battements de cœur d'Apache Spark driver/executor et d'Apache Hadoop YARN. Ne choisir aucun filtre vous donne tous les messages de journaux.

`--enable-glue-datacatalog`

Vous permet d'utiliser le catalogue de données AWS Glue comme métastore Apache Spark Hive. Pour activer cette fonctionnalité, définissez la valeur sur `true`.

`--enable-job-insights`

Permet une surveillance supplémentaire de l'analyse des erreurs grâce à AWS Glue Job Run Insights. Pour en savoir plus, consultez the section called "Surveillance à l'aide AWS Glue d'informations sur l'exécution des tâches". Par défaut, cette valeur est définie sur `true` et les informations sur l'exécution des tâches sont activées.

Cette option est disponible pour les versions 2.0 et 3.0 de AWS Glue.

`--enable-lakeformation-fine-grained-access`

Permet un contrôle d'accès précis pour les tâches AWS Glue. Pour de plus amples informations, veuillez consulter the section called "Lake Formation pour FGAC".

`--enable-metrics`

Permet de collecter des métriques de profilage de tâche pour l'exécution de cette tâche. Ces statistiques sont disponibles sur les consoles AWS Glue et Amazon CloudWatch . La valeur de ce paramètre n'est pas pertinente. Pour activer cette fonctionnalité, vous pouvez attribuer à ce paramètre n'importe quelle valeur, mais la valeur `true` est recommandée pour des raisons de clarté. Pour désactiver cette fonctionnalité, supprimez ce paramètre de la configuration de votre tâche.

`--enable-observability-metrics`

Active un ensemble de mesures d'observabilité pour générer des informations sur ce qui se passe dans chaque tâche exécutée sur la page Job Runs Monitoring sous AWS la console Glue et la Amazon CloudWatch console. Pour activer cette fonctionnalité, définissez la valeur de ce paramètre sur « true ». Pour désactiver cette fonctionnalité, définissez-la sur `false` ou supprimez ce paramètre de la configuration de votre tâche.

`--enable-rename-algorithm-v2`

Définit la version de l'algorithme de renommage EMRFS sur la version 2. Lorsqu'une tâche Spark utilise le mode d'écrasement de partition dynamique, il est possible qu'une partition en double soit créée. Par exemple, vous pouvez vous retrouver avec une partition dupliquée telle que `s3://`

`bucket/table/location/p1=1/p1=1`. Ici, P1 est la partition en cours de remplacement. Renommer l'algorithme en version 2 résout ce problème.

Cette option n'est disponible que sur la version 1.0 de AWS Glue.

`--enable-s3-parquet-optimized-committer`

Active le validateur EMRFS optimisé pour S3 pour l'écriture de données Parquet dans Amazon S3. Vous pouvez fournir la parameter/value paire via la console AWS Glue lors de la création ou de la mise à jour d'une tâche AWS Glue. Définir la valeur sur **`true`** active le validateur. Par défaut, le drapeau est activé dans AWS Glue 3.0 et désactivé dans AWS Glue 2.0.

Pour de plus amples informations, veuillez consulter Utilisation du validateur EMRFS optimisé pour S3.

`--enable-spark-ui`

Lorsque ce paramètre est défini sur`true`, active la fonctionnalité permettant d'utiliser l'interface utilisateur de Spark pour surveiller et déboguer les tâches AWS Glue ETL.

`--executor-cores`

Nombre de tâches Spark pouvant être exécutées en parallèle. Cette option est prise en charge sur AWS Glue 3.0+. La valeur ne doit pas dépasser le double du nombre de v CPUs sur le type de travailleur, qui est 8 activé `G.1X`G.2X, 16 activé`G.4X`, 32 activé`G.8X`, 64 activé`G.12X`, 96 activé`G.16X`, 128 activé et 8 activé`R.1X`, 16 activé`R.2X`, 32 activé`R.4X`, 64 activé`R.8X`. Vous devez faire preuve de prudence lors de la mise à jour de cette configuration, car cela peut avoir un impact sur les performances des tâches. En effet, l'augmentation du parallélisme des tâches entraîne une surcharge de la mémoire et des disques, ainsi qu'un ralentissement des systèmes source et cible (par exemple : cela entraînerait davantage de connexions simultanées sur Amazon RDS).

`--extra-files`

Chemins Amazon S3 vers des fichiers supplémentaires (par exemple, des fichiers de configuration) que AWS Glue copie vers le répertoire de travail de votre script sur le nœud du pilote avant de l'exécuter. Plusieurs valeurs doivent être des chemins entiers séparés par une virgule (`,`). La valeur peut être un fichier individuel ou des emplacements de répertoire. Cette option n'est pas prise en charge pour les types de tâches shell Python.

`--extra-jars`

Les chemins Amazon S3 mènent à des fichiers supplémentaires que AWS Glue copie vers le pilote et les exécuteurs. AWS Glue ajoute également ces fichiers au chemin de classe Java avant

d'exécuter votre script. Plusieurs valeurs doivent être des chemins entiers séparés par une virgule (`,`). Il n'est pas nécessaire que l'extension soit `.jar`.

`--extra-py-files`

Les chemins Amazon S3 vers des modules Python supplémentaires que AWS Glue ajoute au chemin Python sur le nœud pilote avant d'exécuter votre script. Plusieurs valeurs doivent être des chemins entiers séparés par une virgule (`,`). Seuls les fichiers individuels sont pris en charge, et non un chemin de répertoire.

`--job-bookmark-option`

Contrôle le comportement d'un signet de tâche. Les valeurs d'option suivantes peuvent être définies.

Valeur --job-bookmark-option	Description
`job-bookmark-enable`	Conserver une trace des données précédemm ent traitées. Lorsqu'une tâche s'exécute, traiter les nouvelles données depuis le dernier point de contrôle.
`job-bookmark-disable`	Toujours traiter l'intégralité de l'ensemble de données. Vous êtes responsable de la gestion des résultats des précédentes exécutions de tâches.
`job-bookmark-pause`	Traitez les données progressives+ depuis la dernière exécution réussie ou les données de la plage identifiée par les sous-options suivantes, sans mettre à jour l'état du dernier marque-page. Vous êtes responsable de la gestion des résultats des précédentes exécutions de tâches. Les deux sous-options sont les suivantes : • **`job-bookmark-from`** `<from-value>` est l'ID d'exécution qui représente toute l'entrée qui a été traitée jusqu'à la dernière exécution réussie jusqu'à l'ID d'exécution spécifié (compris). L'entrée correspon dante est ignorée. • **`job-bookmark-to`** `<to-value>` est l'ID d'exécution qui représente toute l'entrée qui a été traitée jusqu'à la dernière exécution réussie jusqu'à

Valeur --job-bookmark-option	Description
	l'ID d'exécution spécifié (compris). L'entrée correspondante, à l'exclusion de l'entrée identifiée par `<from-value>` , est traitée par la tâche. Toute entrée postérieure à cette entrée est également exclue pour traitement. L'état du marque-page de tâche n'est pas mis à jour lorsque cette option est spécifiée. Les sous-options sont facultatives. Cependant, lorsqu'elles sont utilisées, les deux sous-options doivent être fournies.

Par exemple, pour activer un marque-page de tâche, transmettez l'argument suivant :

```
'--job-bookmark-option': 'job-bookmark-enable'
```

`--job-language`

Langage de programmation des scripts. Cette valeur doit être `scala` ou `python`. Si le paramètre n'est pas présent, la valeur par défaut est `python`.

`--python-modules-installer-option`

Une chaîne de texte simple qui définit les options à transmettre à `pip3` lors de l'installation de modules avec --additional-python-modules. Fournissez des options comme vous le feriez dans la ligne de commande, séparées par des espaces et préfixées par des tirets. Pour plus d'informations sur l'utilisation, consultez the section called “Installation de modules Python supplémentaires avec pip dans AWS Glue 2.0 ou version ultérieure”.

> **Note**
>
> Cette option n'est pas prise en charge pour les tâches AWS Glue lorsque vous utilisez Python 3.9.

`--scriptLocation`

Emplacement Amazon Simple Storage Service (Amazon S3) dans lequel votre script ETL est situé (sous la forme `s3://path/to/my/script.py`). Ce paramètre remplace celui du script défini dans l'objet `JobCommand`.

`--spark-event-logs-path`

Spécifie un chemin d'accès Amazon S3. Lors de l'utilisation de la fonction de surveillance de l'interface utilisateur Spark, AWS Glue vide les journaux d'événements Spark vers ce chemin Amazon S3 toutes les 30 secondes dans un compartiment qui peut être utilisé comme répertoire temporaire pour stocker les événements de l'interface utilisateur Spark.

`--TempDir`

Spécifie un chemin Amazon S3 vers un compartiment qui peut être utilisé comme répertoire temporaire pour la tâche.

Par exemple, pour définir un répertoire temporaire, transmettez l'argument suivant :

```
'--TempDir': 's3-path-to-directory'
```

> **Note**
>
> AWS Glue crée un compartiment temporaire pour les tâches s'il n'en existe pas déjà un dans une région. Ce compartiment peut permettre l'accès au public. Vous pouvez soit modifier le compartiment dans Amazon S3 pour définir le bloc d'accès public, soit supprimer le compartiment ultérieurement une fois toutes les tâches de cette région terminées.

`--use-postgres-driver`

Lorsque vous définissez cette valeur sur `true`, cela priorise le pilote JDBC Postgres dans le chemin de classe afin d'éviter tout conflit avec le pilote JDBC Amazon Redshift. Cette option n'est disponible que dans AWS la version 2.0 de Glue.

`--user-jars-first`

Lorsque vous définissez cette valeur sur `true`, cela priorise les fichiers JAR supplémentaires du client dans le chemin de classe. Cette option n'est disponible que dans AWS la version 2.0 ou ultérieure de Glue.

`--conf`

Contrôle les paramètres de configuration de Spark. Elle est destinée aux cas d'utilisation avancés.

`--encryption-type`

Paramètre hérité. Le comportement correspondant doit être configuré à l'aide des configurations de sécurité. Pour plus d'informations sur les configurations de sécurité, consultez the section called "Chiffrement des données écrites par AWS Glue".

AWS Glue utilise les arguments suivants en interne et vous ne devez jamais les utiliser :

- `--debug`— Interne à AWS Glue. Ne pas définir.
- `--mode`— Interne à AWS Glue. Ne pas définir.
- `--JOB_NAME`— Interne à AWS Glue. Ne pas définir.
- `--endpoint`— Interne à AWS Glue. Ne pas définir.

AWS Glue permet de démarrer un environnement à l'aide du `site` module Python `sitecustomize` pour effectuer des personnalisations spécifiques au site. L'amorçage de vos propres fonctions d'initialisation est recommandé uniquement pour les cas d'utilisation avancés et est pris en charge dans toute la mesure du possible sur Glue 4.0. AWS

Le préfixe de la variable d'environnement, `GLUE_CUSTOMER`, est réservé à l'usage du client.

AWS Glue Spark et PySpark jobs

AWS Glue soutenez Spark et PySpark jobs. Une tâche Spark est exécutée dans un environnement Apache Spark géré par AWS Glue. Elle traite les données par lots. Une tâche ETL en streaming est similaire à une tâche Spark, sauf qu'elle exécute ETL sur des flux de données. Elle utilise le cadre Apache Spark Structured Streaming. Certaines fonctionnalités de tâche Spark ne sont pas disponibles pour les tâches ETL en streaming.

Les sections suivantes fournissent des informations sur AWS Glue Spark et les PySpark jobs.

Rubriques

- Configuration des propriétés des tâches pour les tâches Spark dans AWS Glue
- Modification de scripts Spark dans la consoleAWS Glue

- Tâches (hérité)
- Suivi des données traitées à l'aide de signets de tâche
- Stockage des données réorganisées Spark
- Surveillance des tâches Spark AWS Glue
- Résolution des problèmes liés à l'IA générative pour Apache Spark in AWS Glue
- Utilisation de vues matérialisées avec Glue AWS

Configuration des propriétés des tâches pour les tâches Spark dans AWS Glue

Lorsque vous définissez votre tâche sur la AWS Glue console, vous fournissez des valeurs pour les propriétés permettant de contrôler l'environnement AWS Glue d'exécution.

Définition des propriétés des tâches Spark

La liste suivante contient les propriétés d'une tâche Spark. Pour les propriétés d'une tâche shell Python, consultez Définition des propriétés pour les tâches shell Python. Pour les propriétés d'une tâche ETL en streaming, consultez the section called "Définition des propriétés de tâche pour une tâche ETL en streaming".

Les propriétés sont répertoriées dans l'ordre dans lequel elles apparaissent dans l'assistant d'ajout d'une tâche sur AWS Glue la console.

Nom

Fournit une chaîne UTF-8 d'une longueur maximale de 255 caractères

Description

Fournissez une description facultative de 2 048 caractères maximum.

Rôle IAM

Spécifiez le rôle IAM qui permet de définir les autorisations d'accès aux ressources utilisées pour exécuter la tâche et accéder aux magasins de données. Pour plus d'informations sur les autorisations relatives à l'exécution de tâches dans AWS Glue, consultezGestion des identités et des accès pour AWS Glue.

Type

Le type de tâche ETL. Ce paramètre est défini automatiquement en fonction du type de sources de données que vous sélectionnez.

- Spark exécute un script ETL Apache Spark avec la commande de tâche `glueetl`.
- Spark Streaming exécute un script ETL Apache Spark en streaming avec la commande de tâche `gluestreaming`. Pour de plus amples informations, veuillez consulter the section called "Tâches ETL en streaming".
- Python shell (shell Python) exécute un script Python avec la commande de tâche `pythonshell`. Pour de plus amples informations, veuillez consulter Configuration des propriétés pour les tâches shell Python dans AWS Glue.

Version AWS Glue

AWS Glue version détermine les versions d'Apache Spark et de Python disponibles pour la tâche, comme indiqué dans le tableau suivant.

AWS Glue version	Versions Spark et Python prises en charge
5.1	• Spark 3.5.6 • Python 3.11
5.0	• Spark 3.5.4 • Python 3.11
4.0	• Spark 3.3.0 • Python 3.10
3.0	• Spark 3.1.1 • Python 3.7

Language

Le code inclus dans le script ETL permet de définir la logique de votre tâche. Le script peut être codé dans Python ou dans Scala. Vous pouvez choisir si le script exécuté par le job est généré par vous AWS Glue ou fourni par vous. Vous fournissez le nom et l'emplacement du script dans Amazon Simple Storage Service (Amazon S3). Vérifiez qu'il n'existe pas de fichier portant le même nom que le répertoire de script dans le chemin d'accès. Pour en savoir plus sur l'écriture de scripts, consultez AWS Glue guide de programmation.

Type d'employé

Les types d'employé suivantes sont disponibles :

Les ressources disponibles pour les AWS Glue travailleurs sont mesurées en DPUs. Un DPU est une mesure relative de la puissance de traitement composée de 4 V de capacité CPUs de calcul et de 16 Go de mémoire.

- **`G.025X`** – Lorsque vous choisissez ce type, vous devez également fournir une valeur pour Number of workers (Nombre d'employés). Chaque utilisateur est mappé sur 0,25 DPU (2 VCPUs, 4 Go de mémoire) avec un disque de 84 Go (environ 34 Go libres). Nous recommandons ce type d'employé pour les travaux de streaming à faible volume. Ce type de travailleur n'est disponible que pour les tâches de streaming en AWS Glue version 3.0 ou ultérieure.
- **`G.1X`** – Lorsque vous choisissez ce type, vous devez également fournir une valeur pour Number of workers (Nombre d'employés). Chaque utilisateur est mappé sur 1 DPU (4 VCPUs, 16 Go de mémoire) avec un disque de 94 Go (environ 44 Go libres). Nous recommandons ce type de travailleur pour les charges de travail telles que les transformations de données, les jointures et les requêtes, afin de proposer un moyen évolutif et rentable d'exécuter la plupart des tâches.
- **`G.2X`** – Lorsque vous choisissez ce type, vous devez également fournir une valeur pour Number of workers (Nombre d'employés). Chaque utilisateur mappe sur 2 processeurs (8 VCPUs, 32 Go de mémoire) avec un disque de 138 Go (environ 78 Go libres). Nous recommandons ce type de travailleur pour les charges de travail telles que les transformations de données, les jointures et les requêtes, afin de proposer un moyen évolutif et rentable d'exécuter la plupart des tâches.
- **`G.4X`** – Lorsque vous choisissez ce type, vous devez également fournir une valeur pour Number of workers (Nombre d'employés). Chaque utilisateur mappe sur 4 processeurs (16 VCPUs, 64 Go de mémoire) avec un disque de 256 Go (environ 230 Go libres). Nous recommandons ce type de travailleur pour les tâches dont les charges de travail contiennent les transformations, les agrégations, les jointures et les requêtes les plus exigeantes.
- **`G.8X`** – Lorsque vous choisissez ce type, vous devez également fournir une valeur pour Number of workers (Nombre d'employés). Chaque utilisateur mappe sur 8 DPU (32 VCPUs, 128 Go de mémoire) avec un disque de 512 Go (environ 485 Go libres). Nous recommandons ce type de travailleur pour les tâches dont les charges de travail contiennent les transformations, les agrégations, les jointures et les requêtes les plus exigeantes.
- **`G.12X`** – Lorsque vous choisissez ce type, vous devez également fournir une valeur pour Number of workers (Nombre d'employés). Chaque utilisateur mappe sur 12 DPU (48 VCPUs, 192 Go de mémoire) sur un disque de 768 Go (environ 741 Go libres). Nous recommandons

ce type de travailleur pour les tâches comportant des charges de travail très importantes et gourmandes en ressources qui nécessitent une capacité de calcul importante.

- **`G.16X`** – Lorsque vous choisissez ce type, vous devez également fournir une valeur pour Number of workers (Nombre d'employés). Chaque utilisateur mappe sur 16 DPU (64 VCPUs, 256 Go de mémoire) avec un disque de 1024 Go (environ 996 Go libres). Nous recommandons ce type de travailleur pour les tâches comportant les charges de travail les plus importantes et les plus gourmandes en ressources et qui nécessitent une capacité de calcul maximale.
- **`R.1X`** – Lorsque vous choisissez ce type, vous devez également fournir une valeur pour Number of workers (Nombre d'employés). Chaque travailleur mappe vers 1 DPU avec une configuration optimisée pour la mémoire. Nous recommandons ce type de programme de travail pour les charges de travail gourmandes en mémoire qui rencontrent fréquemment des out-of-memory erreurs ou nécessitent des ratios élevés. memory-to-CPU
- **`R.2X`** – Lorsque vous choisissez ce type, vous devez également fournir une valeur pour Number of workers (Nombre d'employés). Chaque travailleur mappe vers 2 DPU avec une configuration optimisée pour la mémoire. Nous recommandons ce type de programme de travail pour les charges de travail gourmandes en mémoire qui rencontrent fréquemment des out-of-memory erreurs ou nécessitent des ratios élevés. memory-to-CPU
- **`R.4X`** – Lorsque vous choisissez ce type, vous devez également fournir une valeur pour Number of workers (Nombre d'employés). Chaque travailleur mappe vers 4 DPU avec une configuration optimisée pour la mémoire. Nous recommandons ce type de programme de travail pour les charges de travail gourmandes en mémoire qui rencontrent fréquemment des out-of-memory erreurs ou nécessitent des ratios élevés. memory-to-CPU
- **`R.8X`** – Lorsque vous choisissez ce type, vous devez également fournir une valeur pour Number of workers (Nombre d'employés). Chaque travailleur mappe vers 8 DPU avec une configuration optimisée pour la mémoire. Nous recommandons ce type de programme de travail pour les charges de travail gourmandes en mémoire très volumineuses qui rencontrent fréquemment des out-of-memory erreurs ou nécessitent des ratios élevés. memory-to-CPU

Spécifications des types de travailleurs

La table suivante fournit des spécifications détaillées pour tous les types de travailleurs G disponibles :

Spécifications du type de travailleur G

Type de travailleur	DPU par nœud	vCPU	Mémoire (Go)	Disque (Go)	Espace disque disponible (Go)	Exécuteurs Spark par nœud
G.025X	0.25	2	4	84	34	1
G.1X	1	4	16	94	44	1
G.2X	2	8	32	138	78	1
G.4X	4	16	64	256	230	1
G.8X	8	32	128	512	485	1
G.12X	12	48	192	768	741	1
G.16 X	16	64	256	1024	996	1

Important : les types de travailleurs G.12X et G.16X, ainsi que tous les types de travailleurs R (R.1X à R.8X), ont une latence de démarrage plus élevée.

Un taux horaire vous est facturé en fonction du nombre de tâches DPUs utilisées pour exécuter vos tâches ETL. Pour plus d'informations, consultez la page de tarification AWS Glue.

Pour les tâches de AWS Glue version 1.0 ou antérieures, lorsque vous configurez une tâche à l'aide de la console et que vous spécifiez un type de travailleur Standard, la capacité maximale est définie et le nombre de travailleurs devient la valeur de Capacité maximale - 1. Si vous utilisez le AWS Command Line Interface (AWS CLI) ou le AWS SDK, vous pouvez spécifier le paramètre Capacité maximale, ou vous pouvez spécifier à la fois le type de travailleur et le nombre de travailleurs.

Pour les tâches de AWS Glue la version 2.0 ou ultérieure, vous ne pouvez pas spécifier de capacité maximale. Au lieu de cela, vous devez spécifier le Worker type (Type d'employé) et le Number of workers (Nombre d'employés).

`G.4X`et les types de **`G.8X`** travailleurs ne sont disponibles que pour les tâches Spark ETL AWS Glue version 3.0 ou ultérieure dans les AWS régions suivantes : USA Est (Ohio), USA

Est (Virginie du Nord), USA Ouest (Californie du Nord), USA Ouest (Oregon), Asie-Pacifique (Mumbai), Asie-Pacifique (Séoul), Asie-Pacifique (Sydney), Asie-Pacifique (Tokyo), Canada (Centre), Europe (Francfort), Europe (Irlande), Europe (Londres), Europe (Espagne), Europe (Stockholm) et Amérique du Sud (São Paulo).

`G.12X`, **`G.16X`**, et les types de **`R.8X`** travailleurs **`R.1X`** intermédiaires ne sont disponibles que pour les jobs Spark ETL AWS Glue version 4.0 ou ultérieure dans les AWS régions suivantes : USA Est (Virginie du Nord), USA Ouest (Oregon), USA Est (Ohio), Europe (Irlande) et Europe (Francfort). D'autres régions seront prises en charge dans les prochaines versions.

Nombre de travailleurs demandé

Pour la plupart des types d'employés, vous devez spécifier le nombre de travailleurs qui sont alloués lors de l'exécution de la tâche.

Marque-page de tâche

Spécifiez la manière dont AWS Glue les informations d'état sont traitées lors de l'exécution de la tâche. Ce signet peut se souvenir des données traitées précédemment, mettre à jour les informations sur l'état ou ignorer les informations sur l'état. Pour de plus amples informations, veuillez consulter the section called "Suivi des données traitées à l'aide de signets de tâche".

Mise en file d'attente des tâches

Spécifie si les exécutions de tâches sont mises en file d'attente pour être exécutées ultérieurement lorsqu'elles ne peuvent pas être exécutées immédiatement en raison de Service Quotas.

Lorsque cette case est cochée, la mise en file d'attente des tâches est activée pour les exécutions de tâches. Si cette option n'est pas cochée, les exécutions de tâches ne seront pas prises en compte pour la mise en file d'attente.

Si ce paramètre ne correspond pas à la valeur définie dans l'exécution de la tâche, la valeur du champ d'exécution de la tâche sera utilisée.

Exécution flexible

Lorsque vous configurez une tâche à l'aide de AWS Studio ou de l'API, vous pouvez spécifier une classe d'exécution de tâche standard ou flexible. Vos tâches peuvent avoir différents degrés de priorité et de sensibilité au temps. La classe d'exécution standard est idéale pour les charges de travail urgentes qui nécessitent un démarrage rapide des tâches et des ressources dédiées.

La classe d'exécution flexible convient aux tâches non urgentes telles que les tâches de pré-production, les tests et les chargements de données uniques. Les exécutions flexibles sont prises en charge pour les tâches utilisant AWS Glue la version 3.0 ou ultérieure et/ou les `G.1X` types de `G.2X` travailleurs. Les nouveaux types de travailleurs (`G.12X`, `G.16X` et `R.1X` à `R.8X`) ne permettent pas une exécution flexible.

Les exécutions de tâches flexibles sont facturées en fonction du nombre de travailleurs en cours d'exécution à un moment donné dans le temps. Le nombre de travailleurs peut être ajouté ou supprimé pour une exécution flexible en cours d'exécution. Au lieu de facturer comme un simple calcul de `Max Capacity` * `Execution Time`, chaque travailleur contribuera pour le temps qu'il a effectué pendant l'exécution du travail. La facture est la somme de (`Number of DPUs per worker` * `time each worker ran`).

Pour plus d'informations, consultez le panneau d'aide de AWS Studio, ou Tâches etExécutions de tâches.

Nombre de nouvelles tentatives

Spécifiez le nombre de fois, de 0 à 10, qui AWS Glue devrait redémarrer automatiquement la tâche en cas d'échec. Les tâches qui atteignent la limite de délai d'expiration ne sont pas redémarrées.

Délai d'expiration de la tâche

Définit le délai d'exécution maximal en minutes. La valeur maximale est de 7 jours ou 10 080 minutes. Dans le cas contraire, les tâches déclencheront une exception.

Si aucune valeur n'est indiquée, le délai d'expiration est défini par défaut sur 2 880 minutes.

Toutes les AWS Glue tâches existantes dont le délai d'expiration est supérieur à 7 jours seront définies par défaut sur 7 jours. Par exemple, si vous avez spécifié un délai d'expiration de 20 jours pour un traitement par lots, celui-ci sera arrêté le 7e jour.

> **Bonnes pratiques relatives aux délais d'expiration des tâches**
>
> Les tâches sont facturées en fonction du temps d'exécution. Pour éviter des frais inattendus, configurez des délais d'expiration adaptés à la durée d'exécution prévue de votre tâche.

Propriétés avancées

Nom du fichier de script

Un nom de script unique pour votre tâche. Il ne peut pas être nommé Tâche sans titre.

Chemin d'accès au script

L'emplacement Amazon S3 du script. Le chemin doit être de la forme `s3://bucket/prefix/path/`. Il doit se terminer par une barre oblique (/) et ne pas inclure de fichiers.

Métriques de tâche

Activez ou désactivez la création de CloudWatch métriques Amazon lors de l'exécution de cette tâche. Pour afficher les données de profilage, vous devez activer cette option. Pour en savoir plus sur l'activation et la visualisation de métriques, consultez Surveillance et débogage des tâches.

Métriques d'observabilité de la tâche

Activez la création de CloudWatch mesures d'observabilité supplémentaires lors de l'exécution de cette tâche. Pour de plus amples informations, veuillez consulter the section called "Surveillance à l'aide de métriques d'observabilité AWS Glue".

Journalisation continue

Activez la journalisation continue sur Amazon CloudWatch. Si cette option n'est pas activée, les journaux ne sont disponibles qu'une fois la tâche terminée. Pour en savoir plus, consultez the section called "Journalisation pour les AWS Glue emplois".

Interface utilisateur Spark

Permettez l'utilisation de l'interface utilisateur Spark pour la surveillance de cette tâche. Pour de plus amples informations, veuillez consulter Activation de l'interface utilisateur web Apache Spark pour les tâches AWS Glue.

Chemin d'accès aux journaux de l'interface utilisateur Spark

Le chemin pour écrire les journaux lorsque l'interface utilisateur Spark est activée.

Configuration de journalisation et de surveillance de l'interface utilisateur Spark

Choisissez l'une des options suivantes :

- Standard : écrivez des journaux en utilisant l'ID AWS Glue d'exécution du job comme nom de fichier. Activez la surveillance de l'interface utilisateur Spark dans la AWS Glue console.

- Hérité : écrivez des journaux en utilisant « spark-application-{timestamp} » comme nom de fichier. N'activez pas la surveillance de l'interface utilisateur Spark.
- Standard et hérité : écrivez des journaux à la fois dans les emplacements standard et hérités. Activez la surveillance de l'interface utilisateur Spark dans la AWS Glue console.

Simultanéité maximum

Définit le nombre maximal d'exécutions simultanées autorisées pour cette tâche. La valeur par défaut est 1. Une erreur est renvoyée lorsque ce seuil est atteint. La valeur maximale que vous pouvez spécifier est contrôlée par une limite de service. Par exemple, si une tâche est toujours en cours d'exécution lorsqu'une nouvelle instance est lancée, vous pouvez souhaiter le renvoi d'une erreur pour empêcher deux instances de la même tâche de s'exécuter simultanément.

Chemin d'accès temporaire

Indiquez l'emplacement d'un répertoire de travail dans Amazon S3 où les résultats intermédiaires temporaires sont écrits lors de l' AWS Glue exécution du script. Vérifiez qu'il n'existe pas de fichier portant le même nom que le répertoire temporaire dans le chemin d'accès. Ce répertoire est utilisé lors des opérations de AWS Glue lecture et d'écriture sur Amazon Redshift et lors de certaines AWS Glue transformations.

> **Note**
>
> AWS Glue crée un compartiment temporaire pour les tâches s'il n'en existe pas déjà un dans une région. Ce compartiment peut permettre l'accès au public. Vous pouvez soit modifier le compartiment dans Amazon S3 pour définir le bloc d'accès public, soit supprimer le compartiment ultérieurement une fois toutes les tâches de cette région terminées.

Seuil de notification de délai (minutes)

Définit le seuil (en minutes) avant l'envoi d'une notification de dépassement de délai. Vous pouvez définir ce seuil pour envoyer des notifications lorsqu'une exécution de tâche `RUNNING`, `STARTING` ou `STOPPING` prend plus de temps que le nombre de minutes prévu.

Configuration de sécurité

Choisissez une configuration de sécurité dans la liste. Une configuration de sécurité spécifie la manière dont les données de la cible Amazon S3 sont chiffrées : pas de chiffrement,

chiffrement côté serveur avec des clés gérées par AWS KMS(SSE-KMS) ou clés de chiffrement gérées par Amazon S3 (SSE-S3).

Chiffrement côté serveur

Si vous sélectionnez cette option, lorsque la tâche ETL écrit sur Amazon S3, les données sont chiffrées au repos à l'aide du chiffrement SSE-S3. Votre cible de données Amazon S3 et toutes les données qui sont écrites dans un répertoire temporaire Amazon S3 sont chiffrées. Cette option est transmise en tant que paramètre de tâche. Pour en savoir plus, consultez la section Protecting Data Using Server-Side Encryption with Amazon S3-Managed Encryption Keys (SSE-S3) (Protection des données à l'aide du chiffrement côté serveur avec des clés de chiffrement (SSE-S3) gérées par Amazon S3) dans le guide de l'utilisateur Amazon Simple Storage Service.

> **Important**
>
> Cette option est ignorée si une configuration de sécurité est spécifiée.

Utiliser le catalogue de données Glue en tant que metastore Hive

Sélectionnez cette option pour utiliser le catalogue de AWS Glue données comme métastore Hive. Le rôle IAM utilisé pour la tâche doit disposer de l'autorisation `glue:CreateDatabase`. Une base de données appelée « default » est créée dans le catalogue de données si elle n'existe pas.

Connexions

Choisissez une configuration VPC pour accéder aux sources de données Amazon S3 situées dans votre cloud privé virtuel (VPC). Vous pouvez créer et gérer une connexion réseau dans AWS Glue. Pour de plus amples informations, veuillez consulter Connexion aux données.

Bibliothèques

Chemin de bibliothèque Python, JARs chemin dépendant et chemin des fichiers référencés

Spécifiez ces options si votre script les requiert. Vous pouvez définir des chemins Amazon S3 séparés par des virgules pour ces options lorsque vous définissez la tâche. Vous pouvez remplacer ces chemins lorsque vous exécutez la tâche. Pour de plus amples informations, veuillez consulter Fournir vos propres scripts personnalisés.

Paramètres du job

Ensemble de paires clé-valeur transmises sous forme de paramètres nommés au script. Il s'agit de valeurs par défaut qui sont utilisées lors de l'exécution du script, mais vous pouvez les remplacer dans les déclencheurs ou lorsque vous exécutez la tâche. Vous devez préfixer le nom de la clé avec `--` ; par exemple : `--myKey`. Vous transmettez les paramètres de la tâche sous forme de carte lorsque vous utilisez le AWS Command Line Interface.

Pour accéder à des exemples, veuillez consulter les paramètres Python dans Transmission de paramètres Python et accès à ces paramètres dans AWS Glue.

Étiquettes

Identifiez votre tâche avec une clé d'identification et avec une valeur d'identification facultative. Lorsque les clés d'identification sont créées, elles sont en lecture seule. Utilisez des identifications sur certaines ressources pour mieux les organiser et les identifier. Pour de plus amples informations, veuillez consulter AWS tags dans AWS Glue.

Restrictions pour les tâches qui accèdent aux tables gérées par Lake Formation

Tenez compte des remarques et restrictions suivantes lorsque vous créez des tâches qui permettent de lire ou d'écrire dans des tables gérées par AWS Lake Formation :

- Les fonctions suivantes ne sont pas prises en charge dans les tâches qui accèdent à des tables comportant des filtres au niveau des cellules :
  - Marque-pages de tâche et exécution limitée
  - Prédicats pushdown
  - Prédicats de partition de catalogue côté serveur
  - enableUpdateCatalog

Modification de scripts Spark dans la consoleAWS Glue

Un script contient le code qui extrait les données des sources, les transforme et les charge dans des cibles. AWS Glue exécute un script lorsqu'il démarre une tâche.

Les scripts ETL AWS Glue peuvent être codés en Python ou Scala. Les scripts Python utilisent un langage qui est une extension du dialecte PySpark Python pour les tâches d'extraction, de transformation et de chargement (ETL). Le script contient des structures étendues pour gérer les

transformations ETL. Lorsque vous générez automatiquement la logique de code source pour votre tâche, un script est créé. Vous pouvez modifier ce script ou fournir votre propre script personnalisé pour effectuer votre travail ETL.

Pour plus d'informations sur la définition et la modification de scripts dans AWS Glue, consultez AWS Glue guide de programmation.

Bibliothèques ou fichiers supplémentaires

Si votre script nécessite des bibliothèques ou des fichiers supplémentaires, vous pouvez les spécifier comme suit :

chemin de la bibliothèque Python

Chemins Amazon Simple Storage Service (Amazon S3) séparés par des virgules vers les bibliothèques Python requises par le script.

> **Note**
>
> Seules les bibliothèques Python pures peuvent être utilisées. Les bibliothèques reposant sur des extensions C, par exemple la bibliothèque d'analyse des données Python pandas, ne sont pas encore prises en charge.

Chemin des fichiers jar dépendants

Chemins Amazon S3 séparés par des virgules vers les fichiers JAR requis par le script.

> **Note**
>
> À l'heure actuelle, seules les bibliothèques pures Java ou Scala (2.11) peuvent être utilisées.

Chemin de fichiers référencés

Chemins Amazon S3 séparés par des virgules vers les fichiers supplémentaires (par exemple, des fichiers de configuration) requis par le script.

Tâches (hérité)

Un script contient le code qui permet d'effectuer un travail ETL (extraction, transformation et chargement). Vous pouvez fournir votre propre script, ou AWS Glue peut générer un script à partir de vos indications. Pour plus d'informations sur la création de vos propres scripts, consultez Fournir vos propres scripts personnalisés.

Vous pouvez modifier un script dans la console AWS Glue. Lorsque vous modifiez un script, vous pouvez ajouter des sources, des cibles et des transformations.

Pour modifier un script

1. Connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/. Ensuite, choisissez l'onglet Jobs (Tâches).
2. Choisissez une tâche dans la liste, puis choisissez Action, Edit script (Modifier le script) pour ouvrir l'éditeur de script.

   Vous pouvez également accéder à l'éditeur de script à partir de la page des détails de la tâche. Sélectionnez l'onglet Script, puis Edit script (Modifier le script).

Éditeur de script

L'éditeur de script AWS Glue vous permet d'insérer, de modifier et de supprimer des sources, des cibles et des transformations dans votre script. L'éditeur de script affiche à la fois le script et un diagramme pour vous aider à visualiser le flux de données.

Pour créer un diagramme pour le script, choisissez Générer un diagramme. AWS Glue utilise des lignes d'annotation dans le script commençant par # # pour afficher le schéma. Pour représenter correctement votre script dans le diagramme, vous devez conserver la synchronisation entre les paramètres dans les annotations et les paramètres dans le code Apache Spark.

L'éditeur de script vous permet d'ajouter des modèles de code à chaque fois que votre curseur est positionné dans le script. Dans la partie supérieure de l'éditeur, choisissez l'une des options suivantes :

- Pour ajouter une table source au script, choisissez Source (Source).
- Pour ajouter une table cible au script, choisissez Target (Cible).
- Pour ajouter un emplacement cible au script, choisissez Target location (Emplacement cible).

- Pour ajouter une transformation au script, choisissez Transform (Transformation). Pour plus d'informations sur les fonctions appelées dans votre script, consultez Programmez les scripts ETL AWS Glue dans PySpark.
- Pour ajouter une transformation Spigot au script, choisissez Spigot.

Dans le code inséré, modifiez l'élément `parameters` dans les annotations et dans le code Apache Spark. Par exemple, si vous ajoutez une transformation Spigot, vérifiez que l'élément `path` est remplacé dans la ligne d'annotation `@args` et dans la ligne de code `output`.

L'onglet Logs (Journaux) présente les journaux associés à votre travail pendant son exécution. Les 1 000 lignes les plus récentes sont affichées.

L'onglet Schema (Schéma) présente le schéma des sources et des cibles sélectionnées, lorsqu'elles sont disponibles dans Data Catalog.

Suivi des données traitées à l'aide de signets de tâche

AWS Glue effectue le suivi des données qui ont déjà été traitées au cours d'une exécution précédente d'une tâche ETL par la persistance des informations d'état à partir de l'exécution de la tâche. Ces informations d'état permanentes sont appelées marque-pages de tâches. Les signes de tâche permettent à AWS Glue de conserver des informations d'état et d'empêcher le retraitement des anciennes données. Avec les marque-pages de tâche, vous pouvez traiter de nouvelles données lors de la réexécution sur un intervalle planifié. Un marque-page de tâche se compose des états des différents éléments de tâches, tels que les sources, les transformations et les cibles. Par exemple, votre tâche ETL peut lire de nouvelles partitions dans un fichier Amazon S3. AWS Glue suit les partitions qui ont été traitées avec succès par la tâche pour éviter le traitement des doublons et les doublons dupliqués dans le magasin de données cible de la tâche.

Les marque-pages de tâche sont implémentés pour les sources de données JDBC, la transformation Relationalize et certaines sources Amazon Simple Storage Service (Amazon S3). Le tableau suivant répertorie les formats source Amazon S3 pris en charge par AWS Glue pour les marque-pages de tâche.

Version de l'AWS Glue	Formats source Amazon S3
Version 0.9	JSON, CSV, Apache Avro, XML
Version 1.0 et ultérieure	JSON, CSV, Apache Avro, XML, Parquet, ORC

Pour plus d'informations sur les versions AWS Glue, veuillez consulter Définition des propriétés des tâches Spark.

La fonction de signets de tâches comporte des fonctionnalités supplémentaires lorsqu'elle est accessible par le biais de AWS Glue scripts. Lorsque vous parcourez le script que vous avez généré, vous pouvez voir des contextes de transformation liés à cette fonctionnalité. Pour de plus amples informations, veuillez consulter the section called "Utilisation des marque-pages de tâche".

Rubriques

- Utilisation des marque-pages de tâche dans AWS Glue
- Détails opérationnels de la fonctionnalité de signets de tâche

Utilisation des marque-pages de tâche dans AWS Glue

L'option de marque-page de tâche est transmise sous forme de paramètre au démarrage de la tâche. Le tableau suivant décrit les options de définition des marque-pages de tâche sur la console AWS Glue.

Marque-page de tâche	Description
Enable	La tâche met à jour l'état après une exécution pour suivre les données traitées précédemment. Si votre tâche a une source avec prise en charge du marque-page de tâche, elle suit les données traitées et, lorsqu'une tâche est exécutée, elle traite les nouvelles données depuis le dernier point de contrôle.
Désactiver	Les marque-pages de tâche ne sont pas utilisés et la tâche traite toujours la totalité du jeu de données. Vous êtes responsable de la gestion des résultats des précédentes exécutions de tâches. Il s'agit de l'option par défaut.
Pause	Traitez les données progressives depuis la dernière exécution réussie ou les données de la plage identifiée par les sous-options suivantes, sans mettre à jour l'état du dernier marque-page. Vous êtes responsable de la gestion des résultats des précédentes exécutions de tâches. Les deux sous-options sont les suivantes :

Marque-page de tâche	Description
	• job-bookmark-from<from-value>est l'ID d'exécution qui représent e toutes les entrées traitées jusqu'à la dernière exécution réussie précédente, y compris l'ID d'exécution spécifié. L'entrée correspon dante est ignorée. • job-bookmark-to<to-value>est l'ID d'exécution qui représente toutes les entrées traitées jusqu'à la dernière exécution réussie précédente, y compris l'ID d'exécution spécifié. L'entrée correspondante, à l'exclusi on de l'entrée identifiée par <from-value>, est traitée par la tâche. Toute entrée postérieure à cette entrée est également exclue pour traitement. L'état du marque-page de tâche n'est pas mis à jour lorsque ce jeu d'options est spécifié. Les sous-options sont facultatives, mais lorsqu'elles sont utilisées, les deux sous-options doivent être fournies.

Pour plus d'informations sur les paramètres transmis à une tâche sur la ligne de commande, et plus particulièrement pour les marque-pages de tâche, consultez Utilisation des paramètres des tâches dans les tâches AWS Glue.

Pour les sources d'entrée Amazon S3, les marque-pages de tâche de AWS Glue vérifient l'heure de la dernière modification des objets afin de contrôler quels objets doivent être retraités. Si les données de la source d'entrée ont été modifiées depuis votre dernière exécution de tâche, les fichiers sont traités de nouveau lorsque vous exécutez la tâche à nouveau.

Pour les sources JDBC, les règles suivantes s'appliquent :

- Pour chaque table, AWS Glue utilise une ou plusieurs colonnes comme clés de marque-page pour déterminer les données nouvelles et les données traitées. Les touches de marque-page se combinent pour former une seule clé composée.
- AWS Glue utilise par défaut la clé primaire comme clé de signet, à condition qu'elle augmente ou diminue de manière séquentielle (sans aucun écart).

- Vous pouvez spécifier les colonnes à utiliser comme clés de signet dans votre AWS Glue script. Pour plus d'informations sur l'utilisation des signets Job dans les AWS Glue scripts, consultezthe section called "Utilisation des marque-pages de tâche".
- AWS Glue ne prend pas en charge l'utilisation de colonnes dont les noms sont sensibles à la casse comme clés de signet de tâche.

Vous pouvez restaurer les marque-pages de tâche pour les tâches AWS Glue Spark ETL sur une exécution de tâche précédente. Les opérations de remplacement automatique de données sont mieux prises en charge : quand un marque-page de tâche déjà exécutée est restauré sur une exécution de tâche précédente, les exécutions de tâche ultérieures ne retraitent que les données à partir de l'exécution de tâche marquée par le marque-page.

Si vous avez l'intention de retraiter toutes les données à l'aide de la même tâche, réinitialisez le marque-page de tâche. Pour réinitialiser l'état d'un marque-page de tâche, utilisez la console AWS Glue, l'opération d'API ResetJobBookmark action (Python : reset_job_bookmark) ou le AWS CLI. Par exemple, entrez la commande suivante à l'aide de l' AWS CLI :

```
aws glue reset-job-bookmark --job-name my-job-name
```

Lorsque vous restaurez ou réinitialisez un marque-page, AWS Glue ne nettoie pas les fichiers cible, car il peut y avoir plusieurs cibles et les cibles ne sont pas suivies avec des marque-pages de tâche. Seuls les fichiers source sont suivis à l'aide de marque-pages de tâche. Vous pouvez créer différentes cibles de sortie lors de la restauration et du retraitement des fichiers source pour éviter les doublons de données dans votre sortie.

AWS Glue assure le suivi des marque-pages de tâche par tâche. Si vous supprimez une tâche, le marque-page de tâche est supprimé.

Dans certains cas, il se peut que vous ayez activé les marque-pages de tâche AWS Glue, mais que votre tâche ETL retraite les données qui ont déjà été traitées lors d'une exécution précédente. Pour plus d'informations sur la résolution des causes courantes de cette erreur, consultez Résolution des erreurs de configuration courantes de Glue.

Détails opérationnels de la fonctionnalité de signets de tâche

Cette section décrit plus de détails opérationnels sur l'utilisation des marque-pages de tâche.

Les marque-pages de tâche stockent les états d'une tâche. Chaque instance de l'état est identifiée par un nom et un numéro de version. Lorsqu'un script appelle `job.init`, il extrait son état et obtient toujours la dernière version. Au sein d'un état, il y a plusieurs éléments d'état, qui sont spécifiques à chaque source, la transformation et l'instance de récepteur dans le script. Ces éléments d'état sont identifiés par un contexte de transformation qui est attaché à l'élément correspondant (source, transformation ou récepteur) dans le script. Les éléments d'état sont enregistrés de manière atomique lorsque `job.commit` est appelé depuis le script utilisateur. Le script obtient le nom de la tâche et l'option de contrôle pour les marque-pages de tâches depuis les arguments.

Les éléments d'état dans le marque-page de tâche sont les données de la source, de la transformation ou des données spécifiques au récepteur. Par exemple, supposons que vous souhaitiez lire des données progressives à partir d'un emplacement Amazon S3 qui est constamment écrit par une tâche ou un processus en amont. Dans ce cas, le script doit déterminer ce qui a été traité jusqu'à présent. L'implémentation du marque-page de tâche pour la source Amazon S3 enregistre des informations pour que, lors de la prochaine exécution de la tâche, elle puisse filtrer uniquement les nouveaux objets à l'aide des informations enregistrées et recalculer l'état de la prochaine exécution de la tâche. Un horodatage est utilisé pour filtrer les nouveaux fichiers.

Outre les éléments d'état, les marque-pages de tâche ont un nombre d'exécutions, un nombre de tentatives et un numéro de version. Le nombre d'exécutions suit l'exécution de la tâche et le nombre de tentatives enregistre les tentatives d'une exécution de tâche. Le nombre d'exécutions d'une tâche est un nombre croissant de façon monotone incrémenté pour chaque exécution réussie. Le nombre de tentatives suit les tentatives de chaque exécution, et il est incrémenté uniquement lorsqu'il y a une exécution après une tentative en échec. Le numéro de version augmente de façon monotone et effectue le suivi des mises à jour d'un marque-page de tâche.

Dans la base de données de AWS Glue service, les états des signets pour toutes les transformations sont stockés ensemble sous forme de paires clé-valeur :

```
{
  "job_name" : ...,
  "run_id": ...,
  "run_number": ..,
  "attempt_number": ...
  "states": {
    "transformation_ctx1" : {
      bookmark_state1
    },
    "transformation_ctx2" : {
      bookmark_state2
```

```
    }
  }
}
```

Bonnes pratiques

Les bonnes pratiques suivantes sont recommandées pour utiliser des signets de tâche.

- Ne modifiez pas la propriété de la source de données lorsque le marque-page est activé. Par exemple, une source de données 0 pointe vers un chemin d'entrée A Amazon S3, et la tâche est en cours de lecture à partir d'une source exécutée pendant plusieurs tours avec le marque-page activé. Si vous remplacez le chemin d'entrée de datasource0 par le chemin B d'Amazon S3 sans le modifier`transformation_ctx`, la AWS Glue tâche utilisera l'ancien état des favoris enregistré. Cela entraînera l'absence ou l'omission de fichiers dans le chemin d'entrée B, comme cela AWS Glue supposerait que ces fichiers ont été traités lors d'exécutions précédentes.
- Utiliser un tableau de catalogue avec des marque-pages pour une meilleure gestion des partitions. Les marque-pages fonctionnent à la fois pour les sources de données du catalogue de données ou à partir d'options. Cependant, il est difficile de créer de remove/add nouvelles partitions avec l'approche des options from. L'utilisation d'un tableau de catalogue avec des crawlers peut fournir une meilleure automatisation pour suivre les partitions nouvellement ajoutées et vous offre la flexibilité nécessaire pour sélectionner des partitions particulières avec un prédicat pushdown.
- Utiliser le listeur de fichiers Amazon S3 AWS Glue pour les jeux de données volumineux. Un marque-page répertorie tous les fichiers sous chaque partition d'entrée et effectue le filtrage. Par conséquent, s'il y a trop de fichiers sous une seule partition, le marque-page peut être exécuté dans le pilote OOM. Utilisez le listeur de fichiers AWS Glue Amazon S3 pour éviter de répertorier tous les fichiers en mémoire en une seule fois.

Stockage des données réorganisées Spark

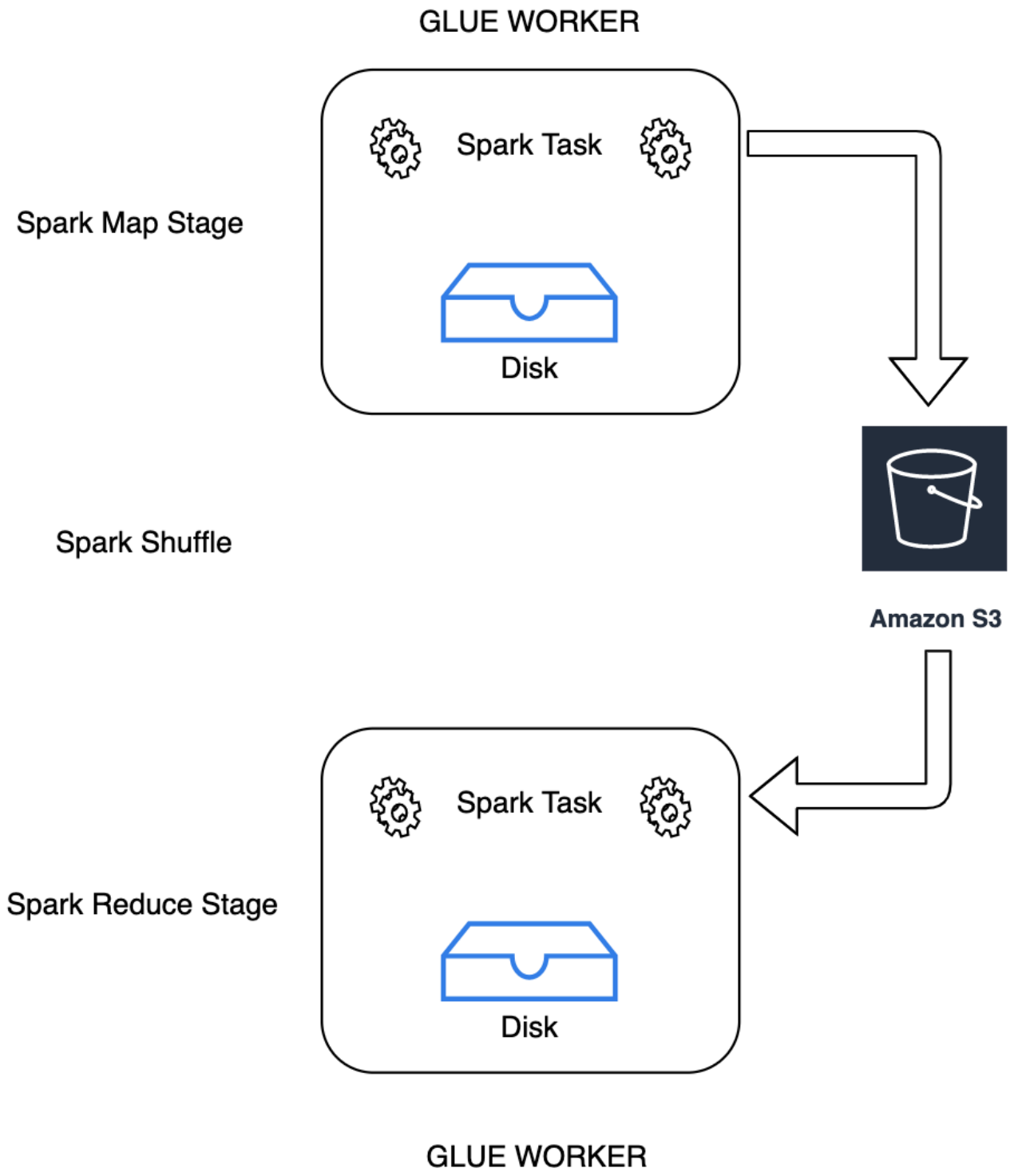
La réorganisation est une étape importante d'une tâche Spark chaque fois que les données sont réarrangées entre des partitions. Elle est nécessaire parce que de larges transformations telles que `join`, `groupByKey`, `reduceByKey` et `repartition` ont besoin d'informations provenant d'autres partitions pour terminer le traitement. Spark rassemble les données requises de chaque partition et les combine dans une nouvelle partition. Lors d'une réorganisation, les données sont écrites sur le disque et transférées via le réseau. Par conséquent, l'opération de réorganisation est liée à la capacité du disque local. Spark émet une erreur `No space left on device` ou

`MetadataFetchFailedException` lorsqu'il n'y a pas assez d'espace disque sur le programme d'exécution et qu'il n'y a pas de récupération.

> **Note**
>
> AWS Glue Le plugin Spark shuffle avec Amazon S3 n'est pris en charge que pour les tâches AWS Glue ETL.

Solution

Avec AWS Glue, vous pouvez désormais utiliser Amazon S3 pour stocker des données de réorganisation Spark. Amazon S3 est un service de stockage d'objets qui offre une évolutivité, une disponibilité des données, une sécurité et des performances de pointe. Cette solution désagrège le calcul et le stockage pour vos tâches Spark, et offre une élasticité totale et un stockage économique de la réorganisation, ce qui vous permet d'exécuter vos applications les plus complexes de manière fiable.



Nous introduisons un nouveau plug-in Cloud Shuffle Storage pour Apache Spark afin d'utiliser Amazon S3. Vous pouvez activer la réorganisation Amazon S3 pour exécuter vos tâches AWS Glue de manière fiable, sans échec, si elles sont liées par la capacité du disque local pour les opérations de réorganisation volumineuses. Dans certains cas, une réorganisation vers Amazon S3

est légèrement plus lente que vers un disque local (ou EBS) si vous avez un grand nombre de petites partitions ou de fichiers réorganisés écrits dans Amazon S3.

Conditions préalables à l'utilisation du plug-in de stockage de réorganisation du cloud.

Pour utiliser le plug-in de stockage Cloud Shuffle avec des tâches AWS Glue ETL, vous avez besoin des éléments suivants :

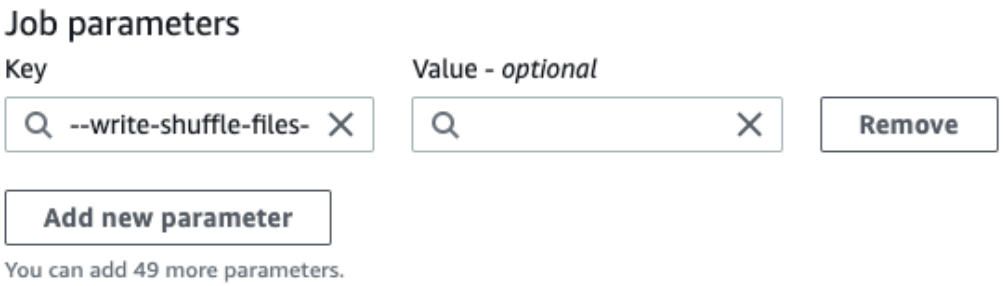
- Un compartiment Amazon S3 situé dans la même région que celle de votre exécution de tâche, pour stocker la réorganisation intermédiaire et les données déversées. Le préfixe Amazon S3 du stockage de réorganisation peut être spécifié avec `--conf spark.shuffle.glue.s3ShuffleBucket=s3://`*`shuffle-bucket`*`/`*`prefix`*`/`, comme dans l'exemple suivant :

  ```
  --conf spark.shuffle.glue.s3ShuffleBucket=s3://glue-shuffle-123456789-us-east-1/glue-shuffle-data/
  ```

- Définissez les stratégies de cycle de vie du stockage Amazon S3 sur le préfixe (tel que `glue-shuffle-data`), car le gestionnaire de réorganisation ne nettoie pas les fichiers une fois la tâche terminée. La réorganisation intermédiaire et les données déversées doivent être supprimées une fois la tâche terminée. Les utilisateurs peuvent définir des stratégies de cycle de vie court pour le préfixe. Les instructions de configuration d'une stratégie de cycle de vie Amazon S3 sont disponibles dans la section Configuration du cycle de vie d'un compartiment dans le Guide de l'utilisateur Amazon Simple Storage Service.

Utilisation du gestionnaire de shuffle AWS Glue Spark depuis la console AWS

Pour configurer le gestionnaire de shuffle AWS Glue Spark à l'aide de la AWS Glue console ou de AWS Glue Studio lors de la configuration d'une tâche : choisissez le paramètre de tâche -- write-shuffle-files-to -s3 pour activer le shuffle Amazon S3 pour la tâche.



Utilisation du plug-in de réorganisation Spark AWS Glue

Les paramètres de tâche suivants activent et affinent le gestionnaire de réorganisation AWS Glue. Ces paramètres étant des indicateurs, les valeurs fournies ne sont pas prises en compte.

- `--write-shuffle-files-to-s3` : l'indicateur principal, qui active le gestionnaire de réorganisation AWS Glue Spark pour utiliser des compartiments Amazon S3 pour écrire et lire des données de réorganisation. Lorsque l'indicateur n'est pas spécifié, le gestionnaire de réorganisation n'est pas utilisé.
- `--write-shuffle-spills-to-s3` – (Pris en charge uniquement dans AWS Glue version 2.0). Indicateur facultatif qui vous permet de décharger des fichiers de déversement dans des compartiments Amazon S3, pour une résilience supplémentaire de votre tâche Spark. Ceci n'est requis que pour les charges de travail volumineuses qui déversent beaucoup de données sur le disque. Lorsque l'indicateur n'est pas spécifié, aucun fichier de déversement intermédiaire n'est écrit.
- `--conf spark.shuffle.glue.s3ShuffleBucket=s3://<shuffle-bucket>` – autre indicateur facultatif spécifiant le compartiment Amazon S3 dans lequel vous écrivez les fichiers de réorganisation. Par défaut, `--TempDir` /shuffle-data. AWS Glue La version 3.0+ prend en charge l'écriture de fichiers shuffle dans plusieurs compartiments en spécifiant des compartiments avec un délimiteur par virgule, comme dans. `--conf spark.shuffle.glue.s3ShuffleBucket=s3://`*`shuffle-bucket-1`*`/`*`prefix`*`,s3://`*`shuffle-bucket-2`*`/`*`prefix`*`/` L'utilisation de plusieurs compartiments améliore les performances.

Vous devez fournir des paramètres de configuration de sécurité pour activer le chiffrement au repos des données de réorganisation. Pour plus d'informations sur les configurations de sécurité, consultez the section called "Configuration du chiffrement ". AWS Glue prend en charge toutes les autres configurations de réorganisation fournies par Spark.

Binaires logiciels pour le plug-in Cloud Shuffle Storage

Vous pouvez également télécharger les fichiers binaires du plug-in Cloud Shuffle Storage pour Apache Spark sous la licence Apache 2.0 et l'exécuter dans n'importe quel environnement Spark. Le nouveau plugin est compatible avec out-of-the Amazon S3 et peut également être facilement configuré pour utiliser d'autres formes de stockage dans le cloud, telles que Google Cloud Storage et Microsoft Azure Blob Storage. Pour plus d'informations, consultez Cloud Shuffle Storage Plugin for Apache Spark (Plug-in Cloud Shuffle Storage pour Apache Spark).

Remarques et limitations

Ci-dessous des remarques ou limitations concernant le gestionnaire de réorganisation AWS Glue.

- AWS Glue le gestionnaire de shuffle ne supprime pas automatiquement les fichiers de données de shuffle (temporaires) stockés dans votre compartiment Amazon S3 une fois la tâche terminée. Pour garantir la protection des données, suivez les instructions fournies dans Conditions préalables à l'utilisation du plug-in de stockage de réorganisation du cloud. avant d'activer le plug-in Cloud Shuffle Storage.
- Vous pouvez utiliser cette fonction si vos données sont biaisées.

Cloud Shuffle Storage Plugin pour Apache Spark

Cloud Shuffle Storage Plugin est un plug-in Apache Spark compatible avec l'API `ShuffleDataIO` qui permet de stocker des données aléatoires sur des systèmes de stockage cloud (tels qu'Amazon S3). Il vous permet de compléter ou de remplacer la capacité de stockage sur disque local pour les opérations de remaniement de grande envergure, généralement déclenchées par des transformations telles que `joinreduceByKey`, `groupByKey` et `repartition` dans vos applications Spark, réduisant ainsi les défaillances ou les price/performance dislocations fréquentes de vos tâches et pipelines d'analyse de données sans serveur.

AWS Glue

Sur les versions 3.0 et 4.0 de AWS Glue, le plug-in est préinstallé et prêt à autoriser la lecture aléatoire dans Amazon S3 sans aucune étape supplémentaire. Pour plus d'informations, consultez AWS Glue Spark shuffle plugin with Amazon S3 afin d'activer cette fonctionnalité pour vos applications Spark.

Autres environnements Spark

Le plug-in exige que les configurations Spark suivantes soient définies sur d'autres environnements Spark :

- `--conf spark.shuffle.sort.io.plugin.class=com.amazonaws.spark.shuffle.io.cloud.Choppe` indique à Spark d'utiliser ce plug-in pour Shuffle IO.
- `--conf spark.shuffle.storage.path=s3://`*`bucket-name`*`/`*`shuffle-file-dir`* : chemin de stockage de vos fichiers de lecture aléatoire.

> **Note**
>
> Le plug-in remplace une classe principale de Spark. Par conséquent, le fichier jar du plug-in doit être chargé avant les fichiers jar de Spark. Pour ce faire, utilisez `userClassPathFirst` dans des environnements YARN sur site si le plug-in est utilisé hors de AWS Glue.

Création d'une offre groupée du plug-in pour vos applications Spark

Vous pouvez intégrer le plug-in à vos applications et distributions Spark (versions 3.1 et supérieures) en ajoutant la dépendance du plug-in dans votre fichier `pom.xml` Maven, tout en développant vos applications Spark en local. Pour plus d'informations sur les versions du plug-in et de Spark, consultez Versions du plug-in.

```
<repositories>
   ...
    <repository>
        <id>aws-glue-etl-artifacts</id>
        <url>https://aws-glue-etl-artifacts.s3.amazonaws.com/release/ </url>
    </repository>
</repositories>
...
<dependency>
    <groupId>com.amazonaws</groupId>
    <artifactId>chopper-plugin</artifactId>
    <version>3.1-amzn-LATEST</version>
</dependency>
```

Vous pouvez également télécharger les fichiers binaires directement depuis les artefacts AWS Glue Maven et les inclure dans votre application Spark comme suit.

```
#!/bin/bash
sudo wget -v https://aws-glue-etl-artifacts.s3.amazonaws.com/release/com/amazonaws/
chopper-plugin/3.1-amzn-LATEST/chopper-plugin-3.1-amzn-LATEST.jar -P /usr/lib/spark/
jars/
```

Exemple d'utilisation de spark-submit

```
spark-submit --deploy-mode cluster \
--conf spark.shuffle.storage.s3.path=s3://<ShuffleBucket>/<shuffle-dir> \
--conf spark.driver.extraClassPath=<Path to plugin jar> \
```

```
--conf spark.executor.extraClassPath=<Path to plugin jar> \
--class <your test class name> s3://<ShuffleBucket>/<Your application jar> \
```

Configurations facultatives

Il s'agit de valeurs de configuration facultatives qui contrôlent le comportement de lecture aléatoire d'Amazon S3.

- `spark.shuffle.storage.s3.enableServerSideEncryption`: Enable/disable S3 SSE pour les fichiers shuffle et spill. La valeur par défaut est `true`.
- `spark.shuffle.storage.s3.serverSideEncryption.algorithm` : algorithme SSE à utiliser. La valeur par défaut est `AES256`.
- `spark.shuffle.storage.s3.serverSideEncryption.kms.key` : ARN de la clé KMS lorsque SSE aws:kms est activé.

Outre ces configurations, vous devrez peut-être définir des configurations telles que `spark.hadoop.fs.s3.enableServerSideEncryption` et d'autres configurations spécifiques à l'environnement pour garantir que le chiffrement approprié est appliqué à votre cas d'utilisation.

Versions du plug-in

Ce plugin est pris en charge pour les versions de Spark associées à chaque AWS Glue version. Le tableau suivant indique la AWS Glue version, la version de Spark et la version du plugin associée avec l'emplacement Amazon S3 du binaire logiciel du plugin.

Version de l'AWS Glue	Version de Spark	Version du plug-in	Emplacement Amazon S3
3.0	3.1	3.1-amzn-LATEST	s3 ://aws-glue-etl-artifacts/release/com/amazonaws/chopper-plugin/3.1-amzn-0/chopper-Plugin-3.1-amzn-Latest.jar
4.0	3.3	3.3-amzn-LATEST	s3 ://aws-glue-etl-artifacts/release/com/amazonaws/choppe

Version de l'AWS Glue	Version de Spark	Version du plug-in	Emplacement Amazon S3
			r-plugin/3.3-amzn-0/chopper-Plugin-3.3-amzn-latest.jar

Licence

Le fichier binaire logiciel de ce plug-in est concédé sous licence dans le cadre de la licence Apache-2.0.

Surveillance des tâches Spark AWS Glue

Rubriques

- Spark Metrics est disponible dans AWS Glue Studio
- Surveillance des tâches à l'aide de l'interface utilisateur web Apache Spark
- Surveillance à l'aide AWS Glue d'informations sur l'exécution des tâches
- Surveillance avec Amazon CloudWatch
- Surveillance et débogage des tâches

Spark Metrics est disponible dans AWS Glue Studio

L'onglet Metrics (Métriques) montre les métriques collectées lorsqu'une tâche s'exécute et que le profilage est activé. Les graphiques suivants sont affichés dans les tâches Spark :

- Déplacement de données ETL
- Profil de la mémoire : pilote et programmes d'exécution

Choisissez View Additional Metrics (Afficher des métriques supplémentaires) pour afficher les graphiques suivants :

- Déplacement de données ETL
- Profil de la mémoire : pilote et programmes d'exécution
- Remaniement de données sur les programmes d'exécution
- Chargement de l'UC : pilote et programmes d'exécution

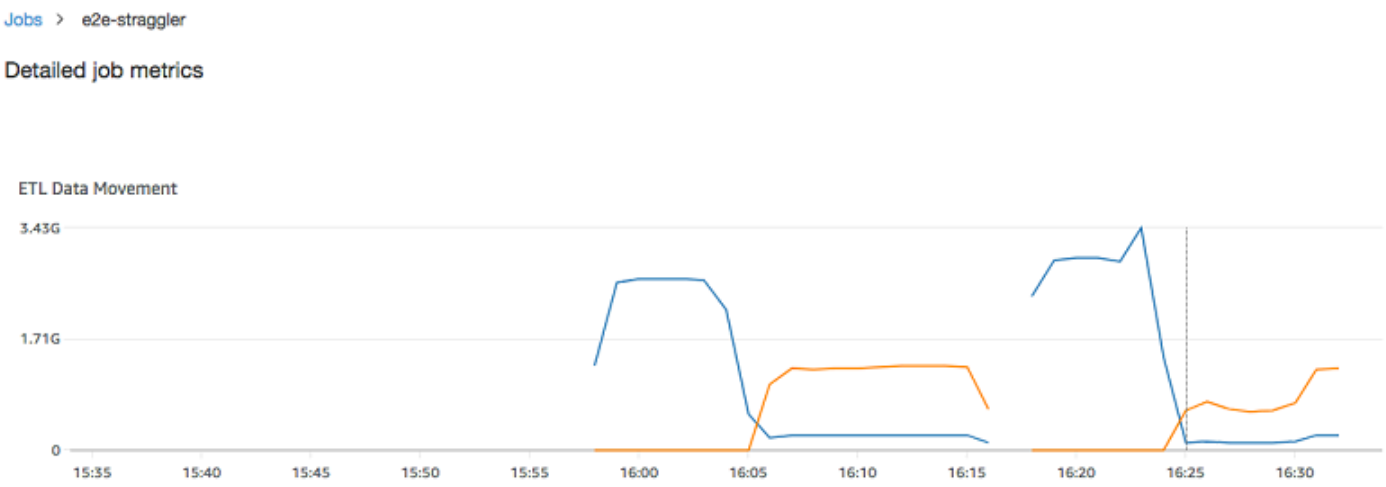
- Exécution de tâche : programmes d'exécution actifs, étapes achevées et nombre de programmes d'exécution maximum nécessaires

Les données de ces graphiques sont transférées vers CloudWatch des métriques si la tâche est configurée pour collecter des métriques. Pour plus d'informations sur l'activation de métriques et l'interprétation de graphiques, veuillez consulter Surveillance et débogage des tâches.

Example Graphique du déplacement de données ETL

Le graphique du Déplacement de données ETL montre les métriques suivantes :

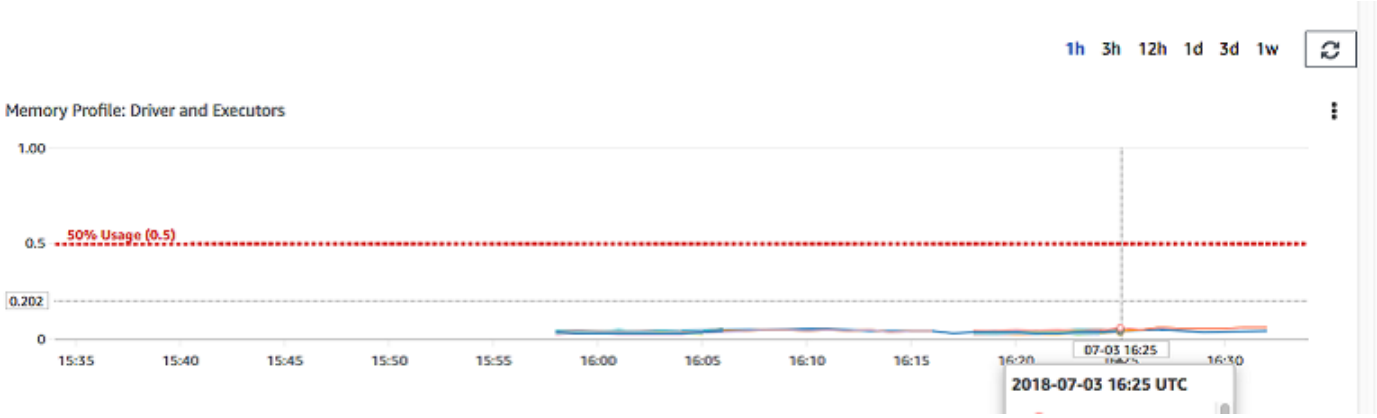
- Le nombre d'octets lus à partir d'Amazon S3 par tous les programmes d'exécution —`glue.ALL.s3.filesystem.read_bytes`
- Le nombre d'octets écrits à partir d'Amazon S3 par tous les programmes d'exécution – `glue.ALL.s3.filesystem.write_bytes`



Example Graphique du profil de mémoire

Le graphique du profil de la mémoire montre les métriques suivantes :

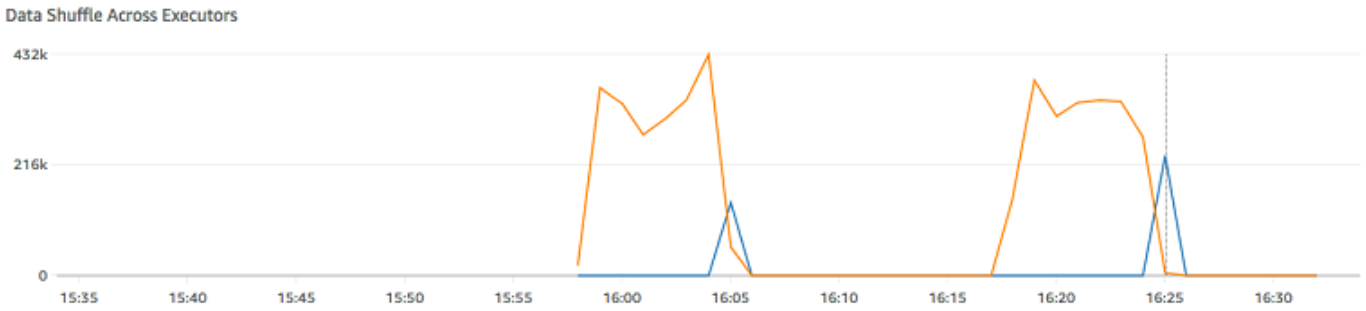
- La portion de mémoire utilisée par la pile de la JVM pour ce pilote (mise à l'échelle : 0-1) par le pilote, un programme d'exécution identifié par executorId ou tous les exécuteurs –
  - `glue.driver.jvm.heap.usage`
  - `glue.executorId.jvm.heap.usage`
  - `glue.ALL.jvm.heap.usage`



Example Graphique de remaniement de données sur les programmes d'exécution

Le graphique de remaniement de données sur les programmes d'exécution montre les métriques suivantes :

- Le nombre d'octets lus par tous les programmes d'exécution pour remanier des données sur ceux-ci - `glue.driver.aggregate.shuffleLocalBytesRead`
- Le nombre d'octets écrits par tous les programmes d'exécution pour remanier des données sur ceux-ci - `glue.driver.aggregate.shuffleBytesWritten`

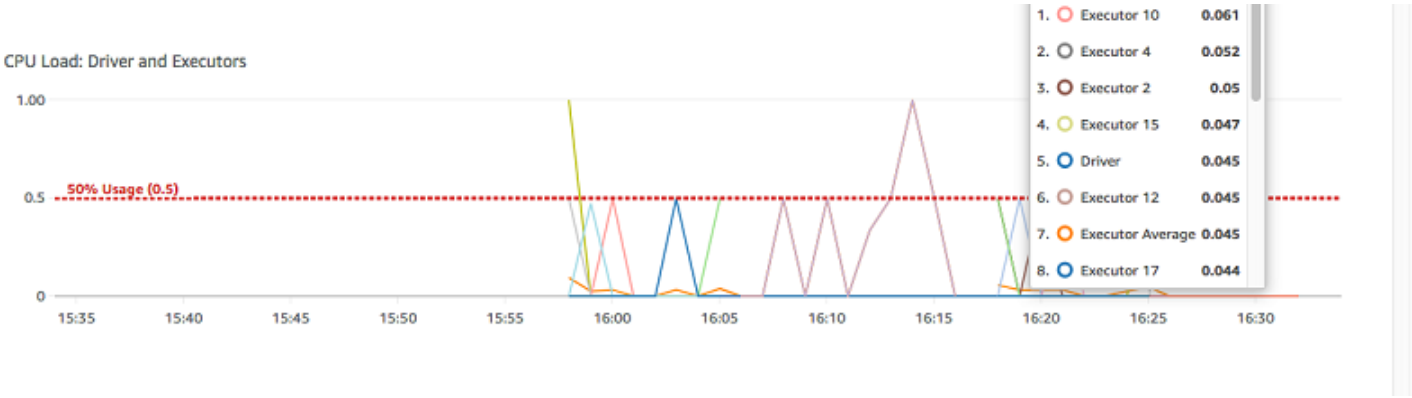


Example Graphique de chargement de l'UC

Le graphique de chargement de l'UC montre les métriques suivantes :

- La fraction de la charge du système CPU utilisée (mise à l'échelle : 0-1) par le pilote, un programme d'exécution identifié par executorId ou tous les exécuteurs –
  - `glue.driver.system.cpuSystemLoad`
  - `glue.executorId.system.cpuSystemLoad`

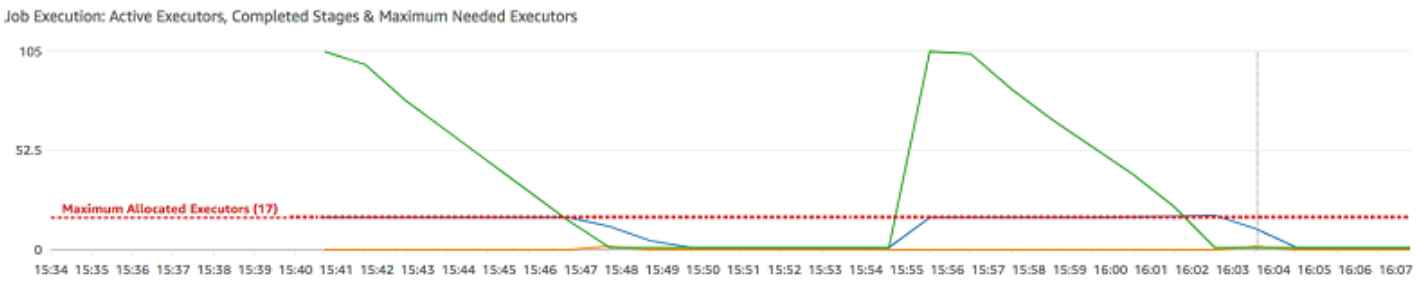
- glue.ALL.system.cpuSystemLoad



Example Graphique d'exécution de tâche

Le graphique d'exécution de tâche montre les métriques suivantes :

- Le nombre de programmes d'exécution actifs en cours d'exécution - glue.driver.ExecutorAllocationManager.executors.numberAllExecutors
- Le nombre d'étapes terminées - glue.aggregate.numCompletedStages
- Le nombre maximum de programmes d'exécution nécessaires - glue.driver.ExecutorAllocationManager.executors.numberMaxNeededExecu



Surveillance des tâches à l'aide de l'interface utilisateur web Apache Spark

Vous pouvez utiliser l'interface utilisateur web Apache Spark pour surveiller et déboguer les tâches ETL AWS Glue en cours d'exécution sur le système de tâches AWS Glue, ainsi que les applications Spark en cours d'exécution sur les points de terminaison de développement AWS Glue. L'interface utilisateur Spark vous permet de vérifier les éléments suivants pour chaque tâche :

- La chronologie des événements de chaque phase Spark

- Un graphe orienté acyclique (DAG) de la tâche
- Les plans physiques et logiques des requêtes SparkSQL
- Les variables environnementales Spark sous-jacentes pour chaque tâche

Pour plus d'informations sur l'utilisation de l'interface utilisateur web Spark, consultez l'Interface utilisateur web dans la documentation Spark. Pour obtenir des conseils sur la façon d'interpréter les résultats de l'interface utilisateur Spark afin d'améliorer les performances de votre tâche, consultez les meilleures pratiques en matière de réglage des performances AWS Glue pour les tâches Apache Spark dans le guide AWS prescriptif.

Vous pouvez voir l'interface utilisateur de Spark dans la AWS Glue console. Ceci est disponible lorsqu'une AWS Glue tâche s'exécute sur des versions AWS Glue 3.0 ou ultérieures avec des journaux générés au format standard (plutôt qu'ancien), qui est le format par défaut pour les nouvelles tâches. Si vos fichiers journaux sont supérieurs à 0,5 Go, vous pouvez activer la prise en charge des journaux progressifs pour les exécutions de tâches sur les versions AWS Glue 4.0 ou ultérieures afin de simplifier l'archivage, l'analyse et le dépannage des journaux.

Vous pouvez activer l'interface utilisateur Spark à l'aide de la AWS Glue console ou du AWS Command Line Interface (AWS CLI). Lorsque vous activez l'interface utilisateur Spark, les tâches ETL AWS Glue et les applications Spark sur les points de terminaison de développement AWS Glue peuvent sauvegarder les journaux d'événements Spark dans un emplacement que vous spécifiez dans Amazon Simple Storage Service (Amazon S3). Vous pouvez utiliser les journaux d'événements sauvegardés dans Amazon S3 avec l'interface utilisateur Spark à la fois en temps réel, lorsque la tâche est en cours de fonctionnement, et une fois celle-ci terminée. Tant que les journaux restent dans Amazon S3, l'interface utilisateur Spark de la AWS Glue console peut les consulter.

Permissions

Pour utiliser l'interface utilisateur Spark dans la AWS Glue console, vous pouvez utiliser `UseGlueStudio` ou ajouter tous les services individuels APIs. Tous APIs sont nécessaires pour utiliser complètement l'interface utilisateur Spark, mais les utilisateurs peuvent accéder aux fonctionnalités de SparkUI en ajoutant son service APIs dans leur autorisation IAM pour un accès détaillé.

`RequestLogParsing` est la plus essentielle, car elle effectue l'analyse des journaux. Le reste APIs est destiné à la lecture des données analysées respectives. Par exemple, `GetStages` permet d'accéder aux données relatives à toutes les étapes d'une tâche Spark.

La liste des services Spark UI APIs `UseGlueStudio` mappés se trouve ci-dessous dans l'exemple de politique. La politique ci-dessous permet d'utiliser uniquement les fonctionnalités de l'interface utilisateur de Spark. Pour ajouter d'autres autorisations, telles qu'Amazon S3 et IAM, consultez la section Création de politiques IAM personnalisées pour. AWS Glue Studio

La liste des services Spark UI APIs mappés se `UseGlueStudio` trouve ci-dessous dans l'exemple de politique. Lorsque vous utilisez une API de service de l'interface utilisateur de Spark, utilisez l'espace de noms suivant : `glue:<ServiceAPI>`.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AllowGlueStudioSparkUI",
      "Effect": "Allow",
      "Action": [
        "glue:RequestLogParsing",
        "glue:GetLogParsingStatus",
        "glue:GetEnvironment",
        "glue:GetJobs",
        "glue:GetJob",
        "glue:GetStage",
        "glue:GetStages",
        "glue:GetStageFiles",
        "glue:BatchGetStageFiles",
        "glue:GetStageAttempt",
        "glue:GetStageAttemptTaskList",
        "glue:GetStageAttemptTaskSummary",
        "glue:GetExecutors",
        "glue:GetExecutorsThreads",
        "glue:GetStorage",
        "glue:GetStorageUnit",
        "glue:GetQueries",
        "glue:GetQuery"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
```

```
}
```

Limitations

- L'interface utilisateur Spark de la AWS Glue console n'est pas disponible pour les exécutions de tâches effectuées avant le 20 novembre 2023 car elles sont dans l'ancien format de journal.
- L'interface utilisateur Spark de la AWS Glue console prend en charge les journaux AWS Glue évolutifs pour la version 4.0, tels que ceux générés par défaut dans les tâches de streaming. La somme maximale de tous les fichiers d'événement du journal propagé est de 2 Go. Pour les AWS Glue tâches ne prenant pas en charge les journaux cumulés, la taille maximale du fichier journal des événements pris en charge par SparkUI est de 0,5 Go.
- L'interface utilisateur de Spark sans serveur n'est pas disponible pour les journaux d'événements Spark stockés dans un compartiment Amazon S3 auquel seul votre VPC peut accéder.

Exemple : interface utilisateur web Apache Spark

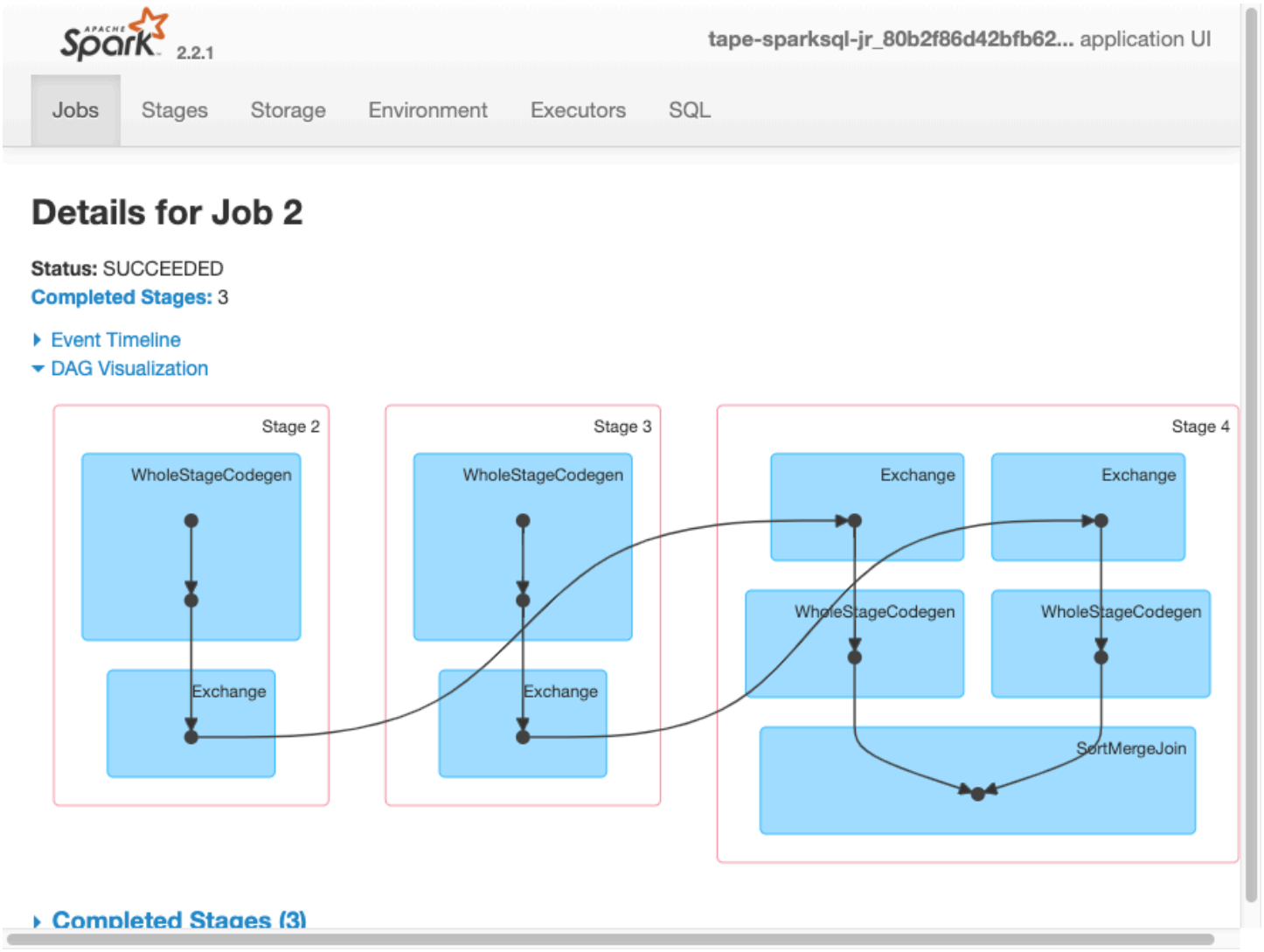
Cet exemple vous montre comment utiliser l'interface utilisateur Spark pour comprendre vos performances pour la tâche. Les captures d'écran montrent l'interface utilisateur web Spark telle que fournie par un serveur d'historique Spark autogéré. L'interface utilisateur Spark de la AWS Glue console fournit des vues similaires. Pour plus d'informations sur l'utilisation de l'interface utilisateur web Spark, consultez l'Interface utilisateur web dans la documentation Spark.

Voici un exemple d'application Spark qui lit à partir de deux sources de données, effectue une transformation de jointure et écrit le tout dans Amazon S3 au format Parquet.

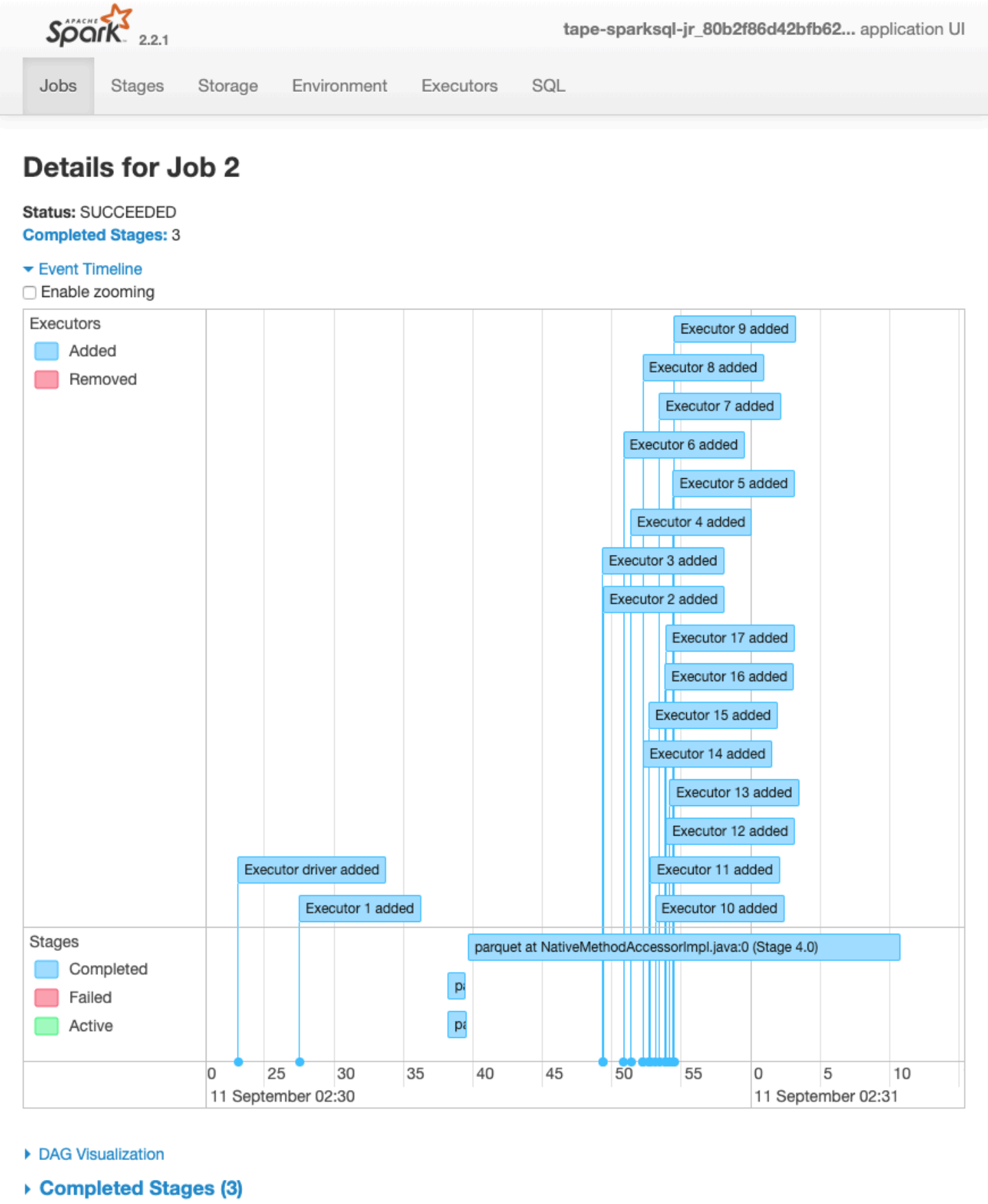
```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job
from pyspark.sql.functions import count, when, expr, col, sum, isnull
from pyspark.sql.functions import countDistinct
from awsglue.dynamicframe import DynamicFrame

args = getResolvedOptions(sys.argv, ['JOB_NAME'])

sc = SparkContext()
glueContext = GlueContext(sc)
```

```
spark = glueContext.spark_session

job = Job(glueContext)
job.init(args['JOB_NAME'])

df_persons = spark.read.json("s3://awsglue-datasets/examples/us-legislators/all/
persons.json")
df_memberships = spark.read.json("s3://awsglue-datasets/examples/us-legislators/all/
memberships.json")

df_joined = df_persons.join(df_memberships, df_persons.id == df_memberships.person_id,
 'fullouter')
df_joined.write.parquet("s3://aws-glue-demo-sparkui/output/")

job.commit()
```

La visualisation DAG suivante montre les différentes phases de cette tâche Spark.



La chronologie d'événements suivante pour une tâche montre le début, l'exécution et la résiliation de différents exécuteurs Spark.


Spark 2.2.1
tape-sparksql-jr_80b2f86d42bfb62... application UI
Jobs
Stages
Storage
Environment
Executors
SQL
Details for Job 2
Status: SUCCEEDED
Completed Stages: 3
Event Timeline
Enable zooming
Executors
Added
Removed
Executor 9 added
Executor 8 added
Executor 7 added
Executor 6 added
Executor 5 added
Executor 4 added
Executor 3 added
Executor 2 added
Executor 17 added
Executor 16 added
Executor 15 added
Executor 14 added
Executor 13 added
Executor 12 added
Executor 11 added
Executor 10 added
Executor driver added
Executor 1 added
Stages
Completed
Failed
Active
parquet at NativeMethodAccessorImpl.java:0 (Stage 4.0)
0
25
30
35
40
45
50
55
0
5
10
11 September 02:30
11 September 02:31
DAG Visualization
Completed Stages (3)

L’écran suivant montre les détails des plans de requête SparkSQL :

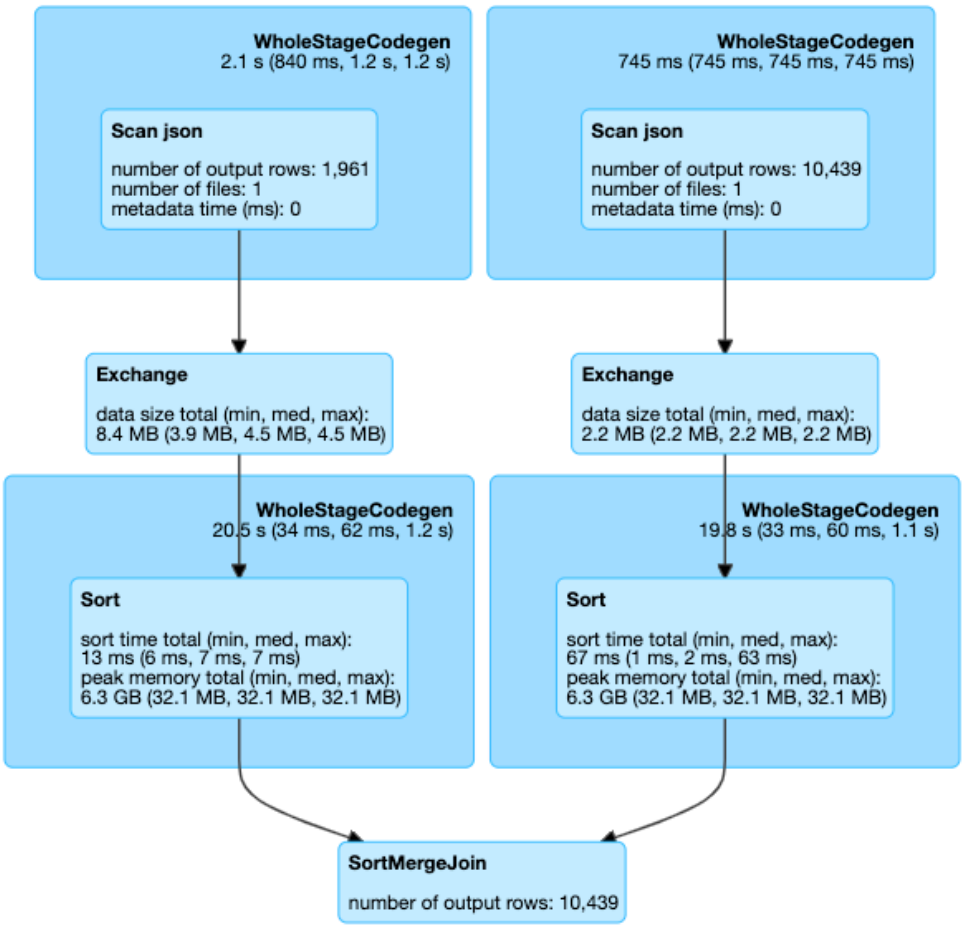
- Plan logique analysé
- Plan logique d’analyse
- Plan logique optimisé
- Plan physique d’exécution

Spark 2.2.1 tape-sparksql-jr_80b2f86d42bfb62... application UI

Jobs Stages Storage Environment Executors SQL

Details for Query 0

Submitted Time: 2019/09/11 02:30:37
Duration: 34 s
Succeeded Jobs: 2



▾ Details

```
== Parsed Logical Plan ==
Join FullOuter, (id#14 = person_id#50)
:-
Relation[birth_date#8,contact_details#9,death_date#10,family_name#11,gender#12,given_name#13,id#14,identifiers#15
,image#16,images#17,links#18,name#19,other_names#20,sort_name#21] json
+-
Relation[area_id#45,end_date#46,legislative_period_id#47,on_behalf_of_id#48,organization_id#49,person_id#50,role#
51,start_date#52] json

== Analyzed Logical Plan ==
birth_date: string, contact_details: array<struct<type:string,value:string>>, death_date: string, family_name:
string, gender: string, given_name: string, id: string, identifiers:
array<struct<identifier:string,scheme:string>>, image: string, images: array<struct<url:string>>, links:
array<struct<note:string,url:string>>, name: string, other_names:
array<struct<lang:string,name:string,note:string>>, sort_name: string, area_id: string, end_date: string,
legislative_period_id: string, on_behalf_of_id: string, organization_id: string, person_id: string, role: string,
start_date: string
Join FullOuter, (id#14 = person_id#50)
```

Rubriques

- Activation de l'interface utilisateur web Apache Spark pour les tâches AWS Glue
- Lancement du serveur d'historique Spark

Activation de l'interface utilisateur web Apache Spark pour les tâches AWS Glue

Vous pouvez utiliser l'interface utilisateur web Apache Spark pour surveiller et déboguer les tâches ETL AWS Glue en cours d'exécution sur le système de tâches AWS Glue. Vous pouvez configurer l'interface utilisateur Spark à l'aide de la console AWS Glue ou de l' AWS Command Line Interface (AWS CLI).

Toutes les 30 secondes, AWS Glue sauvegarde les journaux d'événements Spark à l'emplacement correspondant au chemin Amazon S3 que vous spécifiez.

Rubriques

- Configuration de l'interface utilisateur Spark (console)
- Configuration de l'interface utilisateur Spark (AWS CLI)
- Configuration de l'interface utilisateur Spark pour des sessions utilisant des blocs-notes
- Activer les journaux propagés

Configuration de l'interface utilisateur Spark (console)

Suivez ces étapes pour configurer l'interface utilisateur Spark à l'aide de la AWS Management Console. Lors de la création d'une AWS Glue tâche, l'interface utilisateur Spark est activée par défaut.

Pour activer l'interface utilisateur Spark lorsque vous créez ou modifiez une tâche

1. Connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/.
2. Dans le volet de navigation, sélectionnez Tâches.
3. Choisissez Ajouter une tâche ou sélectionnez-en une existante.
4. Dans Détails de la tâche, ouvrez les Propriétés avancées.
5. Sous l'onglet Interface utilisateur Spark, choisissez Écrire des journaux de l'interface utilisateur Spark sur Amazon S3.

6. Spécifiez un chemin Amazon S3 pour stocker les journaux d'événements Spark pour la tâche. Notez que si vous utilisez une configuration de sécurité dans la tâche, le chiffrement s'applique également au fichier journal de l'interface utilisateur Spark. Pour de plus amples informations, veuillez consulter Chiffrement des données écrites par AWS Glue.
7. Dans la Configuration de journalisation et de surveillance de l'interface utilisateur Spark :
   - Sélectionnez Standard si vous générez des journaux à afficher dans la AWS Glue console.
   - Sélectionnez Hérité si vous générez des journaux à afficher sur un serveur d'historique Spark.
   - Vous pouvez également choisir de générer les deux.

Configuration de l'interface utilisateur Spark (AWS CLI)

Pour générer des journaux à consulter avec l'interface utilisateur de Spark, dans la AWS Glue console, utilisez le AWS CLI pour transmettre les paramètres de tâche suivants aux AWS Glue tâches. Pour de plus amples informations, veuillez consulter the section called "Paramètres du job".

```
'--enable-spark-ui': 'true',
'--spark-event-logs-path': 's3://s3-event-log-path'
```

Pour distribuer les journaux dans leurs anciens emplacements, définissez le paramètre `--enable-spark-ui-legacy-path` sur `"true"`. Si vous ne souhaitez pas générer de journaux dans les deux formats, supprimez le paramètre `--enable-spark-ui`.

Configuration de l'interface utilisateur Spark pour des sessions utilisant des blocs-notes

> **Warning**
>
> AWS Glue les sessions interactives ne prennent actuellement pas en charge l'interface utilisateur Spark dans la console. Configurez un serveur d'historique Spark.

Si vous utilisez des AWS Glue blocs-notes, configurez la configuration de SparkUI avant de démarrer la session. Pour ce faire, utilisez le magic cellulaire de `%%configure` :

```
%%configure { "--enable-spark-ui": "true", "--spark-event-logs-path": "s3://path" }
```

Activer les journaux propagés

L'activation de SparkUI et des fichiers d'événements du journal continu pour les AWS Glue tâches présentent plusieurs avantages :

- Fichiers journaux d'événements continus : lorsque les fichiers journaux d'événements sont activés, des fichiers journaux distincts sont AWS Glue générés pour chaque étape de l'exécution de la tâche, ce qui facilite l'identification et le dépannage des problèmes spécifiques à une étape ou à une transformation en particulier.
- Meilleure gestion des journaux : les fichiers d'événement du journal propagé permettent de gérer plus efficacement les fichiers journaux. Au lieu d'avoir un seul fichier journal potentiellement volumineux, les journaux sont divisés en fichiers plus petits et plus faciles à gérer en fonction des étapes d'exécution des tâches. Cela peut simplifier l'archivage, l'analyse et le dépannage des journaux.
- Tolérance aux pannes améliorée — Si une AWS Glue tâche échoue ou est interrompue, les fichiers d'événements du journal continu peuvent fournir des informations précieuses sur la dernière étape réussie, ce qui permet de reprendre plus facilement la tâche à partir de ce point plutôt que de repartir de zéro.
- Optimisation des coûts : en activant les fichiers d'événement du journal propagé, vous pouvez économiser sur les coûts de stockage associés aux fichiers journaux. Au lieu de stocker un seul fichier journal potentiellement volumineux, vous stockez des fichiers journaux plus petits et plus faciles à gérer, ce qui peut être plus rentable, en particulier pour les tâches complexes ou de longue durée.

Dans un nouvel environnement, les utilisateurs peuvent activer explicitement les journaux propagés via :

```
'–conf': 'spark.eventLog.rolling.enabled=true'
```

or

```
'–conf': 'spark.eventLog.rolling.enabled=true –conf
spark.eventLog.rolling.maxFileSize=128m'
```

Lorsque les journaux propagés sont activés, `spark.eventLog.rolling.maxFileSize` spécifie la taille maximale du fichier journal des événements avant qu'il ne soit reporté. La valeur par défaut de ce paramètre facultatif, s'il n'est pas spécifié, est de 128 Mo. Le minimum est de 10 Mo.

La somme maximale de tous les fichiers d'événement du journal propagé est de 2 Go. Pour les AWS Glue tâches sans support de journalisation continue, la taille maximale du fichier journal des événements pris en charge par SparkUI est de 0,5 Go.

Vous pouvez désactiver les journaux propagés pour une tâche de streaming en fournissant une configuration supplémentaire. Notez que la maintenance de fichiers journaux très volumineux peut coûter cher.

Pour désactiver les journaux propagés, fournissez la configuration suivante :

```
'--spark-ui-event-logs-path': 'true',
'--conf': 'spark.eventLog.rolling.enabled=false'
```

Lancement du serveur d'historique Spark

Vous pouvez utiliser un serveur d'historique Spark pour visualiser les journaux Spark sur votre propre infrastructure. Vous pouvez voir les mêmes visualisations dans la AWS Glue console pour les exécutions de AWS Glue tâches sur les versions AWS Glue 4.0 ou ultérieures avec des journaux générés au format standard (plutôt qu'ancien). Pour de plus amples informations, veuillez consulter the section called "Surveillance à l'aide de l'interface utilisateur Spark".

Vous pouvez lancer le serveur d'historique Spark à l'aide d'un AWS CloudFormation modèle hébergeant le serveur sur une instance EC2, ou le lancer localement à l'aide de Docker.

Rubriques

- Lancement du serveur d'historique Spark et affichage de l'interface utilisateur de Spark à l'aide de AWS CloudFormation
- Lancement du serveur d'historique Spark et affichage de l'interface utilisateur Spark à l'aide de Docker

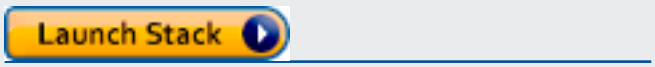
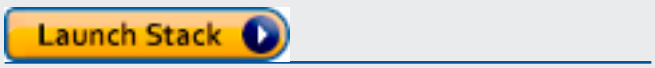
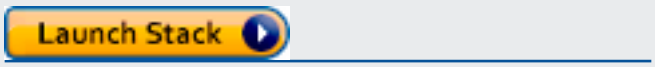
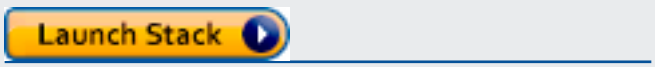
Lancement du serveur d'historique Spark et affichage de l'interface utilisateur de Spark à l'aide de AWS CloudFormation

Vous pouvez utiliser un AWS CloudFormation modèle pour démarrer le serveur d'historique Apache Spark et afficher l'interface utilisateur Web de Spark. Ces modèles sont des exemples que vous devez modifier pour répondre à vos besoins.

Pour démarrer le serveur d'historique Spark et afficher l'interface utilisateur de Spark à l'aide de CloudFormation

1. Choisissez l'un des boutons Launch Stack (Lancer la pile) du tableau suivant. Cela lance la pile sur la CloudFormation console.

Région	Lancer
USA Est (Ohio)	
USA Est (Virginie du Nord)	
USA Ouest (Californie du Nord)	
USA Ouest (Oregon)	
Afrique (Le Cap)	
Asie-Pacifique (Hong Kong)	
Asia Pacific (Mumbai)	
Asie-Pacifique (Osaka)	
Asia Pacific (Seoul)	
Asie-Pacifique (Singapour)	
Asie-Pacifique (Sydney)	
Asie-Pacifique (Tokyo)	

Région	Lancer
Canada (Centre)	
Europe (Francfort)	
Europe (Irlande)	
Europe (Londres)	
Europe (Milan)	
Europe (Paris)	
Europe (Stockholm)	
Moyen-Orient (Bahreïn)	
Amérique du Sud (São Paulo)	

2. Sur la page Specify template (Spécifier un modèle), choisissez Next (Suivant).
3. Sur la page Specify stack details (Spécifier les détails de la pile), entrez le nom de la pile (Stack name). Saisissez des informations supplémentaires sous Paramètres.

   a. Configuration de l'interface utilisateur Spark

      Saisissez les informations suivantes :

      - IP address range (Plage d'adresses IP) – Plage d'adresses IP pouvant être utilisée pour afficher l'interface utilisateur Spark. Pour limiter l'accès à partir d'une plage d'adresses IP spécifique, vous devez utiliser une valeur personnalisée.
      - History server port (Historique du serveur) – Port de l'interface utilisateur Spark. Vous pouvez utiliser la valeur par défaut.

- Event log directory (Répertoire des journaux d'événements) – Choisissez l'emplacement où les journaux d'événements Spark sont stockés en provenance de la tâche AWS Glue ou des points finaux de développement. Vous devez utiliser **`s3a://`** pour le schéma de chemin des journaux d'événements.
- Spark package location (Emplacement du package Spark) – Vous pouvez utiliser la valeur par défaut.
- Chemin du keystore : chemin du SSL/TLS keystore pour HTTPS. Si vous souhaitez utiliser un fichier de magasin de clés personnalisé, vous pouvez spécifier le chemin S3 `s3://path_to_your_keystore_file` ici. Si vous laissez ce paramètre vide, un magasin de clés auto-signé basé sur un certificat est généré et utilisé.
- Mot de passe KeyStore — Saisissez le mot de passe du magasin de clés SSL/TLS pour HTTPS.

b. Configuration des instances EC2

Saisissez les informations suivantes :

- Instance type (Type d'instance) – Type d'instance Amazon EC2 qui héberge le serveur d'historique Spark. Étant donné que ce modèle lance une instance Amazon EC2 dans votre compte, le coût d'Amazon EC2 sera facturé séparément dans votre compte.
- Latest AMI ID (Dernier ID AMI) – ID AMI d'Amazon Linux 2 pour l'instance du serveur d'historique Spark. Vous pouvez utiliser la valeur par défaut.
- VPC ID (ID VPC) – ID du cloud privé virtuel (VPC) pour l'instance du serveur d'historique Spark. Vous pouvez utiliser n'importe lequel des éléments VPCs disponibles dans votre compte. Il n'est pas recommandé d'utiliser un VPC par défaut avec une ACL réseau par défaut. Pour plus d'informations, veuillez consulter les rubriques VPC par défaut et sous-réseaux par défaut et Créer un VPC dans le Guide de l'utilisateur Amazon VPC.
- Subnet ID (ID de sous-réseau) – ID de l'instance du serveur d'historique Spark. Vous pouvez utiliser n'importe quel sous-réseau de votre VPC. Vous devez être en mesure d'atteindre le réseau depuis votre client vers le sous-réseau. Pour y accéder via Internet, vous devez utiliser un sous-réseau public dont la table de routage comporte la passerelle Internet.

c. Choisissez Suivant.

4. Sur la page Configurer les options de la pile, pour utiliser les informations d'identification utilisateur actuelles afin de déterminer CloudFormation comment créer, modifier ou supprimer des ressources dans la pile, choisissez Next. Vous pouvez également spécifier un rôle dans

la section Autorisations à utiliser au lieu des autorisations utilisateur actuelles, puis choisissez Suivant.

5. Sur la page Review (Révision), vérifiez le modèle.

   Sélectionnez Je reconnais que cela CloudFormation pourrait créer des ressources IAM, puis choisissez Create stack.

6. Attendez que la pile soit créée.
7. Ouvrez l'onglet Sorties.

   a. Copiez l'URL de SparkUiPublicUrlsi vous utilisez un sous-réseau public.

   b. Copiez l'URL de SparkUiPrivateUrlsi vous utilisez un sous-réseau privé.

8. Ouvrez un navigateur web et collez-y l'URL. Cela vous permet d'accéder au serveur à l'aide de HTTPS sur le port spécifié. Votre navigateur peut ne pas reconnaître le certificat du serveur ; dans ce cas, vous devez outrepasser sa protection et continuer tout de même.

Lancement du serveur d'historique Spark et affichage de l'interface utilisateur Spark à l'aide de Docker

Si vous préférez un accès local (pour ne pas avoir d'instance EC2 pour le serveur d'historique Apache Spark), vous pouvez également utiliser Docker pour démarrer le serveur d'historique Apache Spark et afficher l'interface utilisateur Spark localement. Ce fichier Dockerfile est un exemple que vous devez modifier pour répondre à vos besoins.

Conditions préalables

Pour plus d'informations sur l'installation de Docker sur votre ordinateur portable, consultez la communauté Docker Engine.

Pour démarrer le serveur d'historique Spark et afficher l'interface utilisateur Spark localement à l'aide de Docker

1. Téléchargez des fichiers depuis GitHub.

   Téléchargez les fichiers Dockerfile et `pom.xml` à partir des exemples de code AWS Glue.

2. Déterminez si vous souhaitez utiliser vos informations d'identification utilisateur ou vos informations d'identification utilisateur fédéré pour accéder à AWS.

- Pour utiliser les informations d'identification utilisateur actuelles pour accéder AWS, obtenez les valeurs à utiliser pour `AWS_ACCESS_KEY_ID` et `AWS_SECRET_ACCESS_KEY` dans la `docker run` commande. Pour plus d'informations, consultez Gestion des clés d'accès pour les utilisateurs IAM dans le Guide de l'utilisateur IAM.
- Pour utiliser les utilisateurs fédérés SAML 2.0 pour l'accès AWS, obtenez les valeurs pour `AWS_ACCESS_KEY_IDAWS_SECRET_ACCESS_KEY`, et. `AWS_SESSION_TOKEN` Pour en savoir plus, consultez Requesting temporary security credentials (Demander d'informations d'identification temporaires de sécurité).

3. Déterminez l'emplacement de votre répertoire de journaux d'événements, à utiliser dans la commande `docker run`.
4. Créez l'image Docker à l'aide des fichiers du répertoire local, en utilisant le nom `glue/sparkui` et l'identification `latest`.

```
$ docker build -t glue/sparkui:latest .
```

5. Créez et démarrez le conteneur Docker.

   Dans les commandes suivantes, utilisez les valeurs obtenues précédemment aux étapes 2 et 3.

   a. Pour créer le conteneur Docker à l'aide de vos informations d'identification utilisateur, utilisez une commande similaire à ce qui suit :

```
docker run -itd -e SPARK_HISTORY_OPTS="$SPARK_HISTORY_OPTS -
Dspark.history.fs.logDirectory=s3a://path_to_eventlog
 -Dspark.hadoop.fs.s3a.access.key=AWS_ACCESS_KEY_ID -
Dspark.hadoop.fs.s3a.secret.key=AWS_SECRET_ACCESS_KEY"
 -p 18080:18080 glue/sparkui:latest "/opt/spark/bin/spark-class
 org.apache.spark.deploy.history.HistoryServer"
```

   b. Pour créer le conteneur Docker à l'aide d'informations d'identification temporaires, utilisez `org.apache.hadoop.fs.s3a.TemporaryAWSCredentialsProvider` en tant que fournisseur, et fournissez les valeurs d'identification obtenues à l'étape 2. Pour plus d'informations, consultez la section Utilisation des informations d'identification de session avec un AWSCredentials fournisseur temporaire dans la documentation Hadoop : intégration avec Amazon Web Services.

```
docker run -itd -e SPARK_HISTORY_OPTS="$SPARK_HISTORY_OPTS -
Dspark.history.fs.logDirectory=s3a://path_to_eventlog
```

```
 -Dspark.hadoop.fs.s3a.access.key=AWS_ACCESS_KEY_ID -
Dspark.hadoop.fs.s3a.secret.key=AWS_SECRET_ACCESS_KEY
 -Dspark.hadoop.fs.s3a.session.token=AWS_SESSION_TOKEN
 -
Dspark.hadoop.fs.s3a.aws.credentials.provider=org.apache.hadoop.fs.s3a.TemporaryAWSCred
 -p 18080:18080 glue/sparkui:latest "/opt/spark/bin/spark-class
 org.apache.spark.deploy.history.HistoryServer"
```

> **Note**
>
> Ces paramètres de configuration proviennent du Hadoop-AWSModule. Il se peut que vous ayez besoin d'ajouter une configuration spécifique en fonction des cas d'utilisation. Par exemple : les utilisateurs des régions isolées devront configurer le `spark.hadoop.fs.s3a.endpoint`.

6. Ouvrez `http://localhost:18080` dans votre navigateur pour afficher localement l'interface utilisateur Spark.

Surveillance à l'aide AWS Glue d'informations sur l'exécution des tâches

AWS Glue Job Run Insights est une fonctionnalité AWS Glue qui simplifie le débogage des tâches et l'optimisation de vos AWS Glue tâches. AWS Glue fournit l'interface utilisateur Spark, ainsi que CloudWatch des journaux et des indicateurs pour surveiller vos AWS Glue tâches. Grâce à cette fonctionnalité, vous obtenez les informations suivantes concernant l'exécution de votre AWS Glue tâche :

- Numéro de ligne de votre script de AWS Glue tâche qui a échoué.
- L'action Spark exécutée en dernier dans le plan de requête Spark juste avant l'échec de votre tâche.
- Les événements d'exception Spark liés à l'échec présenté dans un flux de journal classé par ordre chronologique.
- L'analyse des causes racines et l'action recommandée (telle que le réglage de votre script) pour résoudre le problème.
- Les événements Spark courants (messages de journaux relatifs à une action Spark) avec une action recommandée qui traite la cause racine.

Toutes ces informations sont mises à votre disposition à l'aide de deux nouveaux flux de journaux dans les CloudWatch journaux de vos AWS Glue tâches.

Exigences

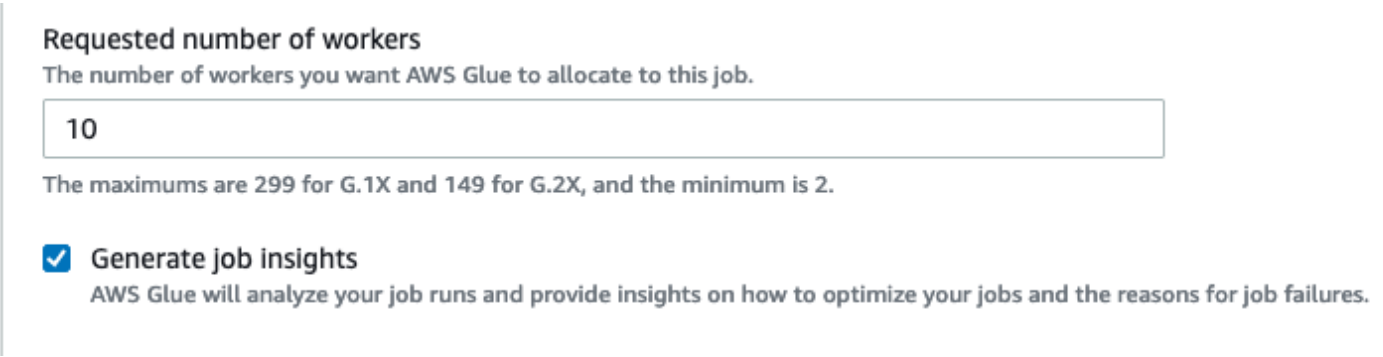
La fonctionnalité AWS Glue Job Run Insights est disponible pour AWS Glue les versions 2.0 et supérieures. Vous pouvez suivre le guide de migration de vos tâches existantes pour les mettre à niveau à partir d'anciennes AWS Glue versions.

Activation des informations relatives à l'exécution des tâches pour une tâche AWS Glue ETL

Vous pouvez activer les informations relatives à l'exécution des tâches via l'interface de ligne de commande AWS Glue Studio ou la CLI.

AWS Glue Studio

Lorsque vous créez une tâche via AWS Glue Studio, vous pouvez activer ou désactiver les informations relatives à l'exécution des tâches dans l'onglet Détails de la tâche. Vérifiez que la case Générer des informations sur les tâches est cochée.



Ligne de commande

Si vous créez une tâche via le CLI, vous pouvez démarrer l'exécution d'une tâche avec un seul nouveau paramètre de tâche :`--enable-job-insights = true`.

Par défaut, les flux de journaux des informations sur l'exécution de la tâche sont créés sous le même groupe de journaux par défaut utilisé par AWS Glue journalisation continue, c'est-à-dire, `/aws-glue/jobs/logs-v2/`. Vous pouvez configurer un nom de groupe de journaux personnalisé, des filtres de journaux et des configurations de groupes de journaux à l'aide du même ensemble d'arguments que pour la journalisation continue. Pour plus d'informations, consultez la section Activation de la journalisation continue pour les AWS Glue tâches.

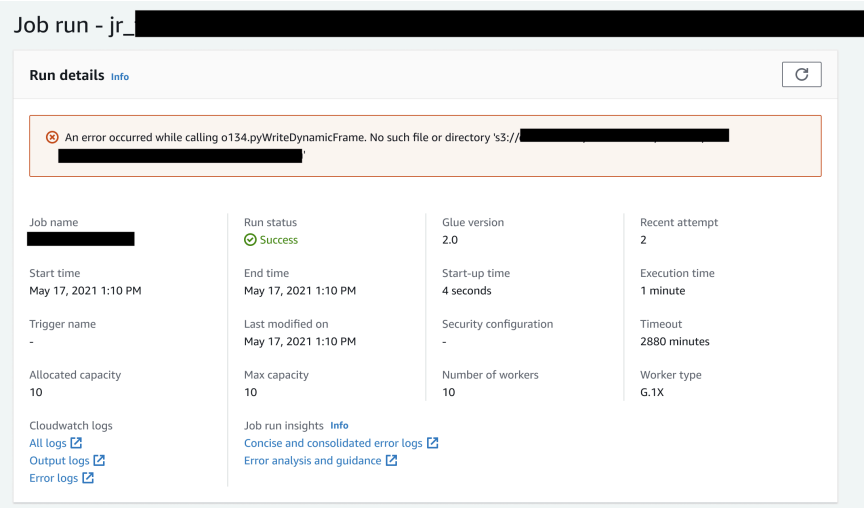
Accès aux flux de journaux Job Run Insights dans CloudWatch

Lorsque la fonctionnalité des informations sur l'exécution de la tâche est activée, deux flux de journaux peuvent être créés en cas d'échec d'une exécution de tâche. Lorsqu'une tâche se termine correctement, aucun des flux n'est généré.

1. Flux de journaux d'analyse des exceptions :`<job-run-id>-job-insights-rca-driver`. Ce flux fournit les informations suivantes :
   - Numéro de ligne du script de AWS Glue tâche à l'origine de l'échec.
   - L'action Spark exécutée en dernier dans le plan de requête Spark (DAG).
   - Les événements chronologiques concis provenant du pilote Spark et les programmes d'exécution associés à l'exception. Vous pourrez trouver des détails tels que des messages d'erreur complets, la tâche Spark qui a échoué et l'ID de ses programmes d'exécution. Ces informations vous aident à vous concentrer sur le flux de journal du programme d'exécution spécifique pour une analyse plus approfondie si nécessaire.
2. Flux d'informations basé sur des règles :
   - Analyse des causes racines et recommandations sur la manière de corriger les erreurs (telles que l'utilisation d'un paramètre de tâche spécifique pour optimiser les performances).
   - Les événements Spark pertinents servant de base à l'analyse des causes racines et à une action recommandée.

> **Note**
>
> Le premier flux n'existera que si des événements Spark d'exception sont disponibles pour une exécution de tâche qui a échoué, et le second flux n'existera que si des informations sont disponibles pour l'exécution de la tâche qui a échoué. Par exemple, si votre tâche se termine correctement, aucun des flux ne sera généré. Si votre tâche échoue, mais qu'aucune règle définie par le service ne correspond à votre scénario d'échec, seul le premier flux sera généré.

Si la tâche est créée à partir de AWS Glue Studio, les liens vers les flux ci-dessus sont également disponibles sous l'onglet Détails de l'exécution du travail (Job run insights) sous la forme « Journaux d'erreurs concis et consolidés » et « Analyse des erreurs et conseils ».



Exemple d'analyse de l'exécution des AWS Glue tâches

Dans cette section, nous vous présentons un exemple de la façon dont la fonctionnalité d'informations sur l'exécution de la tâche peut vous aider à résoudre un problème lié à votre tâche qui a échoué. Dans cet exemple, un utilisateur a oublié d'importer le module requis (tensorflow) dans une AWS Glue tâche pour analyser et créer un modèle d'apprentissage automatique à partir de ses données.

```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job
from pyspark.sql.types import *
from pyspark.sql.functions import udf,col

args = getResolvedOptions(sys.argv, ['JOB_NAME'])

sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
job = Job(glueContext)
job.init(args['JOB_NAME'], args)

data_set_1 = [1, 2, 3, 4]
data_set_2 = [5, 6, 7, 8]

scoresDf = spark.createDataFrame(data_set_1, IntegerType())
```

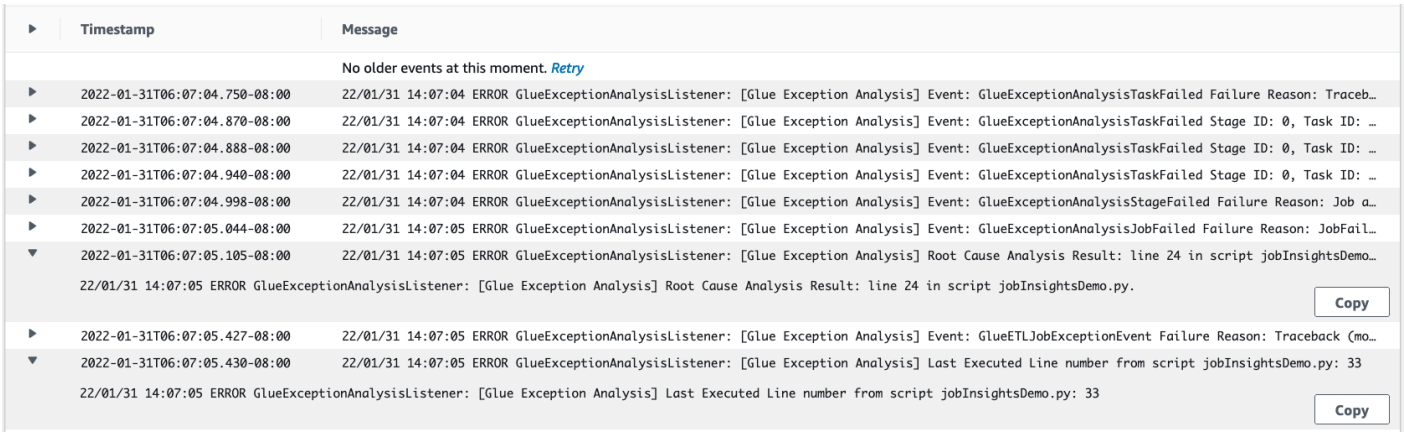
```
def data_multiplier_func(factor, data_vector):
    import tensorflow as tf
    with tf.compat.v1.Session() as sess:
        x1 = tf.constant(factor)
        x2 = tf.constant(data_vector)
        result = tf.multiply(x1, x2)
        return sess.run(result).tolist()

data_multiplier_udf = udf(lambda x:data_multiplier_func(x, data_set_2),
 ArrayType(IntegerType(),False))
factoredDf = scoresDf.withColumn("final_value", data_multiplier_udf(col("value")))
print(factoredDf.collect())
```

La tâche échoue et ,sans la fonctionnalité informations sur l'exécution de la tâche, vous ne voyez que ce message généré par Spark :

`An error occurred while calling o111.collectToPython. Traceback (most recent call last):`

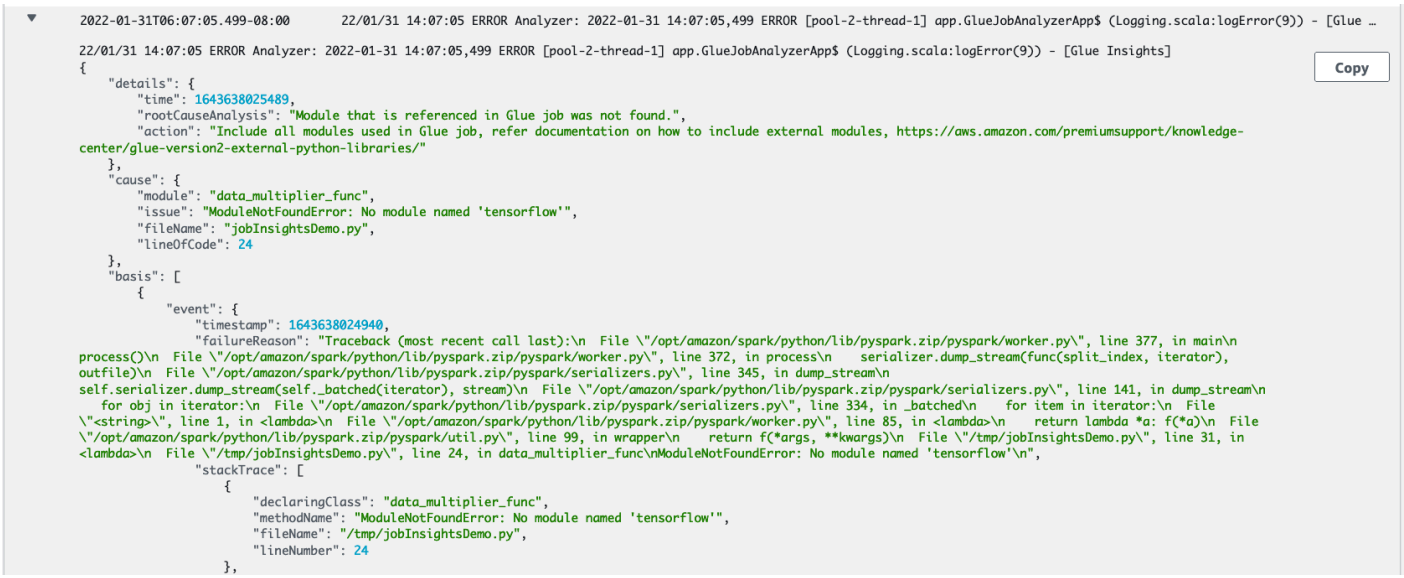
Le message est ambigu et limite votre expérience de débogage. Dans ce cas, cette fonctionnalité vous fournit des informations supplémentaires dans deux flux de CloudWatch journaux :

1. Le flux de journal `job-insights-rca-driver` :
   - Événements d'exception : ce flux de journal fournit les événements d'exception Spark liés à l'échec collecté à partir du pilote Spark et de différents travailleurs distribués. Ces événements vous aident à comprendre la propagation chronologique de l'exception lorsque le code défectueux s'exécute sur les tâches Spark, les exécuteurs et les étapes réparties entre les travailleurs. AWS Glue
   - Numéros de ligne: ce flux de journal identifie la ligne 21, qui a appelé à importer le module Python manquant à l'origine de l'échec. Il identifie également la ligne 24, l'appel à l'action Spark `collect()`, en tant que dernière ligne exécutée dans votre script.



2. Le flux de journal `job-insights-rule-driver` :

   - Cause première et recommandation : Outre le numéro de ligne et le numéro de ligne de la dernière exécution correspondant à l'erreur dans votre script, ce flux de journal indique l'analyse des causes profondes et vous recommande de suivre le AWS Glue document et de configurer les paramètres de tâche nécessaires afin d'utiliser un module Python supplémentaire dans votre AWS Glue tâche.
   - Événement récurrent : ce flux de journal affiche également l'événement d'exception Spark qui a été évalué avec la règle définie par le service afin d'en déduire la cause racine et de fournir une recommandation.



Surveillance avec Amazon CloudWatch

Vous pouvez effectuer un suivi AWS Glue à l'aide d'Amazon CloudWatch, qui collecte et traite les données brutes sous AWS Glue forme de near-real-time métriques lisibles. Ces statistiques sont

enregistrées pour une durée de deux semaines afin de pouvoir accéder aux informations historiques vous permettant d'acquérir un meilleur point de vue sur la façon dont votre service ou application web s'exécute. Par défaut, AWS Glue les données des métriques sont envoyées CloudWatch automatiquement à. Pour plus d'informations, consultez Qu'est-ce qu'Amazon CloudWatch ? dans le guide de CloudWatch l'utilisateur Amazon, etAWS Glue métriques.

Journalisation continue

AWS Glue prend également en charge la journalisation continue en temps réel pour les tâches AWS Glue. Lorsque la journalisation continue est activée pour une tâche, vous pouvez consulter les journaux en temps réel sur la AWS Glue console ou sur le tableau de bord de la CloudWatch console. Pour de plus amples informations, veuillez consulter Journalisation pour les AWS Glue emplois.

Métriques d'observabilité

Lorsque les métriques d'observabilité du Job sont activées, des Amazon CloudWatch métriques supplémentaires sont générées lors de l'exécution du job. Utilisez les métriques d'observabilité AWS Glue pour générer des informations sur ce qui se passe au sein de votre AWS Glue afin d'améliorer le triage et l'analyse des problèmes.

Rubriques

- Surveillance AWS Glue à l'aide CloudWatch des métriques Amazon
- Configuration des CloudWatch alarmes Amazon sur les profils AWS Glue d'emploi
- Journalisation pour les AWS Glue emplois
- Surveillance à l'aide de métriques d'observabilité AWS Glue

Surveillance AWS Glue à l'aide CloudWatch des métriques Amazon

Vous pouvez surveiller les opérations d'AWS Glue à l'aide du profileur de tâche AWS Glue. Il collecte et traite les données brutes des AWS Glue tâches pour en faire des métriques lisibles en temps quasi réel stockées sur Amazon CloudWatch. Ces statistiques sont conservées et agrégées CloudWatch afin que vous puissiez accéder aux informations historiques pour avoir une meilleure idée des performances de votre application.

> **Note**
>
> Des frais supplémentaires peuvent vous être facturés lorsque vous activez les indicateurs relatifs aux tâches et que CloudWatch des indicateurs personnalisés sont créés. Pour plus d'informations, consultez les CloudWatch tarifs Amazon.

Présentation des métriques AWS Glue

Lorsque vous interagissez avecAWS Glue, il envoie des métriques à CloudWatch. Vous pouvez consulter ces statistiques à l'aide de la AWS Glue console (méthode préférée), du tableau de bord de la CloudWatch console ou du AWS Command Line Interface (AWS CLI).

Pour afficher des métriques à l'aide du tableau de bord de la console AWS Glue

Vous pouvez afficher un résumé ou des graphiques détaillés des métriques pour une tâche ou des graphiques détaillés pour l'exécution d'une tâche.

1. Connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/.
2. Dans le panneau de navigation, choisissez Surveillance de l'exécution des tâches.
3. Dans Exécutions de tâches, choisissez Actions pour arrêter une tâche en cours d'exécution, afficher une tâche ou restaurer le signet d'une tâche.
4. Sélectionnez une tâche, puis choisissez Afficher les informations de l'exécution pour afficher des informations supplémentaires sur l'exécution de la tâche.

Pour consulter les statistiques à l'aide du tableau de bord de CloudWatch la console

Les métriques sont d'abord regroupées par espace de noms de service, puis par les différentes combinaisons de dimension au sein de chaque espace de noms.

1. Ouvrez la CloudWatch console à l'adresse https://console.aws.amazon.com/cloudwatch/.
2. Dans le panneau de navigation, sélectionnez Métriques.
3. Sélectionnez l'espace de noms Glue.

Pour consulter les statistiques à l'aide du AWS CLI

- À partir d'une invite de commande, utilisez la commande suivante :

```
aws cloudwatch list-metrics --namespace Glue
```

AWS Gluerapporte les métriques CloudWatch toutes les 30 secondes, et les tableaux de bord des CloudWatch métriques sont configurés pour les afficher toutes les minutes. Les métriques AWS Glue représentent des valeurs delta des valeurs précédemment rapportées. Le cas échéant, les tableaux de bord de métriques regroupent (additionnent) les valeurs de plages de 30 secondes pour obtenir une valeur pour la totalité de la dernière minute.

AWS Glue comportement des métriques pour les tâches Spark

Les métriques AWS Glue sont activées à l'initialisation d'un `GlueContext` dans un script et sont généralement mises à jour uniquement à la fin d'une tâche Apache Spark. Elles représentent les valeurs regroupées sur l'ensemble des tâches Spark terminées jusqu'alors.

Cependant, les métriques Spark AWS Glue transmises à CloudWatch sont généralement des valeurs absolues représentant l'état actuel au moment où elles sont signalées. AWS Glueles rapporte CloudWatch toutes les 30 secondes, et les tableaux de bord des métriques indiquent généralement la moyenne des points de données reçus au cours de la dernière minute.

Les noms des métriques AWS Glue sont tous précédés par l'un des types suivants de préfixe :

- `glue.driver.` – Les métriques dont les noms commencent par ce préfixe représentent des métriques AWS Glue regroupées à partir de tous les programmes d'exécution dans le pilote Spark ou des métriques Spark correspondant au pilote Spark.
- `glue.`executorId`.` – L'executorId correspond au numéro d'un programme d'exécution Spark spécifique. Il correspond aux programmes d'exécution répertoriés dans les journaux.
- `glue.ALL.` - Les métriques dont les noms commencent par ce préfixe regroupent des valeurs de tous les programmes d'exécution Spark.

AWS Glue métriques

AWS Glue établit le profil et envoie les métriques suivantes CloudWatch toutes les 30 secondes, et le tableau de bord AWS Glue des métriques les indique une fois par minute :

Métrique	Description
`glue.driver.aggregate.bytesRead`	Nombre d'octets lus à partir de toutes les sources de données par toutes les tâches Spark exécutées dans tous les programmes d'exécution. Dimensions valides : `JobName` (le nom du AWS Glue Job), `JobRunId` (l' JobRun ID ou`ALL`) et `Type` (le nombre). Statistiques valides : SUM. Cette métrique est une valeur delta par rapport à la dernière valeur signalée. Par conséquent, sur le tableau de bord AWS Glue des mesures, une statistique SUM est utilisée pour l'agrégation. Unité : octets Peut être utilisé pour contrôler ce qui suit : • Octets lus. • Progrès de la tâche. • Source de données JDBC. • Problèmes de marque-page de la tâche. • Variation entre les exécutions de tâches. Cette métrique peut être utilisée de la même manière que la métrique `glue.ALL.s3.filesystem.read_bytes` , à la différence qu'elle est mise à jour à la fin d'une tâche Spark et qu'elle capture également les sources de données non S3.
`glue.driver.aggregate.elapsedTime`	Le temps écoulé ETL en millisecondes (n'inclut pas les temps d'amorçage de la tâche).

Métrique	Description
	Dimensions valides : `JobName` (le nom du AWS Glue Job), `JobRunId` (l' JobRun ID ou`ALL`) et `Type` (le nombre). Statistiques valides : SUM. Cette métrique est une valeur delta par rapport à la dernière valeur signalée. Par conséquent, sur le tableau de bord AWS Glue des mesures, une statistique SUM est utilisée pour l'agrégation. Unité : millisecondes Peut être utilisé pour déterminer combien le temps moyen requis pour une exécution de tâche. Voici quelques façons d'utiliser les données : • Définissez des alarmes pour les ralentisseurs. • Mesurez l'écart entre les exécutions de tâches.

Métrique	Description
`glue.driver.aggregate.numCompletedStages`	Le nombre d'étapes terminées dans la tâche. Dimensions valides : `JobName` (le nom du AWS Glue Job), `JobRunId` (l' JobRun ID ou`ALL`) et `Type` (le nombre). Statistiques valides : SUM. Cette métrique est une valeur delta par rapport à la dernière valeur signalée. Par conséquent, sur le tableau de bord AWS Glue des mesures, une statistique SUM est utilisée pour l'agrégation. Unité : nombre Peut être utilisé pour contrôler ce qui suit : • Progrès de la tâche. • Chronologie par étape de l'exécution des tâches, en cas de corrélation avec d'autres métriques. Voici quelques façons d'utiliser les données : • Identifiez les étapes exigeantes dans l'exécution d'une tâche. • Définissez des alarmes pour les pics corrélés (étapes exigeantes) à travers les exécutions de tâches.

Métrique	Description
`glue.driver.aggregate.numCompletedTasks`	Le nombre de tâches terminées dans la tâche. Dimensions valides : `JobName` (le nom du AWS Glue Job), `JobRunId` (l' JobRun ID ou`ALL`) et `Type` (le nombre). Statistiques valides : SUM. Cette métrique est une valeur delta par rapport à la dernière valeur signalée. Par conséquent, sur le tableau de bord AWS Glue des mesures, une statistique SUM est utilisée pour l'agrégation. Unité : nombre Peut être utilisé pour contrôler ce qui suit : • Progrès de la tâche. • Parallélisme au sein d'une étape.

Métrique	Description
`glue.driver.aggregate.numFailedTasks`	Nombre de tâches ayant échoué. Dimensions valides : `JobName` (le nom du AWS Glue Job), `JobRunId` (l' JobRun ID ou`ALL`) et `Type` (le nombre). Statistiques valides : SUM. Cette métrique est une valeur delta par rapport à la dernière valeur signalée. Par conséquent, sur le tableau de bord AWS Glue des mesures, une statistique SUM est utilisée pour l'agrégation. Unité : nombre Peut être utilisé pour contrôler ce qui suit : • Anomalies de données qui provoquent l'échec des tâches de la tâche. • Anomalies de cluster qui provoquent l'échec des tâches de la tâche. • Anomalies de script qui provoquent l'échec des tâches de la tâche. Les données peuvent être utilisées pour définir des alarmes pour des échecs accrus qui pourraient suggérer des anomalies dans les données, les clusters ou les scripts.

Métrique	Description
`glue.driver.aggregate.numKilledTasks`	Nombre de tâches supprimées. Dimensions valides : `JobName` (le nom du AWS Glue Job), `JobRunId` (l' JobRun ID ou`ALL`) et `Type` (le nombre). Statistiques valides : SUM. Cette métrique est une valeur delta par rapport à la dernière valeur signalée. Par conséquent, sur le tableau de bord AWS Glue des mesures, une statistique SUM est utilisée pour l'agrégation. Unité : nombre Peut être utilisé pour contrôler ce qui suit : • Anomalies dans le biais des données qui se traduisent par des exceptions (OOMs) qui tuent des tâches. • Anomalies du script qui entraînent des exceptions (OOMs) qui tuent des tâches. Voici quelques façons d'utiliser les données : • Définissez des alarmes pour les échecs accrus indiquant des anomalies des données. • Définissez des alarmes pour les échecs accrus indiquant des anomalies de cluster. • Définissez des alarmes pour les échecs accrus indiquant des anomalies de script.

Métrique	Description
`glue.driver.aggregate.recordsRead`	Nombre d'enregistrements lus à partir de toutes les sources de données par toutes les tâches Spark terminées exécutées dans tous les programmes d'exécution. Dimensions valides : `JobName` (le nom du AWS Glue Job), `JobRunId` (l' JobRun ID ou`ALL`) et `Type` (le nombre). Statistiques valides : SUM. Cette métrique est une valeur delta par rapport à la dernière valeur signalée. Par conséquent, sur le tableau de bord AWS Glue des mesures, une statistique SUM est utilisée pour l'agrégation. Unité : nombre Peut être utilisé pour contrôler ce qui suit : • Enregistrements lus. • Progrès de la tâche. • Source de données JDBC. • Problèmes de marque-page de la tâche. • Asymétrie dans les exécutions de tâches au fil des jours. Cette métrique peut être utilisée de manière similaire à la métrique `glue.ALL.s3.filesystem.read_bytes` , à la différence qu'elle est mise à jour à la fin d'une tâche Spark.

Métrique	Description
`glue.driver.aggregate.shuffleBytesWritten`	Nombre d'octets écrits par tous les exécuteurs pour mélanger les données entre eux depuis le rapport précédent (agrégé par le tableau de bord des AWS Glue mesures sous la forme du nombre d'octets écrits à cette fin au cours de la minute précédente). Dimensions valides : `JobName` (le nom du AWS Glue Job), `JobRunId` (l' JobRun ID ou`ALL`) et `Type` (le nombre). Statistiques valides : SUM. Cette métrique est une valeur delta par rapport à la dernière valeur signalée. Par conséquent, sur le tableau de bord AWS Glue des mesures, une statistique SUM est utilisée pour l'agrégation. Unité : octets Peut être utilisé pour contrôler le remaniement de données dans les tâches (jointures volumineuses, GroupBy, répartition, coalesce). Voici quelques façons d'utiliser les données : • Repartitionnez ou décompressez les fichiers d'entrée volumineux avant un traitement ultérieur. • Repartitionnez les données de manière plus uniforme pour éviter les raccourcis clavier. • Filtrez préalablement les données avant les jointures ou les opérations GroupBy.

Métrique	Description
`glue.driver.aggregate.shuffleLocalBytesRead`	Nombre d'octets lus par tous les exécuteurs pour mélanger les données entre eux depuis le rapport précédent (agrégé par le tableau de bord des AWS Glue mesures sous la forme du nombre d'octets lus à cette fin au cours de la minute précédente). Dimensions valides : `JobName` (le nom du AWS Glue Job), `JobRunId` (l' JobRun ID ou`ALL`) et `Type` (le nombre). Statistiques valides : SUM. Cette métrique est une valeur delta par rapport à la dernière valeur signalée. Par conséquent, sur le tableau de bord AWS Glue des mesures, une statistique SUM est utilisée pour l'agrégation. Unité : octets Peut être utilisé pour contrôler le remaniement de données dans les tâches (jointures volumineuses, GroupBy, répartition, coalesce). Voici quelques façons d'utiliser les données : • Repartitionnez ou décompressez les fichiers d'entrée volumineux avant un traitement ultérieur. • Repartitionnez les données de manière plus uniforme à l'aide des raccourcis clavier. • Filtrez préalablement les données avant les jointures ou les opérations GroupBy.

Métrique	Description
`glue.driver.BlockManager.disk.diskSpaceUsed_MB`	Nombre de mégaoctets d'espace disque utilisés sur tous les programmes d'exécution. Dimensions valides : `JobName` (le nom du AWS Glue Job), `JobRunId` (l' JobRun identifiant ou`ALL`) et `Type` (la jauge). Statistiques valides : moyenne Il s'agit d'une métrique Spark, rapportée en tant que valeur absolue. Unité : mégaoctets Peut être utilisé pour contrôler ce qui suit : • Espace disque utilisé pour les blocs qui représent ent des partitions RDD mises en cache. • Espace disque utilisé pour les blocs qui représent ent des sorties de remaniement intermédiaire. • Espace disque utilisé pour les blocs qui représent ent des diffusions. Voici quelques façons d'utiliser les données : • Identifiez les échecs de tâche dus à une utilisation accrue du disque. • Identifiez les larges partitions qui entraînent un déversement ou un remaniement. • Augmentez la capacité DPU allouée pour corriger ces problèmes.

Métrique	Description
`glue.driver.ExecutorAllocationManager.executors.numberAllExecutors`	Le nombre de programmes d'exécution de tâches actifs en cours d'exécution. Dimensions valides : `JobName` (le nom du AWS Glue Job), `JobRunId` (l' JobRun identifiant ou`ALL`) et `Type` (la jauge). Statistiques valides : moyenne Il s'agit d'une métrique Spark, rapportée en tant que valeur absolue. Unité : nombre Peut être utilisé pour contrôler ce qui suit : • Activité de la tâche. • Programmes d'exécution en retard (avec uniquement quelques programmes d'exécution en cours d'exécution) • Parallélisme actuel au niveau du programme d'exécution. Voici quelques façons d'utiliser les données : • Repartitionnez ou décompressez au préalable les fichiers d'entrée volumineux si le cluster est sous-utilisé. • Identifiez les retards d'exécution de l'étape ou de la tâche dus à des scénarios de ralentissement. • • Comparez avec les numberMaxNeeded exécuteurs pour mieux comprendre le backlog en matière de provisionnement. DPUs

Métrique	Description
`glue.driver.ExecutorAllocationManager.executors.numberMaxNeededExecutors`	Nombre maximal de programmes d'exécution de tâches (en cours d'exécution et en attente) nécessaires pour satisfaire la charge actuelle. Dimensions valides : `JobName` (le nom du AWS Glue Job), `JobRunId` (l' JobRun identifiant ou`ALL`) et `Type` (la jauge). Statistiques valides : maximum Il s'agit d'une métrique Spark, rapportée en tant que valeur absolue. Unité : nombre Peut être utilisé pour contrôler ce qui suit : • Activité de la tâche. • Parallélisme actuel au niveau de l'exécuteur et arriéré de tâches en attente non encore planifiés en raison de l'indisponibilité des exécuteurs en raison de la capacité du DPU ou des exécuteurs. killed/failed Voici quelques façons d'utiliser les données : • Identification pending/backlog de la file d'attente de planification. • Identifiez les retards d'exécution de l'étape ou de la tâche dus à des scénarios de ralentissement. • Comparez avec numberAllExecutors pour mieux comprendre le backlog afin de mieux approvisi onner. DPUs • Augmentez la capacité de DPU allouée pour corriger le backlog des programmes d'exécution en attente.

Métrique	Description

`glue.driver.jvm.heap.usage`
 	
`glue.executorId.jvm.heap.usage`
 	
`glue.ALL.jvm.heap.usage`
	La fraction de mémoire utilisée par la pile de la JVM pour ce pilote (échelle : 0-1) pour le pilote, un programme d'exécution identifié par executorId ou TOUS les programmes d'exécution.
	
	Dimensions valides : `JobName` (le nom du AWS Glue Job), `JobRunId` (l' JobRun identifiant ou`ALL`) et `Type` (la jauge).
	
	Statistiques valides : moyenne Il s'agit d'une métrique Spark, rapportée en tant que valeur absolue.
	
	Unité : pourcentage
	
	Peut être utilisé pour contrôler ce qui suit :
	
	• Utilisation out-of-memory `glue.driver.jvm.heap.usage` des conditions du conducteur (OOM).
	• Utilisation des out-of-memory conditions de l'exécuteur (OOM). `glue.ALL.jvm.heap.usage`
	
	Voici quelques façons d'utiliser les données :
	
	• Identifiez les ID et les étapes des programmes d'exécution consommant de la mémoire.
	• Identifiez les ID et les étapes des programmes d'exécution en retard.
	• Identifiez un out-of-memory état du conducteur (OOM).
	• Identifiez une out-of-memory condition d'exécuteur (OOM) et obtenez l'ID d'exécuteur correspondant

Métrique	Description
	afin de pouvoir obtenir une trace de pile à partir du journal de l'exécuteur. • Identifiez les fichiers ou les partitions susceptibles de présenter une distorsion des données, ce qui peut entraîner des retards ou out-of-memory des conditions (). OOMs

Métrique	Description

`glue.driver.jvm.heap.used`
 	
`glue.executorId.jvm.heap.used`
 	
`glue.ALL.jvm.heap.used`
	le nombre d'octets de mémoire utilisés par la pile JVM pour le pilote, le programme d'exécution identifié par executorId, ou TOUS les programmes d'exécution.
	
	Dimensions valides : `JobName` (le nom du AWS Glue Job), `JobRunId` (l' JobRun identifiant ou`ALL`) et `Type` (la jauge).
	
	Statistiques valides : moyenne Il s'agit d'une métrique Spark, rapportée en tant que valeur absolue.
	
	Unité : octets
	
	Peut être utilisé pour contrôler ce qui suit :
	
	• out-of-memoryConditions de conduite (OOM).
	• out-of-memoryConditions de l'exécuteur (OOM).
	
	Voici quelques façons d'utiliser les données :
	
	• Identifiez les ID et les étapes des programmes d'exécution consommant de la mémoire.
	• Identifiez les ID et les étapes des programmes d'exécution en retard.
	• Identifiez un out-of-memory état du conducteur (OOM).
	• Identifiez une out-of-memory condition d'exécuteur (OOM) et obtenez l'ID d'exécuteur correspondant afin de pouvoir obtenir une trace de pile à partir du journal de l'exécuteur.
	• Identifiez les fichiers ou les partitions susceptibles de présenter une distorsion des données, ce qui

Métrique	Description
	peut entraîner des retards ou out-of-memory des conditions (). OOMs

Métrique	Description

`glue.driver.s3.filesystem.read_bytes`
 	
`glue.executorId.s3.filesystem.read_bytes`
 	
`glue.ALL.s3.filesystem.read_bytes`
	Le nombre d'octets lus depuis Amazon S3 par le pilote, un exécuteur identifié par ExecutoriD ou TOUS les exécuteurs depuis le rapport précédent (agrégé par le tableau de bord des métriques sous la forme du nombre d'octets lus AWS Glue au cours de la minute précédente).
	
	Dimensions valides :`JobName`, `JobRunId`, et `Type` (jauge).
	
	Statistiques valides : SUM. Cette métrique est une valeur delta par rapport à la dernière valeur signalée. Par conséquent, sur le tableau de bord AWS Glue des mesures, une statistique SUM est utilisée pour l'agrégation. La zone située sous la courbe sur le tableau de bord AWS Glue des métriques peut être utilisée pour comparer visuellement les octets lus lors de deux exécutions de tâches différentes.
	
	Unité : octets.
	
	Peut être utilisé pour contrôler ce qui suit :
	
	• Déplacement de données ETL.
	• Progrès de la tâche.
	• Problèmes de marque-page de tâche (données traitées, retraitées et ignorées).
	• Comparaison des lectures par rapport au taux d'ingestion à partir de sources de données externes.
	• Variation entre les exécutions de tâches.
	
	Les données résultantes peuvent être utilisées pour ce qui suit :

Métrique	Description
	• Planification des capacités de DPU. • Paramétrage d'alarmes pour les pics importants ou les creux au niveau des données lues pour les exécutions de tâches et les étapes des tâches.

Métrique	Description

`glue.driver.s3.filesystem.write_bytes`
 	
`glue.executorId.s3.filesystem.write_bytes`
 	
`glue.ALL.s3.filesystem.write_bytes`
	Le nombre d'octets écrits sur Amazon S3 par le pilote, un exécuteur identifié par ExecutoriD ou TOUS les exécuteurs depuis le rapport précédent (agrégé par le tableau de bord des métriques sous la forme du nombre d'octets écrits AWS Glue au cours de la minute précédente).
	
	Dimensions valides :`JobName`, `JobRunId`, et `Type` (jauge).
	
	Statistiques valides : SUM. Cette métrique est une valeur delta par rapport à la dernière valeur signalée. Par conséquent, sur le tableau de bord AWS Glue des mesures, une statistique SUM est utilisée pour l'agrégation. La zone située sous la courbe sur le tableau de bord AWS Glue des métriques peut être utilisée pour comparer visuellement les octets écrits lors de deux exécutions de tâches différentes.
	
	Unité : octets
	
	Peut être utilisé pour contrôler ce qui suit :
	
	• Déplacement de données ETL.
	• Progrès de la tâche.
	• Problèmes de marque-page de tâche (données traitées, retraitées et ignorées).
	• Comparaison des lectures par rapport au taux d'ingestion à partir de sources de données externes.
	• Variation entre les exécutions de tâches.
	
	Voici quelques façons d'utiliser les données :
	
	• Planification des capacités de DPU.

Métrique	Description
	• Paramétrage d'alarmes pour les pics importants ou les creux au niveau des données lues pour les exécutions de tâches et les étapes des tâches.
`glue.driver.streaming.numRecords`	Nombre d'enregistrements reçus dans un micro-lot. Cette métrique n'est disponible que pour les jobs de AWS Glue streaming dotés de AWS Glue la version 2.0 ou supérieure. Dimensions valides : `JobName` (le nom de la AWS Glue tâche), `JobRunId` (l' JobRun identifiant ou`ALL`) et `Type` (nombre). Statistiques valides : somme, minimum, maximum, moyenne, percentile Unité : nombre Peut être utilisé pour contrôler ce qui suit : • Enregistrements lus. • Progrès de la tâche.

Métrique	Description
`glue.driver.streaming.batchProcessingTimeInMs`	Temps nécessaire pour traiter les lots en millisecondes. Cette métrique n'est disponible que pour les jobs de AWS Glue streaming dotés de AWS Glue la version 2.0 ou supérieure. Dimensions valides : `JobName` (le nom de la AWS Glue tâche), `JobRunId` (l' JobRun identifiant ou`ALL`) et `Type` (nombre). Statistiques valides : somme, minimum, maximum, moyenne, percentile Unité : nombre Peut être utilisé pour contrôler ce qui suit : • Progrès de la tâche. • Performances de script.

Métrique	Description

`glue.driver.system.cpuSystemLoad`
 	
`glue.executorId.system.cpuSystemLoad`
 	
`glue.ALL.system.cpuSystemLoad`
	La fraction de chargement du système UC utilisée (échelle : 0-1) par le pilote, un programme d'exécution identifié par executorId ou tous les programmes d'exécution.
	
	Dimensions valides : `JobName` (le nom de la AWS Glue tâche), `JobRunId` (l' JobRun identifiant ou`ALL`) et `Type` (la jauge).
	
	Statistiques valides : moyenne Cette métrique est rapportée en tant que valeur absolue.
	
	Unité : pourcentage
	
	Peut être utilisé pour contrôler ce qui suit :
	
	• Charge de l'UC du pilote.
	• Charge de l'UC du programme d'exécution.
	• Détection des programmes d'exécution ou des stages liés à l'UC ou liés à l'IO dans une tâche.
	
	Voici quelques façons d'utiliser les données :
	
	• Planification de la capacité du DPU avec mesures d'E/S (Read/Shuffle octets, parallélisme des tâches) et métrique du nombre maximal d'exécuteurs nécessaires.
	• Identifier le ratio UC/IO lié. Cela permet de repartitionner et d'augmenter la capacité allouée pour les tâches de longue durée avec des jeux de données pouvant être divisés ayant une utilisation moins élevée de l'UC.

Dimensions pour les AWS Glue métriques

AWS Glue les métriques utilisent l'espace de AWS Glue noms et fournissent des métriques pour les dimensions suivantes :

Dimension	Description
`JobName`	Cette dimension filtre les métriques de toutes les exécutions d'une AWS Glue tâche spécifique.
`JobRunId`	Cette dimension filtre les métriques d'une AWS Glue tâche spécifique exécutée par un JobRun ID, ou`ALL`.
`Type`	Cette dimension filtre les métriques par `count` (un nombre agrégé) ou `gauge` (une valeur à un moment donné).

Pour plus d'informations, consultez le guide de CloudWatch l'utilisateur Amazon.

Configuration des CloudWatch alarmes Amazon sur les profils AWS Glue d'emploi

AWS Glueles statistiques sont également disponibles sur Amazon CloudWatch. Vous pouvez configurer des alarmes sur n'importe quelle métrique AWS Glue pour des tâches planifiées.

Voici quelques scénarios courants de configuration d'alarmes :

- Tâches manquant de mémoire (OOM - out of memory) : définissez une alarme lorsque l'utilisation de la mémoire dépasse la moyenne normale pour le pilote ou pour le programme d'exécution d'une tâche AWS Glue.
- Programmes d'exécution en retard : définissez une alarme lorsque le nombre de programmes d'exécution passe au-dessous d'un certain seuil durant un long moment dans une tâche AWS Glue.
- Retard ou retraitement des données : comparez les indicateurs des tâches individuelles d'un flux de travail à l'aide d'une expression CloudWatch mathématique. Vous pouvez ensuite déclencher une alarme sur la valeur résultant de l'expression (comme le ratio entre les octets écrits par une tâche et les octets lus par une tâche suivante).

Pour obtenir des instructions détaillées sur la configuration des alarmes, consultez la section Créer ou modifier une CloudWatch alarme dans le guide de l'utilisateur Amazon CloudWatch Events.

Pour les scénarios de surveillance et de débogage utilisant CloudWatch, voirSurveillance et débogage des tâches.

Journalisation pour les AWS Glue emplois

Dans la AWS Glue version 5.0, toutes les tâches disposent de fonctionnalités de journalisation en temps réel. En outre, vous pouvez spécifier des options de configuration personnalisées pour personnaliser le comportement de la journalisation. Ces options incluent la définition du nom du groupe de Amazon CloudWatch journaux, du préfixe du flux de Amazon CloudWatch journal (qui précédera l'ID et l'ID AWS Glue d'exécution de la tâche) et driver/executor du modèle de conversion des journaux pour les messages de journal. Ces configurations vous permettent d'agréger les journaux dans des groupes de journaux Amazon CloudWatch personnalisés avec différentes politiques d'expiration. En outre, vous pouvez analyser les journaux plus efficacement en utilisant des préfixes de flux de journaux et des modèles de conversion personnalisés. Ce niveau de personnalisation vous permet d'optimiser la gestion et l'analyse des journaux en fonction de vos besoins spécifiques.

Comportement de journalisation en AWS Glue version 5.0

Par défaut, les journaux du système, les journaux du démon Spark et les journaux de l'utilisateur AWS Glue sont écrits dans le groupe de `/aws-glue/jobs/error` journaux. Amazon CloudWatch D'autre part, les journaux utilisateur stdout (sortie standard) et stderr (erreur standard) sont écrits dans le groupe de journaux `/aws-glue/jobs/output` par défaut.

Journalisation personnalisée

Vous pouvez personnaliser le groupe de journaux par défaut et les préfixes de flux de journaux à l'aide des arguments de tâche suivants :

- `--custom-logGroup-prefix` : vous permet de spécifier un préfixe personnalisé pour les groupes de journaux `/aws-glue/jobs/error` et `/aws-glue/jobs/output`. Si vous fournissez un préfixe personnalisé, les noms des groupes de journaux seront au format suivant :
  - `/aws-glue/jobs/error` sera *`<customer prefix>`*`/error`
  - `/aws-glue/jobs/output` sera *`<customer prefix>`*`/output`

- `--custom-logStream-prefix` : vous permet de spécifier un préfixe personnalisé pour les noms des flux de journaux au sein des groupes de journaux. Si vous fournissez un préfixe personnalisé, les noms des flux de journaux seront au format suivant :
  - `jobrunid-driver` sera *`<customer log stream>`*`-driver`
  - `jobrunid-executorNum` sera *`<customer log stream>`*`-executorNum`

Règles de validation et restrictions relatives aux préfixes personnalisés :

- La longueur du nom complet du flux de journaux doit être comprise entre 1 et 512 caractères.
- Le préfixe personnalisé lui-même est limité à 400 caractères.
- Le préfixe personnalisé doit correspondre au modèle d'expression régulière `[^:*] *` (les caractères spéciaux autorisés sont « _ », « - » et « / »).

Journalisation des messages spécifiques aux applications à l'aide de l'enregistreur de script personnalisé

Vous pouvez utiliser l' AWS Glue enregistreur pour enregistrer tous les messages spécifiques à l'application dans le script qui sont envoyés en temps réel au flux de journal du pilote.

L'exemple suivant illustre un script Python.

```
from awsglue.context import GlueContext
from pyspark.context import SparkContext

sc = SparkContext()
glueContext = GlueContext(sc)
logger = glueContext.get_logger()
logger.info("info message")
logger.warn("warn message")
logger.error("error message")
```

L'exemple suivant illustre un script Scala.

```
import com.amazonaws.services.glue.log.GlueLogger

object GlueApp {
  def main(sysArgs: Array[String]) {
    val logger = new GlueLogger
```

```
    logger.info("info message")
    logger.warn("warn message")
    logger.error("error message")
  }
}
```

Activation de la barre de progression pour afficher la progression d'une tâche

AWS Glue fournit une barre de progression en temps réel sous le flux du `JOB_RUN_ID-progress-bar` journal pour vérifier l'état AWS Glue d'exécution des tâches. Actuellement, il prend en charge uniquement les tâches qui initialisent `glueContext`. Si vous exécutez une tâche Spark pure sans l'initialiser`glueContext`, la barre de AWS Glue progression n'apparaît pas.

La barre de progression indique une mise à jour de la progression suivante toutes les 5 secondes.

```
Stage Number (Stage Name): > (numCompletedTasks + numActiveTasks) /
 totalNumOfTasksInThisStage]
```

Configuration de sécurité avec Amazon CloudWatch journalisation

Lorsqu'une configuration de sécurité est activée pour les Amazon CloudWatch journaux, AWS Glue crée des groupes de journaux avec des modèles de dénomination spécifiques qui incorporent le nom de la configuration de sécurité.

Dénomination des groupes de journaux avec configuration de sécurité

Les groupes de journaux par défaut et personnalisés seront les suivants :

- Groupe de journaux d'erreurs par défaut : `/aws-glue/jobs/`*`Security-Configuration-Name`*`-role/`*`glue-job-role`*`/error`
- Groupe de journaux de sortie par défaut : `/aws-glue/jobs/`*`Security-Configuration-Name`*`-role/`*`glue-job-role`*`/output`
- Groupe de journaux d'erreurs personnalisé (AWS Glue 5.0) : *`custom-log-group-prefix`*`/`*`Security-Configuration-Name`*`-role/`*`glue-job-role`*`/error`
- Groupe de journaux de sortie personnalisé (AWS Glue 5.0) : *`custom-log-group-prefix`*`/`*`Security-Configuration-Name`*`-role/`*`glue-job-role`*`/output`

Autorisations IAM nécessaires

Vous devez ajouter l'autorisation `logs:AssociateKmsKey` à vos autorisations de rôle IAM, si vous activez une configuration de sécurité avec Amazon CloudWatch Logs. Si cette autorisation n'est pas incluse, la journalisation continue sera désactivée.

De plus, pour configurer le chiffrement des Amazon CloudWatch journaux, suivez les instructions de la section Chiffrer les données des Amazon CloudWatch journaux dans les journaux à l'aide AWS Key Management Service du guide de l'utilisateur Amazon Amazon CloudWatch Logs.

Informations supplémentaires

Pour plus d'informations sur la création de configurations de sécurité, consultez la section Gestion des configurations de sécurité sur la AWS Glue console.

Rubriques

- Activation de la journalisation continue pour les tâches AWS Glue 4.0 et antérieures
- Afficher les journaux des AWS Glue offres d'emploi

Activation de la journalisation continue pour les tâches AWS Glue 4.0 et antérieures

> **Note**
>
> Dans les versions AWS Glue 4.0 et antérieures, la journalisation continue était une fonctionnalité disponible. Cependant, avec l'introduction de la AWS Glue version 5.0, toutes les tâches disposent d'une fonctionnalité de journalisation en temps réel. Pour plus de détails sur les fonctionnalités de journalisation et les options de configuration de la AWS Glue version 5.0, consultez la section Journalisation des AWS Glue tâches.

Vous pouvez activer la journalisation continue à l'aide de la AWS Glue console ou via le AWS Command Line Interface (AWS CLI).

Vous pouvez activer la journalisation continue lorsque vous créez une tâche, modifiez une tâche existante, ou vous pouvez l'activer via l' AWS CLI.

Vous pouvez également spécifier des options de configuration personnalisées telles que le nom du groupe de Amazon CloudWatch journaux, le préfixe du flux de CloudWatch journal avant l' driver/

executor ID AWS Glue d'exécution de la tâche et le modèle de conversion des journaux pour les messages de journal. Ces configurations vous permettent de définir des journaux agrégés dans des groupes de CloudWatch journaux personnalisés avec différentes politiques d'expiration, et de les analyser plus en détail à l'aide de préfixes de flux de journaux et de modèles de conversion personnalisés.

Rubriques

- À l'aide du AWS Management Console
- Journalisation des messages spécifiques aux applications à l'aide de l'enregistreur de script personnalisé
- Activation de la barre de progression pour afficher la progression d'une tâche
- Configuration de sécurité avec la journalisation continue

À l'aide du AWS Management Console

Procédez comme suit pour utiliser la console afin d'activer la journalisation continue lors de la création ou de la modification d'une AWS Glue tâche.

Pour créer une nouvelle AWS Glue tâche grâce à la journalisation continue

1. Connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/.
2. Dans le volet de navigation, sélectionnez Tâches ETL.
3. Choisissez ETL visuel.
4. Dans l'onglet Détails de la tâche, développez la section Propriétés avancées.
5. Sous Journalisation continue, sélectionnez Activer les connexions CloudWatch.

Pour activer la journalisation continue d'une AWS Glue tâche existante

1. Ouvrez la AWS Glue console à l'adresse https://console.aws.amazon.com/glue/.
2. Dans le volet de navigation, sélectionnez Tâches.
3. Choisissez une tâche existante à partir de la liste des Tâches.
4. Choisissez Action, Modifier la tâche.
5. Dans l'onglet Détails de la tâche, développez la section Propriétés avancées.

6. Sous Journalisation continue, sélectionnez Activer les connexions CloudWatch.

À l'aide du AWS CLI

Pour activer la journalisation continue, vous devez transmettre les paramètres d'une tâche à une AWS Glue tâche. Transmettez les paramètres de tâche spéciaux suivants, similaires aux autres paramètres de AWS Glue tâche. Pour de plus amples informations, veuillez consulter Utilisation des paramètres des tâches dans les tâches AWS Glue.

```
'--enable-continuous-cloudwatch-log': 'true'
```

Vous pouvez spécifier un nom de groupe de CloudWatch journaux Amazon personnalisé. Si aucune valeur n'est spécifiée, le nom par défaut du groupe de journaux est `/aws-glue/jobs/logs-v2`.

```
'--continuous-log-logGroup': 'custom_log_group_name'
```

Vous pouvez spécifier un préfixe de flux de CloudWatch journal Amazon personnalisé. Si aucune valeur n'est spécifiée, le préfixe de flux de journal par défaut est l'ID d'exécution du travail.

```
'--continuous-log-logStreamPrefix': 'custom_log_stream_prefix'
```

Vous pouvez spécifier un modèle de conversion de journalisation continue personnalisé. Si aucune valeur n'est spécifiée, le modèle de conversion par défaut est `%d{yy/MM/dd HH:mm:ss} %p %c{1}: %m%n`. Notez que le modèle de conversion s'applique uniquement aux journaux des pilotes et des programmes d'exécution. Cela n'affecte pas la barre de progression AWS Glue .

```
'--continuous-log-conversionPattern': 'custom_log_conversion_pattern'
```

Journalisation des messages spécifiques aux applications à l'aide de l'enregistreur de script personnalisé

Vous pouvez utiliser l' AWS Glue enregistreur pour enregistrer tous les messages spécifiques à l'application dans le script qui sont envoyés en temps réel au flux de journal du pilote.

L'exemple suivant illustre un script Python.

```
from awsglue.context import GlueContext
```

```
from pyspark.context import SparkContext

sc = SparkContext()
glueContext = GlueContext(sc)
logger = glueContext.get_logger()
logger.info("info message")
logger.warn("warn message")
logger.error("error message")
```

L'exemple suivant illustre un script Scala.

```
import com.amazonaws.services.glue.log.GlueLogger

object GlueApp {
  def main(sysArgs: Array[String]) {
    val logger = new GlueLogger
    logger.info("info message")
    logger.warn("warn message")
    logger.error("error message")
  }
}
```

Activation de la barre de progression pour afficher la progression d'une tâche

AWS Glue fournit une barre de progression en temps réel sous le flux du `JOB_RUN_ID-progress-bar` journal pour vérifier l'état AWS Glue d'exécution des tâches. Actuellement, il prend en charge uniquement les tâches qui initialisent `glueContext`. Si vous exécutez une tâche Spark pure sans l'initialiser`glueContext`, la barre de AWS Glue progression n'apparaît pas.

La barre de progression indique une mise à jour de la progression suivante toutes les 5 secondes.

```
Stage Number (Stage Name): > (numCompletedTasks + numActiveTasks) /
 totalNumOfTasksInThisStage]
```

Configuration de sécurité avec la journalisation continue

Si une configuration de sécurité est activée pour CloudWatch les journaux, AWS Glue créera un groupe de journaux nommé comme suit pour les journaux continus :

```
<Log-Group-Name>-<Security-Configuration-Name>
```

Les groupes de journaux par défaut et personnalisés seront les suivants :

- Le groupe de journaux continus par défaut sera `/aws-glue/jobs/error-<Security-Configuration-Name>`
- Le groupe de journaux continus personnalisé sera `<custom-log-group-name>-<Security-Configuration-Name>`

Vous devez ajouter les autorisations `logs:AssociateKmsKey` à votre rôle IAM, si vous activez une configuration de sécurité avec CloudWatch Logs. Si cette autorisation n'est pas incluse, la journalisation continue sera désactivée. De plus, pour configurer le chiffrement des CloudWatch journaux, suivez les instructions de la section Chiffrer les données des CloudWatch journaux dans les journaux à l'aide AWS Key Management Service du guide de l'utilisateur Amazon CloudWatch Logs.

Pour plus d'informations sur la création de configurations de sécurité, veuillez consulter Gestion des configurations de sécurité sur la console AWS Glue.

> **Note**
>
> Des frais supplémentaires peuvent vous être facturés lorsque vous activez la journalisation et que des événements de CloudWatch journalisation supplémentaires sont créés. Pour plus d'informations, consultez les CloudWatch tarifs Amazon.

Afficher les journaux des AWS Glue offres d'emploi

Vous pouvez consulter les journaux en temps réel à l'aide de la AWS Glue console ou de la CloudWatch console Amazon.

Pour consulter les journaux en temps réel à l'aide du tableau de bord de AWS Glue la console

1. Connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/.
2. Dans le volet de navigation, sélectionnez Tâches.
3. Ajouter ou démarrer une tâche existante. Choisissez Action, Exécuter la tâche.

   Lorsque vous démarrez l'exécution d'une tâche, vous accédez à une page qui contient des informations sur la tâche en cours d'exécution :

   - L'onglet Journaux affiche les journaux des applications agrégés plus anciens.

- L'onglet Journaux affiche une barre de progression en temps réel lorsque la tâche est en cours d'exécution avec `glueContext` initialisé.
- L'onglet Logs contient également les journaux des pilotes, qui capturent les journaux des pilotes Apache Spark en temps réel, et les journaux des applications à partir du script enregistré à l'aide de l'enregistreur d' AWS Glue applications lorsque le job est en cours d'exécution.

4. Pour les anciennes tâches, vous pouvez également afficher les journaux en temps réel sous la vue Job History (Historique des tâches) en choisissant Journaux. Cette action vous amène à la CloudWatch console qui affiche tous les flux de log du pilote Spark, de l'exécuteur et de la barre de progression pour cette tâche exécutée.

Pour consulter les journaux en temps réel à l'aide du tableau de bord de CloudWatch la console

1. Ouvrez la CloudWatch console à l'adresse https://console.aws.amazon.com/cloudwatch/.
2. Dans le volet de navigation, sélectionnez Journaux.
3. Choisissez le groupe de journaux /aws-glue/jobs/error/.
4. Dans la zone Filtre, collez l'ID d'exécution de la tâche

   Vous pouvez afficher les journaux du pilote, les journaux du programme d'exécution et la barre de progression (si vous utilisez le Standard filter (Filtre standard)).

Surveillance à l'aide de métriques d'observabilité AWS Glue

> **Note**
>
> Les métriques d'observabilité AWS Glue sont disponibles dans AWS Glue 4.0 et les versions ultérieures.

Utilisez les métriques d'observabilité AWS Glue pour générer des informations sur ce qui se passe au sein de votre AWS Glue pour les tâches Apache Spark afin d'améliorer le triage et l'analyse des problèmes. Les métriques d'observabilité sont visualisées via des tableaux de bord Amazon CloudWatch et peuvent être utilisées pour analyser les causes racines des erreurs et pour diagnostiquer les goulots d'étranglement en matière de performance. Vous pouvez réduire le temps passé à déboguer les problèmes à l'échelle afin de vous concentrer sur leur résolution plus rapide et plus efficace.

AWS GlueL'observabilité fournit des Amazon CloudWatch métriques classées dans les quatre groupes suivants :

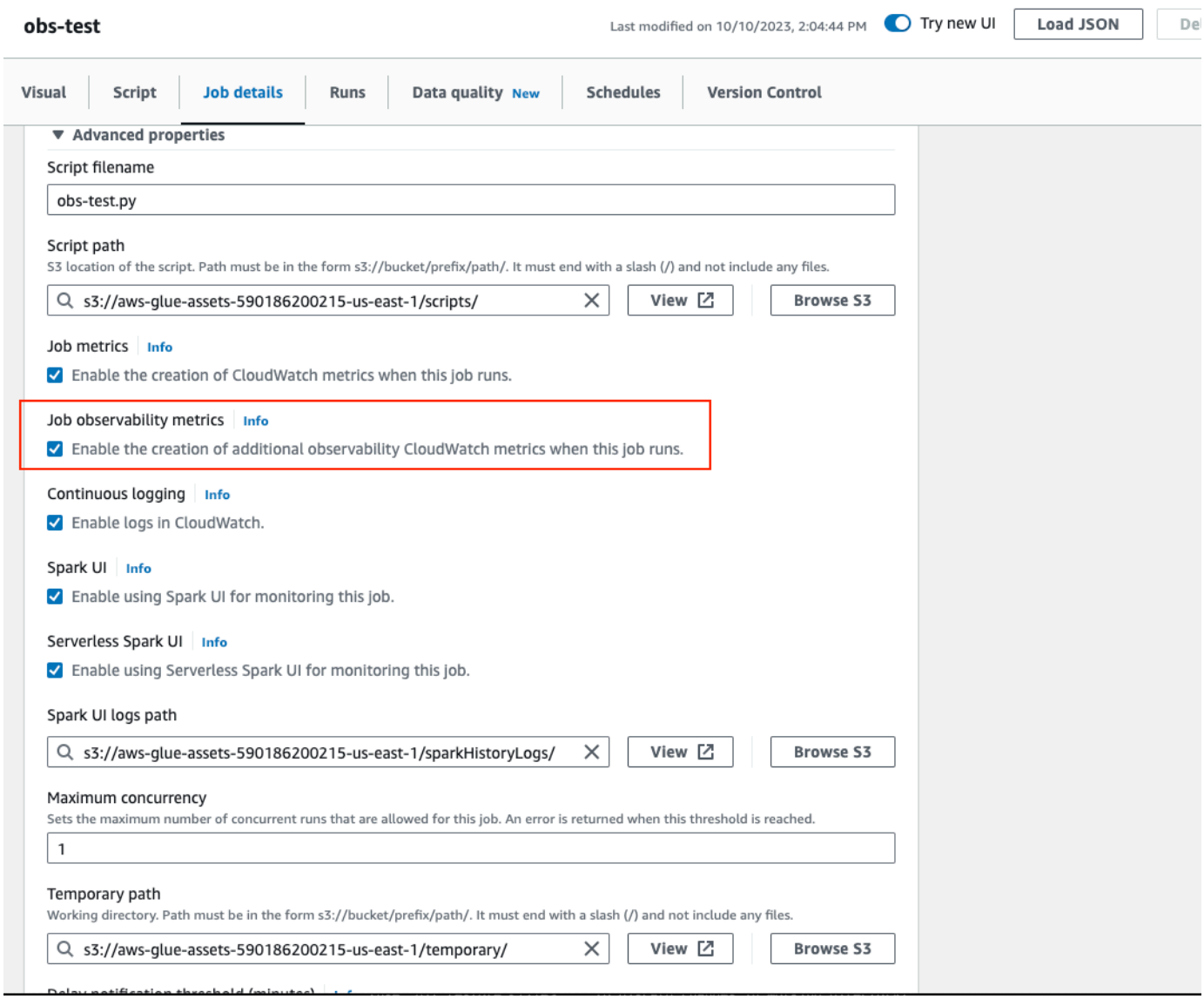
- Fiabilité (par exemple, classes d'erreurs) : identifier facilement les motifs d'échec les plus courants sur une plage de temps donnée que vous souhaiterez remédier.
- Performance (par exemple, asymétrie) : identifier un goulot d'étranglement en matière de performance et appliquer des techniques de réglage. Par exemple, lorsque vos performances sont dégradées en raison de l'asymétrie des tâches, vous pouvez activer l'Exécution adaptative des requêtes Spark et affiner le seuil de jointure asymétrique.
- Débit (c'est-à-dire par source/sink débit) : surveillez les tendances en matière de lecture et d'écriture de données. Vous pouvez également configurer des Amazon CloudWatch alarmes en cas d'anomalie.
- Utilisation des ressources (c'est-à-dire utilisation du personnel, de la mémoire et du disque) : trouver de manière efficace les tâches dont l'utilisation des capacités est faible. Vous souhaiterez peut-être activer la mise à l'échelle automatique AWS Glue pour ces tâches.

Mise en route avec les métriques d'observabilité AWS Glue

> **Note**
>
> Les nouvelles métriques sont disponibles par défaut dans la console AWS Glue Studio.

Pour configurer les métriques d'observabilité dans AWS Glue Studio :

1. Connectez-vous à la console AWS Glue et choisissez les tâches ETL dans le menu de la console.
2. Choisissez une tâche en cliquant sur le nom de la tâche dans la section Vos tâches.
3. Sélectionnez l'onglet Job details (Détails de la tâche).
4. Faites défiler l'écran vers le bas et choisissez Propriétés avancées, puis Metriques d'observabilité de la tâche.



Pour activer les métriques AWS Glue d'observabilité à l'aide AWS CLI de :

- Ajoutez à la carte `--default-arguments` la valeur clé suivante dans le fichier JSON d'entrée :

```
--enable-observability-metrics, true
```

Utiliser l'observabilité AWS Glue

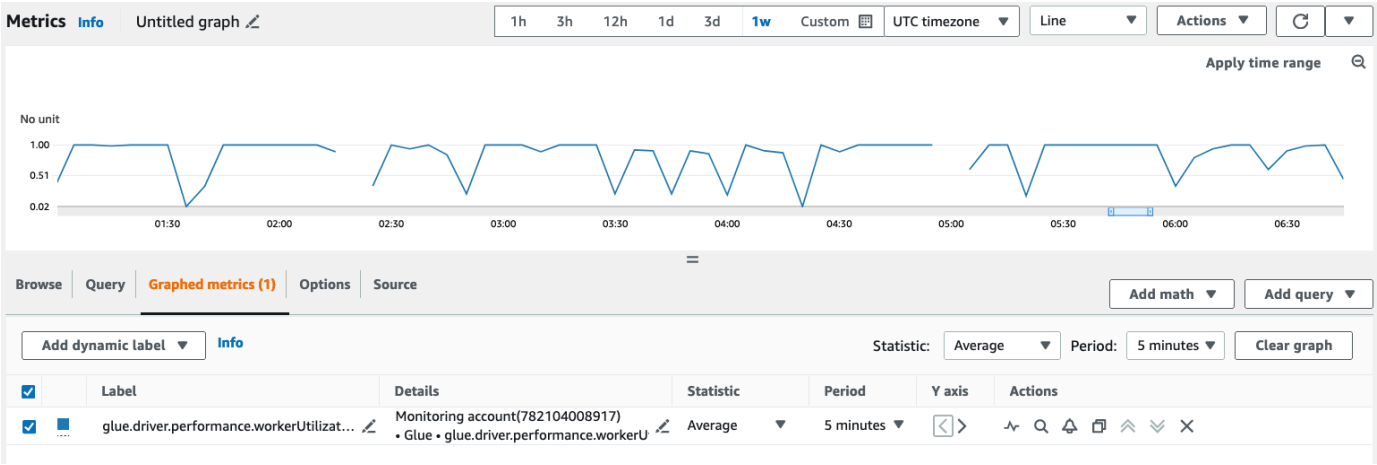
Les métriques AWS Glue d'observabilité étant fournies via Amazon CloudWatch, vous pouvez utiliser la Amazon CloudWatch console AWS CLI, le SDK ou l'API pour interroger les points de

données des métriques d'observabilité. Consultez la rubrique Utilisation de l'observabilité Glue pour surveiller l'utilisation des ressources afin de réduire les coûts pour un exemple de cas d'utilisation des métriques d'observabilité AWS Glue.

Utilisation de l'AWS Glueobservabilité dans la console Amazon CloudWatch

Pour interroger et visualiser les métriques dans la Amazon CloudWatch console :

1. Ouvrez la Amazon CloudWatch console et choisissez Toutes les mesures.
2. Sous Espaces de noms personnalisés, sélectionnez AWS Glue.
3. Choisissez les métriques d'observabilité de la tâche, les métriques d'observabilité par source ou les métriques d'observabilité par récepteur.
4. Recherchez le nom de la métrique, le nom de la tâche, l'ID d'exécution de la tâche spécifiques, puis sélectionnez-les.
5. Sous l'onglet Métriques sous forme de graphique, configurez vos statistiques, périodes et autres options préférées.



Pour interroger une métrique d'observabilité à l'aide AWS CLI de :

1. Créez un fichier JSON de définition de métrique et remplacez `your-Glue-job-name` et `your-Glue-job-run-id` par les vôtres.

```
$ cat multiplequeries.json
[
    {
        "Id": "avgWorkerUtil_0",
        "MetricStat": {
            "Metric": {
```

```
                "Namespace": "Glue",
                "MetricName": "glue.driver.workerUtilization",
                "Dimensions": [
                    {
                        "Name": "JobName",
                        "Value": "<your-Glue-job-name-A>"
                    },
                    {
                        "Name": "JobRunId",
                        "Value": "<your-Glue-job-run-id-A>"
                    },
                    {
                        "Name": "Type",
                        "Value": "gauge"
                    },
                    {
                        "Name": "ObservabilityGroup",
                        "Value": "resource_utilization"
                    }
                ]
            },
            "Period": 1800,
            "Stat": "Minimum",
            "Unit": "None"
        }
    },
    {
        "Id": "avgWorkerUtil_1",
        "MetricStat": {
            "Metric": {
                "Namespace": "Glue",
                "MetricName": "glue.driver.workerUtilization",
                "Dimensions": [
                    {
                        "Name": "JobName",
                        "Value": "<your-Glue-job-name-B>"
                    },
                    {
                        "Name": "JobRunId",
                        "Value": "<your-Glue-job-run-id-B>"
                    },
                    {
                        "Name": "Type",
                        "Value": "gauge"
```

```
                    },
                    {
                        "Name": "ObservabilityGroup",
                        "Value": "resource_utilization"
                    }
                ]
            },
            "Period": 1800,
            "Stat": "Minimum",
            "Unit": "None"
        }
    }
]
```

2. Exécutez la commande `get-metric-data` :

```
$ aws cloudwatch get-metric-data --metric-data-queries file: //multiplequeries.json
 \
     --start-time '2023-10-28T18: 20' \
     --end-time '2023-10-28T19: 10'  \
     --region us-east-1
{
    "MetricDataResults": [
        {
            "Id": "avgWorkerUtil_0",
            "Label": "<your-label-for-A>",
            "Timestamps": [
                "2023-10-28T18:20:00+00:00"
            ],
            "Values": [
                0.06718750000000001
            ],
            "StatusCode": "Complete"
        },
        {
            "Id": "avgWorkerUtil_1",
            "Label": "<your-label-for-B>",
            "Timestamps": [
                "2023-10-28T18:50:00+00:00"
            ],
            "Values": [
                0.5959183673469387
```

```
            ],
            "StatusCode": "Complete"
        }
    ],
    "Messages": []
}
```

Métriques d'observabilité

AWS GlueL'observabilité établit des profils et envoie les métriques suivantes Amazon CloudWatch toutes les 30 secondes, et certaines de ces métriques peuvent être visibles sur la page de surveillance des AWS Glue Studio Job Runs.

Métrique	Description	Catégorie
glue.driver.skewness.stage	Catégorie de métrique : job_performance L'asymétrie d'exécution des étapes Spark : cette métrique indique la durée maximale d'une tâche à une étape donnée par rapport à la durée médiane d'une tâche à cette étape. Elle capture l'asymétrie d'exécution, qui peut être causée par une asymétrie des données d'entrée ou par une transformation (par exemple, une jointure asymétrique). Les valeurs de cette métrique se situent dans l'intervalle [0, infini], où 0 signifie que le rapport entre le temps d'exécution maximum et médian des tâches, parmi toutes les tâches de l'étape,	job_performance

Métrique	Description	Catégorie
	est inférieur à un certain facteur d'asymétrie d'étape. Le facteur d'asymétrie d'étape par défaut est « 5 » et il peut être modifié via la configura tion Spark : spark.metrics.conf .driver.source.glue.jobPerf ormance.skewnessFactor Une valeur d'asymétrie d'étape de 1 signifie que le rapport est le double du facteur d'asymétrie d'étape. La valeur de l'asymétrie d'étape est mise à jour toutes les 30 secondes pour refléter l'asymétrie actuelle. La valeur en fin d'étape reflète l'asymétr ie finale de l'étape. Cette métrique au niveau de l'étape est utilisée pour calculer la métrique `glue.driver.skewne ss.job` au niveau de la tâche. Dimensions valides : JobName (le nom du AWS Glue Job), JobRunId (l' JobRun ID ou ALL), le type (gauge) et ObservabilityGroup (job_performance)	

Métrique	Description	Catégorie
	Statistiques valides : moyenne, maximum, minimum, centile Unité : nombre	

Métrique	Description	Catégorie
glue.driver.skewness.job	Catégorie de métrique : job_performance L'asymétrie de la tâche est le maximum de l'asymétr ie pondérée de toutes les étapes. L'asymétrie d'étape (glue.driver.skewness.stage) est pondérée en fonction de la durée de l'étape. Cela permet d'éviter le cas particulier où une étape très asymétrique s'exécute en réalité pendant un temps très court par rapport aux autres étapes (et donc son asymétrie n'est pas significative pour la performan ce globale de la tâche et ne mérite pas l'effort de tenter de la corriger). Cette métrique est mise à jour à la fin de chaque étape. La dernière valeur reflète donc l'asymétrie globale réelle de la tâche. Dimensions valides : JobName (le nom du AWS Glue Job), JobRunId (l' JobRun ID ou ALL), le type (gauge) et ObservabilityGroup (job_performance)	job_performance

Métrique	Description	Catégorie
	Statistiques valides : moyenne, maximum, minimum, centile Unité : nombre	
glue.succeed.ALL	Catégorie de métrique : erreur Nombre total d'exécutions de tâches réussies, pour compléter le tableau des catégories d'échec Dimensions valides : JobName (le nom du AWS Glue Job), JobRunId (l' JobRun ID ou ALL), le type (nombre) et Observabi lityGroup (erreur) Statistiques valides : SUM Unité : nombre	error

Métrique	Description	Catégorie
glue.error.ALL	Catégorie de métrique : erreur Nombre total d'erreurs d'exécution de tâche, pour compléter le tableau des catégories d'échec Dimensions valides : JobName (le nom du AWS Glue Job), JobRunId (l' JobRun ID ou ALL), le type (nombre) et Observabi lityGroup (erreur) Statistiques valides : SUM Unité : nombre	error

Métrique	Description	Catégorie
glue.error.[error category]	Catégorie de métrique : erreur Il s'agit en fait d'un ensemble de métriques, qui sont mises à jour uniquement lorsqu'une exécution de tâche échoue. La catégorisation des erreurs facilite le triage et le débogage. Lorsqu'une exécution de tâche échoue, l'erreur à l'origine de l'échec est catégorisée et la métrique de la catégorie d'erreur correspondante est définie à 1. Cela permet d'effectuer une analyse des échecs au fil du temps, ainsi qu'une analyse des erreurs sur l'ensemble des tâches, afin d'identifier les catégories d'échec les plus courantes et de commencer à y remédier. AWS Glue compte 28 catégories d'erreur, dont OUT_OF_MEMORY (manque de mémoire) (pilote et exécuteur), PERMISSIO N (autorisation), SYNTAX (syntaxe) et THROTTLING (limitation). Les catégories d'erreur incluent également les catégories d'erreur COMPILATION, LAUNCH et TIMEOUT.	error

Métrique	Description	Catégorie
	Dimensions valides : JobName (le nom du AWS Glue Job), JobRunId (l' JobRun ID ou ALL), le type (nombre) et Observabi lityGroup (erreur) Statistiques valides : SUM Unité : nombre	
glue.driver.workerUtilization	Catégorie de métrique : resource_utilization Le pourcentage des travaille urs affectés qui sont réellemen t utilisés. Si ce n'est pas bon, l'autoscaling peut vous aider. Dimensions valides : JobName (le nom du AWS Glue Job), JobRunId (l' JobRun ID ou ALL), le type (gauge) et ObservabilityGroup (resource_utilization) Statistiques valides : moyenne, maximum, minimum, centile Unité : pourcentage	resource_utilization

Métrique	Description	Catégorie
glue.driver.memory.heap.[available | used]	Catégorie de métrique : resource_utilization Tas disponible/utilisé par le pilote pendant l’exécution de la tâche. Cela aide à comprendre les tendances d’utilisation de la mémoire, particulièrement au fil du temps, ce qui peut contribuer à éviter des échecs potentiels, en plus de déboguer les échecs liés à la mémoire. Dimensions valides : JobName (le nom du AWS Glue Job), JobRunId (l'JobRun ID ou ALL), le type (gauge) et ObservabilityGroup (resource_utilization) Statistiques valides : moyenne Unité : octets	resource_utilization

Métrique	Description	Catégorie
glue.driver.memory.heap.used.percentage	Catégorie de métrique : resource_utilization Tas utilisé (%) par le pilote pendant l’exécution de la tâche. Cela aide à comprendre les tendances d’utilisation de la mémoire, particulièrement au fil du temps, ce qui peut contribuer à éviter des échecs potentiels, en plus de déboguer les échecs liés à la mémoire. Dimensions valides : JobName (le nom du AWS Glue Job), JobRunId (l'JobRun ID ou ALL), le type (gauge) et ObservabilityGroup (resource_utilization) Statistiques valides : moyenne Unité : pourcentage	resource_utilization

Métrique	Description	Catégorie
glue.driver.memory.non-heap.[available | used]	Catégorie de métrique : resource_utilization Mémoire hors tas disponible/utilisée par le pilote pendant l'exécution de la tâche. Cela aide à comprendre les tendances d'utilisation de la mémoire, particuli èrement au fil du temps, ce qui peut contribuer à éviter des échecs potentiels, en plus de déboguer les échecs liés à la mémoire. Dimensions valides : JobName (le nom du AWS Glue Job), JobRunId (l' JobRun ID ou ALL), le type (gauge) et ObservabilityGroup (resource_utilization) Statistiques valides : moyenne Unité : octets	resource_utilization

Métrique	Description	Catégorie
glue.driver.memory.non-heap.used.percentage	Catégorie de métrique : resource_utilization Mémoire hors tas utilisée (%) par le pilote pendant l’exécution de la tâche. Cela aide à comprendre les tendances d’utilisation de la mémoire, particulièrement au fil du temps, ce qui peut contribuer à éviter des échecs potentiels, en plus de déboguer les échecs liés à la mémoire. Dimensions valides : JobName (le nom du AWS Glue Job), JobRunId (l'JobRun ID ou ALL), le type (gauge) et ObservabilityGroup (resource_utilization) Statistiques valides : moyenne Unité : pourcentage	resource_utilization

Métrique	Description	Catégorie
glue.driver.memory.total.[available | used]	Catégorie de métrique : resource_utilization Mémoire totale disponible/utilisée par le pilote pendant l'exécution de la tâche. Cela aide à comprendre les tendances d'utilisation de la mémoire, particuli èrement au fil du temps, ce qui peut contribuer à éviter des échecs potentiels, en plus de déboguer les échecs liés à la mémoire. Dimensions valides : JobName (le nom du AWS Glue Job), JobRunId (l' JobRun ID ou ALL), le type (gauge) et ObservabilityGroup (resource_utilization) Statistiques valides : moyenne Unité : octets	resource_utilization

Métrique	Description	Catégorie
glue.driver.memory.total.used.percentage	Catégorie de métrique : resource_utilization Mémoire totale utilisée (%) par le pilote pendant l’exécution de la tâche. Cela aide à comprendre les tendances d’utilisation de la mémoire, particulièrement au fil du temps, ce qui peut contribuer à éviter des échecs potentiels, en plus de déboguer les échecs liés à la mémoire. Dimensions valides : JobName (le nom du AWS Glue Job), JobRunId (l' JobRun ID ou ALL), le type (gauge) et ObservabilityGroup (resource_utilization) Statistiques valides : moyenne Unité : pourcentage	resource_utilization

Métrique	Description	Catégorie
glue.ALL.memory.heap.[available | used]	Catégorie de métrique : resource_utilization Le tas de mémoire des exécuteurs testamentaires available/used . ALL signifie tous les exécuteurs. Dimensions valides : JobName (le nom du AWS Glue Job), JobRunId (l' JobRun ID ou ALL), le type (gauge) et ObservabilityGroup (resource_utilization) Statistiques valides : moyenne Unité : octets	resource_utilization
glue.ALL.memory.heap.used.percentage	Catégorie de métrique : resource_utilization Tas utilisé (%) par les exécuteurs. ALL signifie tous les exécuteurs. Dimensions valides : JobName (le nom du AWS Glue Job), JobRunId (l' JobRun ID ou ALL), le type (gauge) et ObservabilityGroup (resource_utilization) Statistiques valides : moyenne Unité : pourcentage	resource_utilization

Métrique	Description	Catégorie
glue.ALL.memory.non-heap.[available | used]	Catégorie de métrique : resource_utilization La mémoire available/used hors tas des exécuteurs. ALL signifie tous les exécuteurs. Dimensions valides : JobName (le nom du AWS Glue Job), JobRunId (l' JobRun ID ou ALL), le type (gauge) et ObservabilityGroup (resource_utilization) Statistiques valides : moyenne Unité : octets	resource_utilization
glue.ALL.memory.non-heap.used.percentage	Catégorie de métrique : resource_utilization Mémoire hors tas /utilisée (%) par les exécuteurs. ALL signifie tous les exécuteurs. Dimensions valides : JobName (le nom du AWS Glue Job), JobRunId (l' JobRun ID ou ALL), le type (gauge) et ObservabilityGroup (resource_utilization) Statistiques valides : moyenne Unité : pourcentage	resource_utilization

Métrique	Description	Catégorie
glue.ALL.memory.total.[available | used]	Catégorie de métrique : resource_utilization Mémoire available/used totale des exécuteurs. ALL signifie tous les exécuteurs. Dimensions valides : JobName (le nom du AWS Glue Job), JobRunId (l' JobRun ID ou ALL), le type (gauge) et ObservabilityGroup (resource_utilization) Statistiques valides : moyenne Unité : octets	resource_utilization
glue.ALL.memory.total.used.percentage	Catégorie de métrique : resource_utilization Mémoire totale utilisée (%) par les exécuteurs. ALL signifie tous les exécuteurs. Dimensions valides : JobName (le nom du AWS Glue Job), JobRunId (l' JobRun ID ou ALL), le type (gauge) et ObservabilityGroup (resource_utilization) Statistiques valides : moyenne Unité : pourcentage	resource_utilization

Métrique	Description	Catégorie
glue.driver.disk.[available_GB | used_GB]	Catégorie de métrique : resource_utilization L'espace available/used disque du pilote pendant l'exécution de la tâche. Cela aide à comprendre les tendances d’utilisation du disque, particulièrement au fil du temps, ce qui peut contribuer à éviter des échecs potentiels, en plus de déboguer les échecs liés à un espace disque insuffisant. Dimensions valides : JobName (le nom du AWS Glue Job), JobRunId (l' JobRun ID ou ALL), le type (gauge) et ObservabilityGroup (resource_utilization) Statistiques valides : moyenne Unité : gigaoctets	resource_utilization

Métrique	Description	Catégorie
glue.driver.disk.used.percentage]	Catégorie de métrique : resource_utilization L'espace available/used disque du pilote pendant l'exécution de la tâche. Cela aide à comprendre les tendances d’utilisation du disque, particulièrement au fil du temps, ce qui peut contribuer à éviter des échecs potentiels, en plus de déboguer les échecs liés à un espace disque insuffisant. Dimensions valides : JobName (le nom du AWS Glue Job), JobRunId (l' JobRun ID ou ALL), le type (gauge) et ObservabilityGroup (resource_utilization) Statistiques valides : moyenne Unité : pourcentage	resource_utilization

Métrique	Description	Catégorie
glue.ALL.disk.[available_GB | used_GB]	Catégorie de métrique : resource_utilization L' available/used espace disque des exécuteurs. ALL signifie tous les exécuteurs. Dimensions valides : JobName (le nom du AWS Glue Job), JobRunId (l' JobRun ID ou ALL), le type (gauge) et ObservabilityGroup (resource_utilization) Statistiques valides : moyenne Unité : gigaoctets	resource_utilization
glue.ALL.disk.used.percentage	Catégorie de métrique : resource_utilization Espace disque available/used/used (%) des exécuteurs. ALL signifie tous les exécuteurs. Dimensions valides : JobName (le nom du AWS Glue Job), JobRunId (l' JobRun ID ou ALL), le type (gauge) et ObservabilityGroup (resource_utilization) Statistiques valides : moyenne Unité : pourcentage	resource_utilization

Métrique	Description	Catégorie
glue.driver.bytesRead	Catégorie de métrique : débit Nombre d'octets lus par source d'entrée lors de cette exécution de tâche, ainsi que pour TOUTES les sources. Cela aide à comprendre le volume de données et ses variations au fil du temps, ce qui est utile pour résoudre des problèmes tels que l'asymétrie des données. Dimensions valides : JobName (le nom du AWS Glue Job), JobRunId (l' JobRun ID ou ALL), le type (jauge), ObservabilityGroup (resource_utilization) et la source (emplacement des données source) Statistiques valides : moyenne Unité : octets	débit

Métrique	Description	Catégorie
glue.driver.[recordsRead | filesRead]	Catégorie de métrique : débit Le nombre de records/files lectures par source d'entrée dans ce travail exécuté, ainsi que pour TOUTES les sources. Cela aide à comprendre le volume de données et ses variations au fil du temps, ce qui est utile pour résoudre des problèmes tels que l’asymétrie des données. Dimensions valides : JobName (le nom du AWS Glue Job), JobRunId (l' JobRun ID ou ALL), le type (jauge), ObservabilityGroup (resource_utilization) et la source (emplacement des données source) Statistiques valides : moyenne Unité : nombre	débit

Métrique	Description	Catégorie
glue.driver.partitionsRead	Catégorie de métrique : débit Nombre de partitions lues par source d'entrée Amazon S3 dans cette exécution de tâche, ainsi que pour TOUTES les sources. Dimensions valides : JobName (le nom du AWS Glue Job), JobRunId (l' JobRun ID ou ALL), le type (jauge), ObservabilityGroup (resource_utilization) et la source (emplacement des données source) Statistiques valides : moyenne Unité : nombre	débit

Métrique	Description	Catégorie
glue.driver.bytesWrittten	Catégorie de métrique : débit Nombre d'octets écrits par récepteur de sortie dans cette exécution de tâche, ainsi que pour TOUS les récepteurs. Cela aide à comprendre le volume de données et son évolution au fil du temps, ce qui est utile pour résoudre des problèmes tels que l'asymétrie de traitement. Dimensions valides : JobName (nom du AWS Glue Job), JobRunId (l' JobRun ID ou ALL), Type (gauge), (resource_utilization) et Sink ObservabilityGroup (emplacement des données du récepteur) Statistiques valides : moyenne Unité : octets	débit

Métrique	Description	Catégorie
glue.driver.[recordsWritten | filesWritten]	Catégorie de métrique : débit Le nombre de récepteur s records/files écrits par récepteur de sortie lors de cette exécution de travail, ainsi que pour TOUS les récepteur s. Cela aide à comprendre le volume de données et son évolution au fil du temps, ce qui est utile pour résoudre des problèmes tels que l’asymétrie de traitement. Dimensions valides : JobName (nom du AWS Glue Job), JobRunId (l' JobRun ID ou ALL), Type (gauge), (resource_utilization) et Sink ObservabilityGroup (emplacement des données du récepteur) Statistiques valides : moyenne Unité : nombre	débit

Catégories d’erreur

Catégories d’erreur	Description
COMPILATION_ERROR	Des erreurs surviennent lors de la compilation du code Scala.

Catégories d'erreur	Description
CONNECTION_ERROR	Des erreurs se produisent lors de la connexion à un service/remote hébergeur ou à un service de base de données, etc.
DISK_NO_SPACE_ERROR	Des erreurs surviennent lorsqu'il n'y a plus d'espace disponible sur le disque du pilote/de l'exécuteur.
OUT_OF_MEMORY_ERROR	Des erreurs surviennent lorsqu'il n'y a plus d'espace disponible dans la mémoire du pilote/de l'exécuteur.
IMPORT_ERROR	Des erreurs surviennent lors de l'importation de dépendances.
INVALID_ARGUMENT_ERROR	Des erreurs surviennent lorsque les arguments d'entrée ne sont pas valides/sont illégaux.
PERMISSION_ERROR	Les erreurs surviennent en cas d'absence d'autorisation d'accès au service, aux données, etc.
RESOURCE_NOT_FOUND_ERROR	Des erreurs surviennent lorsque les données, l'emplacement, etc. ne sont pas disponibles.
QUERY_ERROR	Des erreurs surviennent lors de l'exécution de requêtes SQL dans Spark.
SYNTAX_ERROR	Des erreurs surviennent lorsqu'il y a une erreur de syntaxe dans le script.
THROTTLING_ERROR	Des erreurs surviennent lorsqu'une limite de simultanéité de service est atteinte ou qu'une limitation de quota de service est dépassée.

Catégories d'erreur	Description
DATA_LAKE_FRAMEWORK_ERROR	Des erreurs surviennent à partir d'un environnement de lac de données pris en charge nativement par AWS Glue, comme Hudi, Iceberg, etc.
UNSUPPORTED_OPERATION_ERROR	Des erreurs surviennent lors d'une opération non prise en charge.
RESOURCES_ALREADY_EXISTS_ERROR	Des erreurs surviennent lorsqu'une ressource à créer ou à ajouter existe déjà.
GLUE_INTERNAL_SERVICE_ERROR	Les erreurs surviennent lorsqu'il y a un problème de service interne de AWS Glue.
GLUE_OPERATION_TIMEOUT_ERROR	Des erreurs surviennent lorsqu'une opération AWS Glue dépasse le délai imparti.
GLUE_VALIDATION_ERROR	Des erreurs surviennent lorsqu'une valeur requise ne peut pas être validée pour la tâche AWS Glue.
GLUE_JOB_BOOKMARK_VERSION_MISMATCH_ERROR	Des erreurs surviennent lorsque le même job exécute le même bucket source et écrit simultanément sur la same/different destination (simultanéité >1)
LAUNCH_ERROR	Des erreurs surviennent lors de la phase de lancement de la tâche AWS Glue.
DYNAMODB_ERROR	Les erreurs génériques proviennent du Amazon DynamoDB service.
GLUE_ERROR	Les erreurs génériques proviennent du service AWS Glue.
LAKEFORMATION_ERROR	Les erreurs génériques proviennent du AWS Lake Formation service.

Catégories d'erreur	Description
REDSHIFT_ERROR	Les erreurs génériques proviennent du Amazon Redshift service.
S3_ERROR	Les erreurs génériques proviennent du service Amazon S3.
SYSTEM_EXIT_ERROR	Erreur de sortie du système générique.
TIMEOUT_ERROR	Des erreurs génériques surviennent lorsque la tâche échoue en raison du dépassement du délai d'exécution de l'opération.
UNCLASSIFIED_SPARK_ERROR	Les erreurs génériques proviennent de Spark.
UNCLASSIFIED_ERROR	Catégorie d'erreur par défaut.

Limitations

> **Note**
>
> `glueContext` doit être initialisé pour publier les métriques.

Dans la dimension source, la valeur est soit le chemin Amazon S3, soit le nom de la table, selon le type de source. En outre, si la source est JDBC et que l'option de requête est utilisée, la chaîne de requête est définie dans la dimension source. Si la valeur est supérieure à 500 caractères, elle est réduite à 500 caractères maximum. Les limites de la valeur sont les suivantes :

- Les caractères non ASCII seront supprimés.
- Si le nom de la source ne contient aucun caractère ASCII, il est converti en <non-ASCII input>.

Limites et considérations relatives aux métriques de débit

- DataFrame et DataFrame basés DynamicFrame (par exemple JDBC, lecture depuis parquet sur Amazon S3) sont pris en charge, mais les fichiers basés sur RDD DynamicFrame (par exemple,

lecture de csv, de json sur Amazon S3, etc.) ne sont pas pris en charge. Techniquement, toutes les lectures et écritures visibles sur l'interface utilisateur de Spark sont prises en charge.

- La métrique `recordsRead` sera émise si la source de données est une table de catalogue et que le format est JSON, CSV, texte ou Iceberg.
- Les métriques `glue.driver.throughput.recordsWritten`, `glue.driver.throughput.bytesWritten` et `glue.driver.throughput.filesWritten` ne sont pas disponibles dans les tables JDBC et Iceberg.
- Les métriques peuvent être retardées. Si le travail se termine dans environ une minute, il se peut qu'il n'y ait aucune métrique de débit dans Amazon CloudWatch Metrics.

Surveillance et débogage des tâches

Vous pouvez collecter des statistiques sur les AWS Glue offres d'emploi et les visualiser sur les AWS Glue CloudWatch consoles Amazon afin d'identifier et de résoudre les problèmes. Le profilage de vos tâches AWS Glue requiert les étapes suivantes :

1. Activation des métriques :

   a. Activer l'option de métriques de tâche dans la définition de la tâche. Vous pouvez activer le profilage dans la console AWS Glue ou en tant que paramètre de la tâche. Pour plus d'informations, consultez Définition des propriétés des tâches Spark ou Utilisation des paramètres des tâches dans les tâches AWS Glue.

   b. Activez l'option Métriques d'observabilité AWS Glue dans la définition de la tâche. Vous pouvez activer l'observabilité dans la console AWS Glue ou en tant que paramètre de la tâche. Pour de plus amples informations, veuillez consulter Surveillance à l'aide de métriques d'observabilité AWS Glue.

2. Vérifiez que le script de la tâche initialise un `GlueContext`. Par exemple, l'extrait de script suivant initialise un `GlueContext` et indique où le code profilé est placé dans le script. Ce format général est utilisé dans les scénarios de débogage suivants.

```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job
import time
```

```
## @params: [JOB_NAME]
args = getResolvedOptions(sys.argv, ['JOB_NAME'])

sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
job = Job(glueContext)
job.init(args['JOB_NAME'], args)

...
...
code-to-profile
...
...


job.commit()
```

3. Exécutez la tâche.
4. Visualisez les métriques :

   a. Visualisez les métriques sur la console AWS Glue et identifiez les métriques anormales pour le pilote ou un exécuteur.

   b. Consultez les indicateurs d'observabilité sur la page de surveillance des tâches, sur la page des détails de l'exécution des tâches ou sur Amazon CloudWatch. Pour de plus amples informations, veuillez consulter Surveillance à l'aide de métriques d'observabilité AWS Glue.
5. Précisez la cause première à l'aide de la métrique identifiée.
6. Confirmez éventuellement la cause première à l'aide du flux de journaux du pilote identifié ou du programme d'exécution.

Cas d'utilisation des métriques d'observabilité AWS Glue

- Débogage des exceptions OOM et des anomalies de tâches
- Débogage d'étapes exigeantes et de tâches de ralentissement
- Surveillance de la progression des tâches multiples
- Surveillance de la planification des capacités de DPU
- Utilisation de l'observabilité AWS Glue pour surveiller l'utilisation des ressources afin de réduire les coûts

Débogage des exceptions OOM et des anomalies de tâches

Vous pouvez déboguer les exceptions out-of-memory (OOM) et les anomalies de tâches dans. AWS Glue Les sections suivantes décrivent les scénarios de débogage des out-of-memory exceptions du pilote Apache Spark ou d'un exécuteur Spark.

- Débogage d'un pilote d'exception OOM
- Débogage d'un programme d'exécution d'exception OOM

Débogage d'un pilote d'exception OOM

Dans ce scénario, une tâche Spark lit un grand nombre de petits fichiers d'Amazon Simple Storage Service (Amazon S3). Il convertit les fichiers au format Apache Parquet, puis les écrit dans Amazon S3. Le pilote Spark manque de mémoire. Les données d'entrées Amazon S3 comptent plus d'un million de fichiers dans différentes partitions Amazon S3.

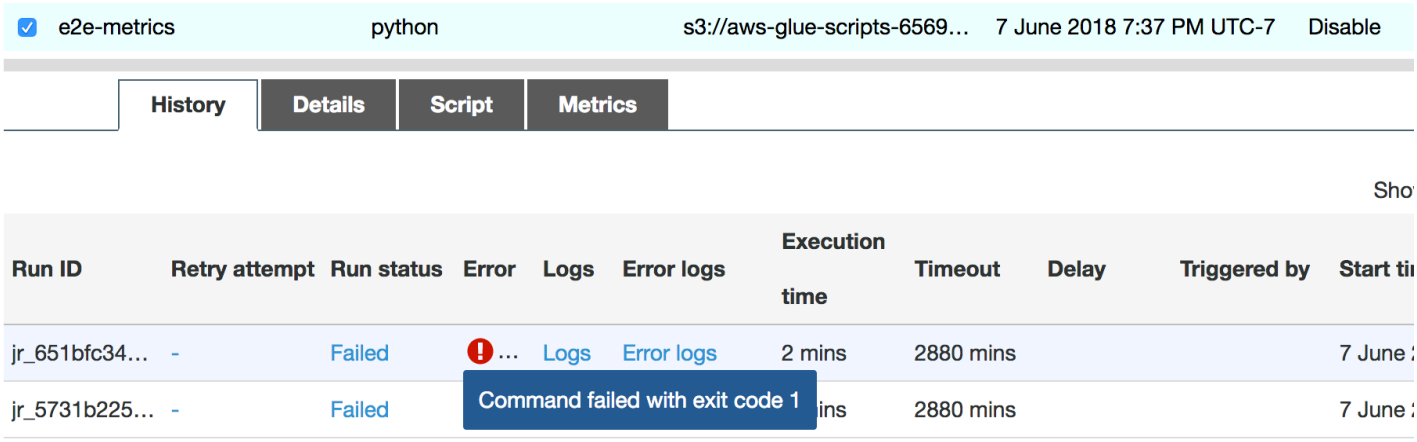
Le code profilé est le suivant :

```
data = spark.read.format("json").option("inferSchema", False).load("s3://input_path")
data.write.format("parquet").save(output_path)
```

Visualisez les métriques profilées sur la console AWS Glue

Le graphique suivant illustre l'utilisation de la mémoire sous forme de pourcentage pour le pilote et les programmes d'exécution. Cette utilisation est déterminée par un point de données qui est moyenné sur les valeurs rapportées au cours de la dernière minute. Vous pouvez voir dans le profil de la mémoire de la tâche que le pilote de la mémoire dépasse le seuil de sécurité de 50 % d'utilisation rapidement. D'autre part, l' utilisation moyenne de la mémoire sur l'ensemble des programmes d'exécution reste inférieure à 4 %. Ceci montre clairement l'anomalie de l'exécution du pilote dans cette tâche Spark.

L'exécution de la tâche ne tarde pas à échouer et l'erreur suivante apparaît dans l'onglet Historique de la console AWS Glue : Command Failed with Exit Code 1. Cette chaîne d'erreur signifie que la tâche a échoué en raison d'une erreur systémique ; dans le cas présent, le pilote manque de mémoire.



Sur la console, cliquez sur le lien Journaux d'erreurs dans l'onglet Historique pour confirmer les conclusions relatives au pilote OOM contenues dans les CloudWatch journaux. Recherchez « **`Error`** » dans les journaux d'erreurs de la tâche pour vérifier que c'est bien à cause d'une exception OOM que la tâche a échoué :

```
# java.lang.OutOfMemoryError: Java heap space
```

```
# -XX:OnOutOfMemoryError="kill -9 %p"
# Executing /bin/sh -c "kill -9 12039"...
```

Dans l'onglet Historique de la tâche, choisissez Journaux. Vous pouvez trouver la trace suivante de l'exécution du pilote dans les CloudWatch journaux au début de la tâche. Le pilote Spark essaie de répertorier tous les fichiers dans tous les répertoires, il génère un `InMemoryFileIndex`et lance une tâche par fichier. Par conséquent, le pilote Spark doit conserver une grande quantité d'état en mémoire pour suivre toutes les tâches. Il met en cache une liste complète comprenant un grand nombre de fichiers pour l'index en mémoire, ce qui se traduit par un pilote OOM.

Corriger le traitement de fichiers multiples grâce au regroupement

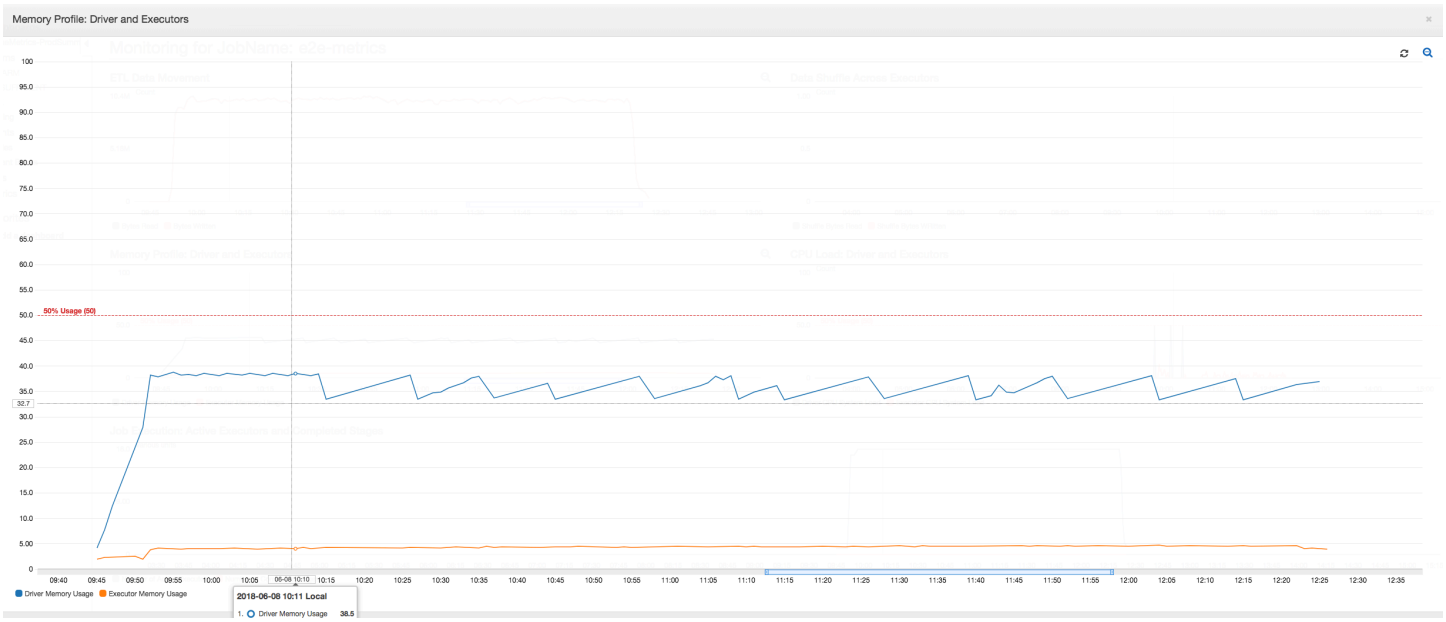
Vous pouvez corriger le traitement de plusieurs fichiers en utilisant la fonction de regroupement dans AWS Glue. Le regroupement est automatiquement activé lorsque vous utilisez des images dynamiques et lorsque l'ensemble des données d'entrée compte un grand nombre de fichiers (plus de 50 000). Le regroupement vous permet de fusionner plusieurs fichiers en un groupe, et il permet à une tâche de traiter l'ensemble du groupe plutôt qu'un seul fichier. Par conséquent, le pilote Spark stocke nettement moins d'état en mémoire pour suivre moins de tâches. Pour plus d'informations sur l'activation manuelle du regroupement pour votre ensemble de données, consultez Lecture des fichiers en entrée dans des groupes de plus grande taille.

Pour consulter le profil de la mémoire de la tâche AWS Glue, profilez le code suivant avec la fonction de regroupement activée :

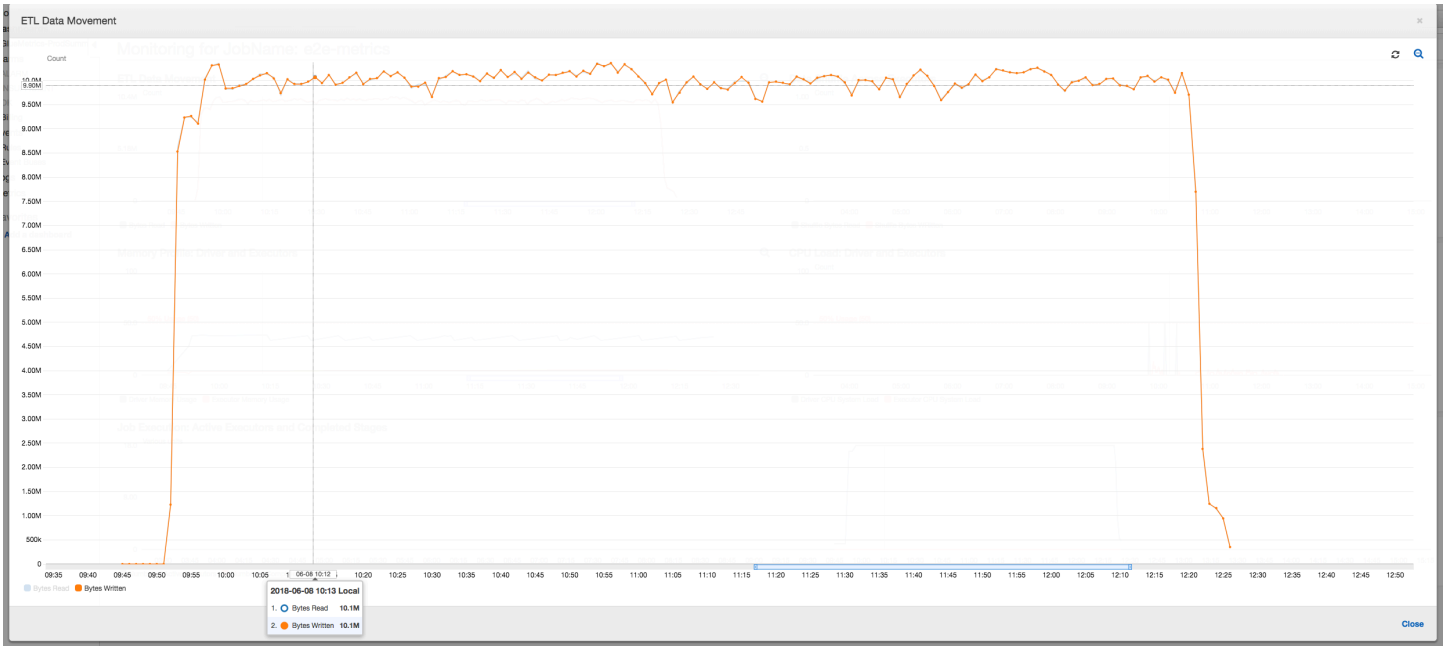
```
df = glueContext.create_dynamic_frame_from_options("s3", {'paths': ["s3://input_path"],
 "recurse":True, 'groupFiles': 'inPartition'}, format="json")
datasink = glueContext.write_dynamic_frame.from_options(frame = df, connection_type
 = "s3", connection_options = {"path": output_path}, format = "parquet",
 transformation_ctx = "datasink")
```

Vous pouvez surveiller le profil de la mémoire et le déplacement des données ETL dans le profil AWS Glue de la tâche.

Le pilote s'exécute sous le seuil de 50 % d'utilisation de la mémoire sur toute la durée de la tâche AWS Glue. Les programmes d'exécution diffusent des données depuis Amazon S3, les traitent et les écrivent dans Amazon S3. Par conséquent, ils consomment moins de 5 % de mémoire, quel que soit le moment.



Le profil de déplacement des données ci-dessous montre le nombre total d'octets d'Amazon S3 lus et écrits au cours de la dernière minute par tous les programmes d'exécution à mesure que la tâche progresse. Les deux suivent un modèle similaire car les données sont diffusées sur tous les programmes d'exécution. La tâche finir de traiter le million de fichiers en moins de trois heures.



Débogage d'un programme d'exécution d'exception OOM

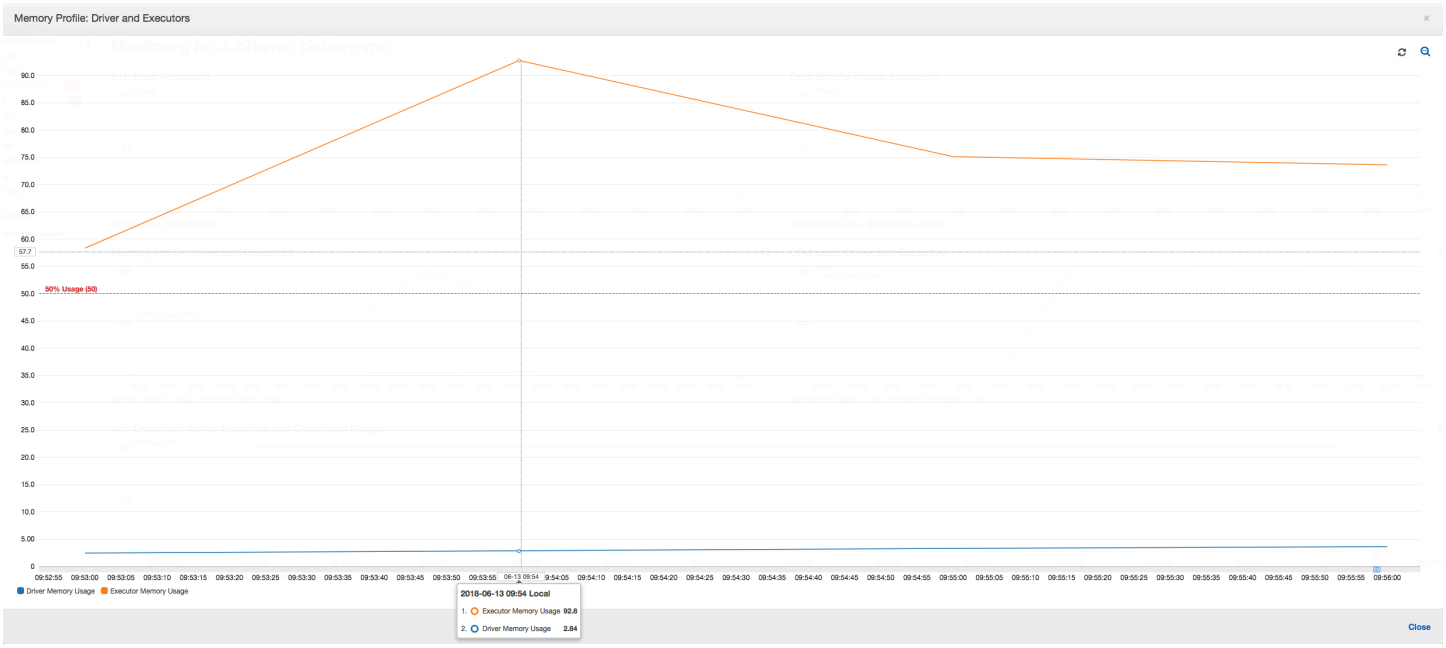
Dans ce scénario, vous pouvez apprendre à déboguer des exceptions OOM pouvant se produire dans les programmes d'exécution d'Apache Spark. Le code suivant utilise le lecteur MySQL Spark pour lire une grande table d'environ 34 millions de lignes dans une tramedonnées Spark. Il l'écrit

ensuite sur Amazon S3 au format Parquet. Vous pouvez fournir les propriétés de connexion et utiliser les configurations Spark par défaut pour lire la table.

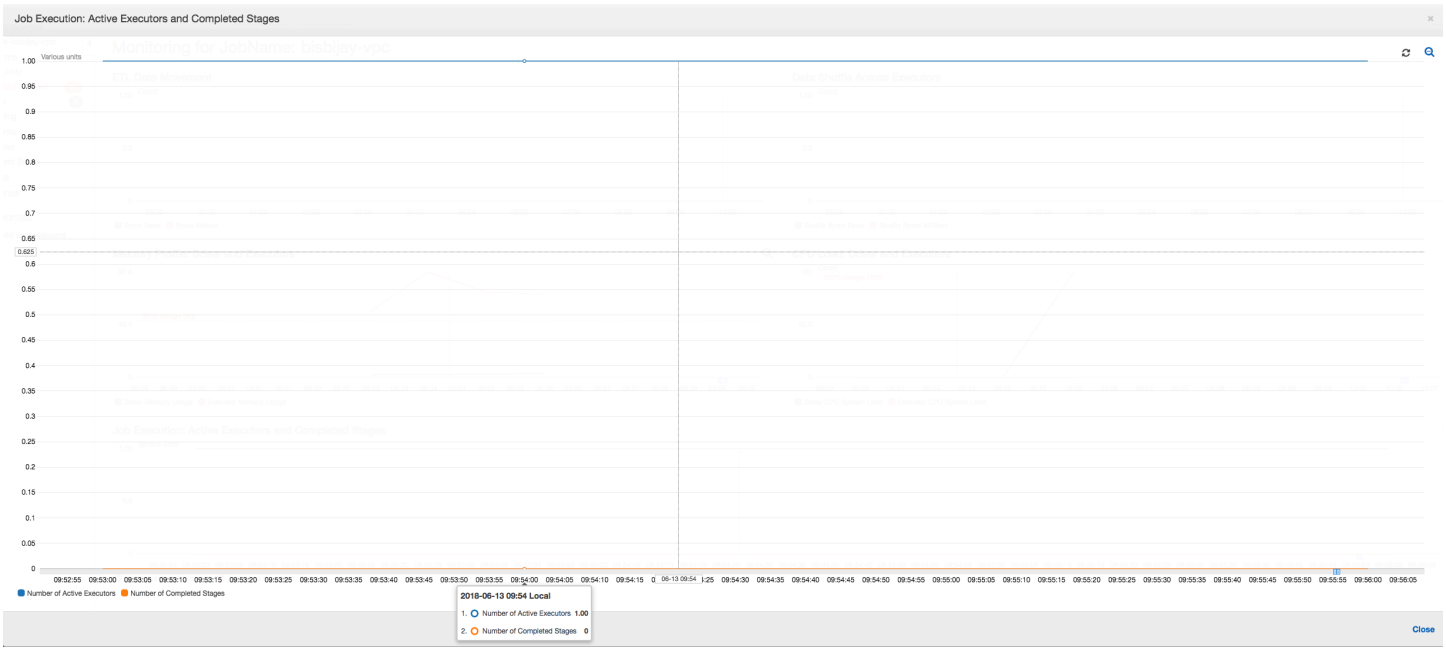
```
val connectionProperties = new Properties()
connectionProperties.put("user", user)
connectionProperties.put("password", password)
connectionProperties.put("Driver", "com.mysql.jdbc.Driver")
val sparkSession = glueContext.sparkSession
val dfSpark = sparkSession.read.jdbc(url, tableName, connectionProperties)
dfSpark.write.format("parquet").save(output_path)
```

Voir les métriques profilées sur la console AWS Glue

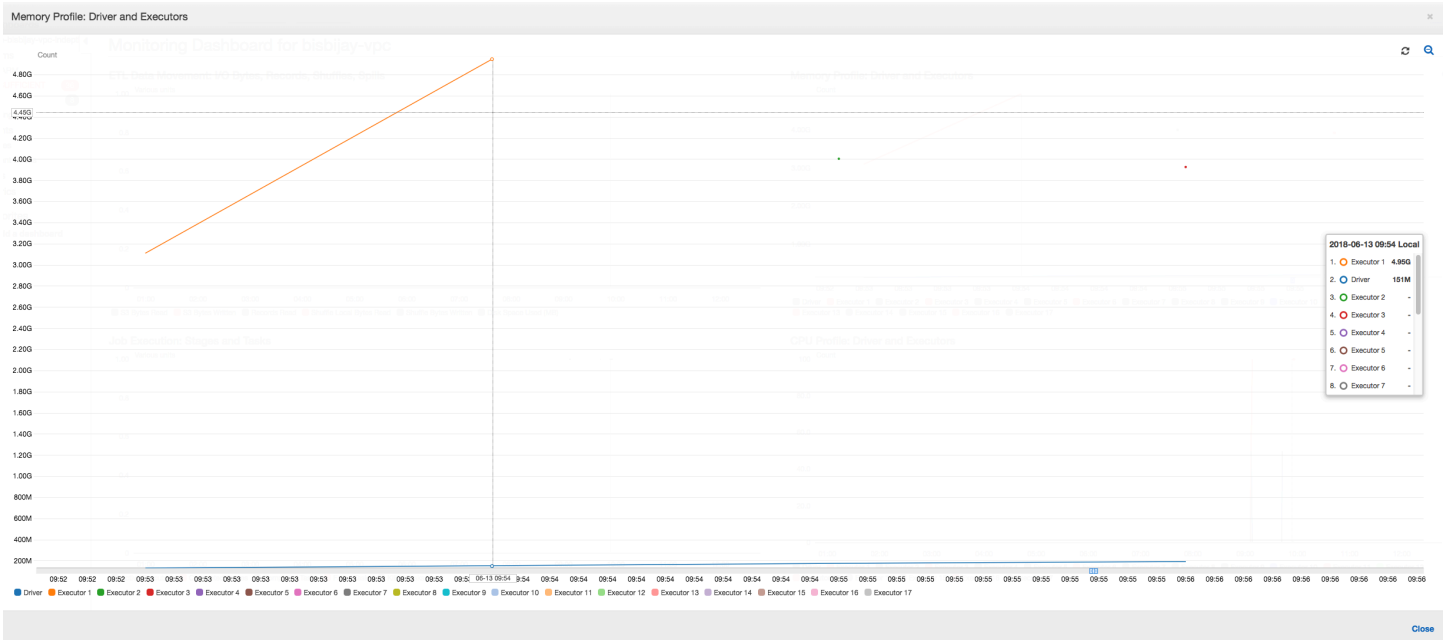
Si la pente du graphique d'utilisation de la mémoire est positive et dépasse 50 %, si la tâche échoue avant l'émission de la métrique suivante, l'épuisement de la mémoire peut en être à l'origine. Le graphique suivant montre qu'en une minute d'exécution, l'utilisation moyenne de la mémoire sur tous les programmes d'exécution dépasse rapidement 50 %. L'utilisation peut atteindre 92 % et le conteneur qui exécute le programme d'exécution est arrêté par Apache Hadoop YARN.



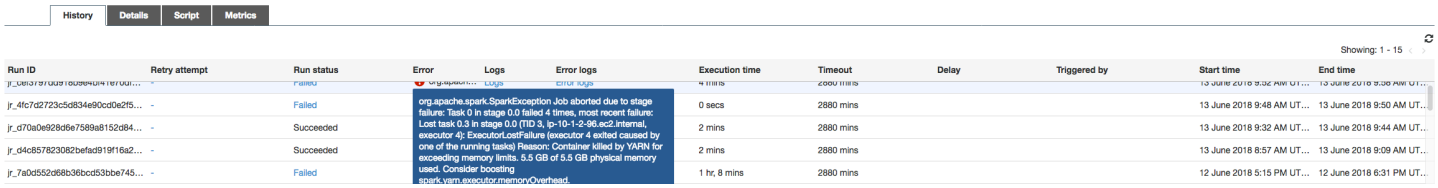
Comme le montre le graphique suivant, un programme d'exécution unique s'exécute toujours jusqu'à ce que la tâche échoue. En effet, un nouveau programme d'exécution est lancé pour remplacer celui qui a été arrêté. Les lectures de la source de données JDBC ne sont pas parallélisées par défaut, car cela nécessiterait le partitionnement de la table sur une colonne et d'ouvrir plusieurs connexions. Par conséquent, seul un programme d'exécution lit dans la table complète de manière séquentielle.



Comme le montre le graphique suivant, Spark tente de lancer une nouvelle tâche quatre fois avant l'échec de la tâche. Vous pouvez voir le profil de la mémoire de trois programmes d'exécution. Chaque programme d'exécution utilise rapidement toute sa mémoire. La quatrième programme d'exécution manque de mémoire et la tâche échoue. Par conséquent, sa métrique n'est pas immédiatement reportée.



À partir de la chaîne d'erreur sur la console AWS Glue, vous pouvez vérifier que la tâche a échoué en raison d'exceptions OOM, comme le montre l'image suivante.



Journaux de sortie des tâches : pour confirmer la découverte d'une exception OOM de l'exécuteur, consultez les CloudWatch journaux. Lorsque vous recherchez **`Error`**, vous trouvez les quatre programme d'exécution arrêtés en à peu près le même temps que la fenêtre l'affiche sur le tableau de bord des métriques. Ils sont tous résiliés par YARN à mesure qu'ils excèdent les limites de mémoire.

Programme d'exécution 1

```
18/06/13 16:54:29 WARN YarnAllocator: Container killed by YARN for exceeding
 memory limits. 5.5 GB of 5.5 GB physical memory used. Consider boosting
 spark.yarn.executor.memoryOverhead.
18/06/13 16:54:29 WARN YarnSchedulerBackend$YarnSchedulerEndpoint: Container killed
 by YARN for exceeding memory limits. 5.5 GB of 5.5 GB physical memory used. Consider
 boosting spark.yarn.executor.memoryOverhead.
18/06/13 16:54:29 ERROR YarnClusterScheduler: Lost executor 1 on
 ip-10-1-2-175.ec2.internal: Container killed by YARN for exceeding
 memory limits. 5.5 GB of 5.5 GB physical memory used. Consider boosting
 spark.yarn.executor.memoryOverhead.
18/06/13 16:54:29 WARN TaskSetManager: Lost task 0.0 in stage 0.0 (TID 0,
 ip-10-1-2-175.ec2.internal, executor 1): ExecutorLostFailure (executor 1
 exited caused by one of the running tasks) Reason: Container killed by YARN for
 exceeding memory limits. 5.5 GB of 5.5 GB physical memory used. Consider boosting
 spark.yarn.executor.memoryOverhead.
```

Programme d'exécution 2

```
18/06/13 16:55:35 WARN YarnAllocator: Container killed by YARN for exceeding
 memory limits. 5.8 GB of 5.5 GB physical memory used. Consider boosting
 spark.yarn.executor.memoryOverhead.
18/06/13 16:55:35 WARN YarnSchedulerBackend$YarnSchedulerEndpoint: Container killed
 by YARN for exceeding memory limits. 5.8 GB of 5.5 GB physical memory used. Consider
 boosting spark.yarn.executor.memoryOverhead.
18/06/13 16:55:35 ERROR YarnClusterScheduler: Lost executor 2 on
 ip-10-1-2-16.ec2.internal: Container killed by YARN for exceeding
 memory limits. 5.8 GB of 5.5 GB physical memory used. Consider boosting
 spark.yarn.executor.memoryOverhead.
18/06/13 16:55:35 WARN TaskSetManager: Lost task 0.1 in stage 0.0 (TID 1,
 ip-10-1-2-16.ec2.internal, executor 2): ExecutorLostFailure (executor 2 exited
```

```
 caused by one of the running tasks) Reason: Container killed by YARN for
 exceeding memory limits. 5.8 GB of 5.5 GB physical memory used. Consider boosting
 spark.yarn.executor.memoryOverhead.
```

Programme d'exécution 3

```
18/06/13 16:56:37 WARN YarnAllocator: Container killed by YARN for exceeding
 memory limits. 5.8 GB of 5.5 GB physical memory used. Consider boosting
 spark.yarn.executor.memoryOverhead.
18/06/13 16:56:37 WARN YarnSchedulerBackend$YarnSchedulerEndpoint: Container killed
 by YARN for exceeding memory limits. 5.8 GB of 5.5 GB physical memory used. Consider
 boosting spark.yarn.executor.memoryOverhead.
18/06/13 16:56:37 ERROR YarnClusterScheduler: Lost executor 3 on
 ip-10-1-2-189.ec2.internal: Container killed by YARN for exceeding
 memory limits. 5.8 GB of 5.5 GB physical memory used. Consider boosting
 spark.yarn.executor.memoryOverhead.
18/06/13 16:56:37 WARN TaskSetManager: Lost task 0.2 in stage 0.0 (TID 2,
 ip-10-1-2-189.ec2.internal, executor 3): ExecutorLostFailure (executor 3
 exited caused by one of the running tasks) Reason: Container killed by YARN for
 exceeding memory limits. 5.8 GB of 5.5 GB physical memory used. Consider boosting
 spark.yarn.executor.memoryOverhead.
```

Programme d'exécution 4

```
18/06/13 16:57:18 WARN YarnAllocator: Container killed by YARN for exceeding
 memory limits. 5.5 GB of 5.5 GB physical memory used. Consider boosting
 spark.yarn.executor.memoryOverhead.
18/06/13 16:57:18 WARN YarnSchedulerBackend$YarnSchedulerEndpoint: Container killed
 by YARN for exceeding memory limits. 5.5 GB of 5.5 GB physical memory used. Consider
 boosting spark.yarn.executor.memoryOverhead.
18/06/13 16:57:18 ERROR YarnClusterScheduler: Lost executor 4 on
 ip-10-1-2-96.ec2.internal: Container killed by YARN for exceeding
 memory limits. 5.5 GB of 5.5 GB physical memory used. Consider boosting
 spark.yarn.executor.memoryOverhead.
18/06/13 16:57:18 WARN TaskSetManager: Lost task 0.3 in stage 0.0 (TID 3,
 ip-10-1-2-96.ec2.internal, executor 4): ExecutorLostFailure (executor 4 exited
 caused by one of the running tasks) Reason: Container killed by YARN for
 exceeding memory limits. 5.5 GB of 5.5 GB physical memory used. Consider boosting
 spark.yarn.executor.memoryOverhead.
```

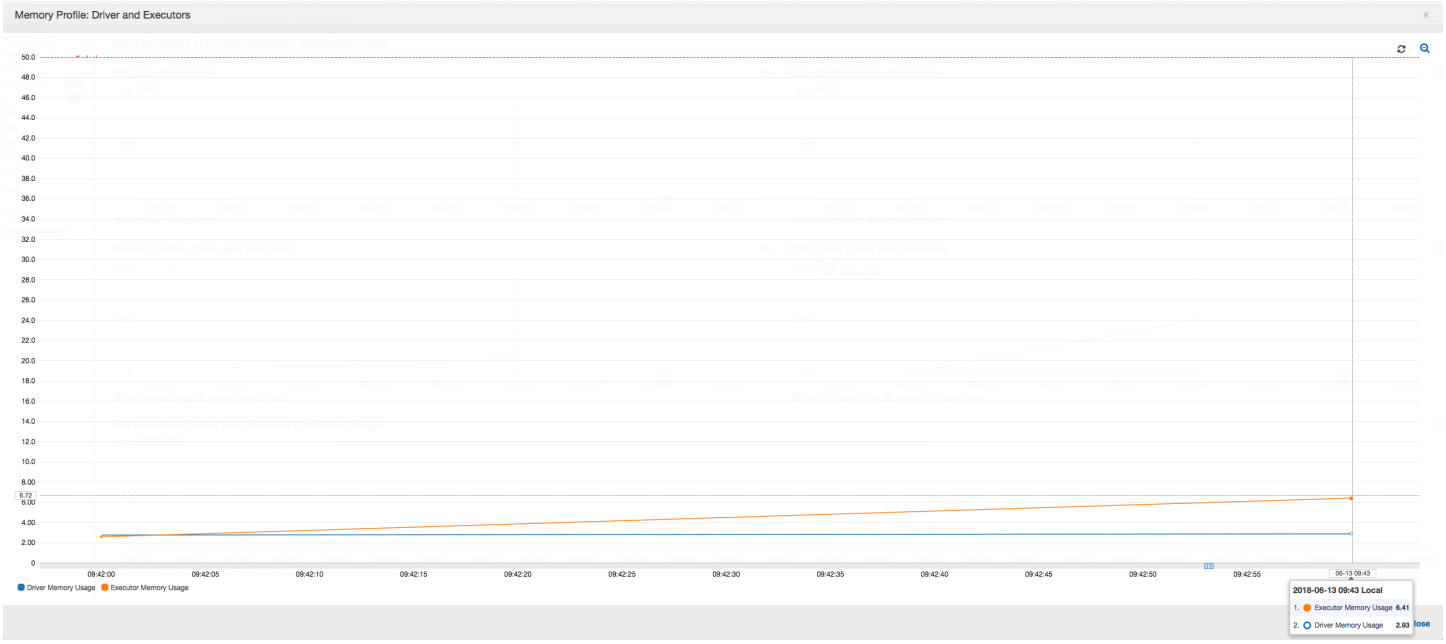
Corriger le paramètre de taille d'extraction à l'aide des images dynamiques AWS Glue

Le programme d'exécution est arrivé à court de mémoire lors de la lecture de la table JDBC, car la configuration par défaut de la taille d'extraction du JDBC de Spark est zéro. Cela signifie que le pilote JDBC sur le programme d'exécution Spark tente d'extraire les 34 millions de lignes de la base de données et de les mettre en cache, même si Spark diffuse les lignes une par une. Avec Spark, vous pouvez éviter ce scénario en définissant le paramètre de taille d'extraction par une valeur autre que zéro.

Vous pouvez également résoudre ce problème en utilisant des images dynamiques AWS Glue. Par défaut, les images dynamiques utilisent une taille d'extraction de 1 000 lignes, qui est généralement une valeur suffisante. Par conséquent, le programme d'exécution n'utilise pas plus de 7 % de la mémoire totale. La tâche AWS Glue s'achève en moins de deux minutes avec un seul programme d'exécution. L'utilisation des images dynamiques AWS Glue est l'approche recommandée, mais il est également possible de définir la taille d'extraction à l'aide de la propriété Apache Spark `fetchsize`. Consultez le guide de Spark SQL DataFrames et des ensembles de données.

```
val (url, database, tableName) = {
 ("jdbc_url", "db_name", "table_name")
 }
val source = glueContext.getSource(format, sourceJson)
val df = source.getDynamicFrame
glueContext.write_dynamic_frame.from_options(frame = df, connection_type = "s3",
 connection_options = {"path": output_path}, format = "parquet", transformation_ctx =
 "datasink")
```

Métriques profilées normales : La mémoire du programme d'exécution avec les images dynamiques d'AWS Glue ne dépasse jamais le seuil de sécurité, comme le montre l'image suivante. Il diffuse dans les lignes depuis la base de données et met en cache 1 000 lignes seulement dans le pilote JDBC quel que soit le moment. Une exception de mémoire insuffisante ne se produit pas.



Débogage d'étapes exigeantes et de tâches de ralentissement

Vous pouvez utiliser le profilage de tâche AWS Glue pour identifier les étapes exigeantes et les tâches de ralentissement dans vos tâches ETL (extraction, transformation et chargement). Un tâche de ralentissement prend beaucoup plus de temps que le reste des tâches dans l'étape d'une tâche AWS Glue. Par conséquent, l'étape s'effectue plus lentement, ce qui retarde aussi la durée d'exécution totale de la tâche.

Fusion de petits fichiers d'entrée pour obtenir de plus grands fichiers de sortie

Une tâche de ralentissement a lieu lorsque la distribution du travail au sein des différentes tâches n'est pas uniforme, ou qu'une asymétrie de données entraîne un plus important traitement de données par une tâche.

Vous pouvez profiler le code suivant un modèle commun dans Apache Spark pour fusionner un grand nombre de petits fichiers en de plus grands fichiers de sortie. Pour cet exemple, l'ensemble de données d'entrée est de 32 Go de fichiers compressés Gzip JSON. L'ensemble de données de sortie a environ 190 Go de fichiers JSON décompressés.

Le code profilé est le suivant :

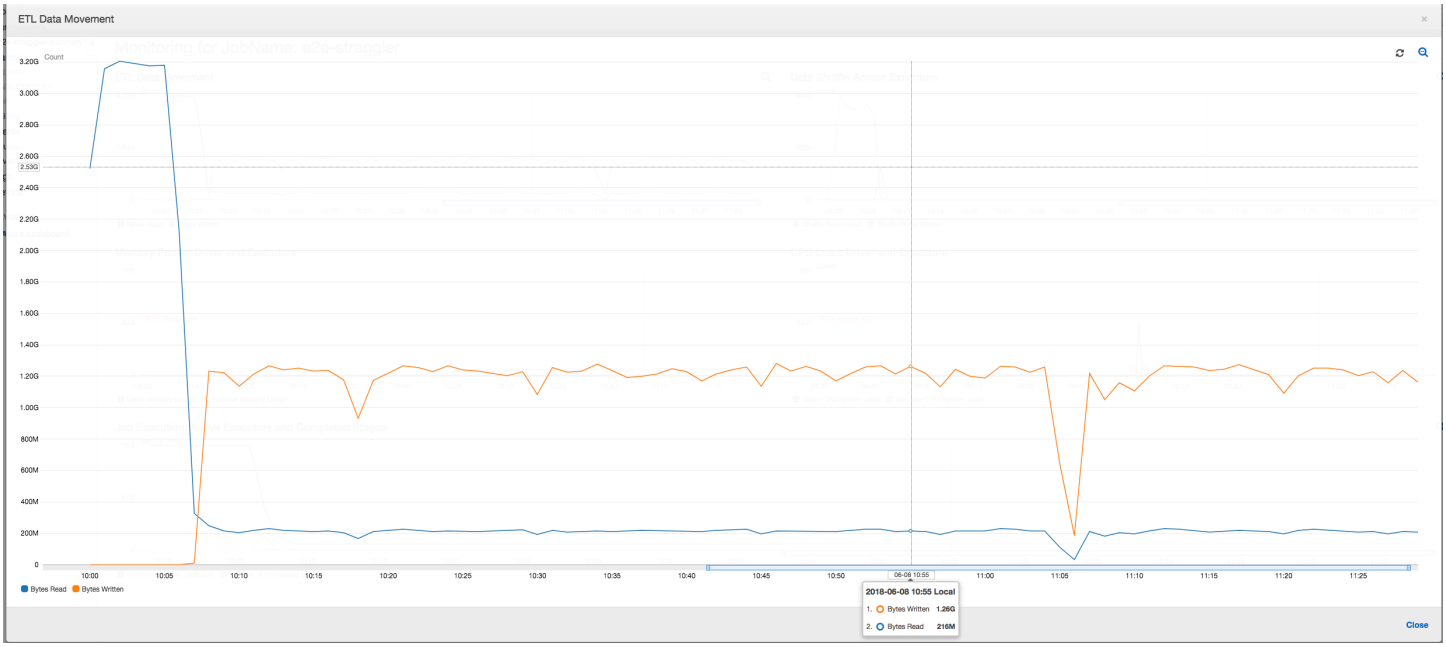
```
datasource0 = spark.read.format("json").load("s3://input_path")
df = datasource0.coalesce(1)
df.write.format("json").save(output_path)
```

Voir les métriques profilées sur la console AWS Glue

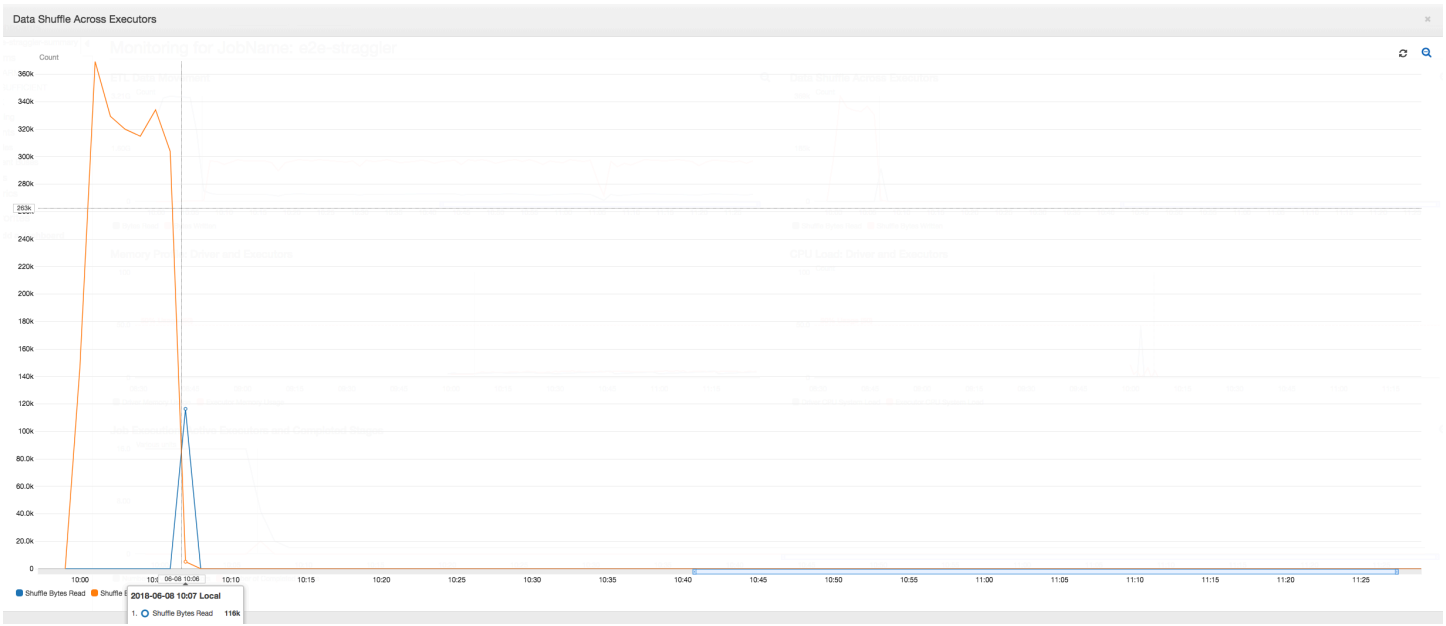
Vous pouvez profiler votre tâche pour examiner quatre différents ensembles de métriques :

- Déplacement de données ETL
- Remaniement de données sur les programmes d'exécution
- Exécution d'une tâche
- Profil de mémoire

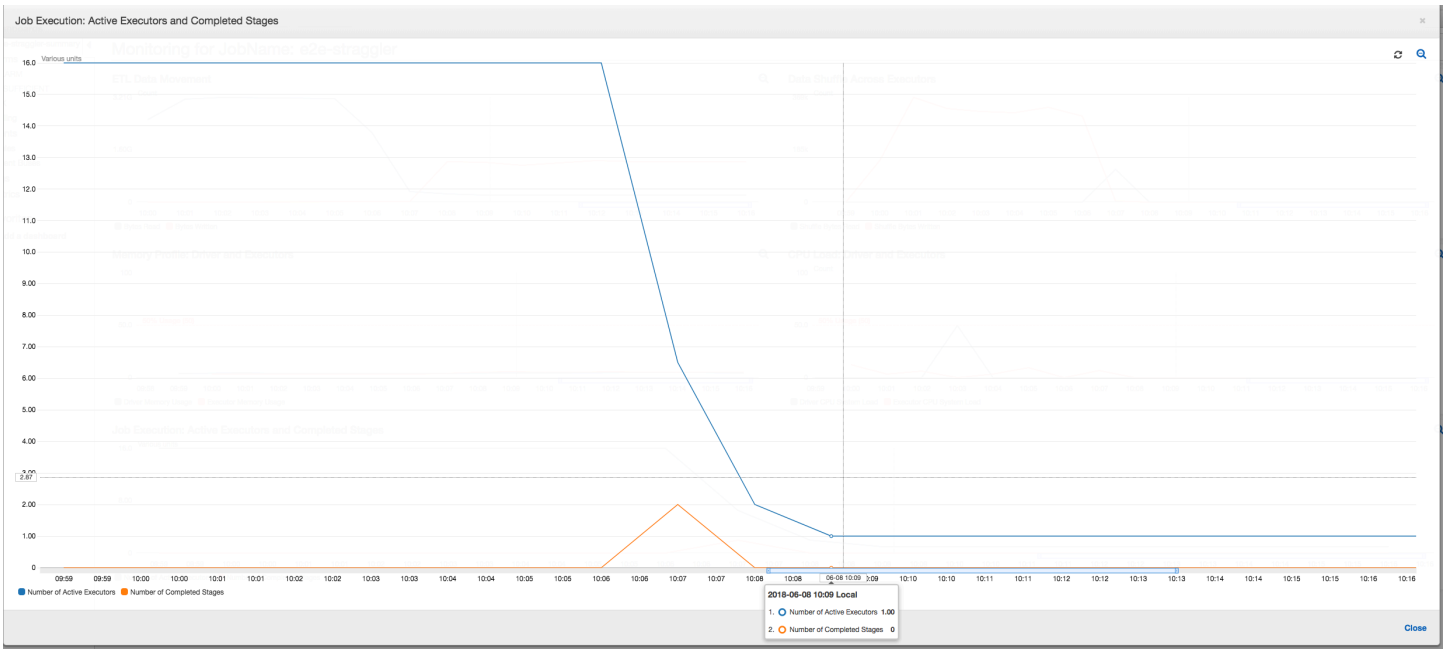
Déplacement des données ETL : Dans le profil du Déplacement des données ETL, les octets sont lus assez rapidement par tous les programmes d'exécution lors de la première étape qui se termine au cours des six premières minutes. Toutefois, la durée totale d'exécution de la tâche est d'environ une heure, et elle est principalement constituée des écritures de données.



Remaniement de données sur les programmes d'exécution : Le nombre d'octets lus et écrits au cours du remaniement montre également un pic avant la fin de l'étape 2, comme le montrent les métriques de l'Exécution de tâche et du Remaniement de données. Suite au remaniement des données de tous les programmes d'exploitation, toutes les lectures et les écritures proviennent uniquement du programme d'exécution n°3.

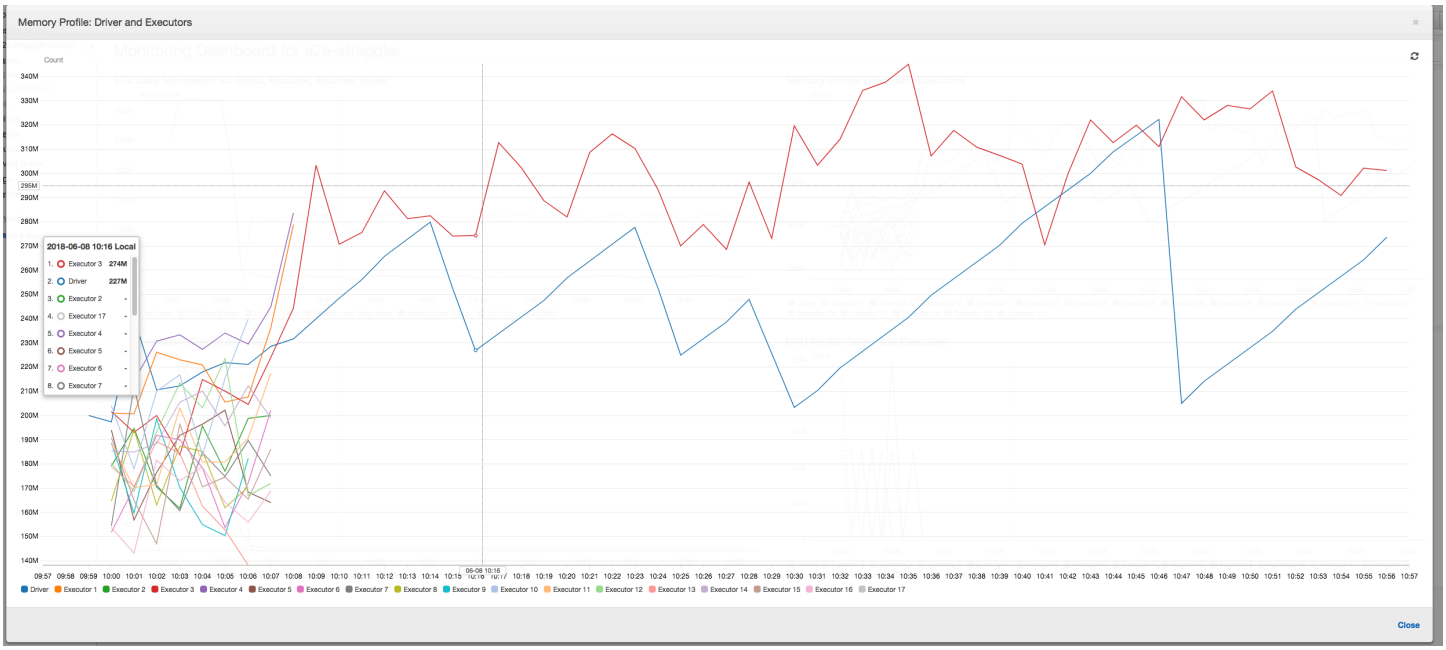


Exécution de tâche : Comme le montre le graphique ci-dessous, tous les autres programmes d'exécution sont inactifs et sont finalement abandonnés avant 10:09. A ce stade, le nombre total de programmes d'exécution diminue et il n'en reste qu'un. Ceci montre clairement que le programme d'exécution numéro 3 se compose de la tâche de ralentissement, dont la durée d'exécution est la plus longue, et contribue à la plus grande partie de la durée d'exécution de la tâche.



Profil de la mémoire : Après les deux premières étapes, seul le programme d'exécution n°3 consomme activement de la mémoire pour traiter les données. Les autres programmes d'exécution

sont simplement inactifs ou ont été abandonnés peu de temps après la fin des deux premières étapes.



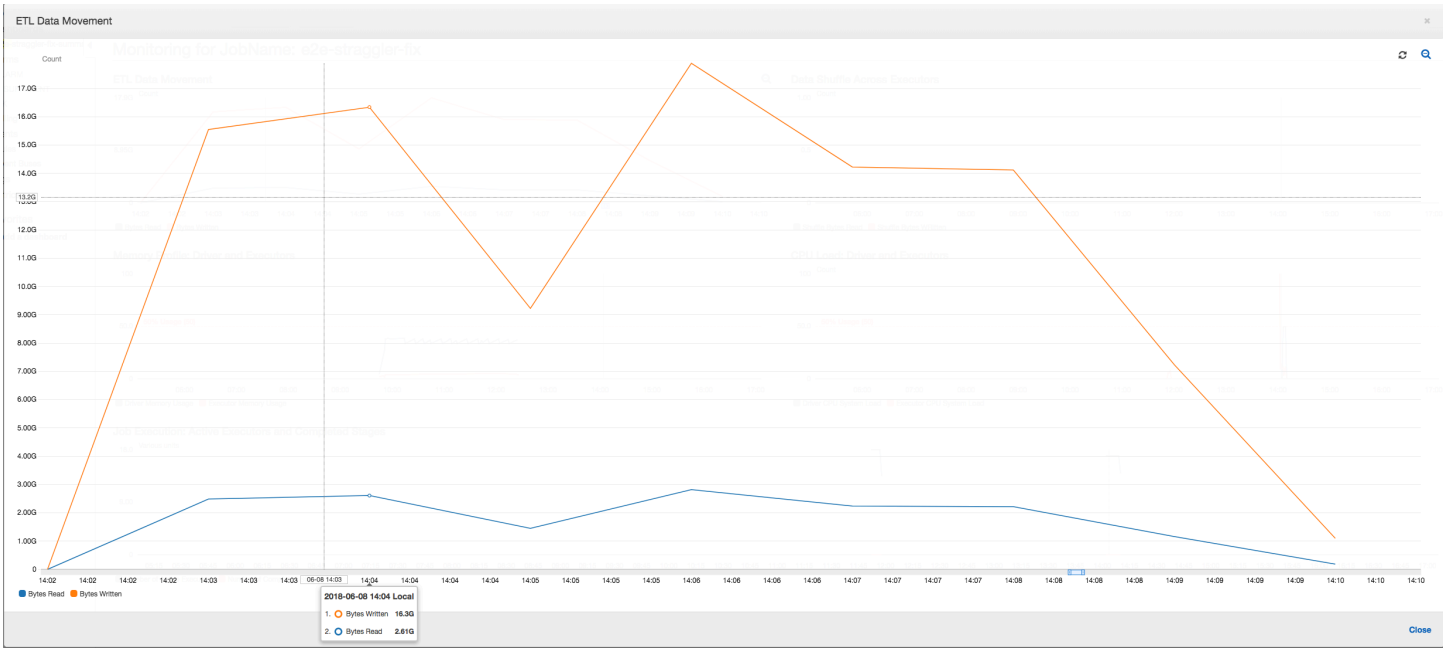
Corriger les programmes d'exécution en retard à l'aide du regroupement

Vous pouvez éviter les programmes d'exécution en retard en utilisant la fonction de regroupement dans AWS Glue. Utilisez le regroupement pour répartir les données uniformément sur tous les programmes d'exécution et fusionner des fichiers en fichiers plus volumineux à l'aide de tous les programmes d'exécution disponibles sur le cluster. Pour de plus amples informations, veuillez consulter Lecture des fichiers en entrée dans des groupes de plus grande taille.

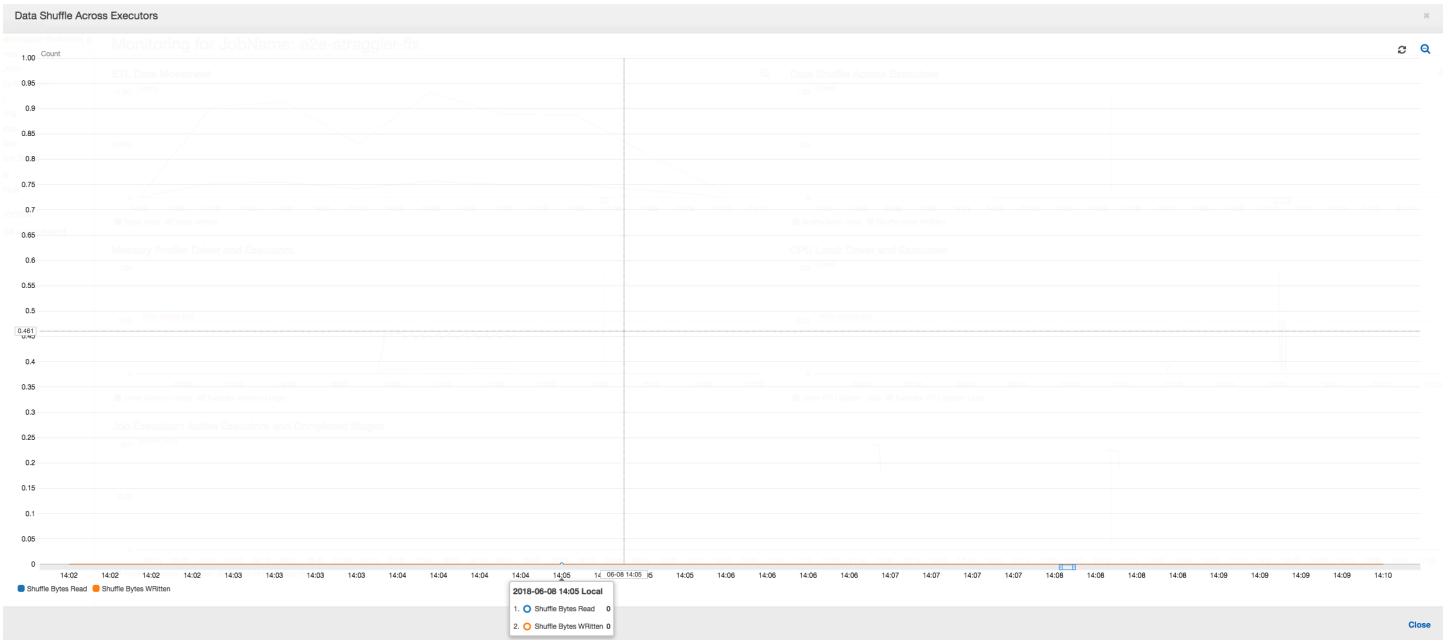
Pour consulter les déplacements de données ETL dans la tâche AWS Glue, profilez le code suivant avec la fonction de regroupement activée :

```
df = glueContext.create_dynamic_frame_from_options("s3", {'paths': ["s3://input_path"],
 "recurse":True, 'groupFiles': 'inPartition'}, format="json")
datasink = glueContext.write_dynamic_frame.from_options(frame = df, connection_type =
 "s3", connection_options = {"path": output_path}, format = "json", transformation_ctx
 = "datasink4")
```

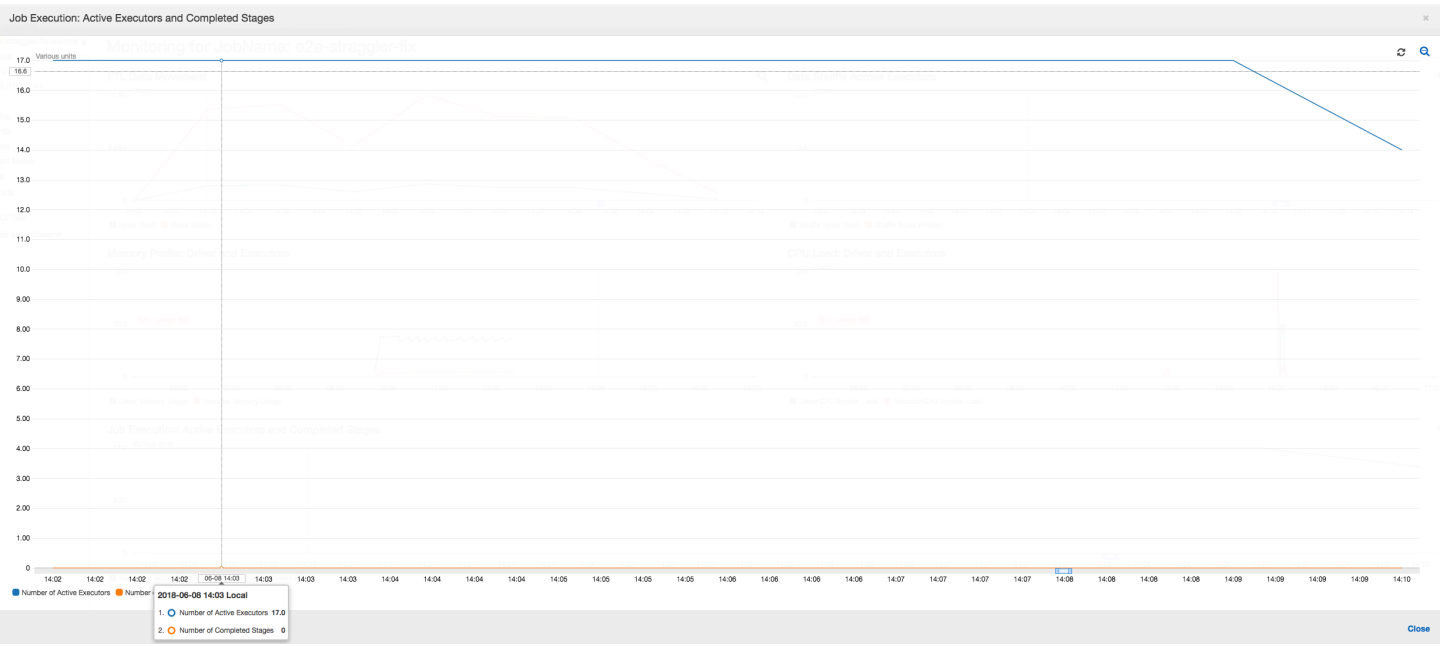
Déplacement de données ETL : Les écritures de données sont désormais diffusées en parallèle avec les lectures de données pendant toute la durée d'exécution de la tâche. Par conséquent, la tâche est réalisée en huit minutes, soit beaucoup plus rapidement qu'avant.



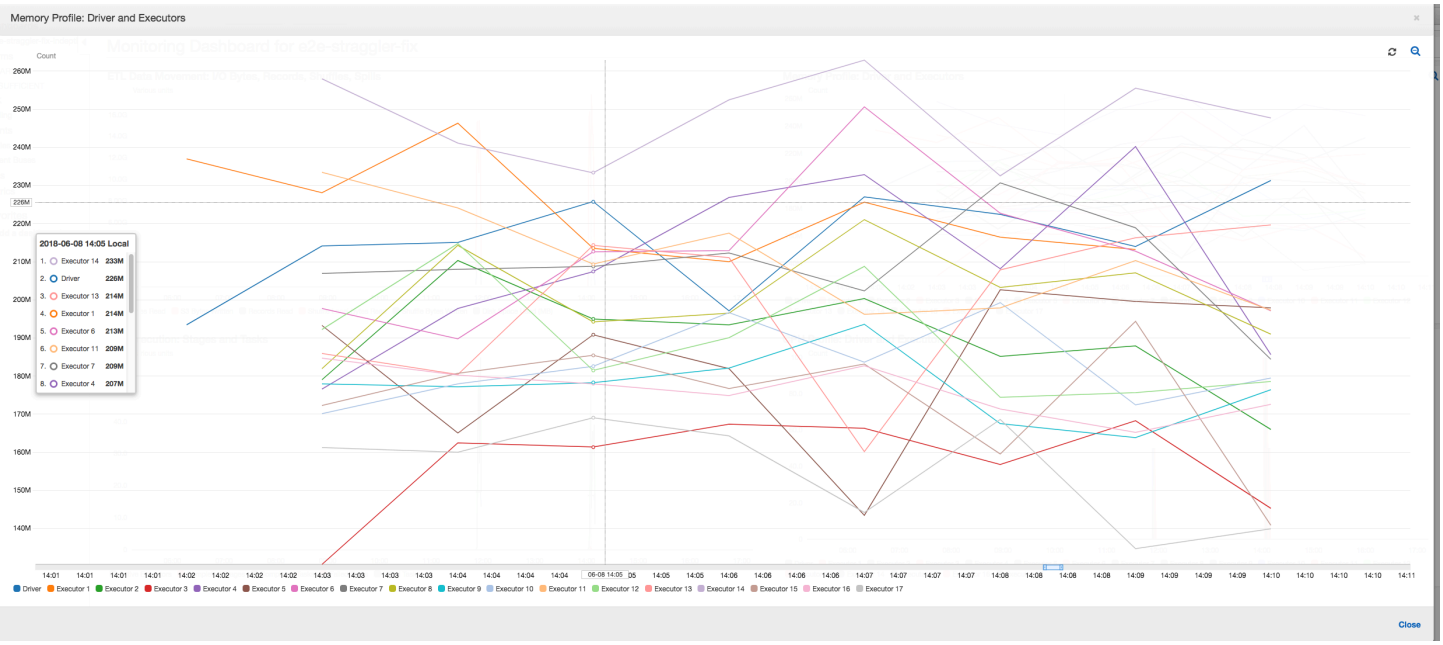
Remaniement des données sur les programmes d'exécution : Les données d'entrée ayant fusionné lors des lectures grâce à la fonction de regroupement, il n'y a pas de remaniement de données coûteux suite aux lectures de données.



Exécution de tâche : Les métriques d'exécution de tâche montrent que le nombre total de programmes d'exécution actifs s'exécutant et traitant des données reste relativement constant. Il n'y a pas de ralentissement dans la tâche. Tous les programmes d'exécution restent actifs et ne sont pas abandonnés tant que la tâche n'est pas terminée. Comme il n'existe pas de remaniement intermédiaire de données sur les programmes d'exécution, il n'y a qu'une seule étape dans la tâche.



Memory profile (Profil de la mémoire) : les métriques montrent la consommation de mémoire active par l'ensemble des programmes d'exécution, confirmant une nouvelle fois qu'il existe une activité sur l'ensemble des programmes d'exécution. Les données étant parallèlement envoyées et reçues, l'espace mémoire total de tous les programmes d'exécution est à peu près uniforme et bien au-dessous du seuil de sécurité pour tous les programmes d'exécution.



Surveillance de la progression des tâches multiples

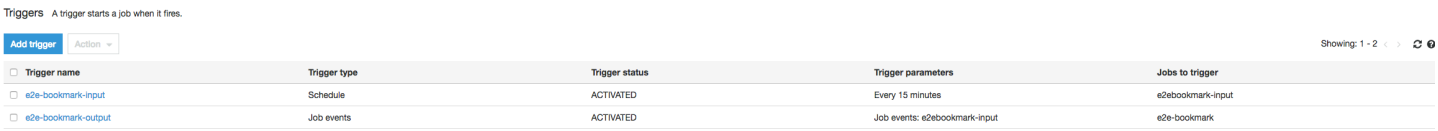
Vous pouvez profiler plusieurs AWS Glue tâches ensemble et surveiller le flux de données entre elles. Il s'agit d'un modèle de flux de travail courant qui nécessite la surveillance de la progression de chaque tâche, du journal de traitement des données, du retraitement des données et des signets de tâche.

Rubriques

- Code profilé
- Voir les métriques profilées sur la console AWS Glue
- Corriger le traitement des fichiers

Code profilé

Ce flux de travail comprend deux tâches : une tâche d'entrée et une tâche de sortie. La tâche d'entrée est planifiée pour s'exécuter toutes les 30 minutes au moyen d'un déclencheur périodique. La tâche de sortie est planifiée pour s'exécuter après chaque exécution réussie de la tâche d'entrée. Ces tâches planifiées sont contrôlées au moyen de déclencheurs de tâche.



Input job (Tâche d'entrée) : cette tâche lit les données à partir d'un emplacement Amazon Simple Storage Service (Amazon S3), les transforme à l'aide d'`ApplyMapping` et les écrit à un emplacement Amazon S3 intermédiaire. Le code suivant est le code profilé pour la tâche d'entrée :

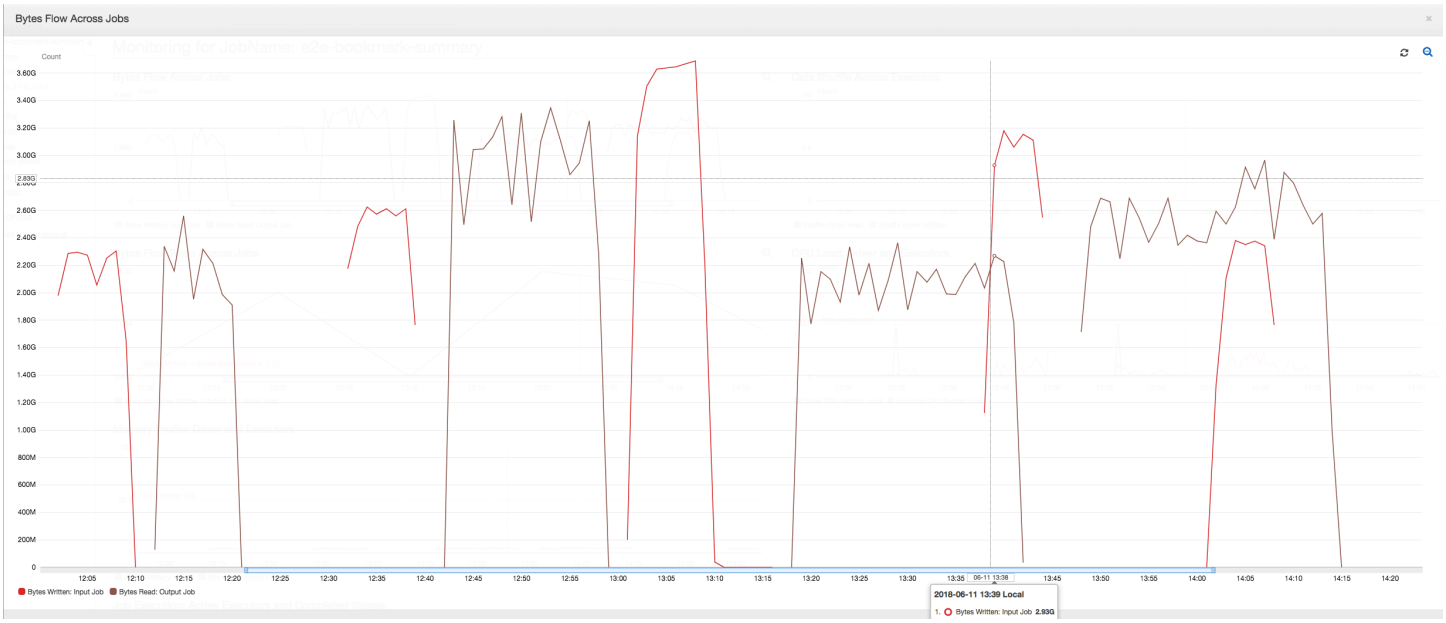
```
datasource0 = glueContext.create_dynamic_frame.from_options(connection_type="s3",
 connection_options = {"paths": ["s3://input_path"],
 "useS3ListImplementation":True,"recurse":True}, format="json")
applymapping1 = ApplyMapping.apply(frame = datasource0, mappings = [map_spec])
datasink2 = glueContext.write_dynamic_frame.from_options(frame = applymapping1,
 connection_type = "s3", connection_options = {"path": staging_path, "compression":
 "gzip"}, format = "json")
```

Output job (Tâche de sortie) : cette tâche lit la sortie de la tâche d'entrée à partir de l'emplacement intermédiaire dans Amazon S3, les transforme à nouveau et les écrit dans une destination :

```
datasource0 = glueContext.create_dynamic_frame.from_options(connection_type="s3",
 connection_options = {"paths": [staging_path],
 "useS3ListImplementation":True,"recurse":True}, format="json")
applymapping1 = ApplyMapping.apply(frame = datasource0, mappings = [map_spec])
datasink2 = glueContext.write_dynamic_frame.from_options(frame = applymapping1,
 connection_type = "s3", connection_options = {"path": output_path}, format = "json")
```

Voir les métriques profilées sur la console AWS Glue

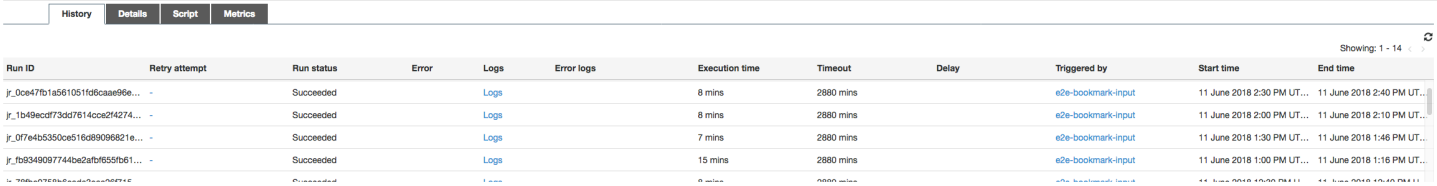
Le tableau de bord suivant superpose la métrique d'octets Amazon S3 écrits de la tâche d'entrée sur la métrique d'octets Amazon S3 lus selon la même chronologie pour la tâche de sortie. La chronologie montre différentes exécutions de tâche des tâches d'entrée et de sortie. La tâche d'entrée (en rouge) démarre toutes les 30 minutes. La tâche de sortie (en marron) démarre lorsque la tâche d'entrée se termine, avec une simultanéité max. de 1.



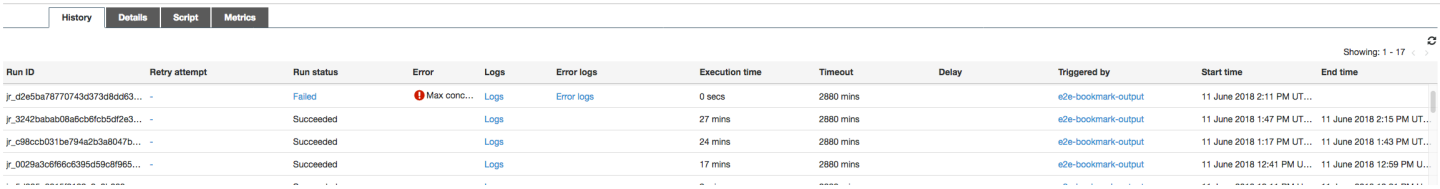
Dans cet exemple, les signets de tâche ne sont pas activés. Aucun contexte de transformation n'est utilisé pour activer les signets de tâche dans le code de script.

Historique de tâche : les tâches d'entrée et de sortie s'exécutent plusieurs fois, à partir de 12 h 00, comme le montre l'onglet Historique.

La tâche d'entrée sur la console AWS Glue se présente comme suit :

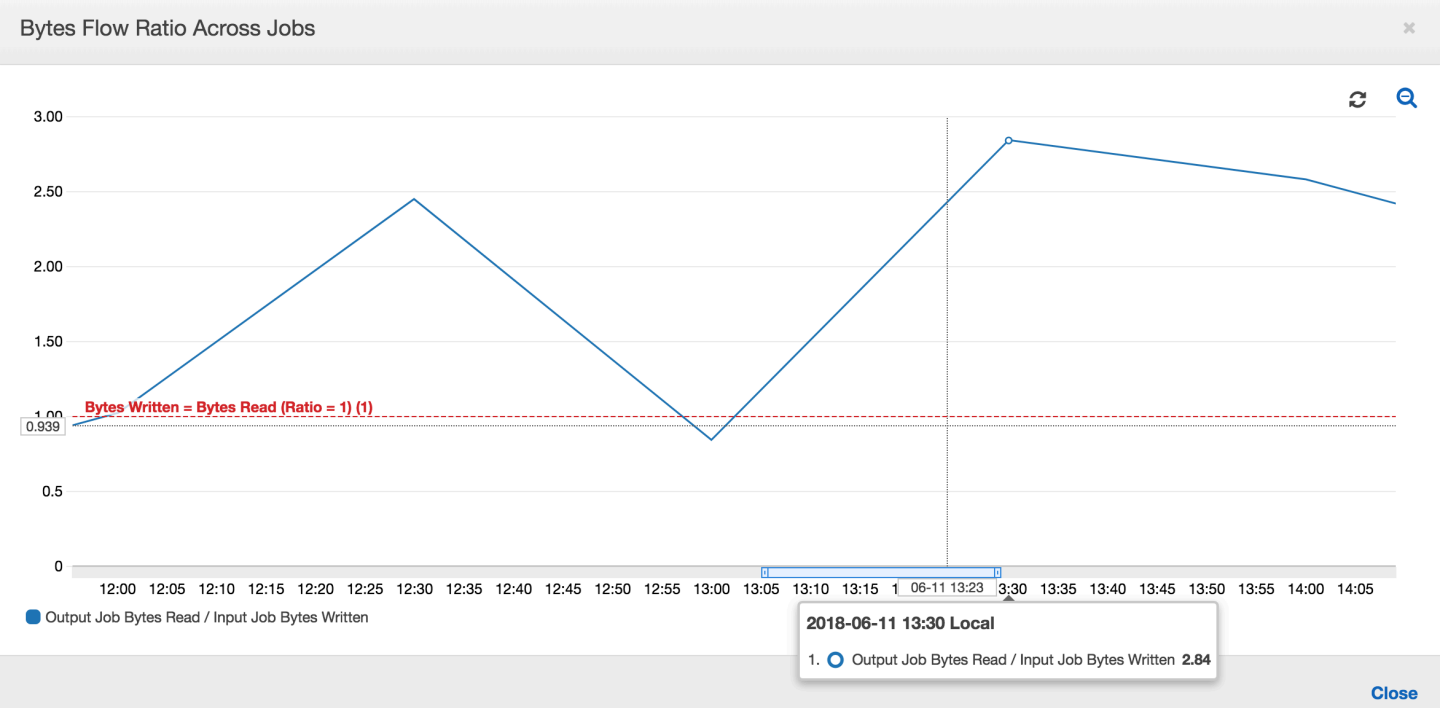


L'image suivante montre la tâche de sortie :



Première exécution de tâche : comme le montre le graphique Octets de données lues et écrites ci-dessous, les premières exécutions des tâches d'entrée et de sortie entre 12 h 00 et 12 h 30 montrent à peu près la même zone sous les courbes. Ces zones représentent les octets Amazon S3 écrits par la tâche d'entrée et les octets Amazon S3 lus par la tâche de sortie. Ces données sont également confirmées par le taux d'octets Amazon S3 écrits (additionnés sur 30 minutes, la fréquence du déclencheur de tâche pour la tâche d'entrée). Le point de données pour le ratio d'exécution de la tâche d'entrée qui a commencé à 12 h 00 est également 1.

Le graphique suivant montre le taux de flux de données pour toutes les exécutions de tâches :



Deuxième exécution de tâche : dans la deuxième exécution de tâche, il existe une différence claire entre le nombre d'octets lus par la tâche de sortie et le nombre d'octets écrits par la tâche d'entrée. (Comparez la zone sous la courbe des deux exécutions de tâche de la tâche de sortie ou comparez les zones de la deuxième exécution des tâches d'entrée et de sortie.) Le ratio entre le nombre d'octets lus et écrits montre que la tâche de sortie lit environ 2,5x les données écrites par la tâche d'entrée dans la seconde période de 30 minutes, de 12 h 30 à 13 h 00. C'est parce que la tâche

de sortie a retraité la sortie de la première exécution de la tâche d'entrée car les signets de tâche n'étaient pas activés. Un ratio supérieur à 1 indique qu'un backlog de données supplémentaire a été traité par la tâche de sortie.

Troisième exécution de tâche : la tâche d'entrée est assez cohérente quant au nombre d'octets écrits (voir la zone sous les courbes rouges). Toutefois, la troisième exécution de la tâche d'entrée a duré plus longtemps que prévu (voir la longue queue de la courbe rouge). Par conséquent, la troisième exécution de la tâche de sortie a démarré en retard. La troisième exécution de tâche n'a traité qu'une fraction des données cumulées à l'emplacement intermédiaire dans les 30 minutes restantes entre 13 h 00 et 13 h 30. Le taux de débit d'octets indique que seules 0,83 des données écrites ont été traitées par la troisième exécution de la tâche d'entrée (voir le ratio à 13 h 00).

Chevauchement des tâches d'entrée et de sortie : la quatrième exécution de la tâche d'entrée a démarré à 13 h 30 comme prévu, avant la fin de la troisième exécution de la tâche de sortie. Ces deux exécutions de tâche se chevauchement partiellement. Toutefois, la troisième exécution de la tâche de sortie ne capture que les fichiers répertoriés dans l'emplacement intermédiaire d'Amazon S3, lorsqu'elle a commencé vers 13:17. Cela comprend toutes les sorties de données de la première exécution de la tâche d'entrée. Le ratio réel à 13 h 30 est d'environ 2,75. La troisième exécution de la tâche de sortie a traité environ 2,75x de données écrites par la quatrième exécution de la tâche d'entrée, de 13 h 30 à 14 h 00.

Comme le montrent ces images, la tâche de sortie retraite des données de toutes les précédentes exécutions de la tâche d'entrée à partir de l'emplacement intermédiaire. Par conséquent, la quatrième exécution de la tâche de sortie est la plus longue et elle chevauche la totalité de la cinquième exécution de la tâche d'entrée.

Corriger le traitement des fichiers

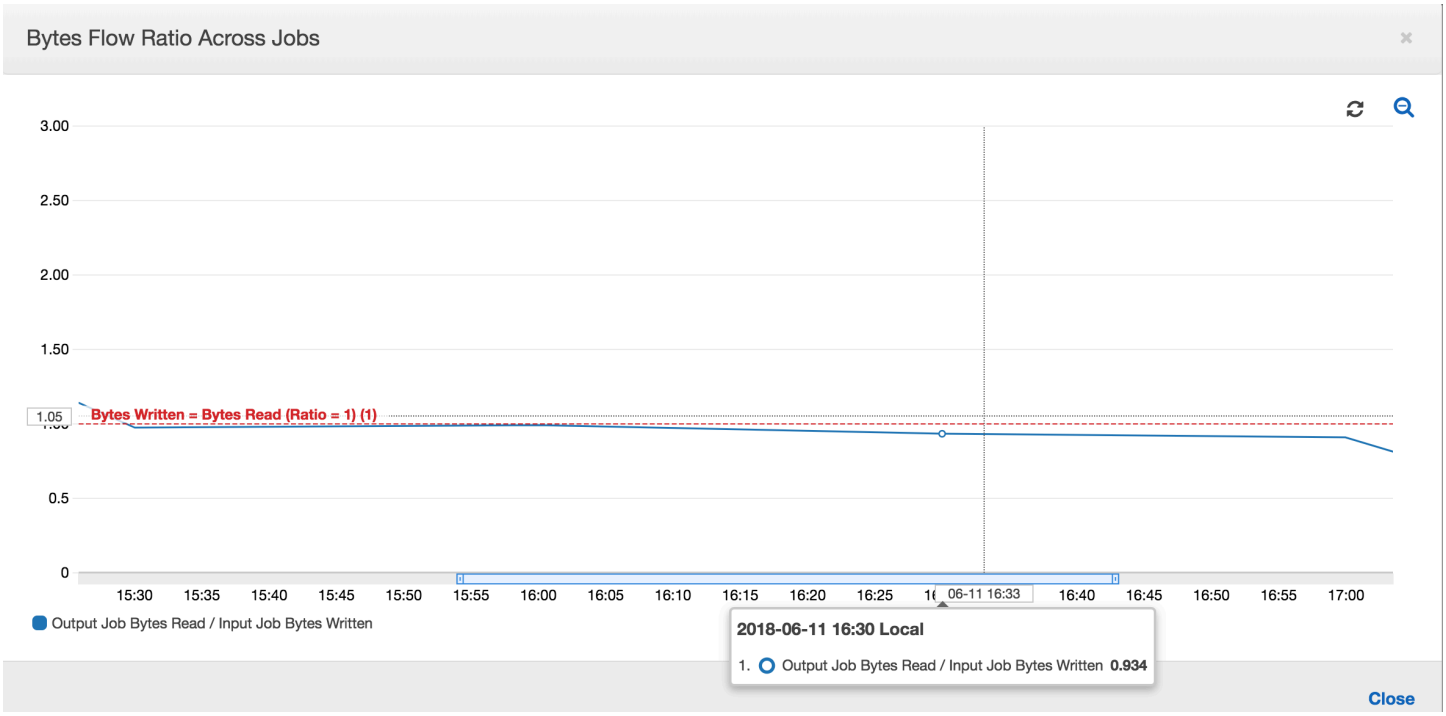
Vous devez vous assurer que la tâche de sortie ne traite les fichiers qui n'ont pas été traités par de précédentes exécutions de la tâche de sortie. Pour ce faire, activez les signets de tâche et configurez le contexte de transformation dans la tâche de sortie de la façon suivante :

```
datasource0 = glueContext.create_dynamic_frame.from_options(connection_type="s3",
 connection_options = {"paths": [staging_path],
 "useS3ListImplementation":True,"recurse":True}, format="json", transformation_ctx =
 "bookmark_ctx")
```

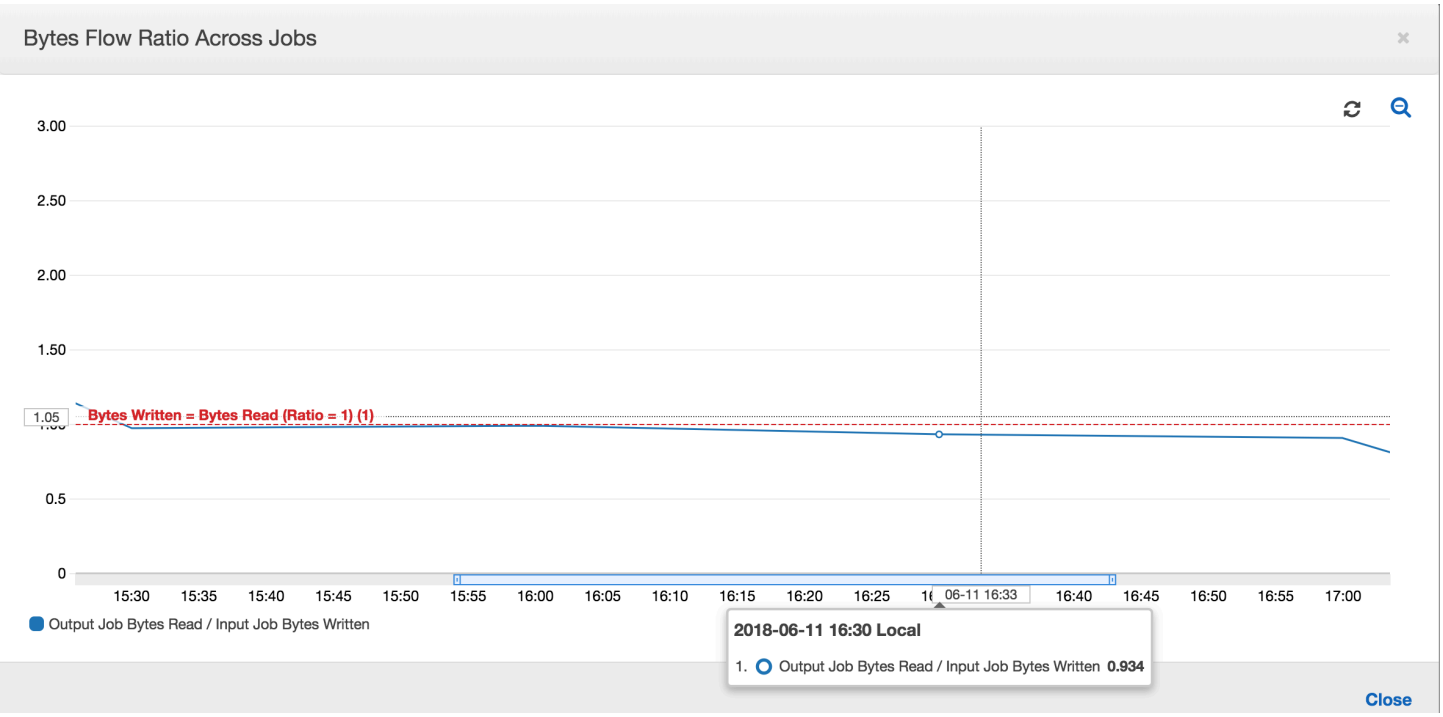
Lorsque les signets de tâche sont activés, la tâche de sortie ne retraite pas les données dans l'emplacement intermédiaire pour toutes les précédentes exécutions de la tâche d'entrée. Dans

l'image suivante montrant les données lues et écrites, la zone sous la courbe marron est assez cohérente et similaire aux courbes rouges.

Le rapports du flux d'octets reste également proche de 1, car des données supplémentaires ne sont pas traitées.



Une exécution de la tâche de sortie démarre et capture les fichiers dans l'emplacement intermédiaire avant que la prochaine exécution de la tâche d'entrée ne commence à placer d'autres données dans l'emplacement intermédiaire. Tant que c'est le cas, elle ne traite que les fichiers capturés lors de la précédente exécution de tâche de sortie, et le ratio reste proche de 1.



Supposons que la tâche d'entrée dure plus longtemps que prévu et que, par conséquent, la tâche de sortie capture des fichiers dans l'emplacement intermédiaire à partir de deux exécutions de tâche d'entrée. Le ratio est supérieur à 1 pour cette exécution de tâche de sortie. Toutefois, les exécutions de tâches de sortie suivantes ne traitent aucun fichier déjà traité par les précédentes exécutions de la tâche de sortie.

Surveillance de la planification des capacités de DPU

Vous pouvez utiliser les métriques des tâches AWS Glue pour estimer le nombre d'unités de traitement des données (DPUs) qui peuvent être utilisées pour étendre une AWS Glue tâche.

> **Note**
>
> Cette page s'applique uniquement à AWS Glue versions 0.9 et 1.0. Les versions ultérieures de AWS Glue contiennent des fonctionnalités économiques qui introduisent des considérations supplémentaires lors de la planification des capacités.

Rubriques

- Code profilé
- Voir les métriques profilées sur la console AWS Glue
- Déterminer la capacité DPU optimale

Code profilé

Le script suivant lit une partition Amazon Simple Storage Service (Amazon S3) contenant 428 fichiers JSON compressés avec gzip. Le script applique un mappage pour modifier les noms de champs, les convertit et les écrit dans Amazon S3 au format Apache Parquet. Vous DPUs en attribuez 10 par défaut et vous exécutez cette tâche.

```
datasource0 = glueContext.create_dynamic_frame.from_options(connection_type="s3",
 connection_options = {"paths": [input_path],
 "useS3ListImplementation":True,"recurse":True}, format="json")
applymapping1 = ApplyMapping.apply(frame = datasource0, mappings = [(map_spec])
datasink2 = glueContext.write_dynamic_frame.from_options(frame = applymapping1,
 connection_type = "s3", connection_options = {"path": output_path}, format =
 "parquet")
```

Voir les métriques profilées sur la console AWS Glue

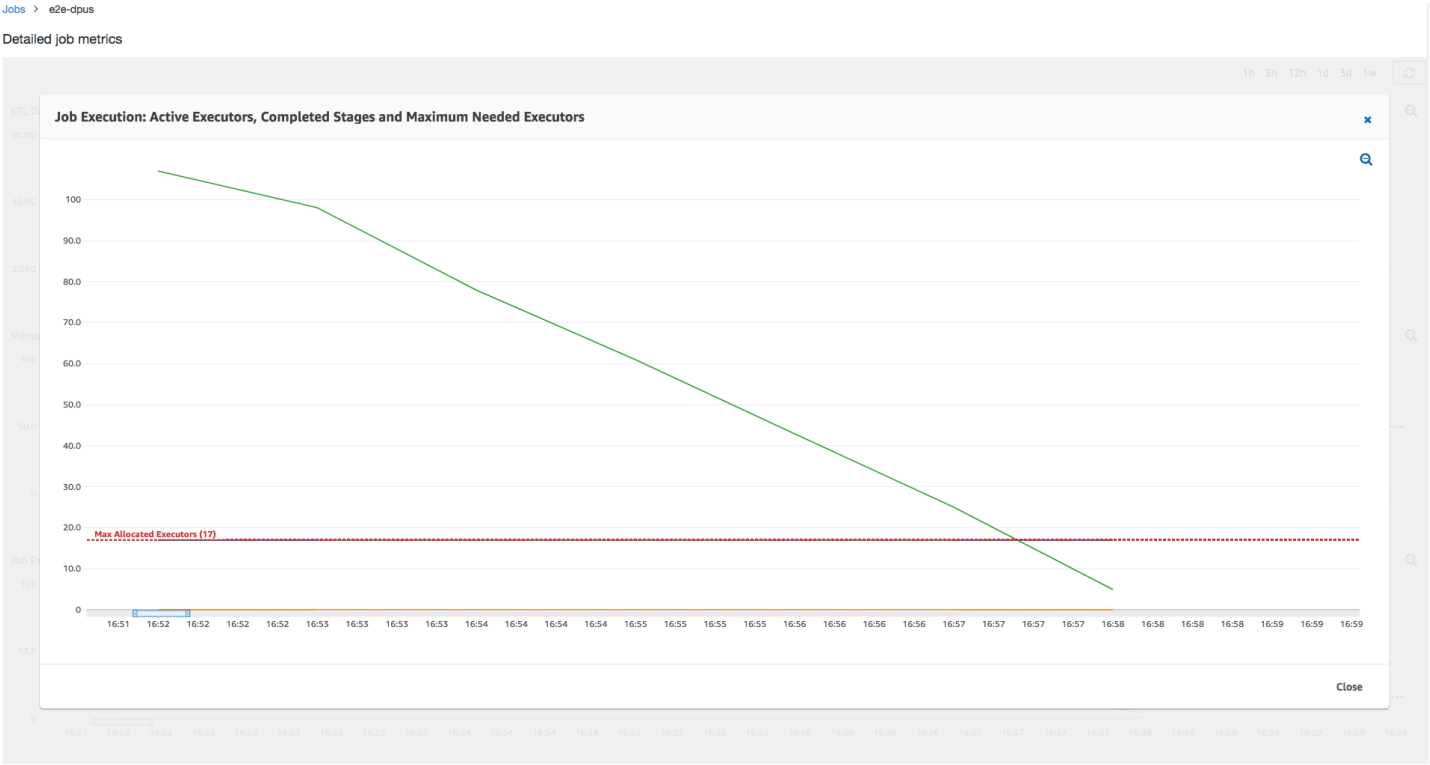
Exécution de tâche 1 : Dans cette exécution de tâche, nous montrons comment déterminer si le cluster est sous-approvisionné DPUs . La fonctionnalité d'exécution de tâches dans AWS Glue indique le nombre total de programmes d'exécution actifs en cours d'exécution, le nombre d'étapes terminées et le nombre maximum de programmes d'exécution nécessaires.

Le nombre maximum de programmes d'exécution nécessaires est calculé en ajoutant le nombre total de tâches en cours d'exécution et de tâches en attente, puis en divisant ce nombre par le nombre de tâches par programme d'exécution. Ce résultat est une mesure du nombre total d'exécuteurs requises pour satisfaire la charge du moment.

En revanche, le nombre de programmes d'exécution actifs en cours d'exécution mesure le nombre de programmes d'exécution qui exécutent des tâches Apache Spark. À mesure que la tâche progresse, le nombre maximum de programmes d'exécution nécessaires peut varier et baisse généralement vers la fin de la tâche à mesure que diminue la file d'attente de tâches en attente.

La ligne rouge horizontale du graphique suivant indique le nombre maximum d'exécuteurs alloués, qui dépend du nombre d'exécuteurs DPUs que vous allouez à la tâche. Dans ce cas, vous allouez

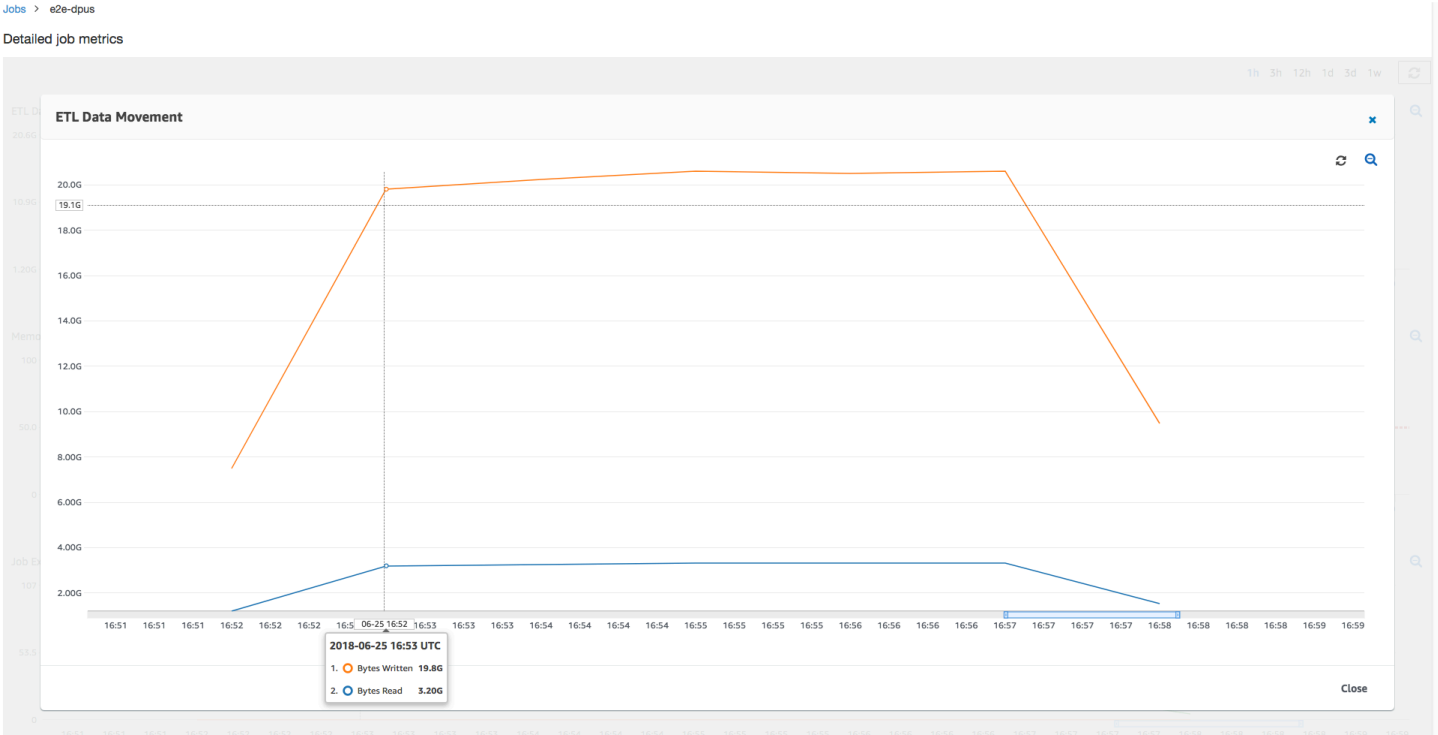
10 points DPUs à l'exécution du job. Une DPU est réservée à la gestion. Neuf DPUs exécuteurs exécutent deux exécuteurs chacun et un exécuteur est réservé au pilote Spark. Le pilote Spark s'exécute dans l'application principale. Par conséquent, le nombre maximum de programmes d'exécution alloués est de 2 x 9 - 1 = 17 programmes d'exécution.



Comme le montre le graphique, le nombre maximum de programmes d'exécution nécessaires commence à 107 au début de la tâche, alors que le nombre de programmes d'exécution actifs reste 17. C'est le même que le nombre maximum d'exécuteurs alloués, soit 10 DPUs. Le ratio entre le nombre maximum de programmes d'exécution nécessaires et le nombre maximum de programmes d'exécution alloués (en ajoutant 1 à chacun des pilotes Spark) donne le facteur sous-alloué : 108/18 = 6x. Vous pouvez en configurer 6 (sous le ratio de provisionnement) *9 (capacité actuelle du DPU - 1) + 1 DPUs = 55 DPUs pour étendre le travail afin de l'exécuter avec un parallélisme maximal et de terminer plus rapidement.

La console AWS Glue affiche le détail des métriques des tâches sous la forme d'une ligne statique qui représente le nombre maximal de programmes d'exécution alloués au départ. Pour ces métriques, la console calcule le nombre maximum de programme d'exécution alloués à partir de la définition de la tâche. En revanche, pour les métriques détaillées d'exécution des tâches, la console calcule le nombre maximum d'exécuteurs alloués à partir de la configuration d'exécution des tâches, en particulier le nombre DPUs alloué pour l'exécution des tâches. Pour afficher les métriques d'une

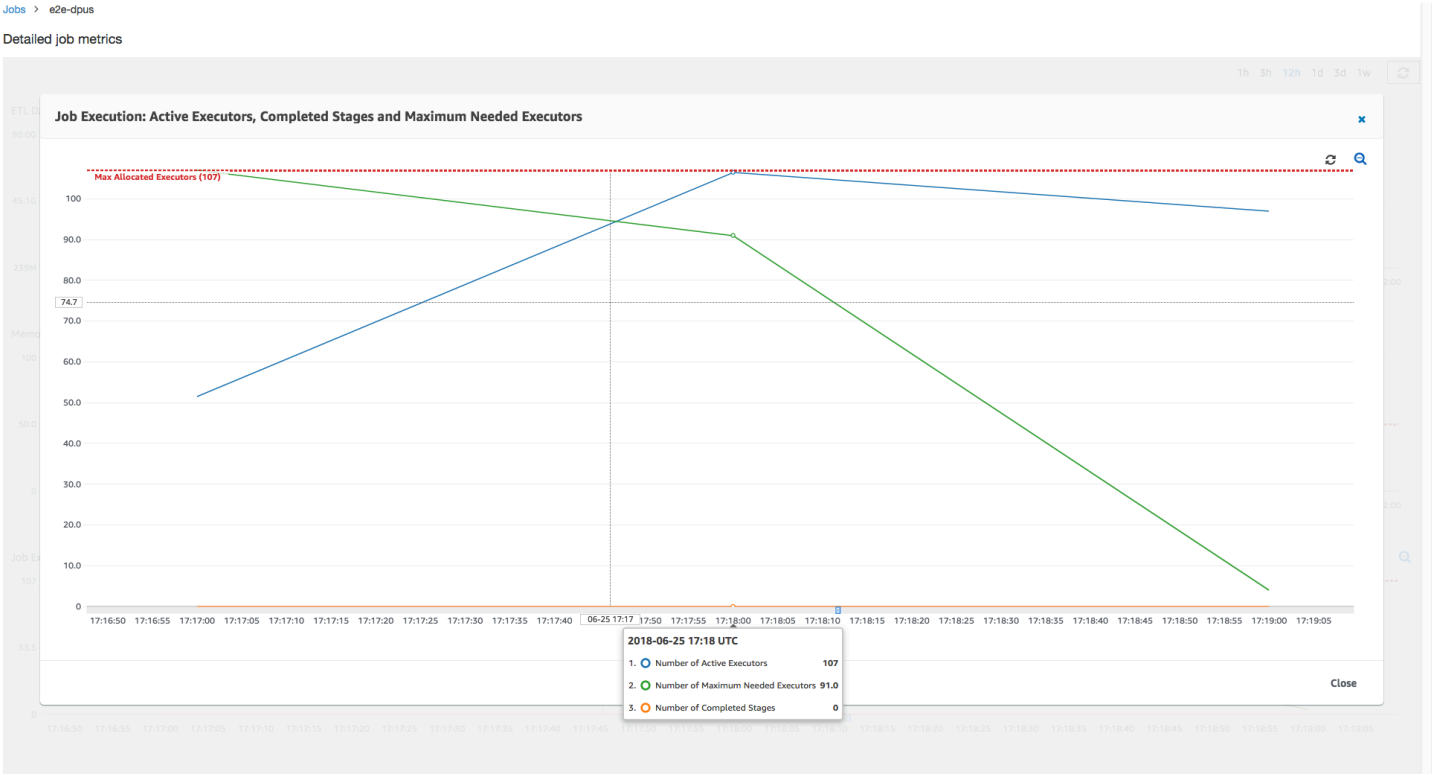
exécution de tâche spécifique, sélectionnez l'exécution et choisissez View run metrics (Afficher les métriques de l'exécution).



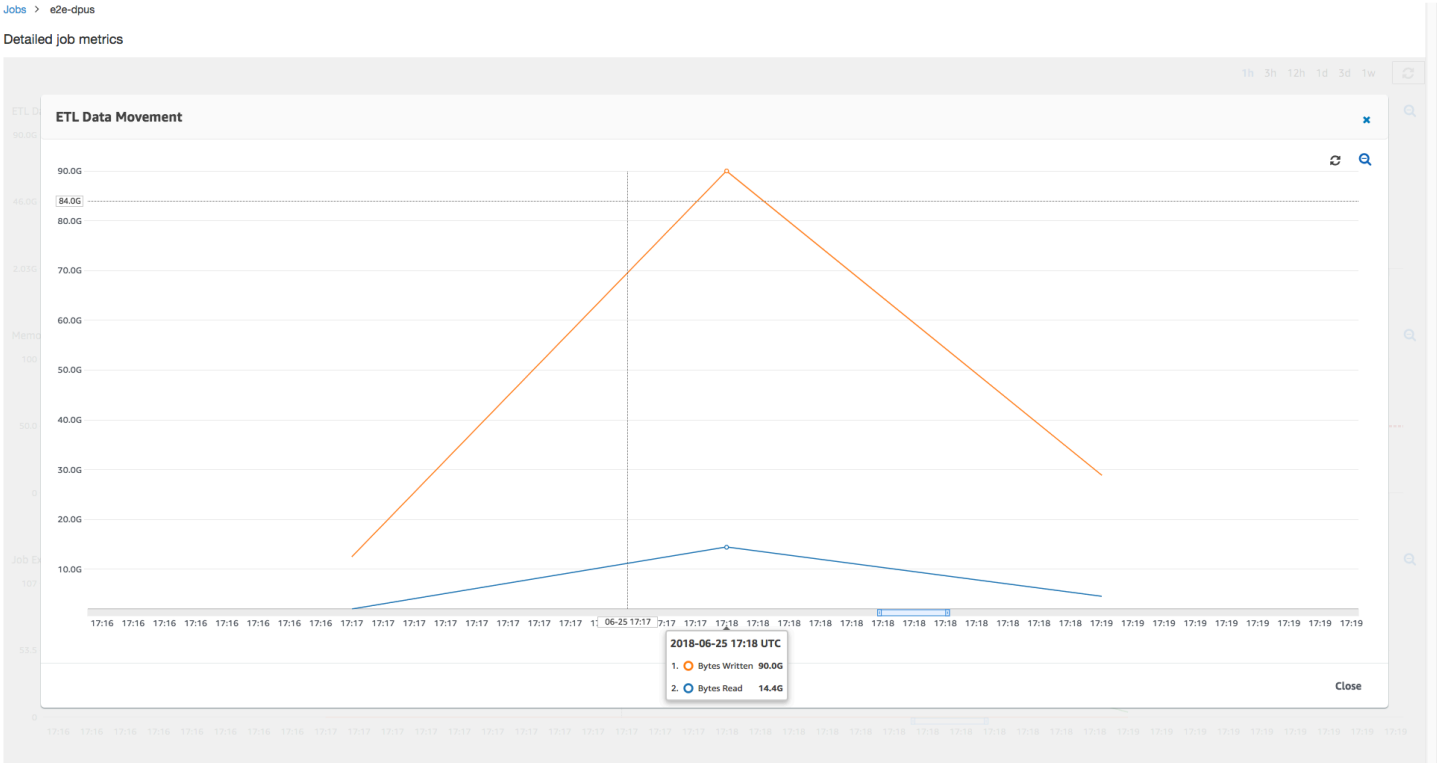
Si l'on regarde les octets Amazon S3 lus et écrits, on remarque que la tâche prend six minutes complètes pour diffuser les données à partir d'Amazon S3 et les écrire en parallèle. Tous les cœurs alloués DPUs lisent et écrivent sur Amazon S3. Le nombre maximum de programmes d'exécution nécessaires (107) correspond également au nombre de fichiers dans le chemin d'accès d'entrée Amazon S3 (428). Chaque programme d'exécution peut lancer quatre tâches Spark pour traiter quatre fichiers d'entrée (JSON compressées avec gzip).

Déterminer la capacité DPU optimale

Sur la base des résultats de l'exécution de la tâche précédente, vous pouvez augmenter le nombre total de tâches allouées DPUs à 55 et suivre les performances de la tâche. La tâche se termine en moins de trois minutes, soit la moitié du temps auparavant nécessaire. La montée en charge de la tâche n'est pas linéaire dans ce cas, car il s'agit d'une tâche d'exécution courte. Les tâches comportant des tâches de longue durée ou un grand nombre de tâches (un grand nombre d'exécuteurs maximum nécessaires) bénéficient d'une accélération des performances du close-to-linear DPU grâce à la scale-out.



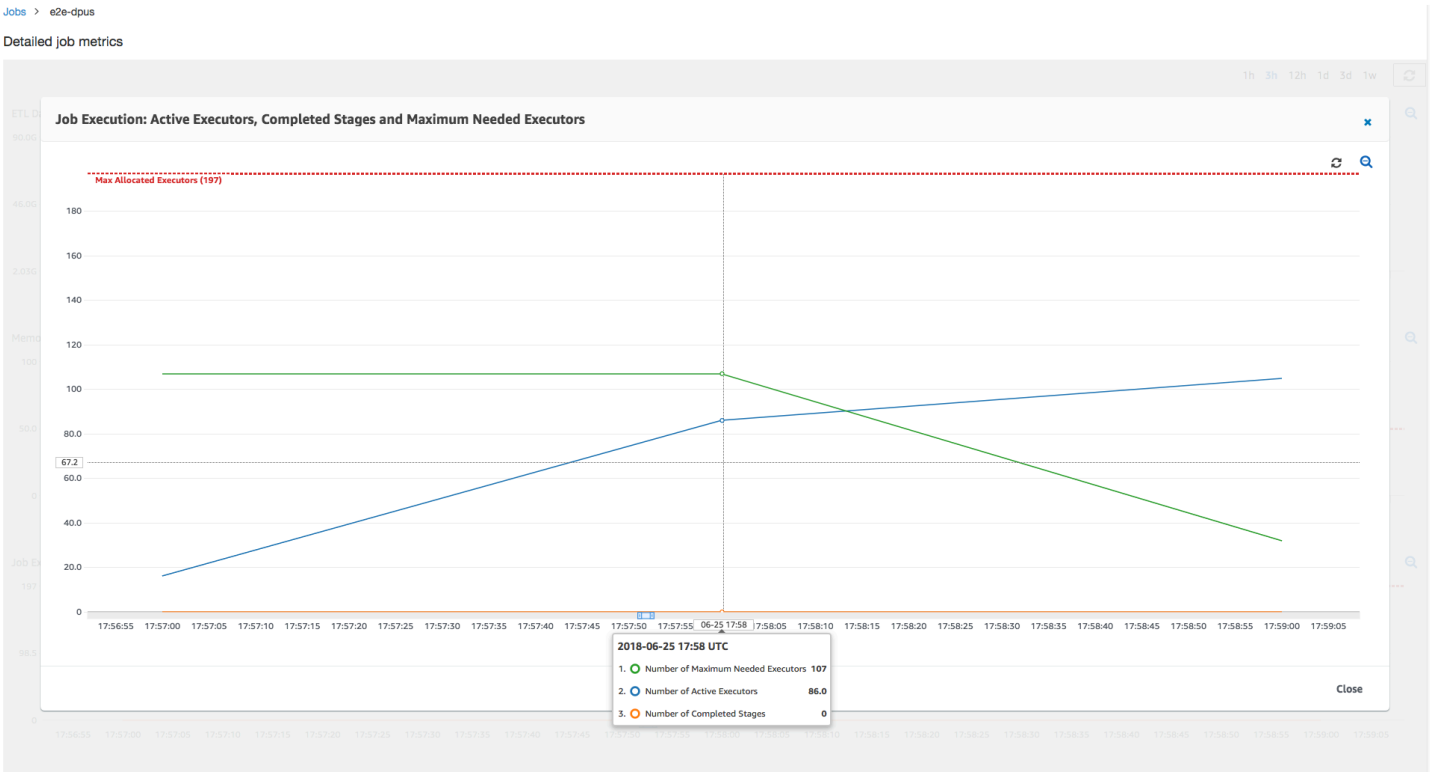
Comme le montre l'image ci-dessus, le nombre total de programmes d'exécution actifs atteint le nombre maximum de programmes d'exécution alloués (107). De même, le nombre maximum de programmes d'exécution nécessaires ne dépasse jamais le nombre maximum de programmes d'exécution alloués. Le nombre maximum de programmes d'exécution nécessaires est calculé à partir du nombre de tâches en cours d'exécution et en attente, il doit donc être inférieur au nombre de programmes d'exécution actifs. Ceci est dû au fait que des programmes d'exécution peuvent être partiellement ou complètement inactifs pour une courte période de temps et ne sont pas encore démagnétisés.



Cette exécution de tâche utilise 6 fois plus de programmes d'exécution pour lire et écrire en parallèle depuis Amazon S3. Par conséquent, cette exécution de tâche utilise plus de bande passante Amazon S3 pour les lectures et les écritures, et se termine plus rapidement.

Identifiez le surprovisionnement DPUs

Ensuite, vous pouvez déterminer si le fait d'étendre la tâche à 100 DPUs (99 * 2 = 198 exécuteurs) permet de poursuivre l'extension. Comme le montre le graphique suivant, la tâche se termine toujours après trois minutes. De même, la tâche ne dépasse pas 107 exécuteurs (DPUsconfiguration 55), et les 91 exécuteurs restants sont surapprovisionnés et ne sont pas du tout utilisés. Cela montre que l'augmentation du nombre de n' DPUs améliore pas toujours les performances, comme en témoigne le nombre maximal d'exécuteurs nécessaires.



Comparer des différences de temps

Les trois cycles de travail présentés dans le tableau suivant résument les temps d'exécution des travaux pour 10 DPUs DPUs, 55 et 100 DPUs. Vous pouvez trouver la capacité de DPU pour améliorer la durée d'exécution de la tâche en utilisant l'estimation que vous avez réalisée en surveillant la première exécution de tâche.

ID de tâche	Nombre de DPUs	Durée d'exécution
jr_c894524c8ef5048a4d9...	10	6 min.
jr_1a466cf2575e7ffe6856...	55	3 min.
jr_34fa1ed4c6aa9ff0a814...	100	3 min.

Résolution des problèmes liés à l'IA générative pour Apache Spark in AWS Glue

La résolution des problèmes liés à l'IA générative pour les tâches Apache Spark dans AWS Glue est une nouvelle fonctionnalité qui aide les ingénieurs de données et les scientifiques à diagnostiquer et à résoudre facilement les problèmes dans leurs applications Spark. En utilisant les technologies de

machine learning et d'IA générative, cette fonctionnalité analyse les problèmes liés aux tâches Spark et fournit une analyse détaillée des causes profondes, ainsi que des recommandations pratiques pour les résoudre. Le dépannage génératif de l'IA pour Apache Spark est disponible pour les jobs exécutés sur AWS Glue version 4.0 ou supérieure.

Transformez votre résolution des problèmes liés à Apache Spark grâce à notre agent de résolution des problèmes basé sur l'IA, qui prend désormais en charge tous les principaux modes de déploiement, notamment AWS Glue, Amazon EMR-, EC2 Amazon EMR-Serverless et Amazon AI Notebooks. SageMaker Ce puissant agent élimine les processus de débogage complexes en combinant des interactions en langage naturel, une analyse de la charge de travail en temps réel et des recommandations de code intelligent dans une expérience fluide. Pour plus de détails sur l'implémentation, reportez-vous à la section Qu'est-ce que l'agent de résolution des problèmes Apache Spark pour Amazon EMR ? Consultez la deuxième démonstration dans la section Exemples de résolution des problèmes liés à l'utilisation de l'agent de dépannage pour AWS Glue.

Comment fonctionne la Résolution des problèmes à l'aide de l'IA générative pour Apache Spark ?

Pour les tâches Spark qui ont échoué, Generative AI Troubleshooting analyse les métadonnées des tâches ainsi que les indicateurs et journaux précis associés à la signature d'erreur de votre tâche afin de générer une analyse des causes premières, et recommande des solutions spécifiques et les meilleures pratiques pour aider à remédier aux échecs de tâches.

Configuration de la Résolution des problèmes à l'aide de l'IA générative pour Apache Spark pour vos tâches

Configuration des autorisations IAM

L'octroi d'autorisations à l'utilisateur APIs utilisé par Spark Troubleshooting pour vos tâches dans AWS Glue nécessite les autorisations IAM appropriées. Vous pouvez obtenir des autorisations en associant la AWS politique personnalisée suivante à votre identité IAM (comme un utilisateur, un rôle ou un groupe).

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
```

```
    {
      "Effect": "Allow",
      "Action": [
        "glue:StartCompletion",
        "glue:GetCompletion"
      ],
      "Resource": [
        "arn:aws:glue:*:*:completion/*",
        "arn:aws:glue:*:*:job/*"
      ]
    }
  ]
}
```

> **Note**
>
> APIs Les deux éléments suivants sont utilisés dans la politique IAM pour permettre cette expérience via la console AWS Glue Studio : `StartCompletion` et`GetCompletion`.

Attribution d'autorisations

Pour activer l'accès, ajoutez des autorisations à vos utilisateurs, groupes ou rôles :

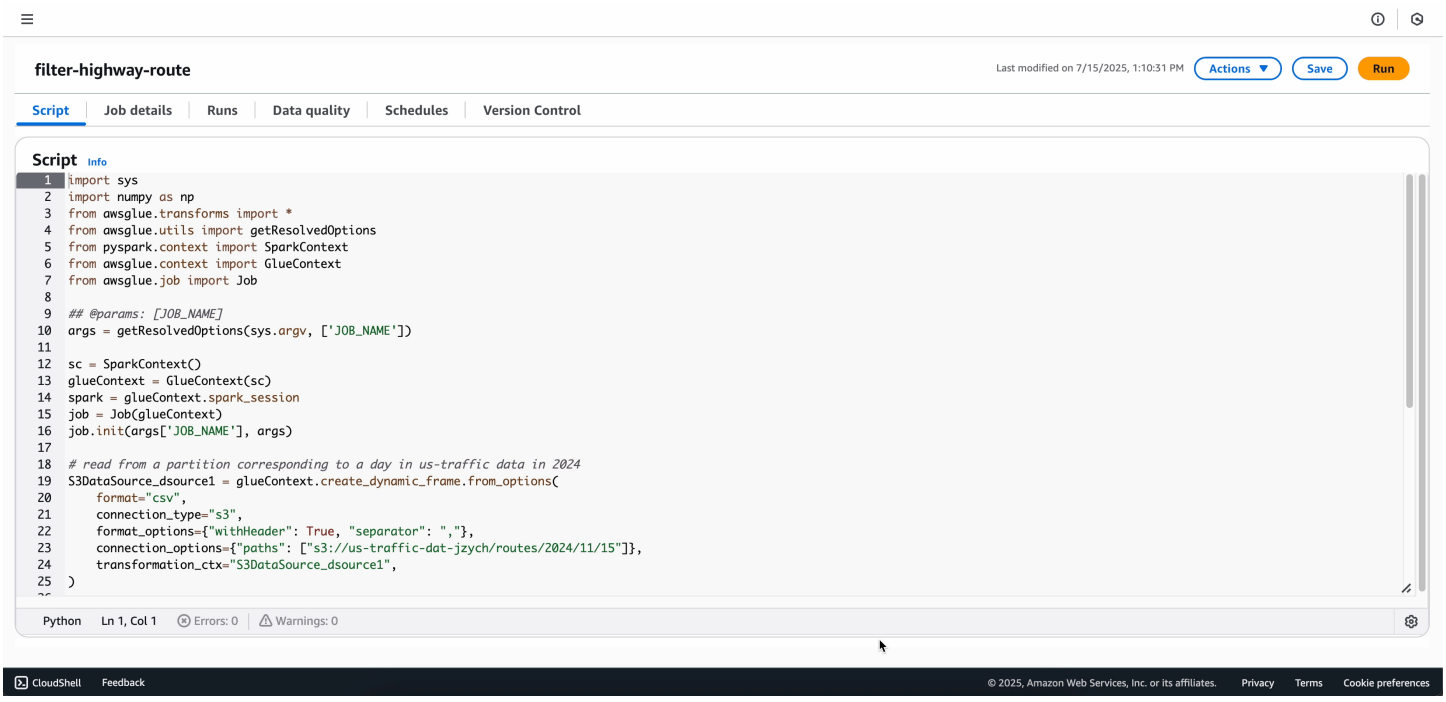
- Pour les utilisateurs et les groupes dans IAM Identity Center : créez un ensemble d'autorisations. Suivez les instructions de la rubrique Create a permission set du Guide de l'utilisateur IAM Identity Center.
- Pour les utilisateurs gérés dans IAM via un fournisseur d'identité : créez un rôle pour la fédération d'identité. Pour plus d'informations, consultez Creating a role for a third-party identity provider (federation) dans le Guide de l'utilisateur IAM.
- Pour les utilisateurs IAM : créez un rôel que votre utilisateur peut endosser. Suivez les instructions de la rubrique Creating a role for an IAM user du Guide de l'utilisateur IAM.

Exécution d'une analyse de résolution des problèmes suite à l'échec de l'exécution d'une tâche

Vous pouvez accéder à la fonctionnalité de résolution des problèmes via plusieurs chemins dans la console AWS Glue. Voici la procédure de démarrage :

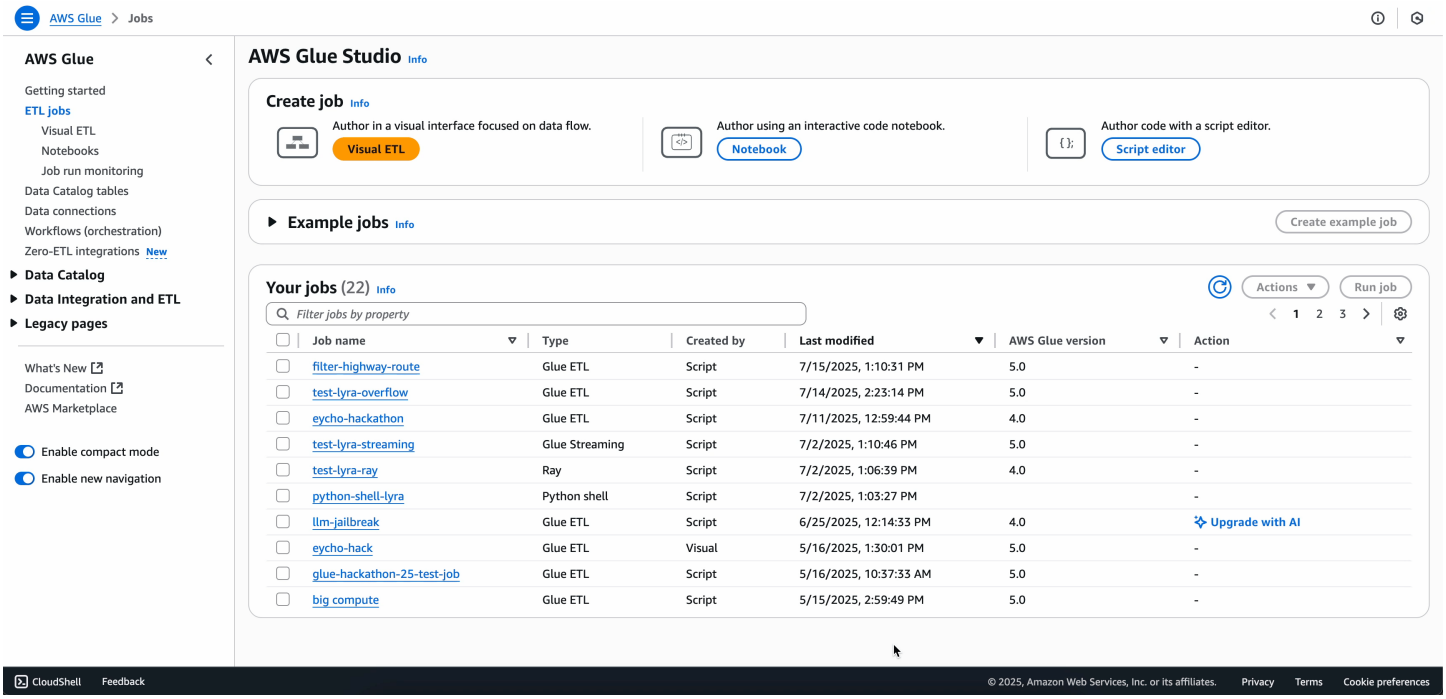
Option 1 : depuis la page Liste des tâches

1. Ouvrez la console AWS Glue à l'adresse https://console.aws.amazon.com/glue/.
2. Dans le volet de navigation, sélectionnez Tâches ETL.
3. Recherchez votre tâche qui a échoué dans la liste des tâches.
4. Sélectionnez l'onglet Exécutions dans la section des informations de la tâche.
5. Cliquez sur la tâche qui a échoué et que vous souhaitez analyser.
6. Choisissez Résoudre les problèmes avec l'IA pour démarrer l'analyse.
7. Lorsque l'analyse de résolution des problèmes est terminée, vous pouvez consulter l'analyse des causes premières et les recommandations dans l'onglet Analyse de résolution des problèmes en bas de l'écran.



Option 2 : utilisation de la page Surveillance de l'exécution des tâches

1. Accédez à la page de Surveillance de l'exécution des tâches.
2. Localisez l'exécution de votre tâche qui a échoué.
3. Choisissez le menu déroulant Actions.
4. Choisissez Résoudre les problèmes avec l'IA.



Option 3 : depuis la page Détails de l'exécution des tâches

1. Accédez à la page de détails de l'exécution de la tâche qui a échoué en cliquant sur Afficher les détails d'une exécution échouée dans l'onglet Exécutions ou en sélectionnant l'exécution de la tâche sur la page Surveillance de l'exécution des tâches.
2. Sur la page des détails de l'exécution des tâches, recherchez l'onglet Analyse de dépannage.

Catégories de dépannage prises en charge

Ce service se concentre sur trois catégories principales de problèmes que les ingénieurs de données et les développeurs rencontrent fréquemment dans leurs applications Spark :

- Erreurs de configuration et d'accès aux ressources : lors de l'exécution d'applications Spark dans AWS Glue, les erreurs de configuration des ressources et d'accès figurent parmi les problèmes les plus courants mais les plus difficiles à diagnostiquer. Ces erreurs se produisent souvent lorsque votre application Spark tente d'interagir avec AWS des ressources mais rencontre des problèmes d'autorisation, des ressources manquantes ou des problèmes de configuration.
- Problèmes de mémoire liés au pilote et à l'exécuteur Spark : les erreurs liées à la mémoire dans les tâches Apache Spark peuvent être complexes à diagnostiquer et à résoudre. Ces erreurs se produisent souvent lorsque vos exigences en matière de traitement des données dépassent les ressources de mémoire disponibles, que ce soit sur le nœud du pilote ou les nœuds d'exécuteur.

- Problèmes de capacité du disque Spark : les erreurs liées au stockage dans les tâches AWS Glue Spark apparaissent souvent lors d'opérations de remaniement, de fuite de données ou lors de transformations de données à grande échelle. Ces erreurs peuvent être particulièrement délicates, car elles peuvent ne pas se manifester avant que votre tâche ne soit en cours d'exécution pendant une certaine durée, ce qui peut entraîner une perte de temps et de ressources de calcul précieux.
- Erreurs d'exécution des requêtes : les échecs de requête dans Spark SQL et les DataFrame opérations peuvent être difficiles à résoudre car les messages d'erreur peuvent ne pas indiquer clairement la cause première, et les requêtes qui fonctionnent correctement avec de petits ensembles de données peuvent soudainement échouer à grande échelle. Ces erreurs deviennent encore plus difficiles lorsqu'elles se produisent au cœur de pipelines de transformation complexes, où le véritable problème peut être dû à des problèmes de qualité des données survenus aux étapes précédentes plutôt qu'à la logique de requête elle-même.

> **Note**
>
> Avant de mettre en œuvre les modifications suggérées dans votre environnement de production, examinez-les attentivement. Le service fournit des recommandations basées sur des modèles et des bonnes pratiques, mais votre cas d'utilisation spécifique peut nécessiter des considérations supplémentaires.

Régions prises en charge

Le dépannage génératif de l'IA pour Apache Spark est disponible dans les régions suivantes :

- Afrique : Le Cap (af-south-1)
- Asie-Pacifique : Hong Kong (ap-east-1), Tokyo (ap-northeast-1), Séoul (ap-northeast-2), Osaka (ap-northeast-3), Mumbai (ap-south-1), Singapour (ap-southeast-1) ap-southeast-1), Sydney (ap-southeast-2) et Jakarta (ap-southeast-3)
- Europe : Francfort (eu-central-1), Stockholm (eu-nord-1), Milan (eu-sud-1), Irlande (eu-west-1), Londres (eu-west-2) et Paris (eu-west-3)
- Moyen-Orient : Bahreïn (me-south-1) et Émirats arabes unis (me-central-1)
- Amérique du Nord : Canada (ca-central-1)
- Amérique du Sud : São Paulo (sa-east-1)
- États-Unis : Virginie du Nord (us-east-1), Ohio (us-east-2), Californie du Nord (us-west-1) et Oregon (us-west-2)

Utilisation de vues matérialisées avec Glue AWS

AWS Les versions 5.1 et ultérieures de Glue permettent de créer et de gérer des vues matérialisées Apache Iceberg dans le catalogue de données AWS Glue. Une vue matérialisée est une table gérée qui stocke le résultat précalculé d'une requête SQL au format Apache Iceberg et qui est mise à jour progressivement à mesure que les tables sources sous-jacentes changent. Vous pouvez utiliser des vues matérialisées pour simplifier les pipelines de transformation des données et accélérer les performances des requêtes pour les charges de travail analytiques complexes.

Lorsque vous créez une vue matérialisée à l'aide de Spark in AWS Glue, la définition de la vue et les métadonnées sont stockées dans le catalogue de données AWS Glue. Les résultats précalculés sont stockés sous forme de tables Apache Iceberg dans des compartiments Amazon S3 Tables ou dans des compartiments à usage général Amazon S3 au sein de votre compte. Le catalogue de données AWS Glue surveille automatiquement les tables sources et actualise les vues matérialisées à l'aide d'une infrastructure informatique gérée.

Rubriques

- Comment fonctionnent les vues matérialisées avec Glue AWS
- Conditions préalables
- Configuration de Spark pour utiliser des vues matérialisées
- Création de vues matérialisées
- Interrogation de vues matérialisées
- Actualisation de vues matérialisées
- Gestion des vues matérialisées
- Autorisations pour les vues matérialisées
- Surveillance des opérations de visualisation matérialisée
- Exemple : flux de travail complet
- Considérations et restrictions

Comment fonctionnent les vues matérialisées avec Glue AWS

Les vues matérialisées s'intègrent à AWS Glue grâce au support Iceberg d'Apache Spark dans les tâches AWS Glue et les blocs-notes AWS Glue Studio. Lorsque vous configurez votre session Spark pour utiliser le catalogue de données AWS Glue, vous pouvez créer des vues matérialisées à l'aide de la syntaxe SQL standard. L'optimiseur Spark peut automatiquement réécrire les requêtes pour

utiliser des vues matérialisées lorsqu'elles offrent de meilleures performances, éliminant ainsi le besoin de modifier manuellement le code de l'application.

Le catalogue de données AWS Glue gère tous les aspects opérationnels de la maintenance des vues matérialisées, notamment :

- Détecter les modifications dans les tables sources à l'aide de la couche de métadonnées d'Apache Iceberg
- Planification et exécution des opérations d'actualisation à l'aide du calcul géré par Spark
- Déterminer s'il convient d'effectuer une actualisation complète ou incrémentielle en fonction des modifications apportées aux données
- Stockage des résultats précalculés au format Apache Iceberg pour un accès multimoteur

Vous pouvez interroger des vues matérialisées à partir de AWS Glue en utilisant les mêmes interfaces SQL Spark que celles que vous utilisez pour les tables classiques. Les données précalculées sont également accessibles depuis d'autres services, notamment Amazon Athena et Amazon Redshift.

Conditions préalables

Pour utiliser des vues matérialisées avec AWS Glue, vous devez :

- Un compte
- AWS Glue version 5.1 ou ultérieure
- Tables sources au format Apache Iceberg enregistrées dans le catalogue de données AWS Glue
- AWS Lake Formation autorisations configurées pour les tables sources et les bases de données cibles
- Un bucket S3 Tables ou un bucket S3 à usage général enregistré auprès duquel sont AWS Lake Formation stockées des données de vue matérialisées
- Rôle IAM autorisé à accéder à AWS Glue Data Catalog et à Amazon S3

Configuration de Spark pour utiliser des vues matérialisées

Pour créer et gérer des vues matérialisées dans AWS Glue, configurez votre session Spark avec les extensions Iceberg et les paramètres de catalogue requis. La méthode de configuration varie selon que vous utilisez des tâches AWS Glue ou des blocs-notes AWS Glue Studio.

Configuration des tâches AWS Glue

Lorsque vous créez ou mettez à jour une tâche AWS Glue, ajoutez les paramètres de configuration suivants en tant que paramètres de tâche :

Pour seaux S3 Tables

```
job = glue.create_job(
    Name='materialized-view-job',
    Role='arn:aws:iam::111122223333:role/GlueServiceRole',
    Command={
        'Name': 'glueetl',
        'ScriptLocation': 's3://amzn-s3-demo-bucket/scripts/mv-script.py',
        'PythonVersion': '3'
    },
    DefaultArguments={
        '--enable-glue-datacatalog': 'true',
        '--conf':
 'spark.sql.extensions=org.apache.iceberg.spark.extensions.IcebergSparkSessionExtensions
 '
        '--conf spark.sql.catalog.glue_catalog=org.apache.iceberg.spark.SparkCatalog '
                  '--conf spark.sql.catalog.glue_catalog.type=glue '
                  '--conf spark.sql.catalog.glue_catalog.warehouse=s3://amzn-s3-demo-
bucket/warehouse '
                  '--conf spark.sql.catalog.glue_catalog.glue.region=us-east-1 '
                  '--conf spark.sql.catalog.glue_catalog.glue.id=111122223333 '
                  '--conf spark.sql.catalog.glue_catalog.glue.account-id=111122223333
 ',
                  '--conf spark.sql.catalog.glue_catalog.glue.lakeformation-
enabled=true ',
                  '--conf
 spark.sql.catalog.s3t_catalog=org.apache.iceberg.spark.SparkCatalog '
                  '--conf spark.sql.catalog.s3t_catalog.type=glue '
                  '--conf
 spark.sql.catalog.s3t_catalog.glue.id=111122223333:s3tablescatalog/my-table-bucket ',
                  '--conf spark.sql.catalog.s3t_catalog.glue.account-id=111122223333 ',
                  '--conf spark.sql.catalog.s3t_catalog.glue.lakeformation-enabled=true
 ',
                  '--conf spark.sql.catalog.s3t_catalog.warehouse=s3://amzn-s3-demo-
bucket/mv-warehouse '
                  '--conf spark.sql.catalog.s3t_catalog.glue.region=us-east-1 '
                  '--conf spark.sql.defaultCatalog=s3t_catalog '
                  '--conf spark.sql.optimizer.answerQueriesWithMVs.enabled=true '
```

```
                  '--conf spark.sql.materializedViews.metadataCache.enabled=true'
    },
    GlueVersion='5.1'
)
```

Pour les godets à usage général S3

```
job = glue.create_job(
    Name='materialized-view-job',
    Role='arn:aws:iam::111122223333:role/GlueServiceRole',
    Command={
        'Name': 'glueetl',
        'ScriptLocation': 's3://amzn-s3-demo-bucket/scripts/mv-script.py',
        'PythonVersion': '3'
    },
    DefaultArguments={
        '--enable-glue-datacatalog': 'true',
        '--conf':
 'spark.sql.extensions=org.apache.iceberg.spark.extensions.IcebergSparkSessionExtensions
 '
                  '--conf
 spark.sql.catalog.glue_catalog=org.apache.iceberg.spark.SparkCatalog '
                  '--conf spark.sql.catalog.glue_catalog.type=glue '
                  '--conf spark.sql.catalog.glue_catalog.warehouse=s3://amzn-s3-demo-
bucket/warehouse '
                  '--conf spark.sql.catalog.glue_catalog.glue.region=us-east-1 '
                  '--conf spark.sql.catalog.glue_catalog.glue.id=111122223333 ',
                  '--conf spark.sql.catalog.glue_catalog.glue.account-id=111122223333
 ',
                  '--conf spark.sql.catalog.glue_catalog.glue.lakeformation-
enabled=true ',
                  '--conf spark.sql.defaultCatalog=glue_catalog '
                  '--conf spark.sql.optimizer.answerQueriesWithMVs.enabled=true '
                  '--conf spark.sql.materializedViews.metadataCache.enabled=true'
    },
    GlueVersion='5.1'
)
```

Configuration des blocs-notes AWS Glue Studio

Dans les blocs-notes AWS Glue Studio, configurez votre session Spark à l'aide de la commande magique %%configure au début de votre bloc-notes :

```
%%configure
{
    "conf": {
        "spark.sql.extensions":
 "org.apache.iceberg.spark.extensions.IcebergSparkSessionExtensions",
        "spark.sql.catalog.glue_catalog": "org.apache.iceberg.spark.SparkCatalog",
        "spark.sql.catalog.glue_catalog.type": "glue",
        "spark.sql.catalog.glue_catalog.warehouse": "s3://amzn-s3-demo-bucket/
warehouse",
        "spark.sql.catalog.glue_catalog.glue.region": "us-east-1",
        "spark.sql.catalog.glue_catalog.glue.id": "111122223333",
        "spark.sql.catalog.glue_catalog.glue.account-id": "111122223333",
        "spark.sql.catalog.glue_catalog.glue.lakeformation-enabled": "true",
        "spark.sql.defaultCatalog": "glue_catalog",
        "spark.sql.optimizer.answerQueriesWithMVs.enabled": "true",
        "spark.sql.materializedViews.metadataCache.enabled": "true"
    }
}
```

Activation de l'actualisation incrémentielle

Pour activer l'optimisation de l'actualisation incrémentielle, ajoutez les propriétés de configuration suivantes aux paramètres de votre tâche ou à la configuration de votre bloc-notes :

```
--conf spark.sql.optimizer.incrementalMVRefresh.enabled=true
--conf spark.sql.optimizer.incrementalMVRefresh.deltaThresholdCheckEnabled=false
```

Paramètres de configuration

Les paramètres de configuration suivants contrôlent le comportement des vues matérialisées :

- `spark.sql.extensions`— Active les extensions de session Iceberg Spark requises pour la prise en charge des vues matérialisées.
- `spark.sql.optimizer.answerQueriesWithMVs.enabled`— Permet la réécriture automatique des requêtes pour utiliser des vues matérialisées. Définissez ce paramètre sur true pour activer cette optimisation.
- `spark.sql.materializedViews.metadataCache.enabled`— Active la mise en cache des métadonnées des vues matérialisées pour l'optimisation des requêtes. Définissez ce paramètre sur true pour améliorer les performances de réécriture des requêtes.

- `spark.sql.optimizer.incrementalMVRefresh.enabled`— Permet une optimisation de l'actualisation incrémentielle. Définissez cette valeur sur true pour traiter uniquement les données modifiées lors des opérations d'actualisation.
- `spark.sql.optimizer.answerQueriesWithMVs.decimalAggregateCheckEnabled`— Contrôle la validation des opérations d'agrégation décimale lors de la réécriture des requêtes. Définissez cette valeur sur false pour désactiver certains contrôles de dépassement de décimales.

Création de vues matérialisées

Vous créez des vues matérialisées à l'aide de l'instruction SQL CREATE MATERIALIZED VIEW dans les jobs ou les AWS blocs-notes Glue. La définition de la vue spécifie la logique de transformation sous la forme d'une requête SQL qui fait référence à une ou plusieurs tables sources.

Création d'une vue matérialisée de base dans les tâches AWS Glue

L'exemple suivant montre comment créer une vue matérialisée dans un script de tâche AWS Glue. Utilisez des noms de table complets avec une convention de dénomination en trois parties dans la définition de la vue :

```
from awsglue.context import GlueContext
from pyspark.context import SparkContext

sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session

# Create materialized view
spark.sql("""
    CREATE MATERIALIZED VIEW customer_orders
    AS
    SELECT
        customer_name,
        COUNT(*) as order_count,
        SUM(amount) as total_amount
    FROM glue_catalog.sales.orders
    GROUP BY customer_name
""")
```

Création d'une vue matérialisée avec actualisation automatique

Pour configurer l'actualisation automatique, spécifiez un calendrier d'actualisation lors de la création de la vue, en utilisant des noms de table complets avec une convention de dénomination en trois parties dans la définition de la vue :

```
spark.sql("""
    CREATE MATERIALIZED VIEW customer_orders
    SCHEDULE REFRESH EVERY 1 HOUR
    AS
    SELECT
        customer_name,
        COUNT(*) as order_count,
        SUM(amount) as total_amount
    FROM glue_catalog.sales.orders
    GROUP BY customer_name
""")
```

Création d'une vue matérialisée avec des références croisées à des catalogues

Lorsque vos tables sources se trouvent dans un catalogue différent de celui de votre vue matérialisée, utilisez des noms de table complets avec une convention de dénomination en trois parties à la fois dans le nom de la vue et dans la définition de la vue :

```
spark.sql("""
    CREATE MATERIALIZED VIEW s3t_catalog.analytics.customer_summary
    AS
    SELECT
        customer_name,
        COUNT(*) as order_count,
        SUM(amount) as total_amount
    FROM glue_catalog.sales.orders
    GROUP BY customer_name
""")
```

Création de vues matérialisées dans les blocs-notes AWS Glue Studio

Dans les blocs-notes AWS Glue Studio, vous pouvez utiliser la commande magique %%sql pour créer des vues matérialisées, en utilisant des noms de table complets avec une convention de dénomination en trois parties pour la définition des vues :

```
%%sql
```

```
CREATE MATERIALIZED VIEW customer_orders
AS
SELECT
    customer_name,
    COUNT(*) as order_count,
    SUM(amount) as total_amount
FROM glue_catalog.sales.orders
GROUP BY customer_name
```

Interrogation de vues matérialisées

Après avoir créé une vue matérialisée, vous pouvez l'interroger comme n'importe quelle autre table à l'aide des instructions SQL SELECT standard dans vos jobs AWS Glue ou vos carnets de notes.

Effectuer des requêtes dans AWS Glue Jobs

```
from awsglue.context import GlueContext
from pyspark.context import SparkContext

sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session

# Query materialized view
result = spark.sql("SELECT * FROM customer_orders")
result.show()
```

Effectuer des requêtes dans les AWS blocs-notes Glue Studio

```
%%sql
SELECT * FROM customer_orders
```

Réécriture automatique des requêtes

Lorsque la réécriture automatique des requêtes est activée, l'optimiseur Spark analyse vos requêtes et utilise automatiquement des vues matérialisées lorsqu'elles peuvent améliorer les performances. Par exemple, si vous exécutez la requête suivante :

```
result = spark.sql("""
    SELECT
        customer_name,
        COUNT(*) as order_count,
```

```
        SUM(amount) as total_amount
    FROM orders
    GROUP BY customer_name
""")
```

L'optimiseur Spark réécrit automatiquement cette requête pour utiliser la vue matérialisée customer_orders au lieu de traiter la table des commandes de base, à condition que la vue matérialisée soit à jour.

Vérification de la réécriture automatique des requêtes

Pour vérifier si une requête utilise la réécriture automatique des requêtes, utilisez la commande EXPLAIN EXTENDED :

```
spark.sql("""
    EXPLAIN EXTENDED
    SELECT customer_name, COUNT(*) as order_count, SUM(amount) as total_amount
    FROM orders
    GROUP BY customer_name
""").show(truncate=False)
```

Dans le plan d'exécution, recherchez le nom de la vue matérialisée dans l' BatchScan opération. Si le plan affiche BatchScan glue_catalog.analytics.customer_orders au lieu de glue_catalog.sales.orders, la requête a été automatiquement réécrite pour BatchScan utiliser la vue matérialisée.

Notez que la réécriture automatique des requêtes nécessite du temps pour que le cache de métadonnées Spark soit rempli après la création d'une vue matérialisée. Ce processus se termine généralement en 30 secondes.

Actualisation de vues matérialisées

Vous pouvez actualiser les vues matérialisées à l'aide de deux méthodes : actualisation complète ou actualisation incrémentielle. L'actualisation complète recalcule l'intégralité de la vue matérialisée à partir de toutes les données de la table de base, tandis que l'actualisation incrémentielle traite uniquement les données modifiées depuis la dernière actualisation.

Actualisation complète manuelle dans les tâches AWS Glue

Pour actualiser complètement une vue matérialisée :

```
spark.sql("REFRESH MATERIALIZED VIEW customer_orders FULL")
```

```
# Verify updated results
result = spark.sql("SELECT * FROM customer_orders")
result.show()
```

Actualisation incrémentielle manuelle dans les tâches AWS Glue

Pour effectuer une actualisation incrémentielle, assurez-vous que l'actualisation incrémentielle est activée dans la configuration de votre session Spark, puis exécutez :

```
spark.sql("REFRESH MATERIALIZED VIEW customer_orders")

# Verify updated results
result = spark.sql("SELECT * FROM customer_orders")
result.show()
```

Le catalogue de données AWS Glue détermine automatiquement si une actualisation incrémentielle est applicable en fonction de la définition de la vue et de la quantité de données modifiées. Si l'actualisation incrémentielle n'est pas possible, l'opération revient à une actualisation complète.

Rafraîchissant dans les blocs-notes AWS Glue Studio

Dans les ordinateurs portables, utilisez la commande magique %%sql :

```
%%sql
REFRESH MATERIALIZED VIEW customer_orders FULL
```

Vérification de l'exécution de l'actualisation incrémentielle

Pour confirmer que l'actualisation incrémentielle a bien été exécutée, activez la journalisation du débogage dans votre tâche AWS Glue :

```
from awsglue.context import GlueContext
from pyspark.context import SparkContext
import logging

sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session

# Enable debug logging
logger = logging.getLogger('org.apache.spark.sql')
logger.setLevel(logging.DEBUG)
```

```
# Execute refresh
spark.sql("REFRESH MATERIALIZED VIEW customer_orders")
```

Recherchez le message suivant dans les journaux des tâches de AWS Glue :

```
DEBUG RefreshMaterializedViewExec: Executed Incremental Refresh
```

Gestion des vues matérialisées

AWS Glue fournit des commandes SQL pour gérer le cycle de vie des vues matérialisées dans vos tâches et carnets de notes.

Décrire une vue matérialisée

Pour afficher les métadonnées relatives à une vue matérialisée, notamment sa définition, son état d'actualisation et l'horodatage de la dernière actualisation :

```
spark.sql("DESCRIBE EXTENDED customer_orders").show(truncate=False)
```

Modification d'une vue matérialisée

Pour modifier le calendrier d'actualisation d'une vue matérialisée existante :

```
spark.sql("""
    ALTER MATERIALIZED VIEW customer_orders
    ADD SCHEDULE REFRESH EVERY 2 HOURS
""")
```

Pour supprimer l'actualisation automatique :

```
spark.sql("""
    ALTER MATERIALIZED VIEW customer_orders
    DROP SCHEDULE
""")
```

Supprimer une vue matérialisée

Pour supprimer une vue matérialisée :

```
spark.sql("DROP MATERIALIZED VIEW customer_orders")
```

Cette commande supprime la définition de la vue matérialisée du catalogue de données AWS Glue et supprime les données de la table Iceberg sous-jacente de votre compartiment S3.

Répertorier les vues matérialisées

Pour répertorier toutes les vues matérialisées d'une base de données :

```
spark.sql("SHOW VIEWS FROM analytics").show()
```

Autorisations pour les vues matérialisées

Pour créer et gérer des vues matérialisées, vous devez configurer AWS Lake Formation les autorisations. Le rôle IAM qui crée la vue matérialisée (le rôle de définition) nécessite des autorisations spécifiques sur les tables sources et les bases de données cibles.

Autorisations requises pour le rôle de définisseur

Le rôle de définition doit disposer des autorisations Lake Formation suivantes :

- Sur les tables sources : autorisations SELECT ou ALL sans filtres de ligne, de colonne ou de cellule
- Sur la base de données cible : autorisation CREATE_TABLE
- Sur le catalogue AWS de données Glue GetTable et les autorisations CreateTable d'API

Lorsque vous créez une vue matérialisée, l'ARN du rôle de définition est stocké dans la définition de la vue. Le catalogue de données AWS Glue assume ce rôle lorsqu'il exécute des opérations d'actualisation automatique. Si le rôle de définition perd l'accès aux tables sources, les opérations d'actualisation échoueront tant que les autorisations ne seront pas restaurées.

Autorisations IAM pour les tâches AWS Glue

Le rôle IAM de votre tâche AWS Glue nécessite les autorisations suivantes :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:GetCatalog",
                "glue:GetCatalogs",
```

```
                "glue:GetTable",
                "glue:GetTables",
                "glue:CreateTable",
                "glue:UpdateTable",
                "glue:DeleteTable",
                "glue:GetDatabase",
                "glue:GetDatabases",
                "cloudwatch:PutMetricData"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:PutObject",
                "s3:DeleteObject"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": [
                "arn:aws:logs:*:*:*:/aws-glue/*"
            ]
        },
        {
            "Effect": "Allow",
```

```
            "Action": [
                "lakeformation:GetDataAccess"
            ],
            "Resource": "*"
        }
    ]
}
```

Le rôle que vous utilisez pour l'actualisation automatique de Materialized View doit disposer de l'PassRole autorisation iam : sur le rôle.

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iam:PassRole"
      ],
      "Resource": [
        "arn:aws:iam::111122223333:role/materialized-view-role-name"
      ]
    }
  ]
}
```

Pour que Glue actualise automatiquement la vue matérialisée pour vous, le rôle doit également respecter la politique de confiance suivante qui permet au service d'assumer le rôle.

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iam:PassRole"
      ],
      "Resource": [
        "arn:aws:iam::111122223333:role/materialized-view-role-name"
```

```
      ]
    }
  ]
}
```

Si la vue matérialisée est stockée dans des compartiments de tables S3, vous devez également ajouter l'autorisation suivante au rôle.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3tables:PutTableMaintenanceConfiguration"
      ],
      "Resource": "arn:aws:s3tables:*:123456789012:*"
    }
  ]
}
```

Octroi de l'accès aux vues matérialisées

Pour autoriser d'autres utilisateurs à interroger une vue matérialisée, accordez l' AWS Lake Formation autorisation SELECT sur la table des vues matérialisées. Les utilisateurs peuvent interroger la vue matérialisée sans avoir besoin d'accéder directement aux tables sources sous-jacentes.

Pour des informations détaillées sur la configuration des autorisations de Lake Formation, consultez la section Octroi et révocation d'autorisations sur les ressources du catalogue de données dans le manuel du AWS Lake Formation développeur.

Surveillance des opérations de visualisation matérialisée

Le AWS Glue Data Catalog publie des statistiques et des journaux pour les opérations d'actualisation des vues matérialisées sur Amazon CloudWatch. Vous pouvez surveiller l'état, la durée et le volume de données traités par le biais de CloudWatch métriques d'actualisation.

Afficher les journaux des tâches

Pour consulter les journaux des tâches AWS Glue qui créent ou actualisent des vues matérialisées, procédez comme suit :

1. Ouvrez la console AWS Glue.
2. Choisissez Jobs dans le volet de navigation.
3. Sélectionnez votre tâche et choisissez Runs.
4. Sélectionnez une exécution spécifique et choisissez Logs pour afficher CloudWatch les journaux.

Configuration des alarmes

Pour recevoir des notifications lorsque les opérations d'actualisation échouent ou dépassent la durée prévue, créez des CloudWatch alarmes sur les métriques des vues matérialisées. Vous pouvez également configurer les EventBridge règles Amazon pour déclencher des réponses automatisées afin d'actualiser les événements.

Exemple : flux de travail complet

L'exemple suivant illustre un flux de travail complet pour créer et utiliser une vue matérialisée dans AWS Glue.

Exemple de script AWS de tâche Glue

```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job

args = getResolvedOptions(sys.argv, ['JOB_NAME'])
sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
job = Job(glueContext)
job.init(args['JOB_NAME'], args)

# Create database and base table
spark.sql("CREATE DATABASE IF NOT EXISTS sales")
spark.sql("USE sales")
```

```
spark.sql("""
    CREATE TABLE IF NOT EXISTS orders (
        id INT,
        customer_name STRING,
        amount DECIMAL(10,2),
        order_date DATE
    )
""")

# Insert sample data
spark.sql("""
    INSERT INTO orders VALUES
        (1, 'John Doe', 150.00, DATE('2024-01-15')),
        (2, 'Jane Smith', 200.50, DATE('2024-01-16')),
        (3, 'Bob Johnson', 75.25, DATE('2024-01-17'))
""")

# Create materialized view
spark.sql("""
    CREATE MATERIALIZED VIEW customer_summary
    AS
    SELECT
        customer_name,
        COUNT(*) as order_count,
        SUM(amount) as total_amount
    FROM glue_catalog.sales.orders
    GROUP BY customer_name
""")

# Query the materialized view
print("Initial materialized view data:")
spark.sql("SELECT * FROM customer_summary").show()

# Insert additional data
spark.sql("""
    INSERT INTO orders VALUES
        (4, 'Jane Smith', 350.00, DATE('2024-01-18')),
        (5, 'Bob Johnson', 100.25, DATE('2024-01-19'))
""")

# Refresh the materialized view
spark.sql("REFRESH MATERIALIZED VIEW customer_summary FULL")
```

```
# Query updated results
print("Updated materialized view data:")
spark.sql("SELECT * FROM customer_summary").show()

job.commit()
```

Exemple de bloc-notes AWS Glue Studio

```
%%configure
{
    "conf": {
        "spark.sql.extensions":
 "org.apache.iceberg.spark.extensions.IcebergSparkSessionExtensions",
        "spark.sql.catalog.glue_catalog": "org.apache.iceberg.spark.SparkCatalog",
        "spark.sql.catalog.glue_catalog.type": "glue",
        "spark.sql.catalog.glue_catalog.warehouse": "s3://amzn-s3-demo-bucket/
warehouse",
        "spark.sql.catalog.glue_catalog.glue.region": "us-east-1",
        "spark.sql.catalog.glue_catalog.glue.id": "111122223333",
        "spark.sql.catalog.glue_catalog.glue.account-id": "111122223333",
        "spark.sql.catalog.glue_catalog.glue.lakeformation-enabled": "true",
        "spark.sql.defaultCatalog": "glue_catalog",
        "spark.sql.optimizer.answerQueriesWithMVs.enabled": "true",
        "spark.sql.materializedViews.metadataCache.enabled": "true"
    }
}
```

```
%%sql
CREATE DATABASE IF NOT EXISTS sales
```

```
%%sql
USE sales
```

```
%%sql
CREATE TABLE IF NOT EXISTS orders (
    id INT,
    customer_name STRING,
    amount DECIMAL(10,2),
    order_date DATE
)
```

```
%%sql
INSERT INTO orders VALUES
    (1, 'John Doe', 150.00, DATE('2024-01-15')),
    (2, 'Jane Smith', 200.50, DATE('2024-01-16')),
    (3, 'Bob Johnson', 75.25, DATE('2024-01-17'))
```

```
%%sql
CREATE MATERIALIZED VIEW customer_summary
AS
SELECT
    customer_name,
    COUNT(*) as order_count,
    SUM(amount) as total_amount
FROM glue_catalog.sales.orders
GROUP BY customer_name
```

```
%%sql
SELECT * FROM customer_summary
```

```
%%sql
INSERT INTO orders VALUES
    (4, 'Jane Smith', 350.00, DATE('2024-01-18')),
    (5, 'Bob Johnson', 100.25, DATE('2024-01-19'))
```

```
%%sql
REFRESH MATERIALIZED VIEW customer_summary FULL
```

```
%%sql
SELECT * FROM customer_summary
```

Considérations et restrictions

Lorsque vous utilisez des vues matérialisées avec AWS Glue, tenez compte des points suivants :

- Les vues matérialisées nécessitent AWS la version 5.1 ou ultérieure de Glue.
- Les tables sources doivent être des tables Apache Iceberg enregistrées dans le catalogue de données AWS Glue. Les tables Apache Hive, Apache Hudi et Linux Foundation Delta Lake ne sont pas prises en charge au lancement.

- Les tables sources doivent résider dans la même région et dans le même compte que la vue matérialisée.
- Toutes les tables sources doivent être régies par AWS Lake Formation. Les autorisations IAM uniquement et l'accès hybride ne sont pas pris en charge.
- Les vues matérialisées ne peuvent pas faire référence aux vues AWS Glue Data Catalog, aux vues multidialectes ou à d'autres vues matérialisées en tant que tables sources.
- Le rôle de définition de vues doit disposer d'un accès complet en lecture (autorisation SELECT ou ALL) sur toutes les tables sources sans appliquer de filtres de ligne, de colonne ou de cellule.
- Les vues matérialisées sont finalement cohérentes avec les tables sources. Pendant la fenêtre d'actualisation, les requêtes peuvent renvoyer des données périmées. Exécutez une actualisation manuelle pour une cohérence immédiate.
- L'intervalle d'actualisation automatique minimum est d'une heure.
- L'actualisation incrémentielle prend en charge un sous-ensemble restreint d'opérations SQL. La définition de la vue doit être un seul bloc SELECT-FROM-WHERE-GROUP BY-HAVING et ne peut pas contenir d'opérations définies, de sous-requêtes, le mot clé DISTINCT dans SELECT ou de fonctions d'agrégation, de fonctions de fenêtre ou de jointures autres que INNER JOIN.
- L'actualisation incrémentielle ne prend pas en charge les fonctions définies par l'utilisateur ni certaines fonctions intégrées. Seul un sous-ensemble des fonctions intégrées de Spark SQL est pris en charge.
- La réécriture automatique des requêtes ne prend en compte que les vues matérialisées dont les définitions appartiennent à un sous-ensemble SQL restreint similaire aux restrictions d'actualisation incrémentielle.
- Les identifiants contenant des caractères spéciaux autres que des caractères alphanumériques et des traits de soulignement ne sont pas pris en charge dans les requêtes CREATE MATERIALIZED VIEW. Cela s'applique à tous les types d'identifiant, y compris les catalog/namespace/table noms, les noms de colonnes et de champs de structure CTEs, ainsi que les alias.
- Les colonnes de vues matérialisées commençant par le préfixe __ivm sont réservées à l'utilisation du système. Amazon se réserve le droit de modifier ou de supprimer ces colonnes dans les futures versions.
- Les clauses SORT BY, LIMIT, OFFSET, CLUSTER BY et ORDER BY ne sont pas prises en charge dans les définitions de vues matérialisées.
- Les tables sources entre régions et entre comptes ne sont pas prises en charge.

- Les tables référencées dans la requête de vue doivent utiliser une convention de dénomination en trois parties (par exemple, glue_catalog.my_db.my_table) car l'actualisation automatique n'utilise pas les paramètres de catalogue et de base de données par défaut.
- Les opérations d'actualisation complète remplacent l'intégralité du tableau et rendent les instantanés précédents indisponibles.
- Les fonctions non déterministes telles que rand () ou current_timestamp () ne sont pas prises en charge dans les définitions de vues matérialisées.

AWS Glue types de travailleurs

Présentation de

AWS Glue propose plusieurs types de travailleurs pour répondre aux différentes exigences de charge de travail, qu'il s'agisse de petites tâches de streaming ou de tâches de traitement de données à grande échelle nécessitant beaucoup de mémoire. Cette section fournit des informations complètes sur tous les types de travailleurs disponibles, leurs spécifications et les recommandations d'utilisation.

Catégories de types de travailleur

AWS Glue propose deux catégories principales de types de travailleurs :

- Types de travailleurs G : travailleurs de calcul polyvalents optimisés pour les charges de travail ETL standard
- Types de travailleurs R : travailleurs optimisés pour la mémoire conçus pour les applications Spark gourmandes en mémoire

Unités de traitement de données (DPUs)

Les ressources disponibles pour les AWS Glue travailleurs sont mesurées en DPUs. Un DPU est une mesure relative de la puissance de traitement composée de 4 V de capacité CPUs de calcul et de 16 Go de mémoire.

Optimisé pour la mémoire DPUs (M-DPUs) : les travailleurs de type R utilisent M-DPUs, qui fournit le double de l'allocation de mémoire pour une taille donnée par rapport à la norme. DPUs Cela signifie qu'alors qu'une DPU standard fournit 16 Go de mémoire, une M-DPU dans des travailleurs de type R fournit 32 Go de mémoire optimisée pour les applications Spark gourmandes en mémoire.

Types de travailleurs disponibles

G.1X

- DPU : 1 DPU (4 VCPUs, 16 Go de mémoire)
- Stockage : disque de 94 Go (environ 44 Go libres)
- Cas d'utilisation : transformations, jointures et requêtes de données : évolutifs et économiques pour la plupart des tâches

G.2X

- DPU : 2 DPU (8 VCPUs, 32 Go de mémoire)
- Stockage : disque de 138 Go (environ 78 Go libres)
- Cas d'utilisation : transformations, jointures et requêtes de données : évolutifs et économiques pour la plupart des tâches

G.4X

- Processeur : 4 processeurs (16 VCPUs, 64 Go de mémoire)
- Stockage : disque de 256 Go (environ 230 Go libres)
- Cas d'utilisation : transformations, agrégations, jointures et requêtes exigeantes

G.8X

- DPU : 8 DPU (32 VCPUs, 128 Go de mémoire)
- Stockage : disque de 512 Go (environ 485 Go libres)
- Cas d'utilisation : transformations, agrégations, jointures et requêtes exigeantes

G.12X

- DPU : 12 DPU (48 VCPUs, 192 Go de mémoire)
- Stockage : disque de 768 Go (environ 741 Go libres)
- Cas d'utilisation : charges de travail très importantes et gourmandes en ressources nécessitant une capacité de calcul importante

G.16X

- DPU : 16 DPU (64 VCPUs, 256 Go de mémoire)
- Stockage : disque de 1 024 Go (environ 996 Go libres)
- Cas d'utilisation : charges de travail les plus importantes et gourmandes en ressources nécessitant une capacité de calcul maximale

R.1X - Mémoire optimisée *

- DPU : 1 M-DPU (4 VCPUs, 32 Go de mémoire)
- Cas d'utilisation : charges de travail gourmandes en mémoire associées à des out-of-memory erreurs fréquentes ou à des exigences de ratio élevées memory-to-CPU

R.2X - Mémoire optimisée *

- DPU : 2 M-DPU (8 VCPUs, 64 Go de mémoire)
- Cas d'utilisation : charges de travail gourmandes en mémoire associées à des out-of-memory erreurs fréquentes ou à des exigences de ratio élevées memory-to-CPU

R.4X - Mémoire optimisée *

- DPU : 4 M-DPU (16 VCPUs, 128 Go de mémoire)
- Cas d'utilisation : grandes charges de travail gourmandes en mémoire avec des out-of-memory erreurs fréquentes ou des exigences de ratio élevées memory-to-CPU

R.8X - Mémoire optimisée *

- DPU : 8 M-DPU (32 VCPUs, 256 Go de mémoire)
- Cas d'utilisation : très grandes charges de travail gourmandes en mémoire avec des out-of-memory erreurs fréquentes ou des exigences de ratio élevées memory-to-CPU

* Vous pouvez rencontrer une latence de démarrage plus élevée avec ces travailleurs. Pour résoudre ce problème, essayez ce qui suit :

- Attendez quelques minutes et soumettez à nouveau votre tâche.

- Soumettez une nouvelle tâche avec un nombre réduit de travailleurs.
- Soumettez une nouvelle tâche en utilisant un type ou une taille de travailleur différents.

Table de spécifications des types de travailleur

Spécifications des types de travailleur

Type de travailleur	DPU par nœud	vCPU	Mémoire (Go)	Disque (Go)	Espace disque disponible approximatif (Go)	Exécuteurs Spark par nœud
G.1X	1	4	16	94	44	1
G.2X	2	8	32	138	78	1
G.4X	4	16	64	256	230	1
G.8X	8	32	128	512	485	1
G.12X	12	48	192	768	741	1
G.16X	16	64	256	1 024	996	1
R.1 X	1	4	32	94	44	1
R.2X	2	8	64	138	78	1
R.4X	4	16	128	256	230	1
R.8X	8	32	256	512	485	1

Remarque : Les types de travailleurs R ont des configurations optimisées pour la mémoire avec des spécifications optimisées pour les charges de travail gourmandes en mémoire.

Considérations importantes

Latence de démarrage

> **Important**
>
> Les types de travailleurs G.12X et G.16X, ainsi que tous les types de travailleurs R (R.1X à R.8X), peuvent subir une latence de démarrage plus élevée. Pour résoudre ce problème, essayez ce qui suit :
>
> - Attendez quelques minutes et soumettez à nouveau votre tâche.
> - Soumettez une nouvelle tâche avec un nombre réduit de travailleurs.
> - Soumettez une nouvelle tâche en utilisant un type et une taille de travailleur différents.

Choix du type de travailleur approprié

Pour les charges de travail ETL standard

- G.1X ou G.2X : solution la plus rentable pour les transformations de données, les jointures et les requêtes classiques
- G.4X ou G.8X : pour les charges de travail plus exigeantes avec des jeux de données plus volumineux

Pour les charges de travail à grande échelle

- G.12X : très grands jeux de données nécessitant d'importantes ressources de calcul
- G.16X : capacité de calcul maximale pour les charges de travail les plus exigeantes

Pour les charges de travail gourmandes en mémoire

- R.1X ou R.2X : tâches gourmandes en mémoire de petite ou moyenne taille
- R.4X ou R.8X : charges de travail gourmandes en mémoire, associées à des erreurs OOM fréquentes

Considérations en matière d'optimisation des coûts

- Travailleurs G standard : fournissent un équilibre entre les ressources de calcul, de mémoire et de mise réseau. Ils peuvent être utilisés pour diverses charges de travail à moindre coût.
- Travailleurs R : spécialisés dans les tâches gourmandes en mémoire avec des performances rapides pour les charges de travail qui traitent de larges jeux de données en mémoire.

Bonnes pratiques

Consignes de sélection des travailleurs

1. Commencez avec des travailleurs standard (G.1X, G.2X) pour la plupart des charges de travail.
2. Utilisez R Workers en cas d' out-of-memoryerreurs ou de charges de travail fréquentes liées à des opérations gourmandes en mémoire telles que la mise en cache, le remaniement et l'agrégation
3. Optez pour G.12X/G.16X pour les charges de travail gourmandes en ressources informatiques nécessitant un maximum de ressources.
4. Tenez compte des contraintes de capacité lors de l'utilisation de nouveaux types de travailleurs dans des flux de travail urgents.

Optimisation des performances

- Surveillez CloudWatch les indicateurs pour comprendre l'utilisation des ressources
- Utiliser un nombre de travailleurs approprié en fonction de la taille et de la complexité des données
- Envisager des stratégies de partitionnement des données pour optimiser l'efficacité des travailleurs

Tâches ETL en streaming dans AWS Glue

Vous pouvez créer des tâches Extract-transform-load (ETL) qui s'exécutent continu et consomment des données provenant de sources en streaming telles que Amazon Kinesis Data Streams, Apache Kafka et Amazon Managed Streaming pour Apache Kafka (Amazon MSK). Les tâches nettoient et transforment les données, puis chargent les résultats dans les lacs de données Amazon S3 ou les magasins de données JDBC.

En outre, vous pouvez produire des données pour des flux Amazon Kinesis Data Streams. Cette fonctionnalité n'est disponible que lors de l'écriture de AWS Glue scripts. Pour de plus amples informations, veuillez consulter the section called "Connexions Kinesis".

Par défaut, AWS Glue traite et écrit les données dans des fenêtres de 100 secondes. Cela permet de traiter les données de manière efficace et d'effectuer des agrégations sur les données qui arrivent plus tard que prévu. Vous pouvez modifier cette taille de fenêtre temporelle pour augmenter la ponctualité ou la précision de l'agrégation. Les tâches de streaming AWS Glue utilisent des points de contrôle au lieu de signets de tâche pour suivre les données lues.

> **Note**
>
> AWS Glue facture à l'heure les tâches ETL en streaming.

Cette vidéo aborde les défis liés aux coûts du streaming ETL et les fonctionnalités de réduction des coûts dans AWS Glue.

La création d'une tâche ETL en streaming implique les étapes suivantes :

1. Pour une source de streaming Apache Kafka, créez une connection AWS Glue à la source Kafka ou au cluster Amazon MSK.
2. Créez manuellement une table du catalogue de données pour la source en streaming.
3. Créez une tâche ETL pour la source de données de streaming. Définissez des propriétés de tâche spécifiques au streaming et fournissez votre propre script ou modifiez éventuellement le script généré.

Pour de plus amples informations, consulter Streaming d'ETL en AWS Glue.

Lorsque vous créez une tâche ETL en streaming pour Amazon Kinesis Data Streams, vous n'avez pas besoin de créer une connexion AWS Glue. Toutefois, s'il existe une connexion attachée à la tâche ETL AWS Glue en streaming pointant sur une source Kinesis Data Streams, puis un point de terminaison de Virtual Private Cloud (VPC) vers Kinesis est requis. Pour plus d'informations, consultez Création d'un point de terminaison d'interface dans le Guide de l'utilisateur Amazon VPC. Lorsque vous spécifiez un Amazon Kinesis Data Streams dans un autre compte, vous devez configurer les rôles et les stratégies pour autoriser l'accès entre comptes. Pour de plus amples informations, veuillez consulter la rubrique Exemple : Lire à partir d'un flux Kinesis dans un autre compte.

Les tâches ETL de streaming AWS Glue peuvent détecter automatiquement les données compressées, décompresser les données de streaming de manière transparente, effectuer les modifications habituelles sur la source d'entrée et charger vers le stockage de sortie.

AWS Glue prend en charge la décompression automatique pour les types de compression suivants, compte tenu du format d'entrée :

Type de compression	Fichier Avro	Référence Avro	JSON	CSV	Grok
BZIP2	Oui	Oui	Oui	Oui	Oui
GZIP	Non	Oui	Oui	Oui	Oui
SNAPPY	Oui (format Snappy brut)	Oui (format Snappy encadré)	Oui (format Snappy encadré)	Oui (format Snappy encadré)	Oui (format Snappy encadré)
XZ	Oui	Oui	Oui	Oui	Oui
ZSTD	Oui	Non	Non	Non	Non
DEFLATE	Oui	Oui	Oui	Oui	Oui

Rubriques

- Création d'une connexion AWS Glue pour un flux de données Apache Kafka
- Création d'un tableau de catalogue de données pour une source en streaming
- Notes et restrictions pour les sources en streaming Avro
- Application de modèles Grok à des sources de streaming
- Définition des propriétés de tâche pour une tâche ETL en streaming
- Restrictions et notes sur ETL en streaming

Création d'une connexion AWS Glue pour un flux de données Apache Kafka

Pour lire à partir d'un flux Apache Kafka, vous devez créer une connexion AWS Glue.

Pour créer une connexion AWS Glue pour une source Kafka (Console)

1. Ouvrez la AWS Glue console à l'adresse https://console.aws.amazon.com/glue/.
2. Dans le volet de navigation, sous Data catalog (Catalogue de données), choisissez Connections (Connexions).

3. Choisissez Add connection (Ajouter une connexion), et dans la page Set up your connection's properties (Configurer les propriétés de votre connexion), saisissez un nom de connexion.

   > **Note**
   >
   > Pour plus d'informations sur la spécification des propriétés de connexion, veuillez consulter Propriétés de connexion AWS Glue.

4. Pour Type de connexion, choisissez Kafka.
5. Pour les serveurs bootstrap Kafka URLs, entrez l'hôte et le numéro de port des courtiers bootstrap pour votre cluster Amazon MSK ou votre cluster Apache Kafka. Utilisez uniquement des points de terminaison utilisant le protocole TLS (Transport Layer Security) pour établir la connexion initiale au cluster Kafka. Les points de terminaison en texte brut ne sont pas pris en charge.

   Voici un exemple de liste de paires de nom d'hôte et de numéro de port pour un cluster Amazon MSK.

   ```
   myserver1.kafka.us-east-1.amazonaws.com:9094,myserver2.kafka.us-
   east-1.amazonaws.com:9094,
   myserver3.kafka.us-east-1.amazonaws.com:9094
   ```

   Pour plus d'informations sur l'obtention des informations du broker d'amorçage, veuillez consulter la rubrique Obtenir les brokers d'amorçage pour un cluster Amazon MSK dans le Guide du développeur Amazon Managed Streaming for Apache Kafka.
6. Si vous souhaitez une connexion sécurisée à la source de données Kafka, sélectionnez Require SSL connection (Connexion SSL obligatoire), et pour Kafka private CA certificate location (Emplacement du certificat d'autorité de certification privée Kafka), entrez un chemin d'accès Amazon S3 valide vers un certificat SSL personnalisé.

   Pour une connexion SSL à Kafka autogérée, le certificat personnalisé est obligatoire. Il est facultatif pour Amazon MSK.

   Pour plus d'informations sur la spécification d'un certificat personnalisé pour Kafka, consulter the section called "Propriétés de connexion SSL".
7. Utilisez AWS Glue Studio la AWS CLI pour spécifier une méthode d'authentification du client Kafka. Pour y accéder, AWS Glue Studio AWS Gluesélectionnez dans le menu ETL du volet de navigation de gauche.

Pour plus d'informations sur les méthodes d'authentification client Kafka, consultez AWS Glue Propriétés de connexion Kafka pour l'authentification du client.

8. Saisissez éventuellement une description, puis choisissez Suivant.
9. Pour un cluster Amazon MSK, spécifiez son Virtual Private Cloud (VPC), son sous-réseau et son groupe de sécurité. Les informations du VPC sont facultatives pour Kafka autogéré.
10. Choisissez Next (Suivant) pour vérifier toutes les propriétés de connexion, puis choisissez Finish (Terminer).

Pour plus d'informations sur les connexions AWS Glue, consultez Connexion aux données.

AWS Glue Propriétés de connexion Kafka pour l'authentification du client

Authentification SASL/GSSAPI (Kerberos)

Le choix de cette méthode d'authentification vous permettra de spécifier les propriétés Kerberos.

Kerberos Keytab

Choisissez l'emplacement du fichier keytab. Un keytab stocke les clés à long terme pour un ou plusieurs principaux. Pour en savoir plus, consultez Documentation du MIT Kerberos : Keytab.

Fichier Kerberos krb5.conf

Choisissez le fichier krb5.conf. Il contient le domaine par défaut (un réseau logique, similaire à un domaine, qui définit un groupe de systèmes sous le même KDC) et l'emplacement du serveur KDC. Pour en savoir plus, consultez Documentation MIT Kerberos : krb5.conf.

Principal Kerberos et nom de service Kerberos

Saisissez le principal Kerberos et le nom de service Kerberos. Pour plus d'informations, consultez MIT Kerberos Documentation: Kerberos principal (Documentation du MIT Kerberos : principal Kerberos).

Authentification SASL/SCRAM-SHA-512

Le choix de cette méthode d'authentification vous permettra de spécifier les informations d'identification d'authentification.

AWS Secrets Manager

Recherchez votre jeton dans la zone Search (Rechercher) en saisissant son nom ou son ARN.

Nom d'utilisateur et mot de passe du fournisseur directement

Recherchez votre jeton dans la zone Search (Rechercher) en saisissant son nom ou son ARN.

Authentification SSL client

Le choix de cette méthode d'authentification vous permet de sélectionner l'emplacement du centre de stockage des clés client Kafka en naviguant sur Amazon S3. Vous pouvez également entrer le mot de passe du centre de stockage des clés client Kafka et le mot de passe de la clé client Kafka.

Authentification IAM

Cette méthode d'authentification ne nécessite aucune spécification supplémentaire et n'est applicable que lorsque la source de diffusion en direct est MSK Kafka.

Authentification SASL/PLAIN

Le choix de cette méthode d'authentification vous permet de spécifier les informations d'identification d'authentification.

Création d'un tableau de catalogue de données pour une source en streaming

Une table du catalogue de données qui spécifie les propriétés du flux de données source, y compris le schéma de données, peut être créée manuellement pour une source de streaming. Cette table est utilisée comme source de données pour la tâche ETL en streaming.

Si vous ne connaissez pas le schéma des données dans le flux de données source, vous pouvez créer la table sans schéma. Ensuite, lorsque vous créez la tâche ETL en streaming, vous pouvez activer la fonction de détection de schéma AWS Glue. AWS Gluedétermine le schéma à partir des données en streaming.

Utilisez la AWS Glueconsole, le AWS Command Line Interface (AWS CLI) ou l'AWS GlueAPI pour créer la table. Pour plus d'informations sur la création manuelle d'une table avec la console AWS Glue, consultez the section called "Création de tables".

> **Note**
>
> Vous ne pouvez pas utiliser la AWS Lake Formation console pour créer la table ; vous devez utiliser la AWS Glue console.

Tenez également compte des informations suivantes pour les sources en streaming au format Avro ou pour les données de journal auxquelles vous pouvez appliquer des modèles Grok.

- the section called "Notes et restrictions pour les sources en streaming Avro"
- the section called "Application de modèles Grok à des sources de streaming"

Rubriques

- Source de données Kinesis
- Source de données Kafka
- Source AWS Glue de la table Schema Registry

Source de données Kinesis

Lors de la création de la table, définissez les propriétés de streaming ETL suivantes (console).

Type de source

Kinesis

Pour une source Kinesis dans le même compte :

Région

AWS Région dans laquelle réside le service Amazon Kinesis Data Streams. Le nom de la région et du flux Kinesis sont convertis en un ARN de flux.

Exemple : https://kinesis.us-east-1.amazonaws.com

Nom du flux Kinesis

Nom du flux comme décrit dans Création d'un flux dans le Guide du développeur d'Amazon Kinesis Data Streams.

Pour obtenir une source Kinesis dans un autre compte, reportez-vous à Cet exemple pour configurer les rôles et les stratégies pour autoriser un accès entre comptes. Configurez ces paramètres :

ARN du flux de diffusion

ARN du flux de données Kinesis auprès duquel le consommateur est enregistré. Pour plus d'informations, consultez les sections Amazon Resource Names (ARNs) et AWS Service Namespaces dans le Références générales AWS.

ARN du rôle assumé

Amazon Resource Name (ARN) du rôle à assumer.

Nom de la séance (facultatif)

Identifiant de la séance du rôle assumé.

Utilisez le nom de séance de rôle pour identifier de manière unique une séance lorsque le même rôle est assumé par différents principaux ou pour différentes raisons. Dans les scénarios impliquant plusieurs comptes, le nom de séance de rôle est visible et peut être enregistré par le compte propriétaire du rôle. Le nom de la séance de rôle est également utilisé dans l'ARN du principal de rôle présumé. Cela signifie que les demandes d'API inter-comptes ultérieures utilisant les informations d'identification de sécurité temporaires exposeront le nom de session du rôle au compte externe dans ses AWS CloudTrail journaux.

Pour définir les propriétés ETL de streaming pour Amazon Kinesis Data Streams (API AWS Glue ou AWS CLI)

- Pour configurer les propriétés ETL de streaming pour une source Kinesis dans le même compte, spécifiez les paramètres `streamName` et `endpointUrl` dans la structure `StorageDescriptor` de l'opération d'API `CreateTable` ou de la commande `create_table` de la CLI.

```
"StorageDescriptor": {
 "Parameters": {
  "typeOfData": "kinesis",
  "streamName": "sample-stream",
  "endpointUrl": "https://kinesis.us-east-1.amazonaws.com"
 }
 ...
}
```

Ou, spécifiez le `streamARN`.

Example

```
"StorageDescriptor": {
 "Parameters": {
  "typeOfData": "kinesis",
  "streamARN": "arn:aws:kinesis:us-east-1:123456789:stream/sample-stream"
```

```
 }
 ...
}
```

- Pour configurer les propriétés ETL de streaming pour une source Kinesis dans un autre compte, spécifiez les paramètres `streamARN`, `awsSTSRoleARN` et `awsSTSSessionName` (facultatif) dans la structure`StorageDescriptor` de l'opération d'API `CreateTable` ou de la commande `create_table` de la CLI.

```
"StorageDescriptor": {
 "Parameters": {
  "typeOfData": "kinesis",
  "streamARN": "arn:aws:kinesis:us-east-1:123456789:stream/sample-stream",
  "awsSTSRoleARN": "arn:aws:iam::123456789:role/sample-assume-role-arn",
  "awsSTSSessionName": "optional-session"
 }
 ...
}
```

Source de données Kafka

Lors de la création de la table, définissez les propriétés de streaming ETL suivantes (console).

Type de source

Kafka

Pour une source Kafka :

Nom du sujet

Nom de la rubrique tel que spécifié dans Kafka.

Connexion

Une connexion AWS Glue qui fait référence à une source Kafka, comme décrit dans the section called "Création d'une connexion pour un flux de données Kafka".

Source AWS Glue de la table Schema Registry

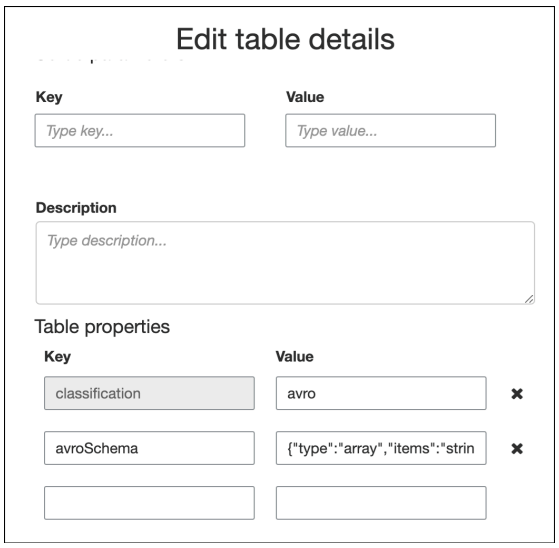
Pour utiliser Registre de schémas pour les tâches de streaming AWS Glue, suivez les instructions de Cas d'utilisation : AWS Glue Data Catalog pour créer ou mettre à jour une table de Registre de schémas.

Présentement, le streaming AWS Glue prend uniquement en charge le format Glue Schema Registry Avro avec l'inférence de schéma définie sur `false`.

Notes et restrictions pour les sources en streaming Avro

Les notes et restrictions suivantes s'appliquent aux sources en streaming au format Avro :

- Lorsque la détection du schéma est activée, le schéma Avro doit être inclus dans la charge utile. Lorsqu'elle est désactivée, la charge utile ne doit contenir que des données.
- Certains types de données Avro ne sont pas pris en charge dans les trames dynamiques. Vous ne pouvez pas spécifier ces types de données lors de la définition du schéma avec la page Define a schema (Définir un schéma) de l'assistant création de table dans la console AWS Glue. Lors de la détection de schéma, les types non pris en charge dans le schéma Avro sont convertis en types pris en charge comme suit :
  - `EnumType => StringType`
  - `FixedType => BinaryType`
  - `UnionType => StructType`
- Si vous définissez le schéma de table à l'aide de la page Define a schema (Définir un schéma) dans la console, le type de l'élément racine implicite pour le schéma est `record`. Si vous voulez un type d'élément racine autre que `record`, par exemple, `array` ou `map`, vous ne pouvez pas spécifier le schéma à l'aide de la page Define a schema (Définir un schéma). Au lieu de cela, vous devez ignorer cette page et spécifier le schéma en tant que propriété de table ou dans le script ETL.
  - Pour spécifier le schéma dans les propriétés de la table, complétez l'assistant de création de table, modifiez les détails de la table et ajoutez une nouvelle paire clé-valeur sous Table properties (Propriétés de la table). Utilisez la clé `avroSchema` et saisissez un objet de schéma JSON pour la valeur, comme illustré dans la capture d'écran suivante.



- Pour spécifier le schéma dans le script ETL, modifiez l'instruction d'affectation `datasource0` et ajoutez la clé `avroSchema` à l'argument `additional_options`, comme indiqué dans les exemples Python et Scala suivants.

  Python

  ```
  SCHEMA_STRING = '{"type":"array","items":"string"}'
  datasource0 = glueContext.create_data_frame.from_catalog(database =
   "database", table_name = "table_name", transformation_ctx = "datasource0",
   additional_options = {"startingPosition": "TRIM_HORIZON", "inferSchema":
   "false", "avroSchema": SCHEMA_STRING})
  ```

  Scala

  ```
  val SCHEMA_STRING = """{"type":"array","items":"string"}"""
  val datasource0 = glueContext.getCatalogSource(database = "database", tableName
   = "table_name", redshiftTmpDir = "", transformationContext = "datasource0",
   additionalOptions = JsonOptions(s"""{"startingPosition": "TRIM_HORIZON",
   "inferSchema": "false", "avroSchema":"$SCHEMA_STRING"}""")).getDataFrame()
  ```

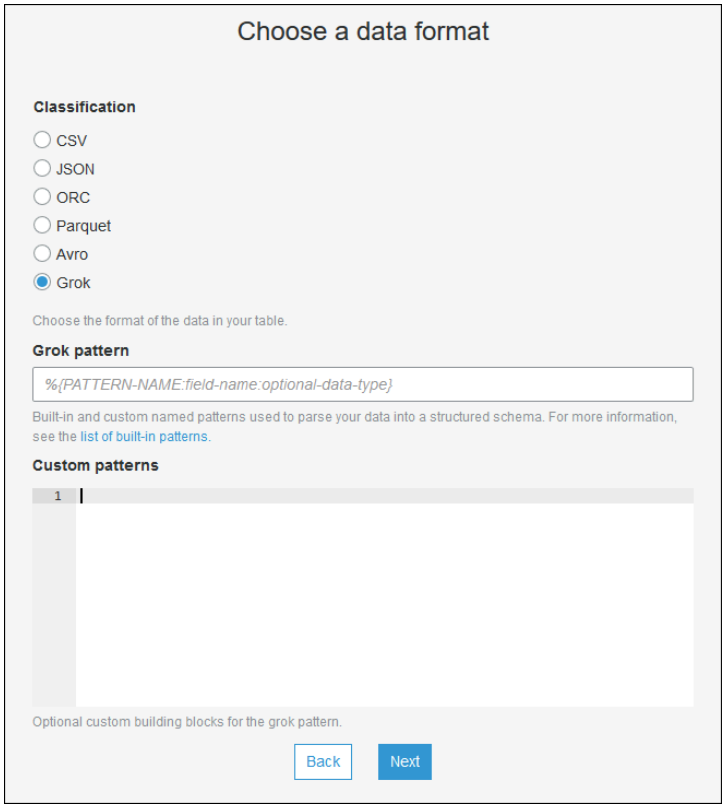
Application de modèles Grok à des sources de streaming

Vous pouvez créer une tâche ETL en streaming pour une source de données de journalisation et utiliser des modèles Grok pour convertir les journaux en données structurées. La tâche ETL traite ensuite les données en tant que source de données structurée. Vous spécifiez les modèles Grok à appliquer lorsque vous créez la table du catalogue de données pour la source en streaming.

Pour plus d'informations sur les modèles Grok et les valeurs de chaînes de modèle personnalisées, veuillez consulter Écriture de classifieurs personnalisés Grok.

Pour ajouter des modèles Grok au tableau du catalogue de données (console)

- Utilisez l'assistant de création de table et créez la table avec les paramètres spécifiés dans the section called "Création d'un tableau de catalogue de données pour une source en streaming". Spécifiez le format de données en tant que Grok, remplissez le champ Grok pattern (Modèle Grok), et éventuellement ajouter des modèles personnalisés sous Custom patterns (optional) (Modèles personnalisés – facultatif).



  Appuyez sur Entrée après chaque modèle personnalisé.

Pour ajouter des modèles de grok à la table du catalogue de données (AWS GlueAPI ou AWS CLI)

- Ajoutez le paramètre `GrokPattern` et, éventuellement, le paramètre `CustomPatterns` à l'opération d'API `CreateTable` ou à la commande `create_table` de la CLI.

```
 "Parameters": {
...
    "grokPattern": "string",
```

```
    "grokCustomPatterns": "string",
...
},
```

Formulez `grokCustomPatterns` en tant que chaîne et utilisez « \n » comme séparateur entre les motifs.

Voici un exemple de ces paramètres dans un fichier.

Example

```
"parameters": {
...
    "grokPattern": "%{USERNAME:username} %{DIGIT:digit:int}",
    "grokCustomPatterns": "digit \d",
...
}
```

Définition des propriétés de tâche pour une tâche ETL en streaming

Lorsque vous définissez une tâche ETL en streaming dans la console AWS Glue, fournissez les propriétés spécifiques aux flux suivantes. Pour obtenir la description d'autres propriétés de la tâche, consulter Définition des propriétés des tâches Spark.

Rôle IAM

Spécifiez le rôle Gestion des identités et des accès AWS (IAM) utilisé pour autoriser les ressources utilisées pour exécuter la tâche, accéder aux sources de streaming et accéder aux magasins de données cibles.

Pour accéder à Amazon Kinesis Data Streams, associez `AmazonKinesisFullAccess` AWS la politique gérée au rôle ou associez une politique IAM similaire qui permet un accès plus précis. Pour obtenir des exemples de stratégies, consultez Controlling Access to Amazon Kinesis Data Streams Resources Using IAM (Contrôle de l'accès aux ressources Amazon Kinesis Data Streams à l'aide d'IAM).

Pour plus d'informations sur les autorisations requises pour exécuter des tâches dans AWS Glue, consultez Gestion des identités et des accès pour AWS Glue.

Type

Choisissez Spark Streaming.

Version AWS Glue

La version AWS Glue détermine les versions d'Apache Spark et Python ou Scala qui sont disponibles pour la tâche. Choisissez une sélection qui spécifie la version de Python ou Scala disponible pour la tâche. AWS Glue version 2.0 avec prise en charge de Python 3 est la valeur par défaut pour les tâches ETL en streaming.

Fenêtre de maintenance

Spécifie une fenêtre dans laquelle une tâche de streaming peut être redémarrée. Consultez the section called "Fenêtres de maintenance".

Délai d'expiration de la tâche

Vous pouvez saisir une durée en minutes. La valeur par défaut est vide.

- Les tâches de streaming doivent avoir un délai d'expiration inférieur à 7 jours ou 10 080 minutes.
- Si aucun délai n'est indiqué, la tâche sera redémarrée au bout de sept jours, si vous n'avez pas défini de fenêtre de maintenance. Si vous avez défini une fenêtre de maintenance, la tâche sera redémarrée au cours de celle-ci au bout de sept jours.

Source de données

Spécifiez la table que vous avez créée dans the section called "Création d'un tableau de catalogue de données pour une source en streaming".

Cible de données

Effectuez l'une des actions suivantes :

- Choisissez Create tables in your data target (Créer des tables dans votre cible de données) et spécifiez les propriétés de cible de données suivantes.

  Banque de données

  Choisissez Amazon S3 ou JDBC.

  Format

  Choisissez n'importe quel format. Tous sont pris en charge pour le streaming.

- Choisissez Use tables in the data catalog and update your data target (Utiliser les tables du catalogue de données et mettre à jour votre cible de données) et choisissez une table pour un magasin de données JDBC.

Définition du schéma en sortie

Effectuez l'une des actions suivantes :

- Choisissez Automatically detect schema of each record (Détection automatique du schéma de chaque enregistrement) pour activer la détection de schéma. AWS Glue détermine le schéma à partir des données en streaming.
- Choisissez Specify output schema for all records (Spécifier le schéma de sortie pour tous les enregistrements) pour utiliser la transformation Apply Mapping (Appliquer le mappage) pour définir le schéma en sortie.

Script

Vous pouvez également fournir votre propre script ou modifier le script généré pour effectuer des opérations prises en charge par le moteur Apache Spark Structured Streaming. Pour plus d'informations sur les opérations disponibles, consultez la section Opérations sur le DataFrames streaming/Datasets.

Restrictions et notes sur ETL en streaming

Gardez à l'esprit les notes et restrictions suivantes :

- La décompression automatique pour les tâches ETL de streaming AWS Glue est uniquement disponible pour les types de compression pris en charge. Veuillez prendre en compte les points suivants :
  - Le format Framed Snappy fait référence au format de trame officiel pour Snappy.
  - L'algorithme Deflate est pris en charge dans la version 3.0 de Glue et non dans la version 2.0.
- Lorsque vous utilisez la détection de schéma, vous ne pouvez pas effectuer de jointures de données en streaming.
- Les tâches ETL AWS Glue en streaming ne prennent pas en charge le type de données Union pour le Registre de schéma AWS Glue avec le format Avro.
- Votre script ETL peut utiliser les transformations intégrées d'AWS Glue et les transformations natives d'Apache Spark Structured Streaming. Pour plus d'informations, consultez Opérations sur le DataFrames streaming/Datasets sur le site Web Apache Spark ou. AWS Glue PySpark transforme la référence

- Les tâches ETL AWS Glue en streaming utilisent des points de contrôle pour garder la trace des données qui ont été lues. Par conséquent, une tâche arrêtée et redémarrée reprend là où elle s'est arrêtée dans le flux. Si vous souhaitez retraiter les données, vous pouvez supprimer le dossier du point de contrôle référencé dans le script.
- Les signets de tâche ne sont pas pris en charge.
- Pour utiliser la fonctionnalité de diffusion améliorée de Kinesis Data Streams dans tâche, consultez the section called "Utilisation de la diffusion améliorée dans les tâches de streaming Kinesis".
- Si vous utilisez une table de catalogue de données créée à partir d'AWS Glue Schema Registry, lorsqu'une nouvelle version de schéma devient disponible, pour refléter le nouveau schéma, vous devez effectuer les opérations suivantes :
  1. Arrêtez les tâches associées à la table.
  2. Mettez à jour le schéma de la table Catalogue de données.
  3. Redémarrez les tâches associées à la table.

Record correspondant à AWS Lake Formation FindMatches

> **Note**
>
> Le rapprochement des records n'est actuellement pas disponible dans les régions suivantes de la AWS Glue console : Moyen-Orient (Émirats arabes unis), Europe (Espagne), Asie-Pacifique (Jakarta) et Europe (Zurich).

AWS Lake Formation fournit des fonctionnalités d'apprentissage automatique pour créer des transformations personnalisées afin de nettoyer vos données. Il existe actuellement une transformation disponible nommée FindMatches. La FindMatches transformation vous permet d'identifier les enregistrements dupliqués ou correspondants dans votre ensemble de données, même lorsque les enregistrements n'ont pas d'identifiant unique commun et qu'aucun champ ne correspond exactement. Cela ne nécessitera pas d'écrire de code ou de connaître le fonctionnement de l'apprentissage automatique. FindMatches peut être utile pour résoudre de nombreux problèmes, tels que :

- Correspondance de clients : liaison des enregistrements clients entre différentes bases de données client, même lorsque de nombreux champs client ne correspondent pas exactement entre les bases de données (par exemple, orthographe de nom différentes, adresses différentes, données manquantes ou inexactes, etc.).

- Correspondance de produits : correspondance des produits de votre catalogue par rapport à d'autres sources de produits, telle qu'un catalogue de produits par rapport au catalogue d'un concurrent, où les entrées sont structurées différemment.
- Amélioration de la détection des fraudes : identification de comptes client en double, afin de déterminer quand un nouveau compte est (ou peut être) une correspondance pour un utilisateur frauduleux précédemment connu.
- Autres problèmes de correspondance : correspondance d'adresses, de films, de listes de pièces, etc. En général, si un être humain pouvait examiner les lignes de votre base de données et déterminer si elles correspondent, il y a de fortes chances que la FindMatches transformation puisse vous aider.

Vous pouvez créer ces transformations lorsque vous créez une tâche. La transformation que vous créez est basée sur un schéma de magasin de données source et des exemples de données du jeu de données source que vous labélisez (nous appelons ce processus l'« enseignement » d'une transformation). Les enregistrements que vous labélisez doivent être présents dans le jeu de données source. Dans ce processus, nous générons un fichier que vous labélisez et chargez ensuite pour que la transformation apprenne d'une manière ou d'une autre. Après avoir enseigné votre transformation, vous pouvez l'appeler depuis votre AWS Glue tâche basée sur Spark (PySpark ou Scala Spark) et l'utiliser dans d'autres scripts dotés d'un magasin de données source compatible.

Une fois la transformation créée, elle est stockée dans AWS Glue. Sur la console AWS Glue, vous pouvez gérer les transformations que vous créez. Dans le volet de navigation sous Intégration des données et ETL, Outils de classification des données > Correspondance des enregistrements, vous pouvez modifier et continuer à entraîner votre transformation de machine learning. Pour de plus amples informations sur la gestion des transformations sur la console, veuillez consulter Utilisation de transformations du machine learning.

> **Note**
>
> AWS Glueles FindMatches tâches de la version 2.0 utilisent le compartiment Amazon S3 `aws-glue-temp-<accountID>-<region>` pour stocker des fichiers temporaires pendant que la transformation traite des données. Vous pouvez supprimer ces données une fois l'exécution terminée, soit manuellement, soit en définissant une règle de cycle de vie Amazon S3.

Types de transformations Machine Learning

Vous pouvez créer des transformations de machine learning pour nettoyer vos données. Vous pouvez appeler ces transformations à partir de votre script ETL. Vos données passent d'une transformation à l'autre dans une structure de données appelée a DynamicFrame, qui est une extension d'un SQL Apache Spark`DataFrame`. Le `DynamicFrame` contient vos données, et vous référencez son schéma pour traiter vos données.

Les types de transformations de machine learning suivants sont disponibles :

Recherche de correspondances

Recherche des enregistrements en double dans les données source. Vous enseignez cette transformation Machine Learning par l'étiquetage des ensembles de données afin d'indiquer les lignes qui correspondent. La transformation de machine learning apprend quelles lignes doivent être des correspondances à mesure que vous l'entraînez à l'aide d'exemples de données étiquetées. En fonction de la manière dont vous configurez la transformation, la sortie peut ressembler à l'une des suivantes :

- Une copie de la table d'entrée et une colonne `match_id` renseignée avec des valeurs qui indiquent des ensembles d'enregistrements correspondants. La colonne `match_id` est un identifiant arbitraire. Tous les enregistrements qui ont le même `match_id` ont été identifiés comme correspondant les uns aux autres. Les enregistrements avec des `match_id` différents ne correspondent pas.
- Une copie de la table d'entrée dans laquelle les lignes en double ont été retirées. Si plusieurs doublons sont trouvés, l'enregistrement avec la clé primaire la plus faible est conservé.

Rechercher des correspondances progressives

La transformation FindMatches peut également être configurée pour rechercher des correspondances entre les trames existantes et progressives et renvoyer en sortie une colonne contenant un ID unique par groupe de correspondances.

Pour plus d'informations, voir : Recherche de correspondances progressives

Utilisation de la FindMatches transformation

Vous pouvez utiliser la transformation `FindMatches` pour rechercher les enregistrements en double dans les données source. Un fichier d'étiquetage des données est généré ou fourni pour vous aider à enseigner à la transformation.

> **Note**
>
> Actuellement, les transformations `FindMatches` utilisant une clé de chiffrement personnalisée ne sont pas prises en charge dans les régions suivantes :
>
> - Asie-Pacifique (Osaka) - `ap-northeast-3`

Pour commencer la FindMatches transformation, vous pouvez suivre les étapes ci-dessous. Pour un exemple plus avancé et détaillé, consultez le blog AWS Big Data : Harmonize data using AWS Glue and AWS Lake Formation FindMatches ML to build a customer 360 view.

Commencer à utiliser la transformation Recherche de correspondances

Suivez les étapes ci-après pour commencer à utiliser la transformation `FindMatches` :

1. Créez une table dans le AWS Glue Data Catalog pour les données source à nettoyer. Pour plus d'informations sur la façon de créer un crawler, veuillez consulter Travailler avec des crawlers sur la console AWS Glue.

   Si vos données source sont dans un fichier texte, par exemple un fichier CSV (valeurs séparées par une virgule), tenez compte des éléments suivants :

   - Placez votre fichier CSV des enregistrements d'entrée et le fichier d'étiquetage des données dans des dossiers distincts. Dans le cas contraire, le crawler AWS Glue pourrait les considérer comme plusieurs parties de la même table et créer des tables dans le catalogue de données de manière incorrecte.
   - Sauf si votre fichier CSV inclut des caractères ASCII uniquement, assurez-vous que l'encodage UTF-8 sans marque d'ordre d'octet (BOM) est utilisé pour les fichiers CSV. Microsoft Excel ajoute souvent une marque d'ordre d'octet (BOM) au début de fichiers CSV UTF-8. Pour la retirer, ouvrez le fichier CSV dans un éditeur de texte, puis réenregistrez le fichier au format UTF-8 sans marque d'ordre d'octet (BOM).
2. Sur la console AWS Glue, créez une tâche, puis choisissez le type de transformation Recherche de correspondances.

   > **Important**
   >
   > La table de sources de données que vous choisissez pour la tâche ne peut pas avoir plus de 100 colonnes.

3. Demandez à AWS Glue de générer un fichier d'étiquetage des données en choisissant Generate labeling file (Générer un fichier d'étiquetage). AWS Glue effectue une première passe en regroupant les enregistrements similaires pour chaque `labeling_set_id` afin que vous puissiez vérifier ces regroupements. Vous labélisez les correspondances dans la colonne `label`.
   - Si vous disposez déjà d'un fichier d'étiquetage, c'est-à-dire un exemple d'enregistrements qui indiquent des lignes qui correspondent l'une à l'autre, téléchargez le fichier dans Amazon Simple Storage Service (Amazon S3). Pour en savoir plus sur le format du fichier d'étiquetage, consultez Format du fichier d'étiquetage. Passez à l'étape 4.
4. Téléchargez le fichier d'étiquetage et labélisez le fichier comme décrit dans la section Étiquetage.
5. Chargez le fichier étiqueté corrigé. AWS Glue exécute des tâches pour enseigner à la transformation comment trouver des correspondances.

   Sur la page de liste Machine learning transforms(Transformations Machine Learning), choisissez l'onglet History (Historique). Cette page indique quand AWS Glue effectue les tâches suivantes :
   - Importer des étiquettes
   - Exporter des étiquettes
   - Générer des étiquettes
   - Estimation de la qualité
6. Pour créer une meilleure transformation, vous pouvez télécharger, étiqueter, puis charger de manière itérative le fichier étiqueté. Lors des premières exécutions, beaucoup d'enregistrements peuvent ne pas correspondre. Cependant, AWS Glue apprend dès lors que vous continuez à lui enseigner en vérifiant le fichier d'étiquetage.
7. Évaluez et ajustez votre transformation en évaluant les performances et les résultats des correspondances trouvées. Pour de plus amples informations, veuillez consulter Réglage des transformations Machine Learning dans AWS Glue.

Étiquetage

Lorsque `FindMatches` génère un fichier d'étiquetage, les enregistrements sont sélectionnés à partir de votre table source. En fonction de l'entraînement précédent, `FindMatches` identifie les enregistrements les plus significatifs pouvant servir de base d'apprentissage.

L'acte d'étiquetage consiste à modifier un fichier d'étiquetage (nous vous suggérons d'utiliser une feuille de calcul Microsoft Excel, par exemple) et à ajouter des identifiants, ou étiquettes, dans la colonne `label` qui identifie les enregistrements correspondants ou non correspondants. Il est important d'avoir une définition claire et cohérente d'une correspondance dans vos données source.

`FindMatches` apprend à partir des enregistrements que vous désignez comme correspondants (ou pas) et utilise vos décisions pour apprendre à rechercher des enregistrements en double.

Lorsqu'un fichier d'étiquetage est généré par `FindMatches`, environ 100 enregistrements sont générés. Ces 100 enregistrements sont généralement divisés en 10 jeux d'étiquetage, chaque jeu étant identifié par un `labeling_set_id` unique généré par `FindMatches`. Chaque jeu d'étiquetage doit être considéré comme une tâche d'étiquetage distincte indépendante des autres jeux d'étiquetage. Votre tâche consiste à identifier les enregistrements correspondants et non correspondants dans chaque jeu d'étiquetage.

Conseils pour modifier les fichiers d'étiquetage dans une feuille de calcul

Lorsque vous modifiez le fichier d'étiquetage dans une application de feuille de calcul, tenez compte des éléments suivants :

- Le fichier peut ne pas s'ouvrir avec les champs de colonne entièrement développés. Vous devrez peut-être développer les colonnes `label` et `labeling_set_id` pour voir le contenu de ces cellules.
- Si la colonne de clé primaire comporte un nombre, comme un type de données `long`, la feuille de calcul peut l'interpréter comme un nombre et modifier la valeur. La valeur de cette clé doit être traitée en tant que texte. Pour corriger ce problème, définissez le format Texte pour toutes les cellules de la colonne de clé primaire.

Format du fichier d'étiquetage

Le fichier d'étiquetage généré par AWS Glue pour l'apprentissage de votre transformation `FindMatches` utilise le format suivant. Si vous générez votre propre fichier pour AWS Glue, il doit également suivre ce format :

- Il s'agit d'un fichier CSV (valeurs séparées par des virgules).
- Il doit être encodé en `UTF-8`. Si vous modifiez le fichier à l'aide de Microsoft Windows, il peut être encodé avec `cp1252`.
- Il doit se trouver dans un emplacement Amazon S3 pour sa transmission à AWS Glue.
- Utilisez un nombre modéré de lignes pour chaque tâche d'étiquetage. Il est recommandé d'utiliser 10 à 20 lignes par tâche, même si 2 à 30 lignes par tâche sont acceptées. Les tâches de plus de 50 lignes ne sont pas recommandées et peuvent entraîner de mauvais résultats ou une défaillance du système.

- Si vous disposez de données composées de paires d'enregistrements déjà marqués comme « correspondants » ou « non correspondants », cela ne pose pas de problème. Ces paires étiquetées peuvent être représentées sous forme de jeux d'étiquetage de taille 2. Dans ce cas, labélisez les deux enregistrements avec, par exemple, la lettre « A » s'ils correspondent, ou labélisez l'une avec la lettre « A » et l'autre avec la lettre « B » s'ils ne correspondent pas.

  > **Note**
  >
  > Étant donné qu'il comporte des colonnes supplémentaires, le fichier d'étiquetage a un schéma différent d'un fichier qui contient vos données source. Placez le fichier d'étiquetage dans un dossier différent de celui de tout fichier CSV d'entrée de transformation, de sorte que le crawler AWS Glue n'en tienne pas compte lorsqu'il crée des tables dans le catalogue de données. Sinon, les tables créées par le crawler AWS Glue pourraient ne pas représenter correctement vos données.

- Les deux premières colonnes (`labeling_set_id`, `label`) sont requises par AWS Glue. Les colonnes restantes doivent correspondre au schéma des données à traiter.
- Pour chaque `labeling_set_id`, vous identifiez tous les enregistrements correspondants à l'aide de la même étiquette. Une étiquette est une chaîne unique placée dans la colonne `label`. Nous vous recommandons d'utiliser les étiquettes contenant des caractères simples, par exemple A, B, C, et ainsi de suite. Les étiquettes sont sensibles à la casse et sont saisies dans la colonne `label`.
- Les lignes qui contiennent le même `labeling_set_id` et la même étiquette sont considérées comme formant une correspondance.
- Les lignes qui contiennent le même `labeling_set_id` et une étiquette différente sont considérées comme ne formant pas une correspondance.
- Les lignes qui contiennent un `labeling_set_id` différent sont considérées comme ne fournissant aucune information pour ou contre la correspondance.

  Voici un exemple d'étiquetage de données :

labeling_set_id	étiquette	first_name	last_name	Anniversaire
ABC123	A	John	Doe	04/01/1980
ABC123	B	Jane	Smith	04/03/1980
ABC123	A	Johnny	Doe	04/01/1980

labeling_set_id	étiquette	first_name	last_name	Anniversaire
ABC123	A	Jon	Doe	04/01/1980
DEF345	A	Richard	Jones	12/11/1992
DEF345	A	Rich	Jones	11/12/1992
DEF345	B	Sarah	Jones	12/11/1992
DEF345	C	Richie	Jones Jr.	05/06/2017
DEF345	B	Sarah	Jones-Walker	12/11/1992
GHI678	A	Robert	Miller	1/3/1999
GHI678	A	Bob	Miller	1/3/1999
XYZABC	A	William	Robinson	2/5/2001
XYZABC	B	Andrew	Robinson	2/5/1971

- Dans l'exemple ci-dessus, nous identifions John/Johnny/Jon Doe as being a match and we teach the system that these records do not match Jane Smith. Separately, we teach the system that Richard and Rich Jones are the same person, but that these records are not a match to Sarah Jones/Jones -Walker et Richie Jones Jr.
- Comme vous pouvez le voir, la portée des étiquettes est limitée à l'élément `labeling_set_id`. Ainsi, les étiquettes ne dépassent pas les limites de `labeling_set_id`. Par exemple, une étiquette « A » dans `labeling_set_id` 1 n'a aucune relation avec l'étiquette « A » dans `labeling_set_id` 2.
- Si un enregistrement n'a aucune correspondance dans un jeu d'étiquettes, affectez-lui une étiquette unique. Par exemple, Jane Smith ne correspond à aucun enregistrement du jeu d'étiquettes ABC123, c'est donc le seul enregistrement de ce jeu d'étiquettes portant l'étiquette B.
- Le jeu d'étiquettes « GHI678 » indique qu'un ensemble d'étiquettes peut être composé uniquement de deux enregistrements auxquels est attribuée la même étiquette pour montrer qu'ils correspondent. De même, « XYZABC » montre deux enregistrements portant des étiquettes différentes pour montrer qu'ils ne correspondent pas.
- Notez que parfois un jeu d'étiquetage peut ne pas contenir de correspondances (c'est-à-dire que vous donnez à chaque enregistrement du jeu d'étiquetage une étiquette différente) ou qu'un jeu

d'étiquetage peut être globalement « identique » (vous leur donnez le même label). Ceci est correct tant que vos jeux d'étiquetage contiennent collectivement des exemples d'enregistrements qui sont et ne sont pas « identiques » selon vos critères.

> **Important**
>
> Confirmez que le rôle IAM que vous transmettez à AWS Glue a accès au compartiment Amazon S3 qui contient le fichier d'étiquetage. Par convention, les stratégies AWS Glue accordent une autorisation sur les enregistrements ou les dossiers Amazon S3 dont les noms sont préfixés avec aws-glue-. Si vos fichiers d'étiquetage se trouvent dans un emplacement différent, ajoutez l'autorisation sur cet emplacement dans le rôle IAM.

Réglage des transformations Machine Learning dans AWS Glue

Vous pouvez régler vos transformations Machine Learning dans AWS Glue pour améliorer les résultats de vos tâches de nettoyage de données afin d'atteindre vos objectifs. Pour améliorer votre transformation, vous pouvez l'entraîner en générant un ensemble d'étiquetage, en ajoutant des étiquettes, puis en répétant ces étapes plusieurs fois jusqu'à obtention des résultats souhaités. Vous pouvez également procéder à un réglage en modifiant certains paramètres de machine learning.

Pour de plus amples informations sur les transformations Machine Learning, veuillez consulter Record correspondant à AWS Lake Formation FindMatches.

Rubriques

- Mesures de Machine Learning
- Choisir entre la précision et le rappel
- Choisir entre l'exactitude et le coût
- Estimation de la qualité des correspondances à l'aide des notes de confiance des correspondances
- Entraîner la transformation à trouver des correspondances

Mesures de Machine Learning

Pour comprendre les mesures qui sont utilisés pour régler votre transformation Machine Learning, vous devez vous familiariser avec la terminologie suivante :

Vrai positif (TP, True positive)

Correspondance dans les données correctement trouvée par la transformation, parfois appelée occurrence.

Vrai négatif (TN, True Negative)

Non correspondance dans les données que la transformation a correctement rejetée.

Faux positif (FP, False positive)

Non correspondance dans les données que la transformation a classifiée de manière incorrecte en tant que correspondance, parfois appelée fausse alarme.

Faux négatif (FN, False negative)

Correspondance dans les données que la transformation n'a pas trouvée, parfois appelée échec.

Pour de plus amples informations sur la terminologie utilisée dans le Machine Learning, consultez Confusion matrix dans Wikipedia.

Pour régler vos transformations Machine Learning, vous pouvez modifier la valeur des mesures suivantes dans les propriétés avancées de la transformation.

- Précision (Précision) évalue dans quelle mesure la transformation trouve de vrais positifs parmi le nombre total d'enregistrements qu'elle identifie comme positifs (vrais positifs et faux positifs). Pour en savoir plus, consultez Précision et rappel dans Wikipédia.
- Recall (Rappel) mesure comment la transformation trouve des vrais positifs à partir du nombre total d'enregistrements dans les données source. Pour en savoir plus, consultez Précision et rappel dans Wikipédia.
- Accuracy (Exactitude) mesure comment la transformation trouve des vrais positifs et des vrais négatifs. L'augmentation de la précision nécessite plus de ressources machine et des coûts supérieurs. Mais elle entraîne également une augmentation du rappel. Pour en savoir plus, consulter Précision et rappel dans Wikipédia.
- Cost (Coût) mesure le nombre de ressources de calcul (et par conséquent le coût) consommées pour l'exécution de la transformation.

Choisir entre la précision et le rappel

Chaque transformation `FindMatches` contient un paramètre `precision-recall`. Vous utilisez ce paramètre pour spécifier l'un des éléments suivants :

- Si vous êtes plus préoccupé par le fait que la transformation indique de manière erronée que deux enregistrements sont des correspondances alors que ce n'est pas le cas, vous devez alors mettre l'accent sur la précision.
- Si vous êtes plus préoccupé par le fait que la transformation ne parvienne pas à détecter les enregistrements qui sont de réelles correspondances, vous devez mettre l'accent sur le rappel.

Vous pouvez effectuer un tel compromis sur la console AWS Glue ou à l'aide d'opérations d'API de Machine Learning AWS Glue.

Dans quel cas favoriser la précision

Privilégiez la précision si vous êtes plus préoccupé par le risque que `FindMatches` se traduise par une paire d'enregistrements correspondants alors qu'il n'y a aucune correspondance réelle. Pour favoriser la précision, choisissez une valeur de compromis précision-rappel plus élevée. Avec une valeur plus élevée, la transformation `FindMatches` a besoin de plus de preuves pour décider qu'une paire d'enregistrements doivent correspondre. La transformation est réglée sur une tendance à indiquer que les enregistrements ne correspondent pas.

Par exemple, supposons que vous utilisez `FindMatches` pour détecter des articles en double dans un catalogue vidéo, et que vous fournissez une valeur précision-rappel plus élevée à la transformation. Si votre transformation détecte de manière incorrecte détecte qu' Star Wars : Un nouvel espoir est identique à Star Wars : L'Empire contre-attaque, un client qui souhaite Un nouvel espoir peut voir s'afficher L'Empire contre-attaque. Cela constituerait une mauvaise expérience client.

Toutefois, si la transformation ne parvient pas à détecter que Star Wars : Un nouvel espoir et Star Wars : Episode IV - Un nouvel espoir sont un seul et même article, le client peut être un peu perdu au début mais il peut au final les reconnaître comme étant identiques. Il s'agirait là d'une erreur, mais pas aussi grave que dans le scénario précédent.

Dans quel cas favoriser le rappel

Privilégiez le rappel si vous êtes plus préoccupé par le risque que les résultats de la transformation `FindMatches` ne parviennent pas à détecter une paire d'enregistrements qui correspondent réellement. Pour favoriser le rappel, choisissez une valeur de compris précision-rappel plus faible. Avec une valeur plus faible, la transformation `FindMatches` a besoin de moins de preuves pour décider qu'une paire d'enregistrements doivent correspondre. La transformation est réglée sur une tendance à indiquer que les enregistrements correspondent.

Cela peut, par exemple, être une priorité pour une organisation de sécurité. Supposons que vous établissez des correspondances entre des clients et une liste de fraudeurs connus, et il est important de déterminer si un client est un fraudeur. Vous utilisez `FindMatches` pour établir des correspondances entre la liste de fraudeurs et la liste de clients. Chaque fois que `FindMatches` détecte une correspondance entre les deux listes, un auditeur humain est affecté pour vérifier que la personne est effectivement un fraudeur. Il se peut que votre organisation préfère le rappel à la précision. En d'autres termes, vous préférez plutôt avoir des auditeurs qui vérifient et rejettent manuellement certains cas lorsque le client n'est pas un fraudeur que de ne pas parvenir à identifier qu'un client est, effectivement, sur la liste de fraudeurs.

Comment favoriser à la fois la précision et le rappel

Le meilleur moyen d'améliorer à la fois la précision et le rappel est d'étiqueter davantage de données. Au fur et à mesure que vous étiquetez plus de données, l'exactitude globale de la transformation `FindMatches` s'améliore, ce qui améliore à la fois la précision et le rappel. Néanmoins, même avec la transformation la plus précise possible, il demeure toujours une zone grise où vous devez essayer de favoriser la précision ou le rappel, ou choisir une valeur intermédiaire.

Choisir entre l'exactitude et le coût

Chaque transformation `FindMatches` contient un paramètre `accuracy-cost`. Vous pouvez utiliser ce paramètre pour spécifier l'un des éléments suivants :

- Si vous êtes plus préoccupé par le fait que la transformation indique précisément que deux enregistrements correspondent, vous devez mettre l'accent sur la précision.
- Si vous êtes plus préoccupé par le coût ou la vitesse d'exécution de la transformation, vous devez mettre l'accent sur la réduction des coûts.

Vous pouvez effectuer un tel compromis sur la console AWS Glue ou à l'aide d'opérations d'API de Machine Learning AWS Glue.

Dans quel cas favoriser l'exactitude

Privilégiez la précision si vous êtes plus préoccupé par le risque que les résultats de `find matches` ne contiennent des correspondances. Pour favoriser l'exactitude, choisissez une valeur de compromis exactitude-coût plus élevée. Avec une valeur plus élevée, la transformation `FindMatches` a besoin de plus de temps pour effectuer une recherche plus approfondie afin de trouver des enregistrements qui correspondent correctement. Notez que ce paramètre ne réduit pas le risque d'appeler de manière erronée une correspondance une paire d'enregistrements non

correspondants. La transformation est réglée sur une tendance à consacrer plus de temps à la recherche de correspondances.

Dans quel cas favoriser le coût

Privilégiez le coût si vous être plus préoccupé par le coût d'exécution de la transformation `find matches` et moins préoccupé par le nombre de correspondances trouvées. Pour favoriser le coût, choisissez une valeur de compromis exactitude-coût plus faible. Avec une valeur plus faible, la transformation `FindMatches` a besoin de moins de ressources pour son exécution. La transformation est réglée sur une tendance à rechercher moins de correspondances. Si le fait de favoriser la réduction des coûts produit des résultats acceptables, utilisez ce paramètre.

Comment favoriser à la fois l'exactitude et la réduction des coûts

Un temps machine supérieur est nécessaire pour examiner davantage de paires d'enregistrements afin de déterminer s'il peut s'agir de correspondances. Si vous souhaitez réduire les coûts sans réduire la qualité, voici quelques conseils :

- Éliminez dans votre source de données les enregistrements pour lesquels vous ne cherchez pas de correspondances.
- Éliminez dans votre source de données les colonnes qui sont de manière certaine inutiles à la prise d'une décision correspondance/non correspondance. Une bonne manière de déterminer cela consiste à éliminer les colonnes qui n'affectent en rien votre décision quant il s'agit de déterminer si un ensemble d'enregistrements sont « identiques ».

Estimation de la qualité des correspondances à l'aide des notes de confiance des correspondances

Les scores de confiance des correspondances fournissent une estimation de la qualité des correspondances trouvées FindMatches afin de faire la distinction entre les enregistrements correspondants dans lesquels le modèle d'apprentissage automatique est très sûr, incertain ou peu probable. Une note de confiance des correspondances sera comprise entre 0 et 1, alors qu'une note plus élevée signifie une similitude plus élevée. L'examen des notes de confiance des correspondances vous permet de distinguer les clusters de correspondances pour lesquels le système est très confiant (que vous pouvez décider de fusionner), les clusters dont le système est incertain (que vous pouvez décider de faire vérifier par un humain) et les clusters que le système juge improbables (que vous pouvez décider de rejeter).

Vous voudrez peut-être ajuster vos données d'entraînement dans des situations où vous constatez une note de confiance des correspondances élevées, mais déterminez qu'il n'y a pas

de correspondances, ou lorsque vous voyez une note faible, mais déterminez qu'il y a en fait des correspondances.

Les scores de confiance sont particulièrement utiles lorsqu'il existe des ensembles de données industriels de grande taille, pour lesquels il est impossible de revoir chaque FindMatches décision.

Les notes de confiance des correspondances sont disponibles dans la version 2.0 ou ultérieure de AWS Glue.

Génération des notes de confiance des correspondances

Vous pouvez générer des notes de confiance des correspondances en définissant la valeur booléenne de `computeMatchConfidenceScores` sur true (VRAI) lorsque vous appelez l'API `FindMatches` ou `FindIncrementalMatches`.

AWS Glue ajoute un nouveau `column match_confidence_score` à la sortie.

Exemples de notation des correspondances

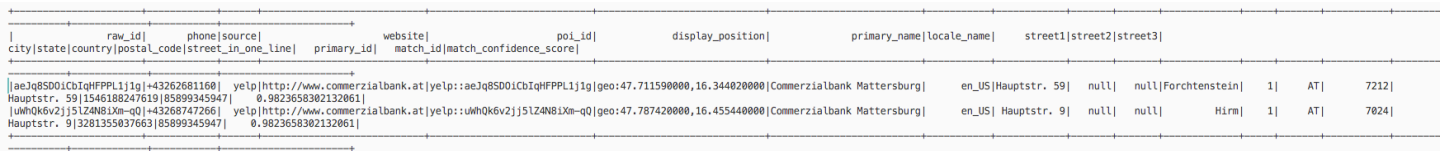
Prenons l'exemple des registres de correspondance suivant :

Note >= 0,9

Résumé des registres correspondants :

```
  primary_id  |   match_id  | match_confidence_score

3281355037663    85899345947   0.9823658302132061
1546188247619    85899345947   0.9823658302132061
```

Détails :



```
+--------------------+----------+-----+--------------------+--------------------+--------------------+--------------------+------------+-----------+------+-----+-------+-----------+
|              raw_id|     phone|source|             website|              poi_id|    display_position|        primary_name|locale_name|    street1|street2|street3|
city|state|country|postal_code|street_in_one_line|  primary_id|  match_id|match_confidence_score|
+--------------------+----------+-----+--------------------+--------------------+--------------------+--------------------+------------+-----------+------+-----+-------+-----------+
|aeJq8SDOiCbIqHFPPL1j1g|+43262681160|  yelp|http://www.commerzialbank.at|yelp::aeJq8SDOiCbIqHFPPL1j1g|geo:47.711590000,16.344020000|Commerzialbank Mattersburg|     en_US|Hauptstr. 59|   null|   null|Forchtenstein|    1|     AT|      7212|
Hauptstr. 59|1546188247619|85899345947|    0.9823658302132061|
|uWhQk6v2jj5lZ4N8iXm-qQ|+432687472666|  yelp|http://www.commerzialbank.at|yelp::uWhQk6v2jj5lZ4N8iXm-qQ|geo:47.787420000,16.455440000|Commerzialbank Mattersburg|     en_US| Hauptstr. 9|   null|   null|         Hirm|    1|     AT|      7024|
Hauptstr. 9|3281355037663|85899345947|    0.9823658302132061|
+--------------------+----------+-----+--------------------+--------------------+--------------------+--------------------+------------+-----------+------+-----+-------+-----------+
```

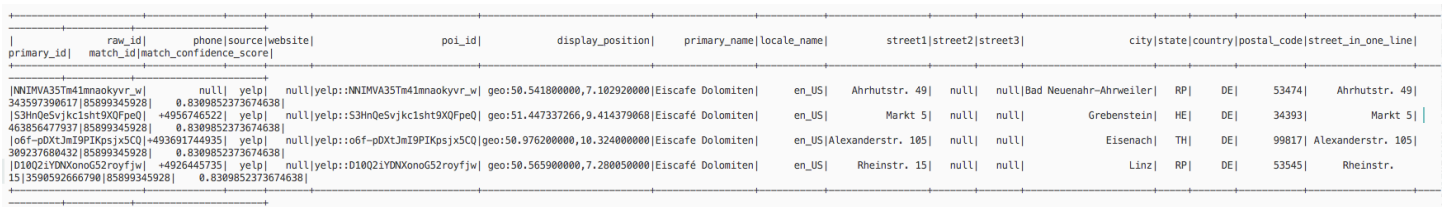
Dans cet exemple, nous pouvons voir que deux registres sont très similaires et partagent `display_position`, `primary_name`, et `street name`.

Note >= 0,8 et note < 0,9

Résumé des registres correspondants :

```
  primary_id  |   match_id  | match_confidence_score

309237680432     85899345928   0.8309852373674638
3590592666790    85899345928   0.8309852373674638
343597390617     85899345928   0.8309852373674638
249108124906     85899345928   0.8309852373674638
463856477937     85899345928   0.8309852373674638
```

Détails :



```
|              raw_id|       phone|source|website|                    poi_id|           display_position|     primary_name|locale_name|         street1|street2|street3|               city|state|country|postal_code|street_in_one_line|
primary_id|   match_id|match_confidence_score|
|NNIMVA35Tm41mnaokyvr_w|        null|  yelp|   null|yelp::NNIMVA35Tm41mnaokyvr_w| geo:50.541800000,7.102920000|Eiscafe Dolomiten|      en_US|    Ahrhutstr. 49|   null|   null|Bad Neuenahr~Ahrweiler|   RP|     DE|      53474|     Ahrhutstr. 49|
343597390617|85899345928|    0.8309852373674638|
|S3HnQeSvjkc1sht9XQFpeQ|  +4956746522|  yelp|   null|yelp::S3HnQeSvjkc1sht9XQFpeQ| geo:51.447337266,9.414379068|Eiscafé Dolomiten|      en_US|          Markt 5|   null|   null|           Grebenstein|   HE|     DE|      34393|           Markt 5|
463856477937|85899345928|    0.8309852373674638|
|o6f-pDXtJmI9PIKpsjx5CQ|+493691744935|  yelp|   null|yelp::o6f-pDXtJmI9PIKpsjx5CQ|geo:50.976200000,10.324000000|Eiscafe Dolomiten|      en_US|Alexanderstr. 105|   null|   null|              Eisenach|   TH|     DE|      99817| Alexanderstr. 105|
309237680432|85899345928|    0.8309852373674638|
|D10Q2iYDNXonoG52royfjw|  +4926445735|  yelp|   null|yelp::D10Q2iYDNXonoG52royfjw| geo:50.565900000,7.280050000|Eiscafé Dolomiten|      en_US|     Rheinstr. 15|   null|   null|                Linz|   RP|     DE|      53545|      Rheinstr.
15|3590592666790|85899345928|    0.8309852373674638|
```

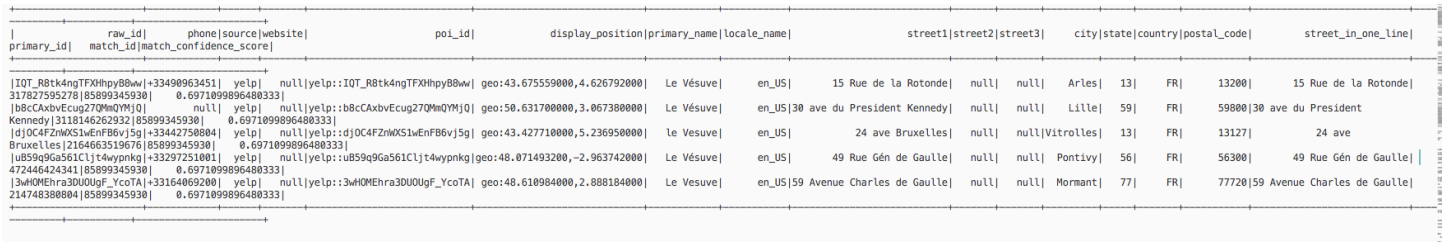
Dans cet exemple, nous pouvons voir que ces registres partagent le même`primary_name`, et `country`.

Note >= 0,6 et note < 0,7

Résumé des registres correspondants :

```
  primary_id  |   match_id  | match_confidence_score

2164663519676    85899345930   0.6971099896480333
 317827595278    85899345930   0.6971099896480333
 472446424341    85899345930   0.6971099896480333
3118146262932    85899345930   0.6971099896480333
 214748380804    85899345930   0.6971099896480333
```

Détails :



```
|              raw_id|       phone|source|website|                    poi_id|           display_position|primary_name|locale_name|                street1|street2|street3|     city|state|country|postal_code|        street_in_one_line|
primary_id|   match_id|match_confidence_score|
|IQT_R8tk4ngTFXHhpyB8ww|+33490963451|  yelp|   null|yelp::IQT_R8tk4ngTFXHhpyB8ww| geo:43.675559000,4.626792000|   Le Vésuve|      en_US|       15 Rue de la Rotonde|   null|   null|    Arles|   13|     FR|      13200|       15 Rue de la Rotonde|
317827595278|85899345930|    0.6971099896480333|
|b8cCAxbvEcug27QMmQYMjQ|        null|  yelp|   null|yelp::b8cCAxbvEcug27QMmQYMjQ| geo:50.631700000,3.067380000|   Le Vésuve|      en_US|30 ave du President Kennedy|   null|   null|    Lille|   59|     FR|      59800|30 ave du President
Kennedy|3118146262932|85899345930|    0.6971099896480333|
|djOC4FZnWXS1wEnFB6vj5g|+33442750804|  yelp|   null|yelp::djOC4FZnWXS1wEnFB6vj5g| geo:43.427710000,5.236950000|   le Vesuve|      en_US|          24 ave Bruxelles|   null|   null|Vitrolles|   13|     FR|      13127|          24 ave
Bruxelles|2164663519676|85899345930|    0.6971099896480333|
|uB59q9Ga561Cljt4wypnkg|+33297251001|  yelp|   null|yelp::uB59q9Ga561Cljt4wypnkg|geo:48.071493200,-2.963742000|   Le Vésuve|      en_US|      49 Rue Gén de Gaulle|   null|   null|  Pontivy|   56|     FR|      56300|      49 Rue Gén de Gaulle|
472446424341|85899345930|    0.6971099896480333|
|3wHOMEhra3DUOUgF_YcoTA|+33164069200|  yelp|   null|yelp::3wHOMEhra3DUOUgF_YcoTA| geo:48.610984000,2.888184000|   Le Vesuve|      en_US|59 Avenue Charles de Gaulle|   null|   null|  Mormant|   77|     FR|      77720|59 Avenue Charles de Gaulle|
214748380804|85899345930|    0.6971099896480333|
```

Dans cet exemple, nous pouvons voir que ces registres ne partagent que le même `primary_name`.

Pour en savoir plus, consultez :

- Étape 5 : Ajouter et exécuter une tâche avec votre transformation Machine Learning
- PySpark: FindMatches classe
- PySpark: FindIncrementalMatches classe
- Scala : FindMatches classe
- Scala : FindIncrementalMatches classe

Entraîner la transformation à trouver des correspondances

Il est nécessaire d'entraîner chaque transformation `FindMatches` à déterminer ce qui est une correspondance et ce qui ne l'est pas. Vous entraînez votre transformation en ajoutant des étiquettes dans un fichier et en chargeant vos choix dans AWS Glue.

Vous pouvez effectuer un tel compromis sur la console AWS Glue ou à l'aide d'opérations d'API de Machine Learning AWS Glue.

Combien de fois dois-je ajouter des étiquettes ? De combien d'étiquettes ai-je besoin ?

Les réponses à ces questions dépendent essentiellement de vous. Vous devez évaluer si `FindMatches` offre le niveau d'exactitude dont vous avez besoin et si vous pensez qu'un effort d'étiquetage supplémentaire est utile pour vous. La meilleure façon de procéder pour cela consiste à examiner les métriques « Precision » « Recall » et « Area under the precision recall curve » que vous pouvez générer lorsque vous choisissez Estimate quality (Estimation de la qualité) sur la console AWS Glue. Une fois que vous avez étiqueté plusieurs ensembles de tâches, réexécutez ces métriques et vérifiez si elles présentent des améliorations. Si, après l'étiquetage de quelques ensembles de tâches, vous ne constatez pas une amélioration au niveau de la métrique concernée, il est vraisemblable que la qualité de la transformation a atteint son niveau maximum.

Pourquoi les étiquettes Vrai positif et Vrai négatif sont-elles nécessaires ?

La transformation `FindMatches` nécessite à la fois des exemples positifs et négatifs pour vous permettre d'enseigner ce que vous estimez être une correspondance. Si vous étiquetez des données de formation générées par `FindMatches` (par exemple, à l'aide de l'option I do not have labels (Je ne dispose pas d'étiquettes), `FindMatches` essaie de générer un ensemble d'« ID d'ensemble d'étiquettes » pour vous. Au sein de chaque tâche, vous donnez la même « étiquette » à certains enregistrements et des « étiquettes » différentes à d'autres enregistrements. En d'autres termes, les

tâches ne sont généralement pas toutes identiques ou toutes différentes (mais c'est normal si une tâche particulière est entièrement « identique » ou entièrement « non identique »).

Si vous entraînez votre transformation `FindMatches` à l'aide de l'option Upload labels from S3 (Télécharger des étiquettes de S3), essayez d'inclure à la fois des exemples d'enregistrements correspondants et non correspondants. Il est acceptable d'avoir un seul type. Ces étiquettes vous aident à créer une transformation `FindMatches` plus exacte, mais vous devez quand-même étiqueter certains enregistrements que vous générez à l'aide de l'option Generate labeling file (Générer un fichier d'étiquetage).

Comment puis-je m'assurer que la transformation fasse exactement ce que je lui ai enseigné ?

La transformation `FindMatches` apprend à partir des étiquettes que vous fournissez, elle peut donc générer des paires d'enregistrements qui ne respectent pas les étiquettes fournies. Pour garantir que la `FindMatches` transformation respecte vos étiquettes, sélectionnez « EnforceProvidedLabelsin FindMatchesParameter».

Quelles techniques puis-je utiliser lorsqu'une transformation ML identifie des éléments en tant que correspondances alors qu'elles n'en sont pas vraiment ?

Vous pouvez utiliser les techniques suivantes :

- Définir `precisionRecallTradeoff` sur une valeur plus élevée. Cela peut se traduire par la production d'un nombre de correspondances inférieur, mais cela peut également permettre de fractionner votre gros cluster lorsqu'il atteint une valeur suffisamment élevée.
- Prenez les lignes de sortie correspondant aux résultats incorrects et reformatez-les en tant qu'ensemble d'étiquetage (en retirant la colonne `match_id` et en ajoutant une colonne `labeling_set_id` et `label`). Si nécessaire, fractionnez (subdivisez) en plusieurs ensembles d'étiquetage afin de vous assurer que l'étiqueteur peut continuer en gardant chaque étiquetage à l'esprit tout en affectant des étiquettes. Ensuite, étiquetez correctement les ensembles correspondants, puis chargez le fichier d'étiquettes et ajoutez-le à vos étiquettes existantes. Cela peut suffisamment entraîner votre transformateur à ce qu'il doit rechercher pour comprendre le modèle.
- (Avancé) Enfin, examinez ces données pour voir si vous pouvez détecter un modèle que le système n'a pas remarqué. Prétraitez ces données à l'aide de fonctions AWS Glue standard pour normaliser les données. Mettez en évidence ce sur quoi l'algorithme doit se baser pour apprendre en séparant les données que vous jugez différemment importantes dans leurs propres colonnes. Vous pouvez aussi créer des colonnes combinées à partir de colonnes contenant des données connexes.

Utilisation de transformations du machine learning

Vous pouvez l'utiliser AWS Glue pour créer des transformations d'apprentissage automatique personnalisées qui peuvent être utilisées pour nettoyer vos données. Vous pouvez utiliser ces transformations lorsque vous créez une tâche sur la console AWS Glue .

Pour de plus amples informations sur la création d'une transformation Machine Learning, veuillez consulter Record correspondant à AWS Lake Formation FindMatches.

Rubriques

- Propriétés de transformation
- Ajout et modification de transformations Machine Learning
- Affichage des détails d'une transformation
- Entraîner les transformations à l'aide d'étiquettes

Propriétés de transformation

Pour consulter une transformation d'apprentissage automatique existante, connectez-vous à la AWS Management ConsoleAWS Glue console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/. Dans le volet de navigation, sous Intégration des données et ETL, choisissez Outils de classification des données > Correspondance des enregistrements.

Les propriétés de chaque transformation :

Nom de transformation

Nom unique donné à la transformation lors de sa création.

ID

Identifiant unique de la transformation.

Nombre d'étiquettes

Le nombre d'étiquettes dans le fichier d'étiquetage qui a été fourni pour aider à entraîner la transformation.

Statut

Indique si la transformation a le statut Ready (Prêt) ou Needs teaching (Entraînement nécessaire). Pour qu'une transformation machine learning puisse s'exécuter dans une tâche, elle doit avoir le statut Ready (Prêt).

Créé

Date de création de la transformation.

Modifié

Date de la dernière mise à jour de la transformation.

Description

Description fournie pour la transformation, le cas échéant.

Version de l'AWS Glue

La version de AWS Glue utilisée.

ID d'exécution

Nom unique donné à la transformation lors de sa création.

Type de tâche

Type de transformation Machine Learning ; par exemple, Find matching records (Rechercher des enregistrements correspondants).

Statut

Indique le statut de l'exécution de la tâche. Les statuts possibles incluent :

- Démarrage en cours
- En cours d'exécution
- Arrêt en cours
- Arrêté(e)
- Réussi(e)
- Échec
- Timeout

Erreur

Si le statut est Échec, un message d'erreur indiquant la raison de l'échec s'affiche.

Ajout et modification de transformations Machine Learning

Vous pouvez afficher, supprimer, configurer et entraîner, ou encore régler une transformation sur la console AWS Glue. Cochez la case en regard de la transformation dans la liste, choisissez Action, puis choisissez l'action que vous souhaitez effectuer.

Création d'une nouvelle transformation ML

Pour ajouter une nouvelle transformation machine learning, choisissez Créer une transformation. Suivez les instructions fournies dans l'assistant Ajouter une tâche. Pour de plus amples informations, veuillez consulter Record correspondant à AWS Lake Formation FindMatches.

Étape 1. Définissez les propriétés de la transformation.

1. Saisissez un nom et une description de la tâche (facultatif).
2. Définissez éventuellement la configuration de sécurité. Consultez Utilisation du chiffrement des données avec les transformations machine learning.
3. Définissez éventuellement les paramètres d'exécution des tâches. Les paramètres d'exécution des tâches vous permettent de personnaliser le mode d'exécution de la tâche. Sélectionnez le type de travailleur, le nombre de travailleurs, le délai d'expiration de la tâche (en minutes), le nombre de tentatives et la version de AWS Glue.
4. Vous pouvez éventuellement définir des balises. Les tags sont des libellés que vous pouvez attribuer à une AWS ressource. Chaque balise est constituée d'une clé et d'une valeur facultative. Les balises peuvent être utilisées pour rechercher et filtrer vos ressources ou pour suivre vos AWS coûts.

Étape 2. Choisissez la table et la clé primaire.

1. Choisissez la base de données et la table du catalogue AWS Glue.
2. Choisissez une clé primaire dans le tableau sélectionné. La colonne de clé primaire contient généralement un identifiant unique pour chaque enregistrement de la source de données.

Étape 3. Sélectionnez les options de réglage.

1. Pour Rappel ou précision, choisissez la valeur de réglage pour ajuster la transformation afin de favoriser le rappel ou la précision. Par défaut, Équilibré est sélectionné, mais vous pouvez choisir de privilégier le rappel ou la précision, ou choisir Personnalisé et saisir une valeur comprise entre 0,0 et 1,0 (inclus).
2. Pour Réduire le coût ou la précision, choisissez la valeur de réglage qui favorise la réduction des coûts ou de la précision, ou choisissez Personnalisé et saisissez une valeur comprise entre 0,0 et 1,0 (inclus).

3. Pour Application de la correspondance, choisissez Forcer la sortie à correspondre aux étiquettes si vous souhaitez entraîner la transformation ML en forçant la sortie à correspondre aux étiquettes utilisées.

Étape 4 : Vérifiez et créez.

1. Passez en revue les options des étapes 1 à 3.
2. Choisissez Modifier pour chaque étape qui doit être modifiée. Choisissez Créer une transformation pour terminer l'assistant de création de transformation.

Utilisation du chiffrement des données avec les transformations machine learning

Lors de l'ajout d'une transformation Machine Learning à AWS Glue, vous pouvez éventuellement spécifier une configuration de sécurité associée à la source de données ou à la cible de données. Si le compartiment Amazon S3 utilisé pour stocker les données est chiffré avec une configuration de sécurité, spécifiez la même configuration de sécurité lors de la création de la transformation.

Vous pouvez également choisir d'utiliser le chiffrement côté serveur avec AWS KMS (SSE-KMS) pour chiffrer le modèle et les étiquettes afin d'empêcher des personnes non autorisées de l'inspecter. Si vous choisissez cette option, vous êtes invité à choisir le AWS KMS key nom, ou vous pouvez choisir Enter a key ARN. Si vous choisissez de saisir l'ARN de la clé KMS, un deuxième champ apparaît dans lequel vous pouvez saisir l'ARN de la clé KMS.

> **Note**
>
> Actuellement, les transformations ML utilisant une clé de chiffrement personnalisée ne sont pas prises en charge dans les régions suivantes :
>
> - Asie-Pacifique (Osaka) - `ap-northeast-3`

Affichage des détails d'une transformation

Affichage des propriétés de transformation

La page Propriété de transformation inclut les attributs de votre transformation. Il affiche les détails relatifs à la définition de transformation, y compris les éléments suivants :

- Transform name (Nom de la transformation) indique le nom de la transformation.

- Type répertorie le type de la transformation.
- Status (État) affiche si la transformation est prête à être utilisée dans un script ou une tâche.
- Force output to match labels (Forcer la sortie pour faire correspondre les étiquettes) affiche si la transformation force la sortie afin qu'elle fasse correspondre les étiquettes fournies par l'utilisateur.
- La version Spark est liée à la version AWS Glue que vous avez choisie dans Task run properties (Propriétés d'exécution de la tâche) lors de l'ajout de la transformation. AWS Glue 1.0 et Spark 2.4 sont recommandés pour la plupart des clients. Pour plus d'informations, consultez Version de AWS Glue.

Onglets Historique, Estimation de la qualité et Balises

Les détails d'une transformation incluent les informations que vous avez définies lors de la création de cette transformation. Pour afficher les détails d'une transformation, sélectionnez la transformation dans la liste Machine learning transforms (Transformations Machine learning), puis consultez les informations sur les onglets suivants :

- History (Historique)
- Estimation de la qualité
- Étiquettes

History (Historique)

L'onglet History (Historique) affiche l'historique d'exécution de votre tâche de transformation. Plusieurs types de tâches sont exécutées pour entraîner une transformation. Pour chaque tâche, les métriques d'exécution sont les suivantes :

- Run ID (ID d'exécution) est un identifiant créé par AWS Glue pour chaque exécution de cette tâche.
- Task type (Type de tâche) affiche le type de l'exécution de tâche.
- Run status (Statut d'exécution) indique la réussite de chaque tâche répertoriée avec l'exécution la plus récente en haut de la liste.
- Error (Erreur) affiche les détails d'un message d'erreur si l'exécution a échoué..
- Start time (Heure de début) indique la date et l'heure (heure locale) auxquelles la tâche a démarré.
- Heure de fin indique la date et l'heure (heure locale) auxquelles la tâche s'est terminée.

- Logs (Journaux) fournit des liens vers les journaux écrits sur `stdout` pour cette exécution de tâche.

  Le lien Logs permet d'accéder à Amazon CloudWatch Logs. Vous pouvez y consulter les détails relatifs aux tables créées dans le AWS Glue Data Catalog et aux éventuelles erreurs rencontrées. Vous pouvez gérer la période de conservation des journaux sur la CloudWatch console. La conservation des journaux par défaut est `Never Expire`. Pour plus d'informations sur la modification de la période de conservation, consultez la section Conservation des données du journal des modifications dans CloudWatch les journaux dans le guide de l'utilisateur Amazon CloudWatch Logs.
- Fichier d'étiquettes fournit un lien vers Amazon S3 pour un fichier d'étiquetage généré.

Estimation de la qualité

L'onglet Estimate quality (Estimation de la qualité) affiche les métriques que vous utilisez pour mesurer la qualité de la transformation. Les estimations sont calculées en comparant les prédictions de correspondance de transformation à l'aide d'un sous-ensemble de vos données étiquetées par rapport aux étiquettes que vous avez fournies. Ces estimations sont approximatives. Vous pouvez appeler une tâche Estimate quality (Estimation de la qualité) exécutée à partir de cet onglet.

L'onglet Estimate quality (Estimation de la qualité) affiche les métriques de la dernière estimation de la qualité, y compris les propriétés suivantes :

- Area under the Precision-Recall curve (Zone sous la courbe de précision-rappel est un nombre unique qui estime la limite supérieure de la qualité globale de la transformation. Il est indépendant du choix effectué pour le paramètre de précision-rappel. Des valeurs plus élevées indiquent que vous avez un compromis précision-rappel plus attractif.
- Precision (Précision) évalue la fréquence à laquelle la transformation est correcte lorsqu'elle prédit une correspondance.
- Recall upper limit (Limite supérieure de rappel) évalue, pour une correspondance réelle, la fréquence à laquelle la transformation prédit la correspondance.
- F1 évalue la précision de la transformation comprise entre 0 et 1, où 1 est la meilleure précision. Pour plus d'informations, consultez la page Wikipedia relative au score F1.
- La table Column importance (Importance de la colonne) affiche les noms de colonnes et le score d'importance de chaque colonne. L'importance des colonnes vous aide à comprendre comment elles contribuent à votre modèle, en identifiant les colonnes de vos enregistrements qui sont le

plus utilisées pour effectuer la correspondance. Ces données peuvent vous inciter à ajouter ou à modifier votre jeu d'étiquettes pour augmenter ou réduire l'importance des colonnes.

La colonne Importance fournit un score numérique pour chaque colonne, sous la forme d'une décimale ne dépassant pas 1,0.

Pour de plus amples informations sur la compréhension des estimations de qualité et de la qualité réelle, veuillez consulter Estimations de la qualité par rapport à la end-to-end (vraie) qualité.

Pour de plus amples informations sur le réglage de votre transformation, veuillez consulter Réglage des transformations Machine Learning dans AWS Glue.

Estimations de la qualité par rapport à la end-to-end (vraie) qualité

AWS Glue estime la qualité de vos transformations en présentant le modèle interne appris par machine learning avec un certain nombre de paires d'enregistrements pour lesquels vous avez fourni des étiquettes correspondantes, mais que le modèle n'a jamais vus auparavant. Ces estimations de qualité sont une fonction de la qualité du modèle appris par la machine (qui est influencé par le nombre d'enregistrements que vous étiquetez pour « entraîner » la transformation). Le rappel end-to-end, ou véritable rappel (qui n'est pas automatiquement calculé par le`ML transform`) est également influencé par le mécanisme de `ML transform` filtrage qui propose une grande variété de correspondances possibles avec le modèle appris par machine.

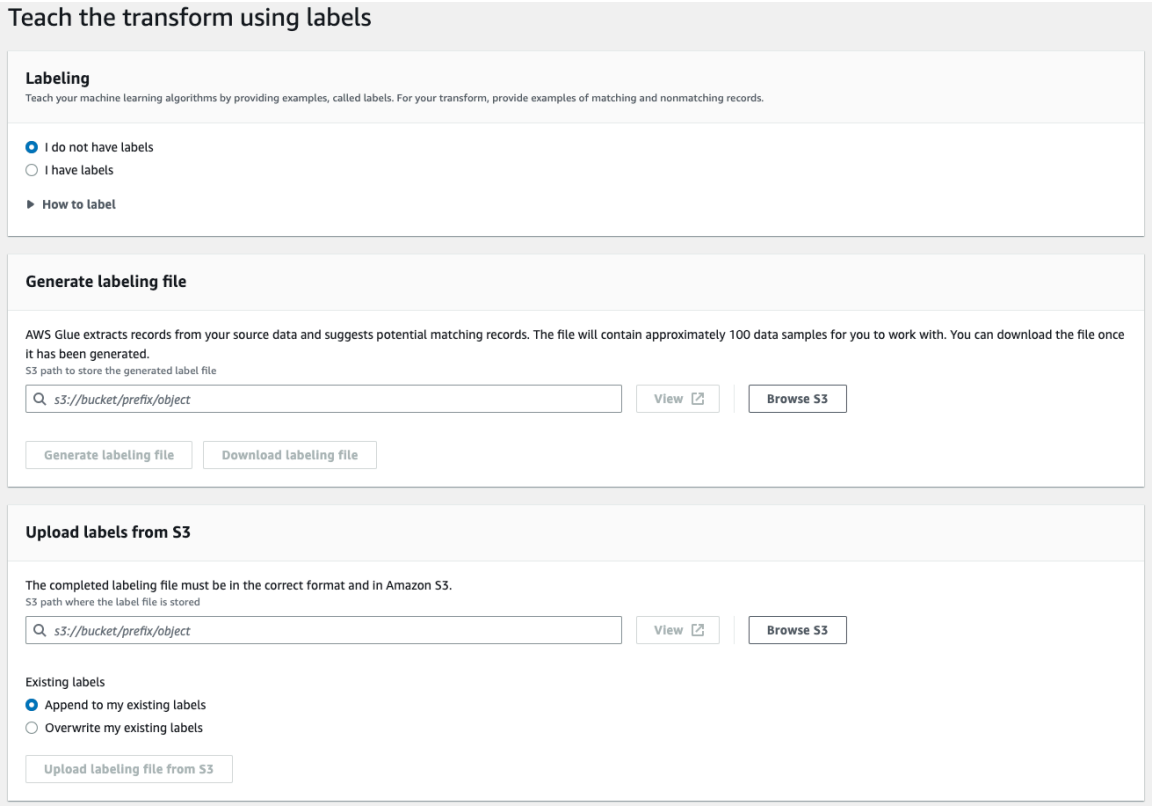
Vous pouvez régler cette méthode de filtrage essentiellement en spécifiant la valeur de réglage Réduction coût-précision. À mesure que la valeur de réglage se rapproche de Précision, le système effectue une recherche plus approfondie et plus coûteuse des paires d'enregistrements qui pourraient être des correspondances. Un plus grand nombre de paires d'enregistrements sont introduites dans votre modèle d'apprentissage automatique, et votre `ML transform` rappel réel se rapproche de la métrique de rappel estimée. end-to-end Par conséquent, les modifications de la end-to-end qualité de vos correspondances résultant de modifications du compromis coût/précision de vos correspondances ne seront généralement pas prises en compte dans l'estimation de la qualité.

Étiquettes

Les tags sont des libellés que vous pouvez attribuer à une AWS ressource. Chaque balise est constituée d'une clé et d'une valeur facultative. Les balises peuvent être utilisées pour rechercher et filtrer vos ressources ou pour suivre vos AWS coûts.

Entraîner les transformations à l'aide d'étiquettes

Vous pouvez entraîner votre transformation ML à l'aide d'étiquettes (exemples) en choisissant Entraîner une transformation sur la page de détails de la transformation ML. Lorsque vous entraînez votre algorithme de machine learning en fournissant des exemples (appelés étiquettes), vous pouvez choisir les étiquettes existantes à utiliser ou créer un fichier d'étiquetage.



- Étiquetage : si vous avez des étiquettes, choisissez J'ai des étiquettes. Si vous n'avez pas d'étiquettes, vous pouvez toujours passer à l'étape suivante pour générer un fichier d'étiquettes.
- Générer un fichier d'étiquetage : AWS Glue extrait les enregistrements de vos données sources et suggère des enregistrements correspondants potentiels. Vous choisissez le compartiment Amazon S3 pour stocker le fichier d'étiquette généré. Choisissez Générer un fichier d'étiquetage pour démarrer le processus. Lorsque vous avez terminé, choisissez Télécharger le fichier d'étiquetage. Le fichier téléchargé comportera une colonne pour les étiquettes dans laquelle vous pourrez remplir les étiquettes.
- Charger des étiquettes depuis Amazon S3 : choisissez le fichier d'étiquetage complet dans le compartiment Amazon S3 dans lequel le fichier d'étiquetage est stocké. Choisissez ensuite d'ajouter les étiquettes à vos étiquettes existantes ou de les remplacer. Choisissez Charger le fichier d'étiquetage depuis Amazon S3.

Tutoriel : Création d'une transformation de machine learning avec AWS Glue

Ce didacticiel vous guide tout au long des actions de création et de gestion d'une transformation Machine Learning (ML) à l'aide de AWS Glue. Avant d'utiliser ce tutoriel, vous devez être familiarisé avec l'utilisation de la console AWS Glue pour ajouter des crawlers et des tâches et modifier des scripts. Vous devez également être familiarisé avec la recherche et le téléchargement de fichiers sur la console Amazon Simple Storage Service (Amazon S3).

Dans cet exemple, vous allez créer une `FindMatches` transformation pour rechercher des enregistrements correspondants, l'entraîner à identifier les enregistrements correspondants et les non correspondants, puis l'utiliser dans une tâche AWS Glue. La tâche AWS Glue écrit un nouveau fichier Amazon S3 avec une colonne supplémentaire nommée `match_id`.

Les données source utilisées par ce tutoriel sont un fichier nommé `dblp_acm_records.csv`. Ce fichier est une version modifiée de publications universitaires (DBLP et ACM) disponibles à partir de l'ensemble de données DBLP ACM d'origine. Le fichier `dblp_acm_records.csv` est un fichier CSV (valeurs séparées par une virgule) au format UTF-8 sans marque d'ordre d'octet (BOM).

Un second fichier, `dblp_acm_labels.csv`, est un exemple de fichier d'étiquetage qui contient des enregistrements correspondants et non correspondants utilisés pour entraîner la transformation dans le cadre du didacticiel.

Rubriques

- Étape 1 : Analyse des données source
- Étape 2 : Ajouter une transformation de Machine Learning
- Étape 3 : Entraîner votre transformation Machine Learning
- Étape 4 : Estimer la qualité de votre transformation Machine Learning
- Étape 5 : Ajouter et exécuter une tâche avec votre transformation Machine Learning
- Étape 6 : vérifier les données de sortie d'Amazon S3

Étape 1 : Analyse des données source

Tout d'abord, analysez le le fichier CSV source Amazon S3 pour créer une table de métadonnées correspondante dans Data Catalog.

> **Important**
>
> Pour indiquer à le crawler de créer une table pour seulement le fichier CSV, stockez les données source CSV dans un autre dossier Amazon S3 à partir d'autres fichiers.

1. Connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/.
2. Dans le panneau de navigation, sélectionnez Crawlers, Ajouter un crawler.
3. Suivez les instructions de l'assistant pour créer et exécuter un crawler nommé `demo-crawl-dblp-acm` avec une sortie dans une base de données `demo-db-dblp-acm`. Lorsque vous exécutez l'assistant, créez la base de données `demo-db-dblp-acm` si elle n'existe pas déjà. Choisissez un chemin d'inclusion Amazon S3 pour obtenir des échantillons de données dans la AWS région actuelle. Par exemple, pour `us-east-1`, le chemin d'inclusion Amazon S3 pour le fichier source est `s3://ml-transforms-public-datasets-us-east-1/dblp-acm/records/dblp_acm_records.csv`.

   Si l'opération aboutit, le crawler crée la table `dblp_acm_records_csv` avec les colonnes suivantes : id, title, authors, venue, year et source.

Étape 2 : Ajouter une transformation de Machine Learning

Ensuite, ajoutez une transformation de Machine Learning basée sur le schéma de votre table source de données créée par le crawler nommé `demo-crawl-dblp-acm`.

1. Sur la console AWS Glue, dans le volet de navigation sous Intégration des données et ETL, choisissez Outils de classification des données > Correspondance des enregistrements, puis Ajouter une transformation. Suivez les instructions de l'assistant pour créer une transformation `Find matches` avec les propriétés suivantes.

   a. Pour Transform name (Nom de transformation), entrez **`demo-xform-dblp-acm`**. Il s'agit du nom de la transformation qui est utilisée pour rechercher des correspondances dans les données source.

   b. Pour IAM role (Rôle IAM), sélectionnez un rôle IAM qui dispose d'une autorisation sur les données source Amazon S3, le fichier d'étiquetage et les opérations d'API AWS Glue. Pour de plus amples informations, veuillez consulter Création d'un rôle IAM pour AWS Glue dans le Guide du développeur AWS Glue .

c. Pour Source de données, choisissez la table nommée dblp_acm_records_csv dans la base de données. demo-db-dblp-acm

d. Pour Primary key (Clé primaire), choisissez la colonne de clé primaire pour la table, id.

2. Dans l'assistant, choisissez, choisissez Finish (Terminer) et revenez à la liste ML transforms (Transformations ML).

Étape 3 : Entraîner votre transformation Machine Learning

Ensuite, vous entraînez votre transformation Machine Learning à l'aide du fichier d'étiquetage de l'exemple de didacticiel.

Vous ne pouvez pas utiliser une transformation Machine Language dans une tâche d'extraction, de transformation et de chargement (ETL) tant que son statut n'est pas Ready for use (Prêt pour utilisation). Pour que votre transformation soit prête, vous devez l'entraîner à identifier les enregistrements correspondants et non correspondants en fournissant des exemples d'enregistrements correspondants et non correspondants. Pour entraîner votre transformation, vous pouvez Générer un fichier d'étiquettes, ajouter des étiquettes, puis Télécharger le fichier d'étiquettes. Dans ce didacticiel, vous pouvez utiliser l'exemple de fichier d'étiquetage nommé `dblp_acm_labels.csv`. Pour plus informations sur le processus d'étiquetage, veuillez consulter Étiquetage.

1. Sur la console AWS Glue, dans le volet de navigation, sélectionnez Correspondance des enregistrements.
2. Choisissez la transformation`demo-xform-dblp-acm`, puis choisissez Action, Teach (Entraîner). Suivez les instructions de l'assistant pour entraîner votre transformation `Find matches`.
3. Sur la page des propriétés de transformation, choisissez I have labels (Je dispose d'étiquettes). Choisissez un chemin Amazon S3 vers l'exemple de fichier d'étiquetage dans la AWS région actuelle. Par exemple, pour `us-east-1`, chargez le fichier d'étiquetage fourni à partir du chemin Amazon S3 `s3://ml-transforms-public-datasets-us-east-1/dblp-acm/labels/dblp_acm_labels.csv` avec l'option de remplacement des étiquettes existantes. Le fichier d'étiquetage doit se trouver dans Amazon S3 dans la même région que la console AWS Glue.

   Lorsque vous chargez un fichier d'étiquetage, une tâche est démarrée dans AWS Glue pour ajouter ou remplacer les étiquettes utilisées pour entraîner la transformation à traiter la source de données.

4. Sur la dernière page de l'assistant, choisissez Terminer, puis revenez à la liste Transformations Machine Learning.

Étape 4 : Estimer la qualité de votre transformation Machine Learning

Ensuite, vous pouvez estimer la qualité de votre transformation Machine Learning. La qualité dépend de la quantité d'étiquetage que vous avez effectuée. Pour en savoir plus sur l'estimation de la qualité, consulter Estimation de la qualité.

1. Sur la console AWS Glue, dans le volet de navigation sous Intégration des données et ETL, choisissez Outils de classification des données > Correspondance des enregistrements.
2. Choisissez la transformation `demo-xform-dblp-acm`, puis choisissez l'onglet Estimate quality (Estimation de la qualité). Cet onglet affiche les estimations de qualité actuelles, le cas échéant, pour la transformation.
3. Choisissez Estimation de la qualité pour démarrer une tâche permettant d'estimer la qualité de la transformation. La précision de l'estimation de la qualité est basée sur l'étiquetage des données source.
4. Accédez à l'onglet History (Historique). Dans ce panneau, les exécutions de tâche sont répertoriées pour la transformation, y compris la tâche Estimation de la qualité. Pour plus de détails sur l'exécution, choisissez Journaux. Vérifiez que le statut d'exécution est Réussite lorsqu'elle se termine.

Étape 5 : Ajouter et exécuter une tâche avec votre transformation Machine Learning

Au cours de cette étape, vous utilisez votre transformation Machine Learning pour ajouter et exécuter une tâche dans AWS Glue. Lorsque la transformation `demo-xform-dblp-acm` a le statut Ready for use (Prête pour utilisation), vous pouvez l'utiliser dans une tâche ETL.

1. Sur la console AWS Glue, dans le panneau de navigation, sélectionnez Tâches.
2. Choisissez Ajouter une tâche, et suivez les étapes de l'assistant pour créer une tâche ETL Spark avec un script généré. Choisissez les valeurs de propriété suivantes pour votre transformation :

    a. Pour Nom, choisissez l'exemple de tâche dans ce didacticiel, demo-etl-dblp-acm.

    b. Pour Rôle IAM, choisissez un rôle IAM qui a une autorisation sur les données source Amazon S3, le fichier d'étiquetage et les opérations d'API AWS Glue. Pour plus informations, veuillez consulter Création d'un rôle IAM pour AWS Glue dans le Guide du développeur AWS Glue .

c. Pour Langage ETL, choisissez Scala. Il s'agit du langage de programmation dans le script ETL.

d. Pour le nom du fichier de script, sélectionnez demo-etl-dblp-acm. Il s'agit du nom de fichier du script Scala (identique au nom de la tâche).

e. Pour Data source (Source de données), choisissez dblp_acm_records_csv. La source de données que vous choisissez doit correspondre au schéma de source de données de la transformation Machine Learning.

f. Pour Type de transformation, choisissez Rechercher des enregistrements correspondants pour créer une tâche à l'aide d'une transformation Machine Learning.

g. Désactivez Enlever les enregistrements en double. Vous ne souhaitez pas retirer les enregistrements en double car les enregistrements de sortie écrits comportent un champ `match_id` supplémentaire ajouté.

h. Pour Transform demo-xform-dblp-acm, choisissez la transformation d'apprentissage automatique utilisée par la tâche.

i. Pour Créer des tables dans votre cible de données, choisissez de créer des tables avec les propriétés suivantes :

- Type de magasin de données — **`Amazon S3`**
- Format – **`CSV`**
- Type de compression – **`None`**
- Chemin cible : chemin Amazon S3 où est écrite la sortie de la tâche (dans la AWS région de console actuelle)

3. Choisissez Enregistrer la tâche et modifier le script pour afficher la page de l'éditeur de script.

4. Modifiez le script pour ajouter une instruction qui entraîne l'écriture de la sortie de tâche du chemin d'accès cible dans un seul fichier de partition. Ajoutez cette instruction immédiatement après l'instruction qui exécute la transformation `FindMatches`. L'instruction est similaire à ce qui suit.

```
val single_partition = findmatches1.repartition(1)
```

Vous devez modifier l'instruction `.writeDynamicFrame(findmatches1)` pour écrire la sortie en tant que `.writeDynamicFrame(single_partion)`.

5. Une fois que vous avez modifié le script, cliquez sur Enregistrer. Le script modifié ressemble au code qui suit, mais personnalisé pour votre environnement.

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.errors.CallSite
import com.amazonaws.services.glue.ml.FindMatches
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._

object GlueApp {
  def main(sysArgs: Array[String]) {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    // @params: [JOB_NAME]
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)
    // @type: DataSource
    // @args: [database = "demo-db-dblp-acm", table_name = "dblp_acm_records_csv",
 transformation_ctx = "datasource0"]
    // @return: datasource0
    // @inputs: []
    val datasource0 = glueContext.getCatalogSource(database = "demo-db-dblp-acm",
 tableName = "dblp_acm_records_csv", redshiftTmpDir = "", transformationContext =
 "datasource0").getDynamicFrame()
    // @type: FindMatches
    // @args: [transformId = "tfm-123456789012", emitFusion = false,
 survivorComparisonField = "<primary_id>", transformation_ctx = "findmatches1"]
    // @return: findmatches1
    // @inputs: [frame = datasource0]
    val findmatches1 = FindMatches.apply(frame = datasource0, transformId
 = "tfm-123456789012", transformationContext = "findmatches1",
 computeMatchConfidenceScores = true)


    // Repartition the previous DynamicFrame into a single partition.
    val single_partition = findmatches1.repartition(1)


    // @type: DataSink
    // @args: [connection_type = "s3", connection_options = {"path": "s3://aws-
glue-ml-transforms-data/sal"}, format = "csv", transformation_ctx = "datasink2"]
    // @return: datasink2
```

```
    // @inputs: [frame = findmatches1]
    val datasink2 = glueContext.getSinkWithFormat(connectionType =
 "s3", options = JsonOptions("""{"path": "s3://aws-glue-ml-transforms-
data/sal"}"""), transformationContext = "datasink2", format =
 "csv").writeDynamicFrame(single_partition)
    Job.commit()
  }
}
```

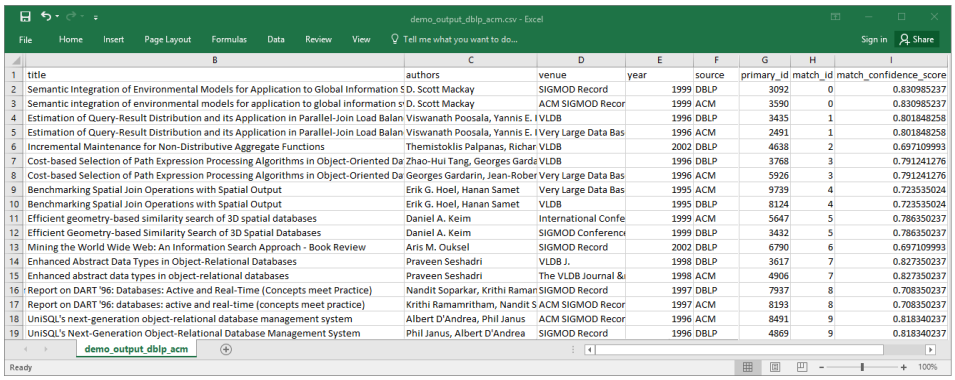
6. Appuyez sur Exécuter la tâche) pour démarrer l'exécution de la tâche. Vérifiez le statut de la tâche dans la liste des tâches. Une fois la tâche terminée, dans l'onglet Transformation ML, Historique, une nouvelle ligne exécution ID de type Tâche ETL est ajoutée.
7. Accédez à l'onglet Jobs (Tâches), Historique. Dans ce panneau, les exécutions de tâches sont répertoriées. Pour plus de détails sur l'exécution, choisissez Journaux. Vérifiez que le statut d'exécution est Réussite lorsqu'elle se termine.

Étape 6 : vérifier les données de sortie d'Amazon S3

Au cours de cette étape, vous vérifiez la sortie de la tâche exécutée dans le compartiment Amazon S3 que vous avez choisi lors de l'ajout à la tâche. Vous pouvez télécharger le fichier de sortie sur votre machine locale et vérifier que les enregistrements correspondants ont été identifiés.

1. Ouvrez la console Amazon S3 à l'adresse https://console.aws.amazon.com/s3/.
2. Téléchargez le fichier de sortie cible de la tâche `demo-etl-dblp-acm`. Ouvrez le fichier dans une application de feuille de calcul (vous devrez peut-être ajouter une extension de fichier `.csv` pour que le fichier s'ouvre correctement).

   L'image suivante montre un extrait de la sortie dans Microsoft Excel.



	B	C	D	E	F	G	H	I
1	title	authors	venue	year	source	primary_id	match_id	match_confidence_score
2	Semantic Integration of Environmental Models for Application to Global Information S	D. Scott Mackay	SIGMOD Record	1999	DBLP	3092	0	0.830985237
3	Semantic integration of environmental models for application to global information s	D. Scott Mackay	ACM SIGMOD Recor	1999	ACM	3590	0	0.830985237
4	Estimation of Query-Result Distribution and its Application in Parallel-Join Load Balan	Viswanath Poosala, Yannis E. I	VLDB	1996	DBLP	3435	1	0.801848258
5	Estimation of Query-Result Distribution and its Application in Parallel-Join Load Balan	Viswanath Poosala, Yannis E. I	Very Large Data Bas	1996	ACM	2491	1	0.801848258
6	Incremental Maintenance for Non-Distributive Aggregate Functions	Themistoklis Palpanas, Richar	VLDB	2002	DBLP	4638	2	0.697109993
7	Cost-based Selection of Path Expression Processing Algorithms in Object-Oriented Da	Zhao-Hui Tang, Georges Garda	VLDB	1996	DBLP	3768	3	0.791241276
8	Cost-based Selection of Path Expression Processing Algorithms in Object-Oriented Da	Georges Gardarin, Jean-Rober	Very Large Data Bas	1996	ACM	5926	3	0.791241276
9	Benchmarking Spatial Join Operations with Spatial Output	Erik G. Hoel, Hanan Samet	Very Large Data Bas	1995	ACM	9739	4	0.723535024
10	Benchmarking Spatial Join Operations with Spatial Output	Erik G. Hoel, Hanan Samet	VLDB	1995	DBLP	8124	4	0.723535024
11	Efficient geometry-based similarity search of 3D spatial databases	Daniel A. Keim	International Confe	1999	ACM	5647	5	0.786350237
12	Efficient Geometry-based Similarity Search of 3D Spatial Databases	Daniel A. Keim	SIGMOD Conferenc	1999	DBLP	3432	5	0.786350237
13	Mining the World Wide Web: An Information Search Approach - Book Review	Aris M. Ouksel	SIGMOD Record	2002	DBLP	6790	6	0.697109993
14	Enhanced Abstract Data Types in Object-Relational Databases	Praveen Seshadri	VLDB J.	1998	DBLP	3617	7	0.827350237
15	Enhanced abstract data types in object-relational databases	Praveen Seshadri	The VLDB Journal &	1998	ACM	4906	7	0.827350237
16	Report on DART '96: Databases: Active and Real-Time (Concepts meet Practice)	Nandit Soparkar, Krithi Raman	SIGMOD Record	1997	DBLP	7937	8	0.708350237
17	Report on DART '96: databases: active and real-time (concepts meet practice)	Krithi Ramamritham, Nandit S	ACM SIGMOD Recor	1997	ACM	8193	8	0.708350237
18	UniSQL's next-generation object-relational database management system	Albert D'Andrea, Phil Janus	ACM SIGMOD Recor	1996	ACM	8491	9	0.818340237
19	UniSQL's Next-Generation Object-Relational Database Management System	Phil Janus, Albert D'Andrea	SIGMOD Record	1996	DBLP	4869	9	0.818340237

   La source de données et le fichier cible comportent tous deux 4 911 enregistrements. Toutefois, la transformation `Find matches` ajoute une autre colonne nommée `match_id` pour identifier

les enregistrements correspondants dans la sortie. Les lignes avec le même `match_id` sont considérées comme des enregistrements correspondants. Le `match_confidence_score` est un nombre compris entre 0 et 1 qui fournit une estimation de la qualité des correspondances trouvées par `Find matches`.

3. Triez le fichier de sortie par `match_id` afin de voir facilement les enregistrements qui sont des correspondances. Comparez les valeurs des autres colonnes pour voir si les résultats de la transformation `Find matches` vous conviennent. Si ce n'est pas le cas, vous pouvez continuer à entraîner la transformation en ajoutant d'autres étiquettes.

   Vous pouvez également trier le fichier sur un autre champ, par exemple `title`, pour voir si les enregistrements avec des titres similaires ont le même `match_id`.

Recherche de correspondances progressives

La fonction Recherche de correspondances vous permet d'identifier les registres en double ou correspondants dans votre jeu de données, même lorsque les registres n'ont pas un identifiant unique commun et qu'aucun champ ne correspond exactement. La version initiale de la recherche de correspondances transforme les registres correspondants identifiés au sein d'un même jeu de données. Lorsque vous ajoutez de nouvelles données au jeu de données, vous deviez les fusionner avec le jeu de données propre existant et exécuter à nouveau la correspondance avec le jeu de données fusionné complet.

La fonction de correspondance progressive facilite la correspondance avec des registres progressifs en comparaison aux jeux de données appariés existants. Supposons que vous souhaitiez associer les données de prospects aux jeux de données clients existants. La fonctionnalité de correspondance progressive vous offre la flexibilité nécessaire pour associer des centaines de milliers de nouveaux prospects à une base de données existante de prospects et de clients en fusionnant les résultats en une seule base de données ou table. En faisant correspondre uniquement les jeux de données nouveaux et existants, l'optimisation de recherche de correspondances progressives réduit le temps de calcul, ce qui réduit également les coûts.

L'utilisation de la correspondance progressive est similaire à celle de la Recherche de correspondances décrite dans Tutoriel : Création d'une transformation de machine learning avec AWS Glue. Cette rubrique identifie uniquement les différences avec la correspondance progressive.

Pour en savoir plus, consultez l'article de blog sur Correspondance progressive des données.

Exécution d'une tâche de correspondance progressive

Pour la procédure suivante, supposons la situation suivante :

- Vous avez exploré le jeu de données existant dans la table first_records. Le jeu de données first_records doit être un jeu de données correspondant, ou la sortie de la tâche correspondante.
- Vous avez créé et entraîné une transformation Recherche de correspondances avec AWS Glue version 2.0. Il s'agit de la seule version de AWS Glue qui prend en charge les correspondances progressives.
- Le langage ETL est Scala. Notez que Python est également pris en charge.
- Le modèle déjà généré s'appelle `demo-xform`.

1. Analyse du jeu de données progressif vers la table second_records.
2. Sur la console AWS Glue, dans le panneau de navigation, sélectionnez Jobs (Tâches).
3. Choisissez Ajouter une tâche, et suivez les étapes de l'assistant pour créer une tâche ETL Spark avec un script généré. Choisissez les valeurs de propriété suivantes pour votre transformation :

    a. Pour Nom, choisissez demo-etl.

    b. Pour Rôle IAM, choisissez un rôle IAM disposant d'une autorisation sur les données source Amazon S3, le fichier d'étiquetage et les opérations d'APIAWS Glue.

    c. Pour Langage ETL, choisissez Scala.

    d. Pour Nom du fichier script, choisissez demo-etl. Il s'agit du nom de fichier du script Scala.

    e. Pour Source de données, choisissez first_records. La source de données que vous choisissez doit correspondre au schéma de source de données de la transformation Machine Learning.

    f. Pour Type de transformation, choisissez Find matching records Rechercher des enregistrements correspondants pour créer une tâche à l'aide d'une transformation Machine Learning.

    g. Sélectionnez l'option de correspondance progressive, et pour Source de données sélectionnez la table nommée second_records.

    h. Pour Transformation, choisissez demo-xform, la transformation de machine learning utilisée par la tâche.

    i. Choisissez Créer des tables dans votre cible de données ou Utiliser les tables du catalogue de données et mettre à jour votre cible de données.

4. Choisissez Enregistrer la tâche et modifier le script pour afficher la page de l'éditeur de script.
5. Choisissez Exécuter la tâche pour démarrer l'exécution de la tâche.

Utilisation FindMatches dans le cadre d'un travail visuel

Pour utiliser la FindMatchestransformationAWS Glue Studio, vous pouvez utiliser le nœud Custom Transform qui appelle l' FindMatches API. Pour plus d'informations sur l'utilisation d'une transformation personnalisée, consultez Creating a custom transformation.

> **Note**
>
> Actuellement, l' FindMatches API ne fonctionne qu'avec`Glue 2.0`. Pour exécuter une tâche avec la transformation personnalisée qui appelle l' FindMatches API, assurez-vous que la AWS Glue version se trouve `Glue 2.0` dans l'onglet Détails de la tâche. Si la version de ne l'AWS Glueest pas`Glue 2.0`, la tâche échouera lors de l'exécution avec le message d'erreur suivant : « Impossible d'importer le nom « FindMatches » depuis « awsglueml.transform ».

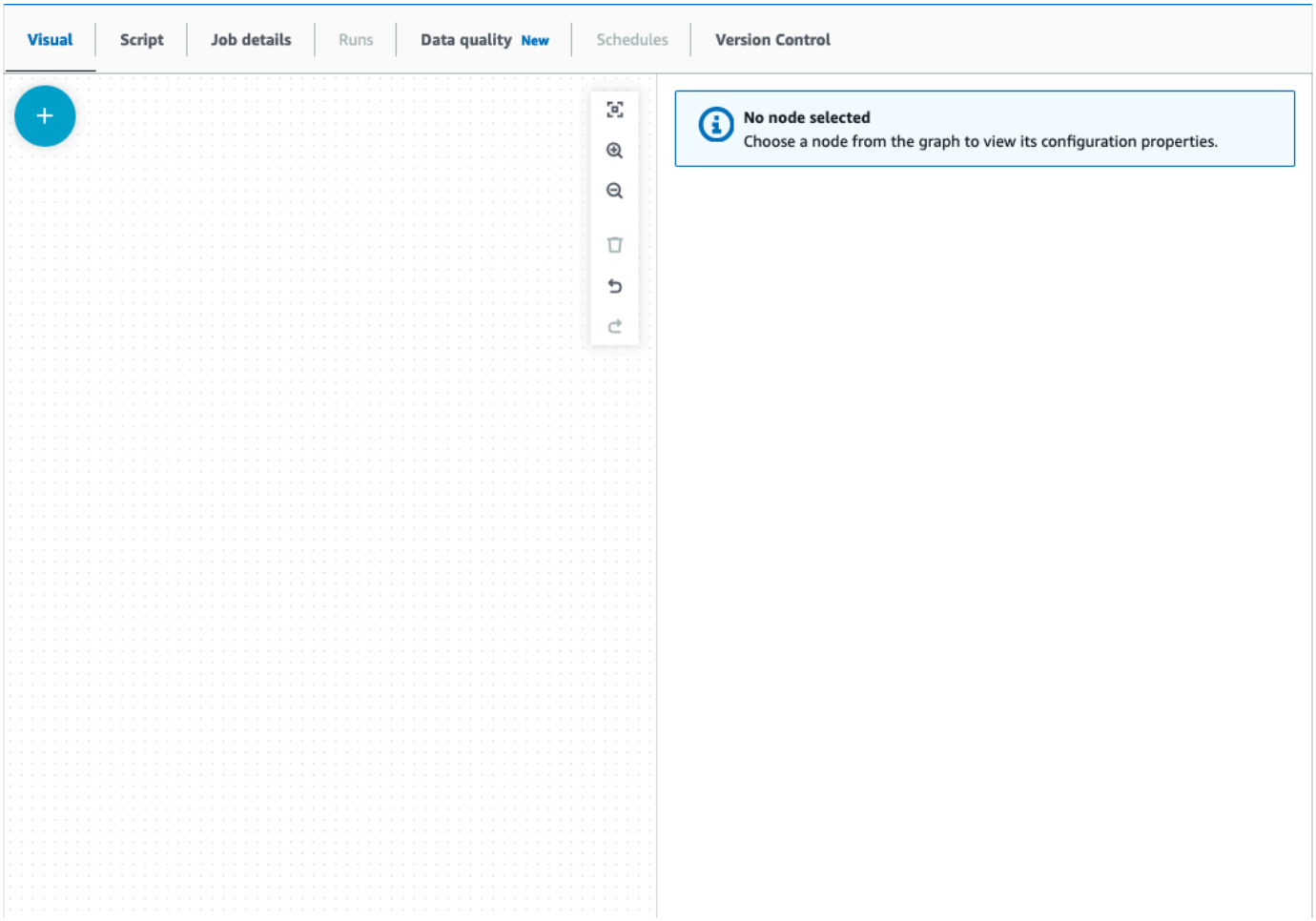
Conditions préalables

- Pour utiliser la transformation Trouver des correspondances, ouvrez la console AWS Glue Studio sur https://console.aws.amazon.com/gluestudio/.
- Créer une transformation de machine learning. Une fois créé, un transformId est généré. Vous aurez besoin de cet identifiant pour les étapes ci-dessous. Pour plus d'informations sur la création d'une transformation de machine learning, consultez Adding and editing machine learning transforms.

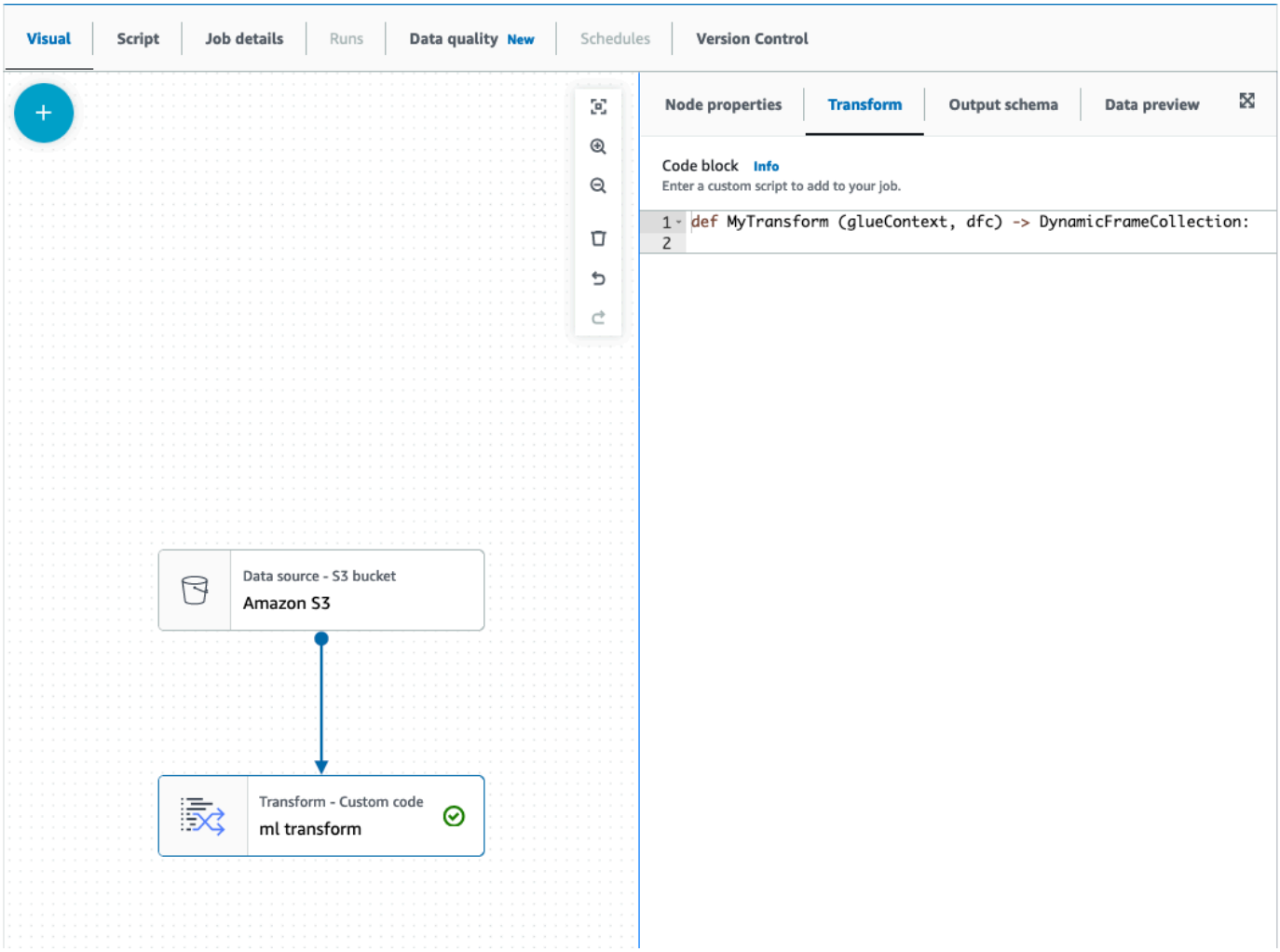
Ajouter une FindMatches transformation

Pour ajouter une FindMatches transformation :

1. Dans l'éditeur de tâches AWS Glue Studio, ouvrez le panneau Ressources en cliquant sur le symbole en forme de croix dans le coin supérieur gauche du graphique visuel des tâches et choisissez une source de données en choisissant l'onglet Données. Il s'agit de la source de données pour laquelle vous souhaitez vérifier les correspondances.



2. Choisissez le nœud de source de données, puis ouvrez le panneau Ressources en cliquant sur le symbole en forme de croix dans le coin supérieur gauche du graphique visuel des tâches et recherchez « transformation personnalisée ». Choisissez le nœud Transformation personnalisée pour l'ajouter au graphique. La Transformation personnalisée est liée au nœud de source de données. Si ce n'est pas le cas, vous pouvez cliquer sur le nœud Transformation personnalisée et choisir l'onglet Propriétés du nœud, puis sous Parents du nœud, choisissez la source de données.
3. Cliquez sur le nœud de Transformation personnalisée dans le graphique visuel, puis choisissez l'onglet Propriétés du nœud et nommez la transformation personnalisée. Il est recommandé de renommer la transformation afin que son nom soit facilement identifiable dans le graphique visuel.
4. Choisissez l'onglet Transformer, dans lequel vous pouvez modifier le bloc de code. C'est ici que le code permettant d'invoquer l' FindMatches API peut être ajouté.



Le bloc de code contient un code prérempli pour vous aider à démarrer. Remplacez le code prérempli par le modèle ci-dessous. Le modèle comporte un espace réservé pour le transformId, que vous pouvez indiquer.

```
def MyTransform (glueContext, dfc) -> DynamicFrameCollection:
    dynf = dfc.select(list(dfc.keys())[0])
    from awsglueml.transforms import FindMatches
    findmatches = FindMatches.apply(frame = dynf, transformId = "<your id>")
    return(DynamicFrameCollection({"FindMatches": findmatches}, glueContext))
```

5. Cliquez sur le nœud Transformation personnalisée dans le graphique visuel, puis ouvrez le panneau Ressources en cliquant sur le symbole en forme de croix dans le coin supérieur gauche du graphique visuel des tâches et recherchez « Collection Sélectionner depuis ». Il n'est pas

nécessaire de modifier la sélection par défaut puisqu'il n'y en a qu'une seule DynamicFrame dans la collection.

6. Vous pouvez continuer à ajouter des transformations ou stocker le résultat, qui est désormais enrichi par les colonnes supplémentaires de recherche de correspondances. Si vous souhaitez référencer ces nouvelles colonnes dans les transformations en aval, vous devez les ajouter au schéma de sortie de la transformation. Le moyen le plus simple est de choisir l'onglet Aperçu des données, puis de choisir « Utiliser le schéma d'aperçu des données » dans l'onglet Schéma.
7. Pour personnaliser FindMatches, vous pouvez ajouter des paramètres supplémentaires à transmettre à la méthode « appliquer ». Voir le FindMatches cours.

Ajouter une transformation FindMatches incrémentielle

Dans le cas de correspondances incrémentielles, le processus est identique à celui de l'ajout d'une FindMatches transformation, avec les différences suivantes :

- Au lieu d'un nœud parent pour la transformation personnalisée, vous avez besoin de deux nœuds parents.
- Le premier nœud parent doit être le jeu de données.
- Le deuxième nœud parent doit être le jeu de données incrémentiel.

  Remplacez le `transformId` par votre `transformId` dans le bloc de code du modèle :

```
def MyTransform (glueContext, dfc) -> DynamicFrameCollection:
    dfs = list(dfc.values())
    dynf = dfs[0]
    inc_dynf = dfs[1]
    from awsglueml.transforms import FindIncrementalMatches
    findmatches = FindIncrementalMatches.apply(existingFrame = dynf, incrementalFrame
 = inc_dynf,
                                    transformId = "<your id>")
    return(DynamicFrameCollection({"FindMatches": findmatches}, glueContext))
```

- Pour les paramètres facultatifs, voir FindIncrementalMatches class.

Migrer les programmes Apache Spark vers AWS Glue

Apache Spark est une plateforme open source pour les charges de travail informatiques distribuées et exécutées sur de vastes jeux de données. AWS Glue tire parti des capacités de Spark pour fournir une expérience optimisée pour l'ETL. Vous pouvez migrer des programmes Spark AWS Glue pour tirer parti de nos fonctionnalités. AWS Glue fournit les mêmes améliorations de performances que celles que vous attendez d'Apache Spark sur Amazon EMR.

Exécuter le code Spark

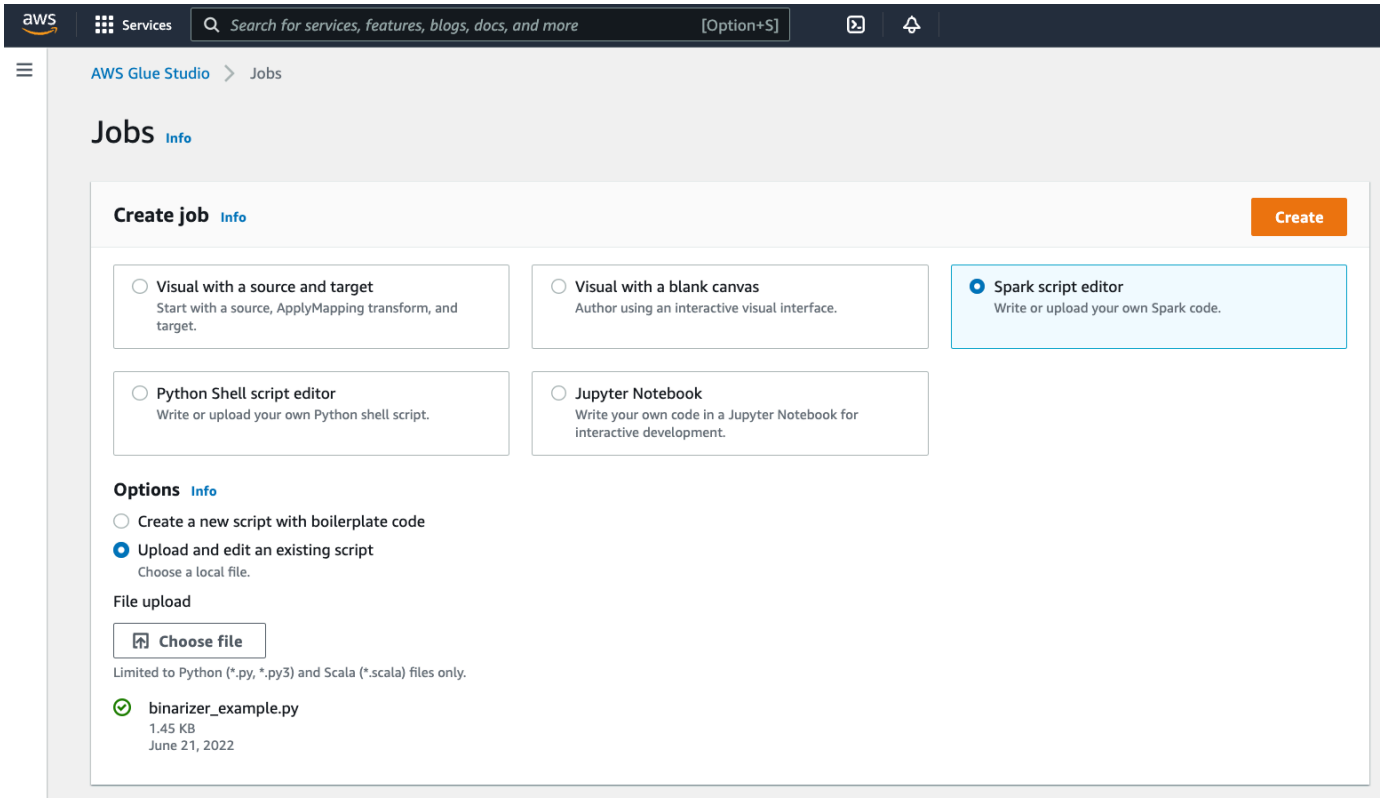
Le code Spark natif peut être exécuté dans un AWS Glue environnement prêt à l'emploi. Les scripts sont souvent développés en modifiant une partie de code de manière itérative, afin d'offrir un flux de travail adapté à une session interactive. Cependant, le code existant est plus adapté à une exécution dans le cadre d'une AWS Glue tâche, ce qui vous permet de planifier et d'obtenir régulièrement des journaux et des métriques pour chaque exécution de script. Vous pouvez télécharger et modifier un script existant en utilisant la console.

1. Obtenir la source de votre script. Pour besoin de cet exemple, vous utiliserez un exemple de script du référentiel Apache Spark. Exemple de binarizer
2. Dans la AWS Glue console, développez le volet de navigation de gauche et sélectionnez ETL > Jobs

   Dans le panneau Créer une tâche, sélectionnez Éditeur de script Spark. Une section Options apparaîtra. Dans cette section Options, sélectionnez Chargement et modification d'un script existant.

   Une section de Chargement du fichier apparaîtra. Dans cette section Chargement du fichier, cliquez sur Choisissez un fichier. Le sélecteur de fichiers de votre système s'affichera. Accédez à l'emplacement où vous avez enregistré `binarizer_example.py`, sélectionnez-le et confirmez votre choix.

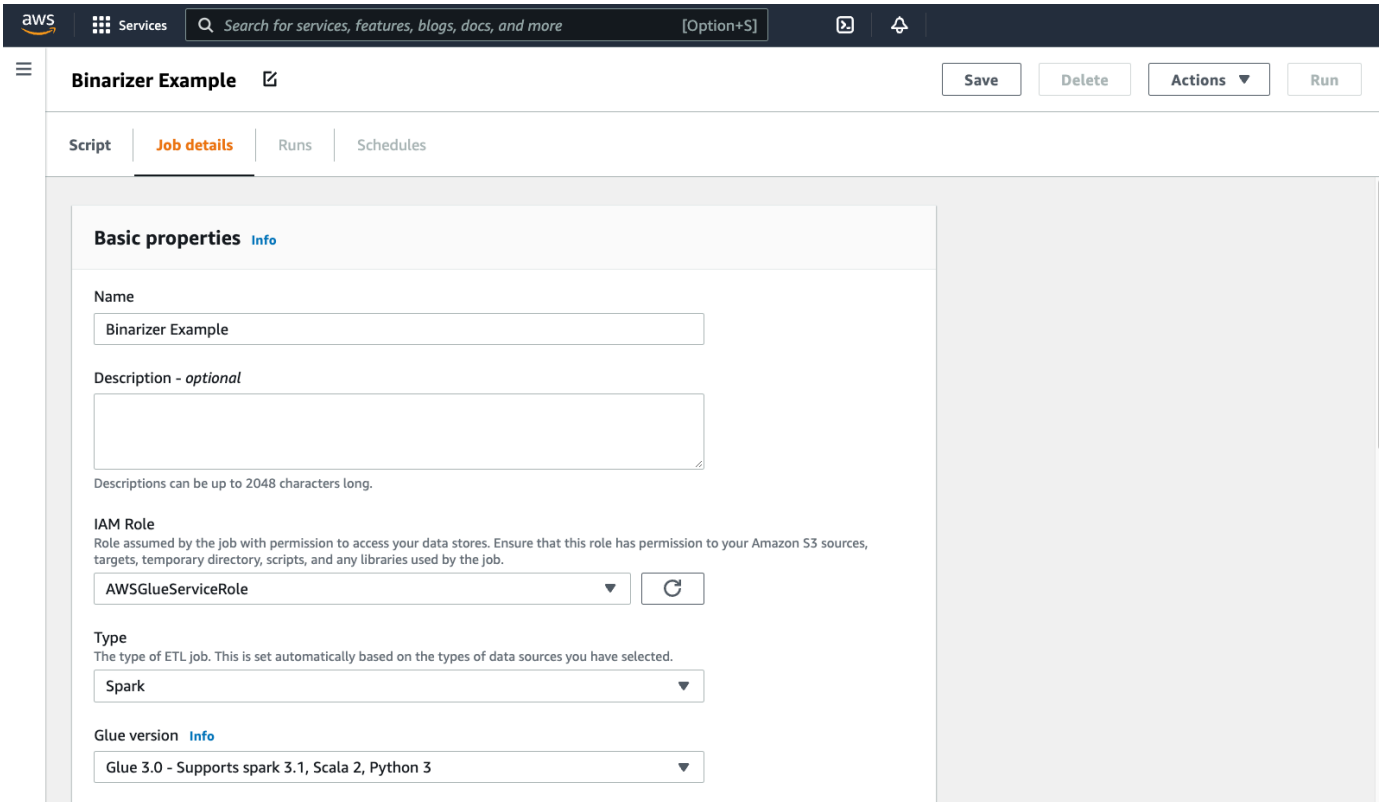
   Une touche Créer apparaîtra dans l'en-tête du panneau Créer une tâche. Cliquez sur cette touche.



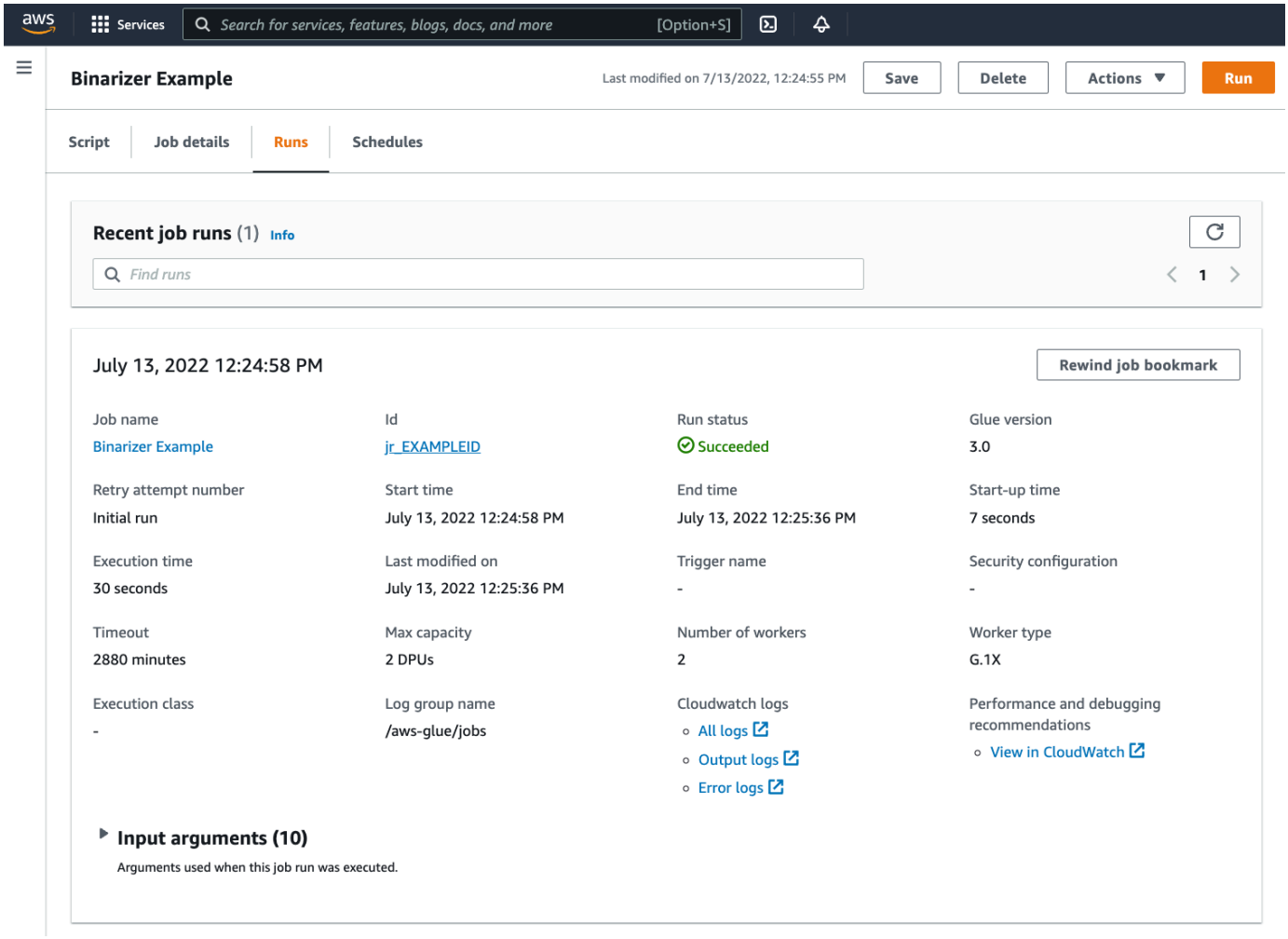
3. Votre navigateur accèdera à l'éditeur de script. Dans l'en-tête, cliquez sur l'onglet Détails de la tâche. Définissez le Nom et le Rôle IAM. Pour obtenir des conseils sur les rôles AWS Glue IAM, consultezthe section called "Configuration des autorisations IAM".

   Facultativement, définissez le Nombre de travailleurs à 2 etle Nombre de nouvelles tentatives à 1. Ces options sont utiles lors de l'exécution de tâches de production. Cependant, leur désactivation simplifiera votre expérience lors du test d'une fonctionnalité.

   Dans la barre de titre, cliquez surEnregistrer, puis sur Exécuter

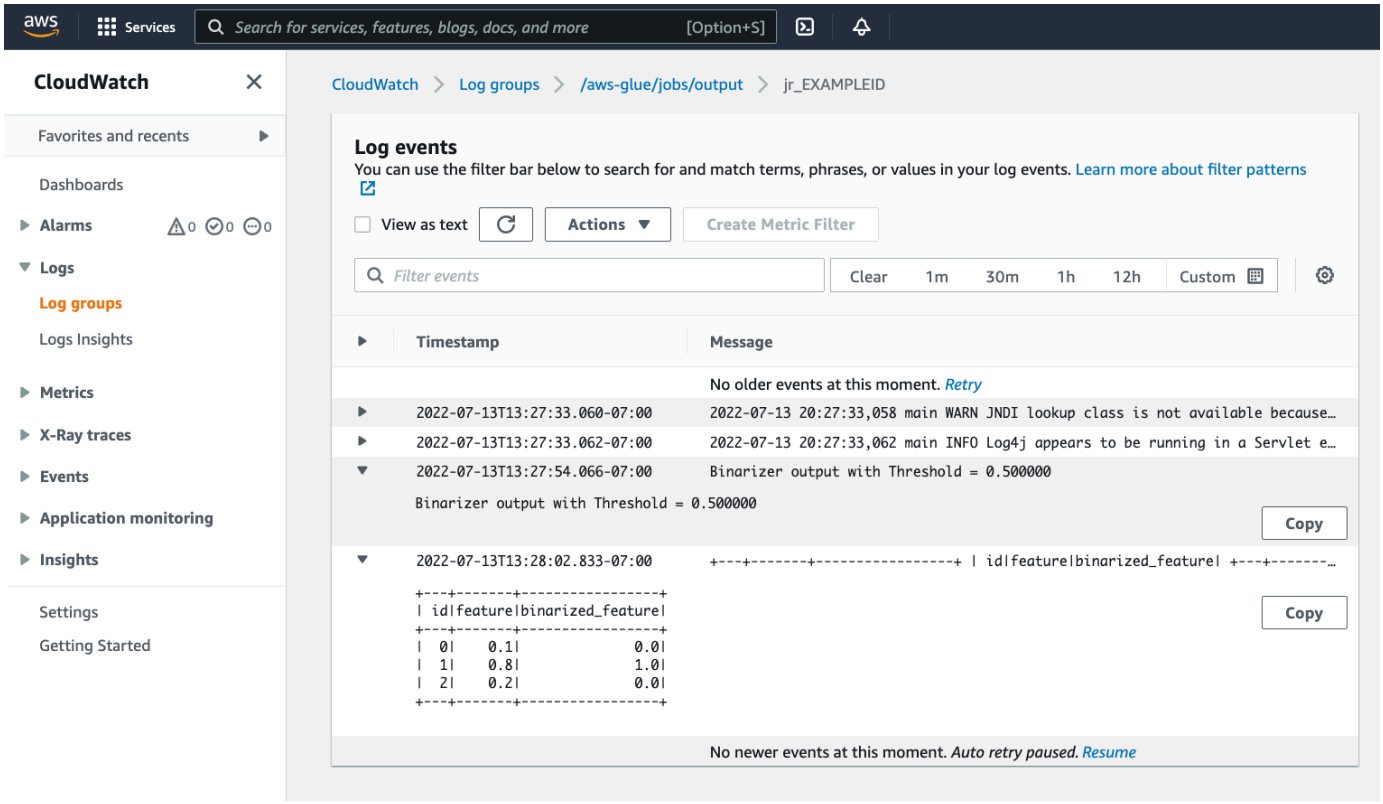


4. Accédez à l'onglet Exécutions. Vous verrez un panneau correspondant à l'exécution de votre tâche. Patientez quelques minutes, la page devrait s'actualiser automatiquement pour afficher Réussie comme Statut de l'exécution.



5. Si vous souhaitez examiner votre sortie pour vérifier que le script Spark s'est exécuté comme prévu. Cet échantillon de script Apache Spark doit écrire une chaîne dans le flux de sortie. Vous pouvez le trouver en accédant aux Journaux sous Cloudwatch logs dans le panneau, pour l'exécution de tâche réussie. Notez l'ID d'exécution de la tâche, un identifiant généré sous le modèle Id commençant par`jr_`.

   Cela ouvrira la CloudWatch console, configurée pour visualiser le contenu du groupe de AWS Glue journaux par défaut`/aws-glue/jobs/output`, filtré en fonction du contenu des flux de journaux pour l'identifiant d'exécution de la tâche. Chaque programme de travail aura généré un flux de journaux, affiché sous forme de lignes sous Flux de journaux. Un programme de travail aurait dû exécuter le code demandé. Vous devrez ouvrir tous les flux de journaux pour identifier le bon programme de travail. Une fois que vous avez trouvé le bon programme de travail, vous devriez voir la sortie du script, comme dans l'image suivante :



Procédures communes requises pour la migration des programmes Spark

Évaluation de la prise en charge de la version de Spark

AWS Glue les versions de version définissent la version d'Apache Spark et de Python disponible pour la AWS Glue tâche. Vous pouvez trouver nos AWS Glue versions et ce qu'elles prennent en charge à l'adresse suivantethe section called “AWS Glue versions”. Vous devrez peut-être mettre à jour votre programme Spark pour qu'il soit compatible avec une version plus récente de Spark afin d'accéder à certaines fonctionnalités AWS Glue .

Intégration des bibliothèques tierces

De nombreux programmes Spark existants dépendront à la fois d'artefacts privés et publics. AWS Glue prend en charge les dépendances de style JAR pour les tâches Scala, ainsi que les dépendances Wheel et source Pure-Python pour les tâches Python.

Python - Pour plus d'informations sur les dépendances Python, consultez the section called “Bibliothèques Python”

Les dépendances Python courantes sont fournies dans l' AWS Glue environnement, y compris la bibliothèque Pandas couramment demandée. Les dépendances sont incluses dans la version 2.0 et

les versions ultérieures de AWS Glue . Pour plus d'informations sur les modules fournis, consultez the section called "Modules Python déjà fournis dans AWS Glue". Si vous devez fournir à une tâche une version différente d'une dépendance incluse par défaut, vous pouvez utiliser `--additional-python-modules`. Pour plus d'informations sur ces arguments, consultez the section called "Paramètres du job".

Vous pouvez fournir des dépendances Python supplémentaires avec l'argument de tâche `--extra-py-files`. Si vous migrez une tâche depuis un programme Spark, ce paramètre est une bonne option car il est fonctionnellement équivalent au `--py-files` flag in PySpark, et il est soumis aux mêmes limites. Pour plus d'informations sur le paramètre `--extra-py-files`, consultez the section called "Y compris des fichiers Python dotés de fonctionnalités PySpark natives"

Pour les nouvelles tâches, vous pouvez gérer les dépendances Python avec l'argument de tâche `--additional-python-modules`. L'utilisation de cet argument offre une expérience plus profonde de la gestion des dépendances. Ce paramètre prend en charge les dépendances de style Wheel, y compris celles comprenant des liaisons de codes natifs compatibles avec Amazon Linux 2.

Scala

Vous pouvez fournir des dépendances Scala supplémentaires avec l'Argument de tâche `--extra-jars`. Les dépendances doivent être hébergées dans Amazon S3, et la valeur de l'argument doit être une liste de chemins non-espacés Amazon S3 délimités par des virgules. Vous trouverez peut-être plus facile de gérer votre configuration en regroupant vos dépendances avant de les héberger et de les configurer. AWS Glue Les dépendances JAR contiennent du bytecode Java, qui peut être généré à partir de n'importe quel langage JVM. Il vous est possible d'utiliser d'autres langages JVM, tels que Java, pour écrire des dépendances personnalisées.

Gestion des informations d'identification de la source de données

Les programmes Spark existants peuvent être fournis avec une configuration complexe ou personnalisée pour l'extraction des données de leurs sources de données. Les flux d'authentification des sources de données courants sont pris en charge par AWS Glue les connexions. Pour plus d'informations sur les connexions AWS Glue , consultez Connexion aux données.

AWS Glue les connexions facilitent la connexion de votre Job à divers types de magasins de données de deux manières principales : via des appels de méthode vers nos bibliothèques et en configurant la connexion réseau supplémentaire dans la AWS console. Vous pouvez également appeler le AWS SDK depuis votre tâche pour récupérer des informations à partir d'une connexion.

Appels de méthode : les AWS Glue connexions sont étroitement intégrées au catalogue de AWS Glue données, un service qui vous permet de gérer les informations relatives à vos ensembles de données, et les méthodes disponibles pour interagir avec AWS Glue les connexions en tiennent compte. Si vous avez une configuration d'authentification existante que vous souhaitez réutiliser, pour les connexions JDBC, vous pouvez accéder à votre configuration de AWS Glue connexion via la `extract_jdbc_conf` méthode indiquée sur le. `GlueContext` Pour de plus amples informations, consultez the section called "extract_jdbc_conf".

Configuration de la console : les AWS Glue tâches utilisent AWS Glue les connexions associées pour configurer les connexions aux sous-réseaux Amazon VPC. Si vous gérez directement votre matériel de sécurité, vous devrez peut-être fournir un `NETWORK` type de connexion réseau supplémentaire dans la AWS console pour configurer le routage. Pour plus d'informations sur la connexion API AWS Glue , consultez the section called "Connexions"

Si vos programmes Spark disposent d'un flux d'authentification personnalisé ou peu commun, vous devrez peut-être gérer vos supports de sécurité manuellement. Si AWS Glue les connexions ne vous conviennent pas, vous pouvez héberger en toute sécurité les documents de sécurité dans Secrets Manager et y accéder via le boto3 ou le AWS SDK, fournis dans le cadre de la tâche.

Configuration d'Apache Spark

Les migrations complexes modifient souvent la configuration de Spark pour s'adapter à leurs charges de travail. Les versions modernes d'Apache Spark permettent de définir la configuration d'exécution à l'aide du`SparkSession`. AWS Glue Les jobs 3.0 et versions ultérieures sont fournis`SparkSession`, qui peuvent être modifiés pour définir la configuration d'exécution. Configuration d'Apache Spark. Le réglage de Spark est complexe et AWS Glue ne garantit pas la prise en charge de la configuration de toutes les configurations de Spark. Si votre migration nécessite une configuration importante au niveau de Spark, contactez l'assistance.

Définition de la configuration

Les programmes Spark migrés peuvent être conçus pour adopter une configuration personnalisée. AWS Glue permet de définir la configuration au niveau de la tâche et de son exécution, via les arguments de la tâche. Pour plus d'informations sur ces arguments, consultez the section called "Paramètres du job". Vous pouvez accéder aux arguments de tâche dans le contexte d'une tâche par le biais de nos bibliothèques. AWS Glue fournit une fonction utilitaire fournissant une vue cohérente entre les arguments définis lors de la tâche et ceux définis lors de l'exécution de la tâche. Voir the section called "getResolvedOptions" dans Python et the section called "GlueArgParser" dans Scala.

Migration du code Java

Comme expliqué dans the section called "Bibliothèques tierces", vos dépendances peuvent contenir des classes générées par les langages JVM, tels que Java ou Scala. Vos dépendances peuvent inclure une Méthode`main`. Vous pouvez utiliser une `main` méthode dans une dépendance comme point d'entrée pour une tâche AWS Glue Scala. Cela vous permet de rédiger votre méthode `main` en langage Java, ou réutilisez une méthode `main` empaquetée selon les normes de votre propre bibliothèque.

Pour utiliser une méthode `main` à partir d'une dépendance, effectuez les opérations suivantes : Effacez le contenu du volet d'édition en fournissant la valeur par défaut de l'objet `GlueApp`. Fournissez le nom complet d'une classe dans une dépendance en tant qu'argument de travail avec la clé `--class`. Vous devez ensuite être en mesure de déclencher une exécution de la tâche.

Vous ne pouvez pas configurer l'ordre ou la structure des AWS Glue arguments transmis à la `main` méthode. Si votre code existant doit lire la configuration définie AWS Glue, cela entraînera probablement une incompatibilité avec le code précédent. Si vous utilisez `getResolvedOptions`, vous ne disposerez pas d'un espace approprié pour appeler cette méthode. Envisagez d'invoquer votre dépendance directement, à partir d'une méthode principale générée par AWS Glue. Le script AWS Glue ETL suivant en montre un exemple.

```
import com.amazonaws.services.glue.util.GlueArgParser

object GlueApp {
  def main(sysArgs: Array[String]) {
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)

    // Invoke static method from JAR. Pass some sample arguments as a String[], one
 defined inline and one taken from the job arguments, using getResolvedOptions
    com.mycompany.myproject.MyClass.myStaticPublicMethod(Array("string parameter1",
 args("JOB_NAME")))

    // Alternatively, invoke a non-static public method.
    (new com.mycompany.myproject.MyClass).someMethod()
  }
}
```

Travailler avec Ray Jobs dans AWS Glue

> **Important**
>
> AWS Glue for Ray ne sera plus ouvert aux nouveaux clients à compter du 30 avril 2026. Si vous souhaitez utiliser AWS Glue for Ray, inscrivez-vous avant cette date. Les clients existants peuvent continuer à utiliser le service normalement. Pour des fonctionnalités similaires à celles de AWS Glue for Ray, explorez Amazon EKS. Pour plus d'informations, consultez la section AWS Glue relative à la fin du support de Ray.

Cette section fournit des informations sur l'utilisation AWS Glue des tâches For Ray. Pour plus d'informations sur l'écriture de scripts AWS Glue pour Ray, consultez la the section called "AWS Glue pour Ray" section.

Rubriques

- Commencer avec AWS Glue For Ray
- Environnements d'exécution Ray pris en charge
- Comptabilité pour les travailleurs dans les tâches Ray
- Utilisation des paramètres de tâche dans les tâches Ray
- Surveiller les tâches Ray à l'aide de métriques

Commencer avec AWS Glue For Ray

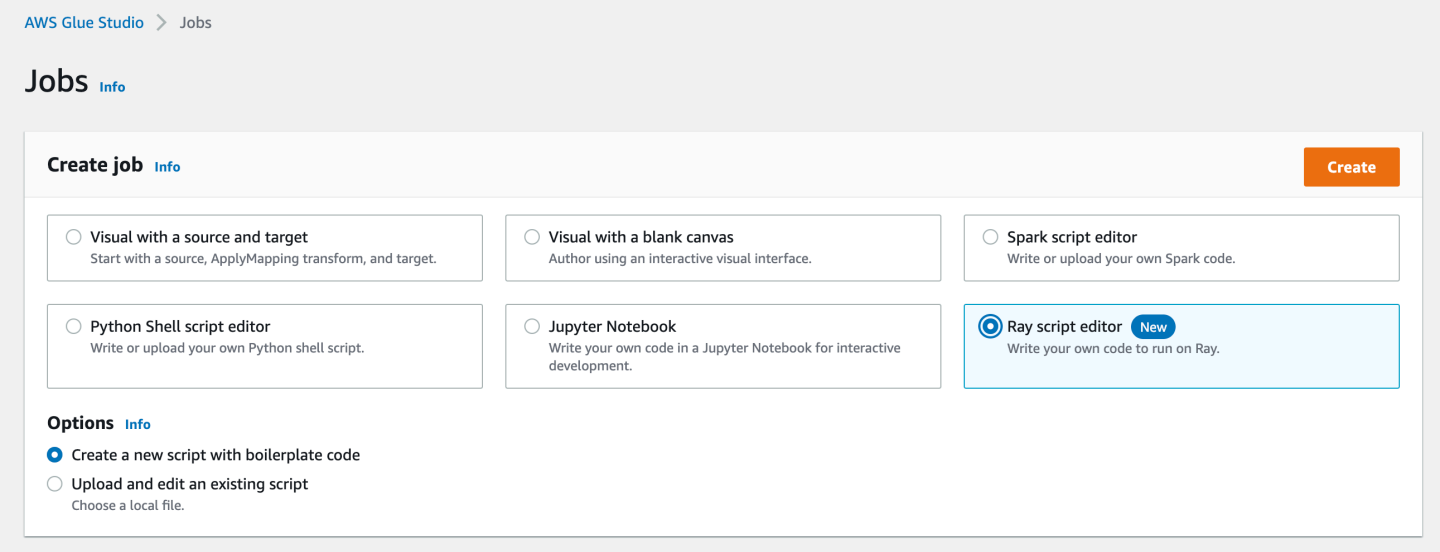
Pour travailler avec AWS Glue for Ray, vous utilisez les mêmes AWS Glue tâches et sessions interactives que AWS Glue pour Spark. AWS Glue les tâches sont conçues pour exécuter le même script à une cadence récurrente, tandis que les sessions interactives sont conçues pour vous permettre d'exécuter des extraits de code de manière séquentielle sur les mêmes ressources provisionnées.

AWS Glue ETL et Ray sont différents en dessous, donc dans votre script, vous avez accès à différents outils, fonctionnalités et configurations. En tant que nouveau framework de calcul géré par AWS Glue, Ray possède une architecture différente et utilise un vocabulaire différent pour décrire ce qu'il fait. Pour plus d'informations, consultez les livres blancs sur l'architecture de la documentation Ray.

> **Note**
>
> AWS Glue for Ray est disponible dans l'est des États-Unis (Virginie du Nord), dans l'est des États-Unis (Ohio), dans l'ouest des États-Unis (Oregon), en Asie-Pacifique (Tokyo) et en Europe (Irlande).

Ray Jobs dans la AWS Glue Studio console

Sur la page Tâches de la AWS Glue Studio console, vous pouvez sélectionner une nouvelle option lorsque vous créez une tâche dans AWS Glue Studio l'éditeur de script Ray. Choisissez cette option pour créer une tâche Ray sur la console. Pour plus d'informations sur ces tâches et leur utilisation, consultez Création de tâches ETL visuelles.



Offres d'emploi Ray dans le SDK AWS CLI et

Les tâches Ray AWS CLI utilisent les mêmes actions et paramètres du SDK que les autres tâches. AWS Glue for Ray introduit de nouvelles valeurs pour certains paramètres. Pour plus d'informations sur l'API Tâches, consultez the section called “Tâches”.

Environnements d'exécution Ray pris en charge

Dans les tâches Spark, `GlueVersion` détermine les versions d'Apache Spark et de Python disponibles dans une tâche AWS Glue pour Spark. La version de Python indique la version qui est prise en charge pour les tâches de type Spark. Les environnements d'exécution Ray ne sont pas configurés de cette manière.

Pour les tâches Ray vous devez définir `GlueVersion` sur `4.0` ou supérieur. Toutefois, les versions de Ray, de Python et des bibliothèques supplémentaires disponibles dans votre tâche Ray sont déterminées par le champ `Runtime` de la définition de la tâche.

L'environnement d'exécution `Ray2.4` sera disponible pendant au moins six mois après sa sortie. Au fur et à mesure de l'évolution rapide de Ray, vous pourrez intégrer les mises à jour et les améliorations de Ray dans les futures versions de l'environnement d'exécution.

Valeurs valides : `Ray2.4`

Valeur d'exécution	Versions Ray et Python
`Ray2.4` (pour la version 4.0 et ultérieure de AWS Glue)	Ray 2.4.0 Python 3.9

Informations supplémentaires

- Pour les notes de publication qui accompagnent AWS Glue les versions de Ray, voirthe section called “AWS Glue versions”.
- Pour les bibliothèques Python fournies dans un environnement d'exécution, consultez the section called “Modules fournis avec les tâches Ray”.

Comptabilité pour les travailleurs dans les tâches Ray

AWS Glue exécute des tâches Ray sur de nouveaux types de travailleurs EC2 basés sur Graviton, qui ne sont disponibles que pour les tâches Ray. Afin de bien approvisionner ces travailleurs pour les charges de travail pour lesquelles Ray est conçu, nous fournissons un rapport différent entre les ressources de calcul et les ressources de mémoire de la plupart des travailleurs. Afin de tenir compte de ces ressources, nous utilisons l'unité de traitement des données à mémoire optimisée (M-DPU) plutôt que l'unité de traitement de données standard (DPU).

- Un M-DPU correspond à 4 V CPUs et à 32 Go de mémoire.
- Un DPU correspond à 4 V CPUs et 16 Go de mémoire. DPUs sont utilisés pour prendre en compte les ressources dans AWS Glue les tâches Spark et les travailleurs correspondants.

Les tâches Ray ont actuellement accès à un type de travailleur : `Z.2X`. Le `Z.2X` worker correspond à 2 M- DPUs (8 VCPUs, 64 Go de mémoire) et dispose de 128 Go d'espace disque. Une machine `Z.2X` fournit huit travailleurs Ray (un par vCPU).

Le nombre de M- DPUs que vous pouvez utiliser simultanément dans un compte est soumis à un quota de service. Pour plus d'informations sur les limites de votre AWS Glue compte, consultez la section AWS Glue Points de terminaison et quotas.

Vous indiquez le nombre de composants master disponibles pour une tâche Ray avec `--number-of-workers (NumberOfWorkers)` dans la définition de la tâche. Pour plus d'informations sur les valeurs Ray dans l'API Tâches, consultez the section called “Tâches”.

Vous pouvez également indiquer le nombre minimum de travailleurs qu'une tâche Ray doit allouer avec le paramètre de tâche `--min-workers`. Pour de plus amples informations sur la définition des paramètres de la tâche, consultez the section called “Référence”.

Utilisation des paramètres de tâche dans les tâches Ray

> **Important**
>
> AWS Glue for Ray ne sera plus ouvert aux nouveaux clients à compter du 30 avril 2026. Si vous souhaitez utiliser AWS Glue for Ray, inscrivez-vous avant cette date. Les clients existants peuvent continuer à utiliser le service normalement. Pour des fonctionnalités similaires à celles de AWS Glue for Ray, explorez Amazon EKS. Pour plus d'informations, consultez la section AWS Glue relative à la fin du support de Ray.

Vous définissez les arguments pour les tâches AWS Glue Ray de la même manière que AWS Glue pour les tâches Spark. Pour plus d'informations sur l' AWS Glue API, consultezthe section called “Tâches”. Vous pouvez configurer les tâches AWS Glue Ray avec différents arguments, qui sont répertoriés dans cette référence. Vous pouvez également fournir vos propres arguments.

Vous pouvez configurer une tâche via la console, dans l'onglet Job details (Détails de la tâche), sous Job Parameters (Paramètres de la tâche). Vous pouvez également configurer une tâche AWS CLI par le biais du paramètre `DefaultArguments` sur une tâche ou `Arguments` sur une exécution de tâche. Les arguments par défaut et les paramètres de la tâche restent associés à la tâche au fil des exécutions.

Par exemple, ce qui suit est la syntaxe pour exécuter une tâche en utilisant `--arguments` pour définir un paramètre spécial.

```
$ aws glue start-job-run --job-name "CSV to CSV" --arguments='--scriptLocation="s3://
my_glue/libraries/test_lib.py",--test-environment="true"'
```

Après avoir défini les arguments, vous pouvez accéder aux paramètres de la tâche depuis votre tâche Ray via des variables d'environnement. Vous pouvez ainsi configurer votre tâche pour chaque exécution. Le nom de la variable d'environnement correspond au nom de l'argument de la tâche sans le préfixe `--`.

Par exemple, dans l'exemple précédent, les noms des variables sont `scriptLocation` et `test-environment`. Vous pouvez ensuite récupérer l'argument via les méthodes disponibles dans la bibliothèque standard : `test_environment = os.environ.get('test-environment')`. Pour plus d'informations sur l'accès aux variables d'environnement avec Python, consultez le module os dans la documentation Python.

Configuration de la façon dont les tâches Ray génèrent des journaux

Par défaut, les tâches Ray génèrent des journaux et des métriques qui sont envoyés à CloudWatch Amazon S3. Vous pouvez utiliser le paramètre `--logging_configuration` pour modifier la façon dont les journaux sont générés. Vous pouvez actuellement l'utiliser pour empêcher les tâches Ray de générer différents types de journaux. Ce paramètre prend un objet JSON dont les clés correspondent à celles que logs/behaviors vous souhaitez modifier. Il prend en charge les clés suivantes :

- `CLOUDWATCH_METRICS`— Configure des séries de CloudWatch métriques qui peuvent être utilisées pour visualiser l'état de santé au travail. Pour plus d'informations sur les métriques, consultez la section the section called "Métriques des tâches Ray".
- `CLOUDWATCH_LOGS`— Configure les CloudWatch journaux qui fournissent des informations détaillées au niveau de l'application Ray sur l'état d'exécution de la tâche. Pour de plus amples informations sur les journaux, veuillez consulter the section called "Dépannage des erreurs Ray".
- `S3`— Configure ce qui est AWS Glue écrit sur Amazon S3, principalement des informations similaires aux CloudWatch journaux, mais sous forme de fichiers plutôt que de flux de journaux.

Pour désactiver un comportement de journalisation de Ray, indiquez la valeur `{\"IS_ENABLED\": \"False\"}`. Par exemple, pour désactiver CloudWatch les métriques et CloudWatch les journaux, fournissez la configuration suivante :

```
"--logging_configuration": "{\"CLOUDWATCH_METRICS\": {\"IS_ENABLED\": \"False\"},
 \"CLOUDWATCH_LOGS\": {\"IS_ENABLED\": \"False\"}}"
```

Référence

Les tâches Ray reconnaissent les noms d'arguments suivants que vous pouvez utiliser afin de configurer l'environnement de script pour vos tâches Ray et vos exécutions de tâche :

- `--logging_configuration` : utilisé pour arrêter la génération de divers journaux créés par les tâches Ray. Ces journaux sont générés par défaut pour toutes les tâches Ray. Format : objet JSON en séquence d'échappement. Pour de plus amples informations, veuillez consulter the section called "Configuration de la façon dont les tâches Ray génèrent des journaux".
- `--min-workers` : nombre minimum de composants master alloués à une tâche Ray. Un composant master peut exécuter plusieurs réplicas, un par processeur virtuel. Format : nombre entier. Minimum : 0. Maximum : valeur spécifiée dans `--number-of-workers` (`NumberOfWorkers`), dans la définition de la tâche. Pour plus d'informations sur la comptabilisation des composants master, consultez the section called "Comptabilité pour les travailleurs dans les tâches Ray".
- `--object_spilling_config`— AWS Glue for Ray prend en charge l'utilisation d'Amazon S3 pour étendre l'espace disponible dans le magasin d'objets de Ray. Pour activer ce comportement, vous pouvez fournir à Ray un objet de configuration JSON pour le déversement d'objets avec ce paramètre. Pour plus d'informations sur la configuration du déversement d'objets dans Ray, consulter Object Spilling dans la documentation Ray. Format : objet JSON.

  AWS Glue for Ray ne prend en charge que le déversement sur le disque ou le déversement sur Amazon S3 en une seule fois. Vous pouvez indiquer plusieurs emplacements pour le déversement, à condition qu'ils respectent cette limite. Lors du déversement sur Amazon S3, vous devez également ajouter des autorisations IAM à votre tâche pour ce compartiment.

  Lorsque vous fournissez un objet JSON en tant que configuration avec la CLI, vous devez le fournir sous forme de chaîne, avec la chaîne de l'objet JSON échappée. Par exemple, une valeur de chaîne pour le déversement vers un chemin Amazon S3 ressemblerait à ce qui suit : `"{\"type\": \"smart_open\", \"params\": {\"uri\":\"s3path\"}}"`. Dans AWS Glue Studio, fournissez ce paramètre sous forme d'objet JSON sans mise en forme supplémentaire.
- `--object_store_memory_head` : mémoire allouée au magasin d'objets Plasma sur le nœud principal Ray. Cette instance exécute des services de gestion de clusters, ainsi que des réplicas de travail. La valeur représente un pourcentage de mémoire disponible sur l'instance après un

démarrage à chaud. Vous utilisez ce paramètre pour régler les charges de travail gourmandes en mémoire. Les valeurs par défaut sont acceptables pour la plupart des cas d'utilisation. Format : entier positif. Minimum : 1. Maximum : 100.

Pour plus d'informations sur Plasma, consultez The Plasma In-Memory Object Store (Le magasin d'objets en mémoire Plasma) dans la documentation Ray.

- `--object_store_memory_worker` : mémoire allouée au magasin d'objets Plasma sur les composants master Ray. Ces instances exécutent uniquement des réplicas de travail. La valeur représente un pourcentage de mémoire disponible sur l'instance après un démarrage à chaud. Ce paramètre est utilisé pour régler les charges de travail gourmandes en mémoire. Les valeurs par défaut sont acceptables pour la plupart des cas d'utilisation. Format : entier positif. Minimum : 1. Maximum : 100.

  Pour plus d'informations sur Plasma, consultez The Plasma In-Memory Object Store (Le magasin d'objets en mémoire Plasma) dans la documentation Ray.

- `--pip-install` : ensemble de packages Python à installer. Vous pouvez installer des packages depuis PyPI en utilisant cet argument. Format : liste délimitée par des virgules.

  Une entrée de package PyPI est au format `package==version`, avec le nom PyPI et la version de votre package cible. Les entrées utilisent la correspondance entre versions de Python pour faire correspondre le package et la version, comme == et non le signe = seul. Il existe d'autres opérateurs de mise en correspondance des versions. Pour plus d'informations, consultez PEP 440 sur le site Internet de Python. Vous pouvez également fournir des modules personnalisés avec `--s3-py-modules`.

- `--s3-py-modules` : un ensemble de chemins Amazon S3 qui hébergent des distributions de modules Python. Format : liste délimitée par des virgules.

  Vous pouvez l'utiliser pour distribuer vos propres modules à votre tâche Ray. Vous pouvez également fournir des modules à partir de PyPI avec `--pip-install`. Contrairement à l' AWS Glue ETL, les modules personnalisés ne sont pas configurés via pip, mais sont transmis à Ray pour distribution. Pour de plus amples informations, veuillez consulter the section called “Modules Python supplémentaires pour les tâches Ray”.

- `--working-dir` : un chemin d'accès à un fichier .zip hébergé sur Amazon S3 contenant les fichiers à distribuer à tous les nœuds exécutant votre tâche Ray. Format : chaîne. Pour de plus amples informations, veuillez consulter the section called “Fournir des fichiers à votre tâche Ray”.

Surveiller les tâches Ray à l'aide de métriques

> **Important**
>
> AWS Glue for Ray ne sera plus ouvert aux nouveaux clients à compter du 30 avril 2026. Si vous souhaitez utiliser AWS Glue for Ray, inscrivez-vous avant cette date. Les clients existants peuvent continuer à utiliser le service normalement. Pour des fonctionnalités similaires à celles de AWS Glue for Ray, explorez Amazon EKS. Pour plus d'informations, consultez la section AWS Glue relative à la fin du support de Ray.

Vous pouvez suivre les jobs de Ray à l'aide AWS Glue Studio d'Amazon CloudWatch. CloudWatch collecte et traite les métriques brutes issues AWS Glue de Ray, ce qui les rend disponibles pour analyse. Ces mesures sont visualisées dans la AWS Glue Studio console, afin que vous puissiez surveiller votre tâche pendant son exécution.

Pour une présentation générale du mode de surveillance AWS Glue, voirthe section called “Utilisation de CloudWatch métriques”. Pour un aperçu général de l'utilisation CloudWatch des métriques publiées par AWS Glue, voirthe section called “Surveillance avec CloudWatch”.

Surveillance des tâches de Ray dans la AWS Glue console

Sur la page de détails d'une exécution de tâche, sous la section Détails de l'exécution, vous pouvez consulter des graphiques agrégés prédéfinis qui visualisent les indicateurs de vos tâches disponibles. AWS Glue Studio envoie des métriques de tâche CloudWatch pour chaque exécution de tâche. Vous pouvez ainsi créer un profil de votre cluster et de vos tâches, ainsi qu'accéder à des informations détaillées sur chaque nœud.

Pour plus d'informations sur les graphiques de métriques disponibles, consultez the section called “Afficher Amazon CloudWatch les statistiques relatives à l'exécution d'une tâche Ray”.

Vue d'ensemble des indicateurs de Ray Jobs dans CloudWatch

Nous publions les métriques Ray lorsque la surveillance détaillée est activée dans CloudWatch. Les métriques sont publiées dans l'espace de `Glue/Ray` CloudWatch noms.

- Métriques des instances

  Nous publions des métriques sur l'utilisation du processeur, de la mémoire et du disque des instances affectées à une tâche. Ces métriques sont identifiées par des fonctionnalités telles

que `ExecutorId`, `ExecutorType` et `host`. Ces métriques constituent un sous-ensemble des métriques standard des CloudWatch agents Linux. Vous trouverez des informations sur les noms et les fonctionnalités des métriques dans la CloudWatch documentation. Pour plus d'informations, consultez la section Mesures collectées par l' CloudWatch agent.

- Métriques du cluster Ray

  Nous transmettons les métriques des processus Ray qui exécutent votre script à cet espace de noms, puis nous vous fournissons celles qui sont les plus essentielles pour vous. Les mesures disponibles peuvent varier selon la version de Ray. Pour plus d'informations sur la version de Ray exécutée par votre tâche, consultez the section called "AWS Glue versions".

  Ray collecte des métriques au niveau de l'instance. Il fournit également des métriques pour les tâches et le cluster. Pour plus d'informations sur la stratégie métrique sous-jacente de Ray, consultez Metrics dans la documentation Ray.

> **Note**
>
> Nous ne publions pas les métriques Ray dans l'espace de noms `Glue/Job Metrics/`, qui n'est utilisé que pour les tâches AWS Glue ETL.

Configuration des propriétés pour les tâches shell Python dans AWS Glue

Vous pouvez utiliser une tâche shell Python pour exécuter des scripts Python en tant que shell dans AWS Glue. Avec une tâche shell Python, vous pouvez exécuter des scripts compatibles avec Python 3.6 ou Python 3.9.

> **Note**
>
> La prise en charge de Pyshell 3.6 prendra fin le 1er mars 2026. Pour migrer vos charges de travail, voir Migrer depuis des tâches shell AWS Glue Python. Si vous souhaitez continuer avec Python shell 3.9, consultez Migration du shell Python 3.6 vers le shell Python 3.9.

Rubriques

- Limitations

- Environnement d'exécution
- Définition des propriétés pour les tâches shell Python
- Bibliothèques prises en charge pour les tâches shell Python
- Ajout de votre propre bibliothèque Python
- À utiliser AWS CloudFormation avec les jobs shell Python dans AWS Glue
- Migration du shell Python 3.6 vers le shell Python 3.9
- Migrer depuis les jobs shell AWS Glue Python

Limitations

Notez les limitations suivantes des tâches shell Python :

- Vous ne pouvez pas utiliser les signets de tâche avec les tâches shell Python.
- Vous ne pouvez pas conditionner les bibliothèques Python sous forme de fichiers `.egg` dans la version 3.9 de Python et les versions ultérieures. Utilisez à la place `.whl`.
- L'option `--extra-files` ne peut pas être utilisée en raison de la limitation des copies temporaires des données S3.

Environnement d'exécution

Les tâches shell Python s'exécutent dans un environnement d'exécution géré qui fournit un accès au stockage local pour le traitement temporaire des données :

Stockage temporaire local

Le `/tmp` répertoire est disponible pour un stockage temporaire pendant l'exécution de la tâche. Ce répertoire fournit environ 14 GiB d'espace libre que vous pouvez utiliser pour :

- Traitement de fichiers temporaires
- Stockage de données intermédiaire
- Mise en cache de petits ensembles de données

> **Note**
>
> Le `/tmp` répertoire est éphémère et est nettoyé une fois le travail terminé. Ne l'utilisez pas pour le stockage permanent de données importantes.

Définition des propriétés pour les tâches shell Python

Ces sections décrivent la définition des propriétés des tâches dans AWS Glue Studio ou à l'aide de la AWS CLI.

AWS Glue Studio

Lorsque vous définissez votre tâche Python shell dans AWS Glue Studio, vous indiquez les propriétés suivantes :

Rôle IAM

Spécifiez le rôle Gestion des identités et des accès AWS (IAM) utilisé pour autoriser les ressources utilisées pour exécuter le travail et accéder aux magasins de données. Pour plus d'informations sur les autorisations requises pour exécuter des tâches dans AWS Glue, consultez Gestion des identités et des accès pour AWS Glue.

Type

Choisissez Python shell (shell Python) pour exécuter un script Python avec la commande de tâche nommée `pythonshell`.

Version de Python

Choisissez la version Python. La valeur par défaut est Python 3.9. Les versions valides sont Python 3.6 et Python 3.9.

Charger des bibliothèques d'analyse courantes (recommandé)

Choisissez cette option pour inclure les bibliothèques courantes pour Python 3.9 dans le Python shell.

Si vos bibliothèques sont personnalisées ou si elles sont en conflit avec les bibliothèques préinstallées, vous pouvez choisir de ne pas installer de bibliothèques courantes. Cependant, vous pouvez installer des bibliothèques supplémentaires en plus des bibliothèques courantes.

Lorsque vous sélectionnez cette option, l'option `library-set` est définie sur `analytics`. Lorsque vous désélectionnez cette option, l'option `library-set` est définie sur `none`.

Nom du fichier de script et chemin du script

Le code inclus dans le script permet de définir la logique procédurale de votre tâche. Vous fournissez le nom et l'emplacement du script dans Amazon Simple Storage Service (Amazon

S3). Vérifiez qu'il n'existe pas de fichier portant le même nom que le répertoire de script dans le chemin d'accès. Pour en savoir plus sur les scripts, consultez AWS Glue guide de programmation.

Script

Le code inclus dans le script permet de définir la logique procédurale de votre tâche. Vous pouvez coder ce script dans Python 3.6 ou Python 3.9. Vous pouvez modifier un script dans AWS Glue Studio.

Unités de traitement de données

Nombre maximal d'unités de traitement de AWS Glue données (DPUs) pouvant être allouées lors de l'exécution de cette tâche. Un DPU est une mesure relative de la puissance de traitement composée de 4 V de capacité CPUs de calcul et de 16 Go de mémoire. Pour en savoir plus, consultez PricingAWS Glue (Tarification).

Vous pouvez affecter la valeur 0.0625 ou 1. La valeur par défaut est 0.0625. Dans les deux cas, le disque local de l'instance sera de 20 Go.

INTERFACE DE LIGNE DE COMMANDE (CLI)

Vous pouvez également créer une tâche shell Python à l'aide de AWS CLI, comme dans l'exemple suivant.

```
aws glue create-job --name python-job-cli --role Glue_DefaultRole
    --command '{"Name" :  "pythonshell", "PythonVersion": "3.9", "ScriptLocation" :
"s3://amzn-s3-demo-bucket/scriptname.py"}'
    --max-capacity 0.0625
```

> **Note**
>
> Il n'est pas nécessaire de spécifier la version d'AWS Glue, puisque le paramètre `--glue-version` ne s'applique pas aux tâches shell AWS Glue. Toute version spécifiée sera ignorée.

Tâches que vous créez avec Python 3 AWS CLI par défaut. Les versions valides de Python sont 3 (correspondant à 3.6), et 3.9. Pour spécifier Python 3.6, ajoutez ce tuple au paramètre `--command` : `"PythonVersion":"3"`

Pour spécifier Python 3.9, ajoutez ce tuple au paramètre `--command` : `"PythonVersion":"3.9"`

Pour définir la capacité maximale utilisée par une tâche shell Python, définissez le paramètre `--max-capacity`. Pour les tâches shell Python, le paramètre `--allocated-capacity` ne peut pas être utilisé.

Bibliothèques prises en charge pour les tâches shell Python

Dans le Python shell utilisant Python 3.9, vous pouvez choisir l'ensemble de bibliothèques qui utilisera les ensembles de bibliothèques préemballés selon vos besoins. Vous pouvez utiliser l'option `library-set` pour choisir l'ensemble de bibliothèques. Les valeurs valides sont `analytics`, et `none`.

L'environnement pour exécuter une tâche shell Python prend en charge les bibliothèques suivantes :

Version de Python	Python 3.6	Python 3.9	
Ensemble de bibliothèques	S/O	analytique	Aucun(e)
avro		1.11.0	
awscli	116,242	1,23,5	1,23,5
awswrangler		2.15.1	
botocore	1,12,232	1,24,21	1,23,5
boto3	1,9,203	1,21,21	
elasticsearch		8.2.0	
numpy	1.16,2	1.22.3	
pandas	0,24,2	1.4.2	
psycopg2		2.9.3	
pyathena		2.5.3	
PyGreSQL	5.0.6		

Version de Python	Python 3.6	Python 3.9	
PyMySQL		1.0.2	
pyodbc		4,0,32	
pyorc		0.6.0	
redshift-connector		2,0.907	
demandes	2.22.0	2.27.1	
scikit-learn	0,20,3	1.0.2	
scipy	1.2.1	1.8.0	
SQLAlchemy		1,4,36	
s3fs		2022.3.0	

Vous pouvez utiliser la bibliothèque NumPy dans une tâche shell Python à des fins de calcul scientifique. Pour de plus amples informations, veuillez consulter NumPy. L'exemple suivant montre un NumPy script qui peut être utilisé dans une tâche shell Python. Le script inscrit « Hello world » et les résultats de plusieurs calculs mathématiques.

```
import numpy as np
print("Hello world")

a = np.array([20,30,40,50])
print(a)

b = np.arange( 4 )

print(b)

c = a-b

print(c)

d = b**2
```

```
print(d)
```

Ajout de votre propre bibliothèque Python

Utilisation de PIP

Le Python shell utilisant Python 3.9 vous permet de fournir des modules Python supplémentaires ou des versions différentes au niveau de la tâche. Vous pouvez utiliser l'option `--additional-python-modules` avec une liste de modules Python séparés par des virgules pour ajouter un nouveau module ou modifier la version d'un module existant. Vous ne pouvez pas fournir de modules Python personnalisés hébergés sur Amazon S3 avec ce paramètre lorsque vous utilisez des tâches shell Python.

Par exemple, pour mettre à jour ou ajouter un nouveau module `scikit-learn`, utilisez la clé et la valeur suivantes : `"--additional-python-modules", "scikit-learn==0.21.3"`.

AWS Glue utilise le programme d'installation de package Python (pip3) pour installer les modules supplémentaires. Vous pouvez transmettre des options pip3 supplémentaires à l'intérieur de la valeur `--additional-python-modules`. Par exemple, `"scikit-learn==0.21.3 -i https://pypi.python.org/simple/"`. Toutes les incompatibilités ou limitations de pip3 s'appliquent.

> **Note**
>
> Pour éviter les incompatibilités à l'avenir, nous vous recommandons d'utiliser des bibliothèques créées pour Python 3.9.

Utilisation d'un fichier Egg ou Whl

Vous avez peut-être déjà une ou plusieurs bibliothèques Python empaquetées sous la forme d'un fichier `.egg` ou `.whl`. Le cas échéant, vous pouvez les spécifier pour votre tâche à l'aide du AWS Command Line Interface (AWS CLI) sous l'indicateur « `--extra-py-files` », comme dans l'exemple suivant.

```
aws glue create-job --name python-redshift-test-cli --role role --command '{"Name" :
 "pythonshell", "ScriptLocation" : "s3://MyBucket/python/library/redshift_test.py"}'
    --connections Connections=connection-name --default-arguments '{"--extra-py-
files" : ["s3://amzn-s3-demo-bucket/EGG-FILE", "s3://amzn-s3-demo-bucket/WHEEL-FILE"]}'
```

Si vous n'êtes pas sûr de la façon de créer un fichier `.egg` ou `.whl` à partir d'une bibliothèque Python, procédez comme suit. Cet exemple est applicable sur macOS, Linux et WSL (sous-système Windows pour Linux).

Pour créer un fichier Python .egg ou .whl

1. Créez un cluster Amazon Redshift dans un Virtual Private Cloud (VPC) et ajoutez des données dans une table.
2. Créez une AWS Glue connexion pour la VPC-SecurityGroup-Subnet combinaison que vous avez utilisée pour créer le cluster. Testez le bon fonctionnement de cette connexion.
3. Créez un répertoire nommé `redshift_example` et un fichier nommé `setup.py`. Collez le code suivant dans .

```
from setuptools import setup

setup(
    name="redshift_module",
    version="0.1",
    packages=['redshift_module']
)
```

4. Dans le répertoire `redshift_example`, créez un répertoire `redshift_module`. Dans le répertoire `redshift_module`, créez les fichiers `__init__.py` et `pygresql_redshift_common.py`.
5. Laissez le fichier `__init__.py` vide. Collez le code suivant dans `pygresql_redshift_common.py`. Remplacez *`portdb_name`*,*`user`*, et par *`password_for_user`* des informations spécifiques à votre cluster Amazon Redshift. Remplacez *`table_name`* par le nom de la table dans Amazon Redshift.

```
import pg


def get_connection(host):
    rs_conn_string = "host=%s port=%s dbname=%s user=%s password=%s" % (
        host, port, db_name, user, password_for_user)

    rs_conn = pg.connect(dbname=rs_conn_string)
    rs_conn.query("set statement_timeout = 1200000")
    return rs_conn
```

```
def query(con):
    statement = "Select * from table_name;"
    res = con.query(statement)
    return res
```

6. Si vous n'êtes pas déjà dans le répertoire `redshift_example`, accédez-y.
7. Effectuez l'une des actions suivantes :

   - Pour créer un fichier `.egg`, exécutez la commande suivante :

     ```
     python setup.py bdist_egg
     ```

   - Pour créer un fichier `.whl`, exécutez la commande suivante.

     ```
     python setup.py bdist_wheel
     ```

8. Installez les dépendances qui sont nécessaires à la commande précédente.
9. La commande crée un fichier dans le répertoire `dist` :

   - Si vous avez créé un fichier egg, il est nommé `redshift_module-0.1-py2.7.egg`.
   - Si vous avez créé un fichier Wheel, il est nommé `redshift_module-0.1-py2.7-none-any.whl`.

   Téléchargez vos fichiers sur Amazon S3.

   Dans cet exemple, le chemin du fichier chargé est *s3://amzn-s3-demo-bucket/EGG-FILE* ou *s3://amzn-s3-demo-bucket/WHEEL-FILE*.
10. Créez un fichier Python à utiliser comme script pour la tâche AWS Glue et ajoutez le code suivant au fichier.

```
from redshift_module import pygresql_redshift_common as rs_common

con1 = rs_common.get_connection(redshift_endpoint)
res = rs_common.query(con1)

print "Rows in the table cities are: "

print res
```

11. Téléchargez le fichier précédent sur Amazon S3. Dans cet exemple, le chemin du fichier chargé est *s3://amzn-s3-demo-bucket/scriptname.py*.
12. Créez une tâche shell Python à l'aide de ce script. Dans la console AWS Glue, dans la page Propriétés de la tâche, spécifiez le chemin d'accès au fichier `.egg/.whl` dans la zone Chemin de la bibliothèque Python. Si vous avez plusieurs fichiers `.egg/.whl` et fichiers Python, indiquez-les dans cette zone sous forme de liste de valeurs séparées par des virgules.

    Lors de la modification ou du changement de nom de fichiers `.egg`, les noms de ces derniers doivent utiliser les noms par défaut générés par la commande « python setup.py bdist_egg » ou doivent respecter les conventions de dénomination du module Python. Pour plus d'informations, consultez Style Guide for Python Code (Guide de style pour le code Python).

    À l'aide de AWS CLI, créez une tâche avec une commande, comme dans l'exemple suivant.

    ```
    aws glue create-job --name python-redshift-test-cli --role Role --command
     '{"Name" :  "pythonshell", "ScriptLocation" : "s3://amzn-s3-demo-bucket/
    scriptname.py"}'
         --connections Connections="connection-name" --default-arguments '{"--extra-
    py-files" : ["s3://amzn-s3-demo-bucket/EGG-FILE", "s3://amzn-s3-demo-bucket/WHEEL-
    FILE"]}'
    ```

    Lorsque le job s'exécute, le script imprime les lignes créées dans la *table_name* table du cluster Amazon Redshift.

À utiliser AWS CloudFormation avec les jobs shell Python dans AWS Glue

Vous pouvez les utiliser AWS CloudFormation avec des jobs shell Python dansAWS Glue. Voici un exemple :

```
AWSTemplateFormatVersion: 2010-09-09
Resources:
  Python39Job:
    Type: 'AWS::Glue::Job'
    Properties:
      Command:
        Name: pythonshell
        PythonVersion: '3.9'
        ScriptLocation: 's3://bucket/location'
```

```
      MaxRetries: 0
      Name: python-39-job
      Role: RoleName
```

Le groupe Amazon CloudWatch Logs pour les résultats des tâches shell Python est`/aws-glue/python-jobs/output`. Pour les erreurs, consultez le groupe de journaux `/aws-glue/python-jobs/error`.

Migration du shell Python 3.6 vers le shell Python 3.9

Pour migrer vos jobs shell Python vers la dernière AWS Glue version :

1. Dans la AWS Glue console (https://console.aws.amazon.com/glue/), choisissez votre job shell Python existant.
2. Dans l'onglet de Informations de la tâche, définissez la version de Python sur `Python 3.9` et choisissez Enregistrer.
3. Assurez-vous que votre script de tâche est compatible avec Python 3.9 et qu'il s'exécute correctement.

Migrer depuis les jobs shell AWS Glue Python

AWS a lancé les jobs shell AWS Glue Python en 2018 AWS a lancé les jobs shell AWS Glue Python en 2018 afin de permettre aux clients d'exécuter facilement des scripts Python pour des tâches ETL de small-to-medium grande taille et de déclencher des requêtes SQL. Cependant, il existe désormais des options plus modernes et plus flexibles pour faire face aux charges de travail actuelles. PythonShell Cette rubrique explique comment migrer vos charges de travail depuis les jobs shell AWS Glue Python vers l'une de ces options alternatives afin de tirer parti des nouvelles fonctionnalités disponibles.

Cette rubrique explique comment migrer des jobs shell AWS Glue Python vers des options alternatives.

Migration de la charge de travail vers les tâches AWS Glue Spark

AWS Glue Spark et les PySpark jobs vous permettent d'exécuter vos charges de travail de manière distribuée. Comme les jobs AWS Glue Python Shell et les jobs AWS Glue Spark s'exécutent sur la même plateforme, la migration est facile et vous pouvez continuer à utiliser les fonctionnalités AWS

Glue existantes que vous utilisez avec les tâches Python Shell, telles que AWS Glue Workflows, AWS Glue Triggers, Amazon EventBridge l'intégration de AWS Glue, l'installation de packages basée sur \ Pip, etc.

Cependant, les tâches AWS Glue Spark sont conçues pour exécuter des charges de travail Spark, et le nombre minimum de travailleurs est de 2. Si vous migrez à partir de tâches shell Python sans modifier vos scripts, un seul travailleur sera réellement utilisé et les autres resteront inactifs. Cela augmentera vos coûts.

Pour que l'opération soit efficace, réécrivez votre script de tâche Python afin d'utiliser les fonctionnalités de Spark et de répartir la charge de travail entre plusieurs travailleurs. Si votre script Python est basé sur Pandas, il est facile de le migrer à l'aide de la nouvelle API Pandas sur Spark. Pour en savoir plus, consultez le blog AWS Big Data : Dive deep into AWS Glue 4.0 pour Apache Spark.

Migration de la charge de travail vers AWS Lambda

AWS Lambda est un service informatique sans serveur qui vous permet d'exécuter du code sans provisionner ni gérer de serveurs. Comme AWS Lambda a des temps de démarrage plus courts et des options de capacité de calcul plus flexibles, vous pouvez bénéficier de ces avantages. Pour gérer des bibliothèques Python supplémentaires, les tâches shell AWS Glue Python utilisent une installation basée sur PIP. Cependant, pour AWS Lambda, vous devez choisir l'une des options suivantes : une archive zip, une image de conteneur ou des couches Lambda.

En revanche, le délai maximum AWS Lambda est de 900 secondes (15 minutes). Si la durée de travail de votre charge de travail AWS Glue Python Shell existante est supérieure à ce montant, ou si votre charge de travail présente un schéma épineux susceptible de prolonger la durée des tâches, nous vous recommandons d'explorer d'autres options au lieu de. AWS Lambda

Migration de la charge de travail vers Amazon ECS/Fargate

Amazon Elastic Container Service (Amazon ECS) est un service entièrement géré qui simplifie le déploiement, la gestion et le dimensionnement des applications conteneurisées. AWS Fargate est un moteur de calcul sans serveur pour les charges de travail conteneurisées exécutées sur Amazon ECS et (Amazon Elastic Kubernetes Service Amazon EKS). Il n'y a pas de délai d'expiration maximal sur Amazon ECS et Fargate, c'est donc une bonne option pour les tâches de longue durée. Comme vous avez le contrôle total sur l'image de votre conteneur, vous pouvez importer votre script Python et des bibliothèques Python supplémentaires dans le conteneur et les utiliser. Cependant, vous devez conteneuriser votre script Python pour utiliser cette approche.

Migration de la charge de travail vers Amazon Managed Workflows pour Apache Airflow Python Operator

Amazon Managed Workflows for Apache Airflow (Managed Workflows for Apache Airflow) est un service d'orchestration géré pour Apache Airflow qui facilite la configuration et l' end-to-endexploitation de pipelines de données dans le cloud à grande échelle. Si vous avez déjà un environnement MWAA, il sera facile d'utiliser l'opérateur Python au lieu des jobs AWS Glue Python Shell. L’opérateur Python est un opérateur qui exécute du code Python dans un flux de travail Airflow. Toutefois, si vous ne disposez pas d’un environnement MWAA existant, nous vous recommandons d’explorer d’autres options.

Migration de la charge de travail vers des postes de formation en Amazon SageMaker AI IA

Amazon SageMaker AI La formation est un service d'apprentissage automatique (ML) entièrement géré Amazon SageMaker AI qui vous aide à entraîner efficacement un large éventail de modèles de machine learning à grande échelle. La conteneurisation des charges de travail de machine learning et la capacité de gérer AWS les ressources informatiques sont au cœur des métiers de l' Amazon SageMaker AI IA. Si vous préférez un environnement sans serveur où il n'y a pas de délai d'attente maximal, les emplois de formation en Amazon SageMaker AI IA pourraient vous convenir. Cependant, la latence de démarrage a tendance à être plus longue que celle des jobs AWS Glue Python Shell. Pour les tâches sensibles à la latence, nous vous recommandons d’explorer d’autres options.

Surveillance AWS Glue

La surveillance joue un rôle important dans le maintien de la fiabilité, de la disponibilité AWS Glue et des performances de vos autres AWS solutions. AWS fournit des outils de surveillance que vous pouvez utiliser pour surveiller AWS Glue, signaler un problème et prendre des mesures automatiquement le cas échéant :

Vous pouvez utiliser les outils de surveillance automatique pour surveiller AWS Glue et signaler en cas de problème :

- Amazon CloudWatch Events fournit un flux en temps quasi réel d'événements système décrivant les modifications apportées aux AWS ressources. CloudWatch Les événements permettent une informatique automatisée axée sur les événements. Vous pouvez rédiger des règles qui surveillent certains événements et déclenchent des actions automatisées dans d'autres AWS services

lorsque ces événements se produisent. Pour plus d'informations, consultez le guide de l'utilisateur d'Amazon CloudWatch Events.

- Amazon CloudWatch Logs vous permet de surveiller, de stocker et d'accéder à vos fichiers journaux à partir d' EC2 instances Amazon et d'autres sources. AWS CloudTrail CloudWatch Les journaux peuvent surveiller les informations contenues dans les fichiers journaux et vous avertir lorsque certains seuils sont atteints. Vous pouvez également archiver vos données de journaux dans une solution de stockage hautement durable. Pour plus d'informations, consultez le guide de l'utilisateur d'Amazon CloudWatch Logs.
- AWS CloudTrailcapture les appels d'API et les événements associés effectués par ou pour le compte de votre AWS compte et envoie les fichiers journaux dans un compartiment Amazon S3 que vous spécifiez. Vous pouvez identifier les utilisateurs et les comptes qui appellent AWS, l'adresse IP source à partir de laquelle les appels sont effectués et le moment où les appels ont lieu. Pour plus d'informations, consultez le Guide de l'utilisateur AWS CloudTrail.

En outre, vous avez accès aux informations suivantes dans la AWS Glue console pour vous aider à déboguer et à profiler les tâches :

- Jobs Spark : vous pouvez voir une visualisation des séries de CloudWatch métriques sélectionnées, et les nouveaux jobs ont accès à l'interface utilisateur de Spark. Pour de plus amples informations, veuillez consulter the section called “Surveillance des tâches Spark”.
- Ray Jobs : vous pouvez voir une visualisation des séries de CloudWatch mesures sélectionnées. Pour de plus amples informations, veuillez consulter the section called “Métriques des tâches Ray”.

Rubriques

- AWS tags dans AWS Glue
- Automatiser AWS Glue avec EventBridge
- AWS Glue Ressources de surveillance
- Journalisation des appels d'AWS GlueAPI avec AWS CloudTrail

AWS tags dans AWS Glue

Pour mieux gérer vos ressources AWS Glue, vous pouvez éventuellement attribuer vos propres balises à certains types de ressources AWS Glue. Une étiquette est une étiquette que vous attribuez à une AWS ressource. Chaque balise est constituée d’une clé et d’une valeur facultative que

vous définissez. Vous pouvez utiliser des balises dans AWS Glue pour identifier et organiser vos ressources. Des balises peuvent être utilisées pour créer des rapports de comptabilisation des coûts et limiter l'accès aux ressources. Si vous en utilisez Gestion des identités et des accès AWS, vous pouvez contrôler quels utilisateurs de votre AWS compte sont autorisés à créer, modifier ou supprimer des tags. Outre les autorisations d'appel liées à l'étiquette APIs, vous devez également être `glue:GetConnection` autorisée à étiqueter les appels APIs sur les connexions et à étiqueter les `glue:GetDatabase` appels sur les bases de données APIs . Pour de plus amples informations, veuillez consulter ABAC avec Glue AWS.

Dans AWS Glue, vous pouvez baliser les ressources suivantes :

- Connexion
- Base de données
- crawler
- Session interactive
- Point de terminaison de développement
- Tâche
- Déclencheur
- Flux de travail
- Plan
- Transformation de machine learning
- Ensemble de règles de qualité des données
- Schémas de flux
- Registres de schémas de flux

> **Note**
>
> La bonne pratique afin d'autoriser le balisage de ces ressources AWS Glue consiste à toujours inclure l'action `glue:TagResource` dans vos politiques.

Tenez compte des éléments suivants lorsque vous utilisez des balises avec AWS Glue.

- Un maximum de 50 balises sont prises en charge par entité.

- Dans AWS Glue, vous spécifiez les balises sous la forme d'une liste de paires clé-valeur au format `{"string": "string" ...}`
- Lorsque vous créez une balise sur un objet, la clé de balise est obligatoire et la valeur de balise est facultative.
- La clé de balise et la valeur de balise sont sensibles à la casse.
- La clé de balise et la valeur de balise ne doivent pas contenir le préfixe aws. Aucune opération n'est autorisée sur ces balises.
- La longueur maximale des clés de balise est de 128 caractères Unicode en UTF-8. La clé de balise ne doit pas être vide ni null.
- La longueur maximale des valeurs de balise est de 256 caractères Unicode en UTF-8. La valeur de balise peut être vide ou null.

Support de balisage pour les connexions AWS Glue

Vous pouvez restreindre l'autorisation d'actions `CreateConnection`, `UpdateConnection`, `GetConnection` et `DeleteConnection` en fonction d'identification de ressources. Cela vous permet de mettre en œuvre le contrôle d'accès avec le moindre privilège sur les AWS Glue tâches comportant des sources de données JDBC qui doivent extraire les informations de connexion JDBC depuis le catalogue de données.

Exemple d’utilisation

Créez une AWS Glue connexion avec le tag ["connection-category », « dev-test"].

Spécifiez la condition de l'identification pour l'action `GetConnection` dans la politique IAM.

```
{
   "Effect": "Allow",
   "Action": [
     "glue:GetConnection"
     ],
   "Resource": "*",
   "Condition": {
     "ForAnyValue:StringEquals": {
       "aws:ResourceTag/tagKey": "dev-test"
     }
   }
 }
```

Exemples

Les exemples suivants créent une tâche avec des balises attribuées.

AWS CLI

```
aws glue create-job --name job-test-tags --role MyJobRole --command
 Name=glueetl,ScriptLocation=S3://aws-glue-scripts//prod-job1
--tags key1=value1,key2=value2
```

CloudFormation JSON

```
{
  "Description": "AWS Glue Job Test Tags",
  "Resources": {
    "MyJobRole": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "AssumeRolePolicyDocument": {
          "Version": "2012-10-17",
          "Statement": [
            {
              "Effect": "Allow",
              "Principal": {
                "Service": [
                  "glue.amazonaws.com"
                ]
              },
              "Action": [
                "sts:AssumeRole"
              ]
            }
          ]
        },
        "Path": "/",
        "Policies": [
          {
            "PolicyName": "root",
            "PolicyDocument": {
              "Version": "2012-10-17",
              "Statement": [
                {
                  "Effect": "Allow",
```

```
                  "Action": "*",
                  "Resource": "*"
                }
              ]
            }
          }
        ]
      }
    },
    "MyJob": {
      "Type": "AWS::Glue::Job",
      "Properties": {
        "Command": {
          "Name": "glueetl",
          "ScriptLocation": "s3://aws-glue-scripts//prod-job1"
        },
        "DefaultArguments": {
          "--job-bookmark-option": "job-bookmark-enable"
        },
        "ExecutionProperty": {
          "MaxConcurrentRuns": 2
        },
        "MaxRetries": 0,
        "Name": "cf-job1",
        "Role": {
           "Ref": "MyJobRole",
           "Tags": {
                "key1": "value1",
                "key2": "value2"
           }
        }
      }
    }
  }
}
```

CloudFormation YAML

```
Description: AWS Glue Job Test Tags
Resources:
  MyJobRole:
    Type: AWS::IAM::Role
    Properties:
```

```
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: Allow
            Principal:
              Service:
                - glue.amazonaws.com
            Action:
              - sts:AssumeRole
      Path: "/"
      Policies:
        - PolicyName: root
          PolicyDocument:
            Version: '2012-10-17'
            Statement:
              - Effect: Allow
                Action: "*"
                Resource: "*"
  MyJob:
    Type: AWS::Glue::Job
    Properties:
      Command:
        Name: glueetl
        ScriptLocation: s3://aws-glue-scripts//prod-job1
      DefaultArguments:
        "--job-bookmark-option": job-bookmark-enable
      ExecutionProperty:
        MaxConcurrentRuns: 2
      MaxRetries: 0
      Name: cf-job1
      Role:
        Ref: MyJobRole
        Tags:
           key1: value1
           key2: value2
```

Pour de plus amples informations, veuillez consulter Politiques de balisage AWS.

Pour obtenir des informations sur la façon de contrôler l'accès à l'aide de balises, consultez ABAC avec Glue AWS.

Automatiser AWS Glue avec EventBridge

Vous pouvez utiliser Amazon EventBridge pour automatiser vos AWS services et répondre automatiquement aux événements du système tels que les problèmes de disponibilité des applications ou les modifications des ressources. Les événements AWS liés aux services sont diffusés EventBridge en temps quasi réel. Vous pouvez écrire des règles simples pour indiquer quels événements vous intéressent et les actions automatisées à effectuer quand un événement correspond à une règle. Les actions pouvant être déclenchées automatiquement sont les suivantes :

- Invoquer une fonction AWS Lambda
- Invocation de la commande Amazon EC2 Run
- Relais de l'événement à Amazon Kinesis Data Streams
- Activation d'une machine à AWS Step Functions états
- Notification d'une rubrique Amazon SNS ou d'une file d'attente Amazon SQS

Voici quelques exemples d'utilisation EventBridge avec AWS Glue :

- Activation d'une fonction Lambda lorsqu'une tâche ETL réussit
- Notification d'une rubrique Amazon SNS lorsqu'une tâche ETL échoue

Les éléments suivants EventBridge sont générés parAWS Glue.

- Les événements pour `"detail-type":"Glue Job State Change"` sont générés pour `SUCCEEDED`, `FAILED`, `TIMEOUT` et `STOPPED`.
- Les événements pour `"detail-type":"Glue Job Run Status"` sont générés pour les exécutions de tâche`RUNNING`, `STARTING` et `STOPPING` lorsqu'ils dépassent le seuil de notification de dépassement de délai de tâche. Vous devez définir la propriété du seuil de notification des retards de tâche pour recevoir ces événements.

  Un seul événement est généré par statut d'exécution de tâche lorsque le seuil de notification de retard de tâche est dépassé.
- Les événements pour `"detail-type":"Glue Crawler State Change"` sont générés pour `Started`, `Succeeded` et `Failed`.
- Les événements pour `“detail_type”:“Glue Scheduled Crawler Invocation Failure”` sont générés lorsque le robot planifié ne démarre pas. Voici les informations figurant dans la notification :

  - `customerId` contient l'identifiant de compte du client.
  - `crawlerName` contient le nom du robot qui n'a pas pu démarrer.
  - `errorMessage` contient le message d'exception signalant l'échec de l'invocation.
- Les événements pour `"detail_type":"Glue Auto Statistics Invocation Failure"` sont générés lorsque l'exécution de la tâche de statistiques de colonne automatiquement gérée ne démarre pas. Voici les informations figurant dans la notification :
  - `catalogId` contient l'ID associé à un catalogue.
  - `databaseName` indique le nom de la base de données concernée.
  - `tableName` indique le nom de la table concernée.
  - `errorMessage` contient le message d'exception signalant l'échec de l'invocation.
- Les événements pour `"detail_type":"Glue Scheduled Statistics Invocation Failure"` sont générés lorsque la tâche de statistiques de colonne planifiée (cron) exécutée ne démarre pas. Voici les informations figurant dans la notification :
  - `catalogId` contient l'ID associé à un catalogue.
  - `databaseName` indique le nom de la base de données concernée.
  - `tableName` indique le nom de la table concernée.
  - `errorMessage` contient le message d'exception signalant l'échec de l'invocation.
- Les événements pour `"detail_type":"Glue Statistics Task Started"` sont générés lorsque l'exécution de la tâche de statistiques de colonne démarre.
- Les événements pour `"detail_type":"Glue Statistics Task Succeeded"` sont générés lorsque l'exécution de la tâche de statistiques de colonne aboutit.
- Les événements pour `"detail_type":"Glue Statistics Task Failed"` sont générés lorsque l'exécution de la tâche de statistiques de colonne échoue.
- Les événements pour `"detail-type":"Glue Data Catalog Database State Change"` sont générés pour `CreateDatabase`, `DeleteDatabase`, `CreateTable`, `DeleteTable` et `BatchDeleteTable`. Par exemple, si une table est créée ou supprimée, une notification est envoyée à EventBridge. Notez que vous ne pouvez pas écrire un programme en fonction de la présence d'événements de notification ou de leur ordre, car il peut ne pas y en avoir ou ils peuvent ne pas être dans l'ordre défini. Les événements sont générés dans la mesure du possible. Voici les informations figurant dans la notification :
  - Le champ `typeOfChange` contient le nom de l'opération d'API.
  - Le champ `databaseName` indique le nom de la base de données concernée.

  - Le champ `changedTables` contient jusqu'à 100 noms de tables concernées par notification. Lorsque les noms de table sont longs, plusieurs notifications peuvent être créées.
- Les événements pour `"detail-type":"Glue Data Catalog Table State Change"` sont générés pour `UpdateTable`, `CreatePartition`, `BatchCreatePartition`, `UpdatePartition`, `DeletePartition`, `BatchUpdatePartition` et `BatchDeletePartition`. Par exemple, si une table ou une partition est mise à jour, une notification est envoyée à EventBridge. Notez que vous ne pouvez pas écrire un programme en fonction de la présence d'événements de notification ou de leur ordre, car il peut ne pas y en avoir ou ils peuvent ne pas être dans l'ordre défini. Les événements sont générés dans la mesure du possible. Voici les informations figurant dans la notification :
  - Le champ `typeOfChange` contient le nom de l'opération d'API.
  - Le champ `databaseName` indique le nom de la base de données dont les ressources sont concernées.
  - Le champ `tableName` indique le nom de la table concernée.
  - Le champ `changedPartitions` spécifie jusqu'à 100 partitions concernées dans une notification. Lorsque les noms de partition sont longs, plusieurs notifications peuvent être créées.

    Par exemple, s'il existe deux clés de partition, `Year` et `Month`, alors `"2018,01", "2018,02"` modifie la partition si `"Year=2018" and "Month=01"` et si `"Year=2018" and "Month=02"`.

```
{
    "version":"0",
    "id":"abcdef00-1234-5678-9abc-def012345678",
    "detail-type":"Glue Data Catalog Table State Change",
    "source":"aws.glue",
    "account":"123456789012",
    "time":"2017-09-07T18:57:21Z",
    "region":"us-west-2",
    "resources":["arn:aws:glue:us-west-2:123456789012:database/default/foo"],
    "detail":{
        "changedPartitions": [
            "2018,01",
            "2018,02"
        ],
        "databaseName": "default",
        "tableName": "foo",
        "typeOfChange": "BatchCreatePartition"
```

```
        }
}
```

Pour plus d'informations, consultez le guide de l'utilisateur d'Amazon CloudWatch Events. Pour des événements spécifiques à AWS Glue, consultez Événements AWS Glue.

AWS Glue Ressources de surveillance

AWS Glue dispose de limites de service afin de protéger les clients contre les provisionnements excessifs inattendus et contre les actions malveillantes visant à augmenter votre facture. Les limites protègent également le service. En se connectant à la console AWS Service Quota, les clients peuvent consulter leurs limites de ressources actuelles et demander une augmentation (le cas échéant).

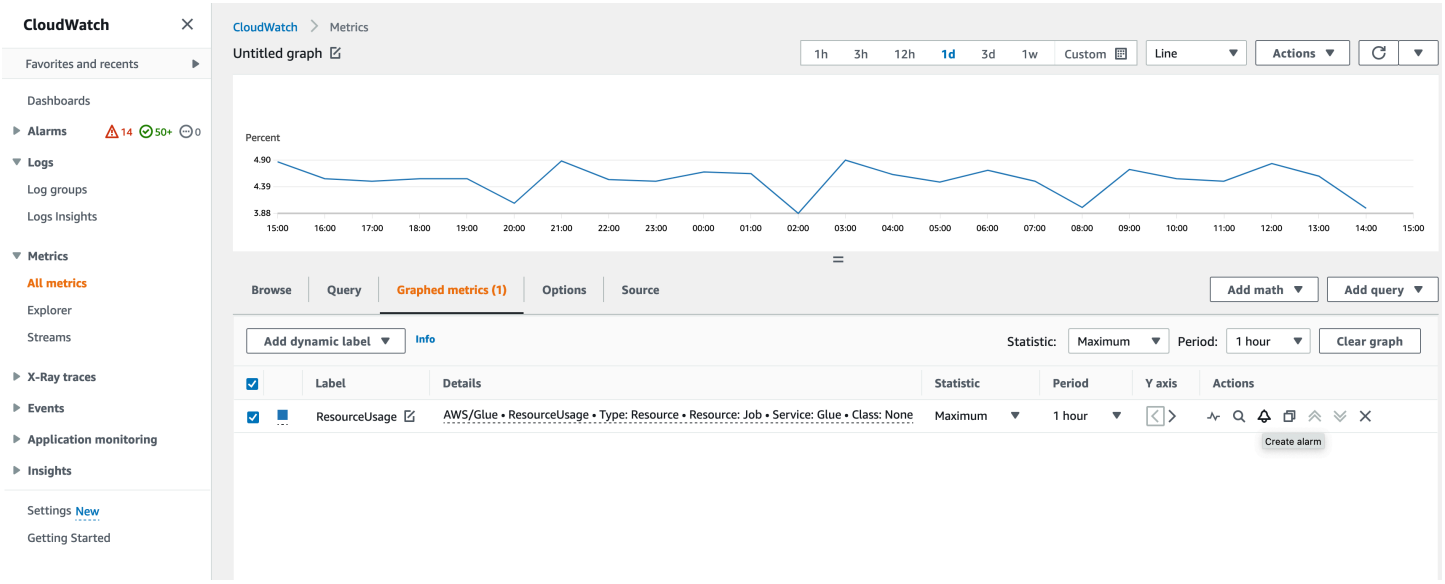
AWS Gluevous permet de visualiser l'utilisation des ressources du service sous forme de pourcentage dans Amazon CloudWatch et de configurer des CloudWatch alarmes pour surveiller l'utilisation. Amazon CloudWatch assure la surveillance des AWS ressources et des applications client exécutées sur l'infrastructure Amazon. Les métriques sont gratuites pour vous. Les métriques suivantes sont prises en charge :

- Nombre de flux de travail par compte
- Nombre de déclencheurs par compte
- Nombre de tâches par compte
- Nombre d'exécutions de tâches simultanées par compte
- Nombre de plans par compte
- Nombre de sessions interactives par compte

Configuration et utilisation des métriques de ressources

Pour utiliser cette fonctionnalité, vous pouvez accéder à la CloudWatch console Amazon pour consulter les statistiques et configurer les alarmes. Les métriques se trouvent sous l'espace de noms AWS/Glue et représentent un pourcentage du nombre réel d'utilisation des ressources divisé par le quota de ressources. Les CloudWatch statistiques sont transmises à vos comptes, sans frais pour vous. Par exemple, si vous avez créé 10 flux de travail et que votre quota de service vous permet d'avoir 200 flux de travail au maximum, votre utilisation est de 10/200 = 5 %, et dans le graphique, vous verrez un point de données de cinq en tant que pourcentage. Pour être plus précis :

```
Namespace: AWS/Glue
Metric name: ResourceUsage
Type: Resource
Resource: Workflow (or Trigger, Job, JobRun, Blueprint, InteractiveSession)
Service: Glue
Class: None
```



Pour créer une alarme sur une métrique dans la CloudWatch console, procédez comme suit :

1. Une fois que vous avez localisé la métrique, accédez à Métriques graphiques.
2. Cliquez sur Créer une alarme sous Actions.
3. Configurez l'alarme selon vos besoins.

Nous émettons des métriques chaque fois que l'utilisation de vos ressources change (augmentation ou diminution, par exemple). Mais si l'utilisation de vos ressources ne change pas, nous émettons des métriques toutes les heures, afin que vous disposiez d'un CloudWatch graphique continu. Pour éviter d'avoir des points de données manquants, nous vous déconseillons de configurer une période inférieure à une heure.

Vous pouvez également configurer les alarmes à AWS CloudFormation l'aide de l'exemple suivant. Dans cet exemple, une fois que l'utilisation des ressources du flux de travail atteint 80 %, une alarme est déclenchée pour envoyer un message à la rubrique SNS existante, à laquelle vous pouvez vous abonner pour recevoir des notifications.

```
{
 "Type": "AWS::CloudWatch::Alarm",
 "Properties": {
  "AlarmName": "WorkflowUsageAlarm",
  "ActionsEnabled": true,
  "OKActions": [],
  "AlarmActions": [
   "arn:aws:sns:af-south-1:085425700061:Default_CloudWatch_Alarms_Topic"
  ],
  "InsufficientDataActions": [],
  "MetricName": "ResourceUsage",
  "Namespace": "AWS/Glue",
  "Statistic": "Maximum",
  "Dimensions": [{
    "Name": "Type",
    "Value": "Resource"
   },
   {
    "Name": "Resource",
    "Value": "Workflow"
   },
   {
    "Name": "Service",
    "Value": "Glue"
   },
   {
    "Name": "Class",
    "Value": "None"
   }
  ],
  "Period": 3600,
  "EvaluationPeriods": 1,
  "DatapointsToAlarm": 1,
  "Threshold": 80,
  "ComparisonOperator": "GreaterThanThreshold",
  "TreatMissingData": "notBreaching"
 }
}
```

Journalisation des appels d'AWS GlueAPI avec AWS CloudTrail

AWS Glueest intégré à AWS CloudTrail un service qui fournit un enregistrement des actions entreprises par un utilisateur, un rôle ou un AWS service dansAWS Glue. CloudTrail capture tous les appels d'API AWS Glue sous forme d'événements. Les appels capturés incluent des appels de la console AWS Glue et les appels de code vers les opérations d'API AWS Glue. Si vous créez un suivi, vous pouvez activer la diffusion continue d' CloudTrail événements vers un compartiment Amazon S3, y compris les événements pourAWS Glue. Si vous ne configurez pas de suivi, vous pouvez toujours consulter les événements les plus récents dans la CloudTrail console dans Historique des événements. À l'aide des informations collectées par CloudTrail, vous pouvez déterminer la demande qui a été faiteAWS Glue, l'adresse IP à partir de laquelle la demande a été faite, qui a fait la demande, quand elle a été faite et des détails supplémentaires.

Pour en savoir plus CloudTrail, consultez le guide de AWS CloudTrail l'utilisateur.

AWS Glueinformations dans CloudTrail

CloudTrail est activé sur votre AWS compte lorsque vous le créez. Lorsqu'une activité se produit dansAWS Glue, cette activité est enregistrée dans un CloudTrail événement avec d'autres événements de AWS service dans l'historique des événements. Vous pouvez consulter, rechercher et télécharger les événements récents dans votre AWS compte. Pour plus d'informations, consultez la section Affichage des événements à l'aide de l'historique des CloudTrail événements.

Pour un enregistrement continu des événements de votre AWS compte, y compris des événements pourAWS Glue, créez un parcours. Un suivi permet CloudTrail de fournir des fichiers journaux à un compartiment Amazon S3. Par défaut, lorsque vous créez un parcours dans la console, celui-ci s'applique à toutes les AWS régions. Le journal enregistre les événements de toutes les régions de la AWS partition et transmet les fichiers journaux au compartiment Amazon S3 que vous spécifiez. En outre, vous pouvez configurer d'autres AWS services pour analyser plus en détail les données d'événements collectées dans les CloudTrail journaux et agir en conséquence. Pour plus d'informations, consultez les ressources suivantes :

- Création d'un parcours pour votre AWS compte
- CloudTrail services et intégrations pris en charge
- Configuration des notifications Amazon SNS pour CloudTrail
- Réception de fichiers CloudTrail journaux de plusieurs régions et réception de fichiers CloudTrail journaux de plusieurs comptes

Toutes les AWS Glue actions sont enregistrées CloudTrail et documentées dans leAWS Glue API. Par exemple, les appels au`CreateDatabase`, `CreateTable` et les `CreateScript` actions génèrent des entrées dans les fichiers CloudTrail journaux.

Chaque événement ou entrée de journal contient des informations sur la personne ayant initié la demande. Les informations relatives à l'identité permettent de déterminer les éléments suivants :

- Si la demande a été effectuée avec des informations d'identification d'utilisateur root ou IAM.
- Si la demande a été effectuée avec des informations d'identification de sécurité temporaires pour un rôle ou un utilisateur fédéré.
- Si la demande a été faite par un autre AWS service.

Pour plus d'informations, consultez la section Élément userIdentity CloudTrail .

Cependant, CloudTrail n'enregistre pas toutes les informations concernant les appels. Par exemple, il n'enregistre pas certaines informations sensibles, telles que celles `ConnectionProperties` utilisées dans les demandes de connexion, et il enregistre un `null` au lieu des réponses renvoyées par les éléments suivants APIs :

```
BatchGetPartition       GetCrawlers          GetJobs          GetTable
CreateScript            GetCrawlerMetrics    GetJobRun        GetTables
GetCatalogImportStatus  GetDatabase          GetJobRuns       GetTableVersions
GetClassifier           GetDatabases         GetMapping       GetTrigger
GetClassifiers          GetDataflowGraph     GetObjects       GetTriggers
GetConnection           GetDevEndpoint       GetPartition     GetUserDefinedFunction
GetConnections          GetDevEndpoints      GetPartitions    GetUserDefinedFunctions
GetCrawler              GetJob               GetPlan
```

Présentation des entrées des fichiers journaux AWS Glue

Un suivi est une configuration qui permet de transmettre des événements sous forme de fichiers journaux à un compartiment Amazon S3 que vous spécifiez. CloudTrail les fichiers journaux contiennent une ou plusieurs entrées de journal. Un événement représente une demande unique provenant de n'importe quelle source et inclut des informations sur l'action demandée, la date et l'heure de l'action, les paramètres de la demande, etc. CloudTrail les fichiers journaux ne constituent pas une trace ordonnée des appels d'API publics, ils n'apparaissent donc pas dans un ordre spécifique.

L'exemple suivant montre une entrée de CloudTrail journal illustrant l'`DeleteCrawler`action.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AKIAIOSFODNN7EXAMPLE",
    "arn": "arn:aws:iam::123456789012:user/johndoe",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "userName": "johndoe"
  },
  "eventTime": "2017-10-11T22:29:49Z",
  "eventSource": "glue.amazonaws.com",
  "eventName": "DeleteCrawler",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "72.21.198.64",
  "userAgent": "aws-cli/1.11.148 Python/3.6.1 Darwin/16.7.0 botocore/1.7.6",
  "requestParameters": {
    "name": "tes-alpha"
  },
  "responseElements": null,
  "requestID": "b16f4050-aed3-11e7-b0b3-75564a46954f",
  "eventID": "e73dd117-cfd1-47d1-9e2f-d1271cad838c",
  "eventType": "AwsApiCall",
  "recipientAccountId": "123456789012"
}
```


Cet exemple montre une entrée de CloudTrail journal illustrant une `CreateConnection` action.


```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AKIAIOSFODNN7EXAMPLE",
    "arn": "arn:aws:iam::123456789012:user/johndoe",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "userName": "johndoe"
  },
  "eventTime": "2017-10-13T00:19:19Z",
  "eventSource": "glue.amazonaws.com",
  "eventName": "CreateConnection",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "72.21.198.66",
```

```
  "userAgent": "aws-cli/1.11.148 Python/3.6.1 Darwin/16.7.0 botocore/1.7.6",
  "requestParameters": {
    "connectionInput": {
      "name": "test-connection-alpha",
      "connectionType": "JDBC",
      "physicalConnectionRequirements": {
        "subnetId": "subnet-323232",
        "availabilityZone": "us-east-1a",
        "securityGroupIdList": [
          "sg-12121212"
        ]
      }
    }
  },
  "responseElements": null,
  "requestID": "27136ebc-afac-11e7-a7d6-ab217e5c3f19",
  "eventID": "e8b3baeb-c511-4597-880f-c16210c60a4a",
  "eventType": "AwsApiCall",
  "recipientAccountId": "123456789012"
}
```

AWS Glue Streaming

AWS Glue Le streaming, un composant de AWS Glue, vous permet de gérer efficacement les données en streaming en temps quasi réel, ce qui vous permet d'effectuer des tâches cruciales telles que l'ingestion, le traitement et l'apprentissage automatique des données. Grâce au framework Apache Spark Streaming, AWS Glue Streaming fournit un service sans serveur capable de gérer les données de streaming à grande échelle. AWS Glue fournit diverses optimisations en plus d'Apache Spark, telles que l'infrastructure sans serveur, l'auto-scaling, le développement visuel des tâches, les blocs-notes instantanés pour les tâches de streaming et d'autres améliorations de performances.

Cas d’utilisation pour le streaming

Parmi les cas d'utilisation courants du AWS Glue streaming, citons :

Near-real-time traitement des données : le AWS Glue streaming permet aux entreprises de traiter les données en streaming en temps quasi réel, ce qui leur permet d'en tirer des informations et de prendre des décisions en temps opportun sur la base des informations les plus récentes.

Détection des fraudes : vous pouvez utiliser le AWS Glue streaming pour analyser en temps réel les données de streaming, ce qui le rend utile pour détecter les activités frauduleuses, telles que les fraudes par carte de crédit, les intrusions sur le réseau ou les escroqueries en ligne. En traitant et en analysant en permanence les données entrantes, vous pouvez rapidement identifier les schémas suspects ou les anomalies.

Analyse des réseaux sociaux : le AWS Glue streaming peut traiter les données des réseaux sociaux en temps réel, telles que les tweets, les publications ou les commentaires, permettant aux entreprises de suivre les tendances, d'analyser les sentiments et de gérer la réputation de la marque en temps réel.

Analyse de l'Internet des objets (IoT) : le AWS Glue streaming convient à la gestion et à l'analyse de flux de données à haute vitesse générés par des appareils IoT, des capteurs et des machines connectées. Il permet le suivi en temps réel, la détection des anomalies, la maintenance prédictive et d’autres cas d’utilisation de l’analytique IoT.

Analyse du flux de clics : le AWS Glue streaming peut traiter et analyser les données du flux de clics en temps réel provenant de sites Web ou d'applications mobiles. Cela permet aux entreprises de mieux comprendre le comportement des utilisateurs, de personnaliser les expériences utilisateur et d’optimiser les campagnes marketing en fonction des données du flux de clics en temps réel.

Surveillance et analyse des journaux : le AWS Glue streaming permet de traiter et d'analyser en continu les données des journaux provenant de serveurs, d'applications ou de périphériques réseau en temps réel. Cela permet de détecter les anomalies, de résoudre les problèmes et de surveiller l’état et les performances du système.

Systèmes de recommandation : le AWS Glue streaming peut traiter les données d'activité des utilisateurs en temps réel et mettre à jour les modèles de recommandation de manière dynamique. Cela permet des recommandations personnalisées et en temps réel basées sur le comportement et les préférences des utilisateurs.

Voici quelques exemples de la diversité des cas d'utilisation dans lesquels le AWS Glue streaming peut être appliqué. Son intégration à l' AWS écosystème et aux services gérés en fait un choix pratique pour le traitement et l'analyse des flux en temps réel dans le cloud.

Quels sont les avantages du AWS Glue streaming ?

Les avantages de l'utilisation du AWS Glue streaming sont les suivants :

- Sans serveur : le AWS Glue streaming se fait sans serveur, ce qui élimine le besoin de gérer l'infrastructure. Cela réduit les frais opérationnels et permet aux utilisateurs de se concentrer sur les tâches de traitement et d’analytique des données plutôt que sur la gestion de l’infrastructure.
- Mise à l'échelle automatique : le AWS Glue streaming fournit des fonctionnalités de mise à l'échelle automatique, ajustant dynamiquement la capacité de traitement en fonction de la charge de travail. Il monte en puissance ou se met à l’échelle horizontale automatiquement pour gérer les fluctuations du volume de données, garantissant ainsi des performances et une utilisation des ressources optimales.
- Développement visuel : le développement des emplois en streaming peut être complexe. AWS Glue Le streaming répond à ce défi en proposant AWS Glue Studio, un outil de création visuelle. AWS Glue Studio simplifie le processus de création de flux de travail de streaming et permet aux développeurs de concevoir et de gérer visuellement des applications de streaming, réduisant ainsi la courbe d'apprentissage et augmentant la productivité.
- Rentable : en tant que service sans serveur, le AWS Glue streaming permet de réduire les coûts en éliminant le besoin de provisionnement et de maintenance de l'infrastructure. Les utilisateurs sont facturés en fonction des ressources consommées lors de l’exécution des tâches de streaming, ce qui permet une optimisation des coûts et une mise à l’échelle en fonction de l’utilisation réelle.
- Gère des charges de travail complexes : le AWS Glue streaming est conçu pour gérer des charges de travail de streaming complexes. Il peut traiter et analyser de grands volumes de données

en temps réel, prendre en charge des transformations avancées et s'intégrer à d'autres AWS services, permettant ainsi des pipelines de données en streaming et des flux de travail d'analyse sophistiqués.

- Aucune dépendance : le AWS Glue streaming apporte de la flexibilité et évite la dépendance vis-à-vis d'un fournisseur. Les utilisateurs peuvent tirer parti du AWS Glue streaming dans le cadre d'un AWS écosystème plus large, en l'intégrant parfaitement à d'autres AWS services. Cela permet une intégration facile avec les sources de données, les applications et les services existants sans être lié à une technologie ou à une plateforme spécifique.

Quand utiliser le AWS Glue streaming ?

Il existe de nombreuses options en ce qui concerne les cas d’utilisation du streaming. Nous recommandons le AWS Glue streaming dans les scénarios suivants.

1. Si vous utilisez AWS Glue déjà Spark pour le traitement par lots, AWS Glue Streaming est le choix idéal pour vous. Il permet une transition fluide vers la création de tâches de streaming sans qu’il soit nécessaire d’apprendre un nouveau langage ou un nouveau cadre. En tirant parti de vos connaissances et de votre infrastructure existantes, AWS Glue Streaming simplifie le processus de développement des tâches et vous permet d'étendre facilement vos capacités de traitement des données à des scénarios de streaming en temps réel.
2. Si vous avez besoin d'un service ou d'un produit unifié pour gérer les charges de travail par lots, de streaming et d'événements, AWS Glue Streaming est la solution qu'il vous faut. Grâce au AWS Glue streaming, vous pouvez regrouper vos besoins en matière de traitement des données dans un cadre unique, éliminant ainsi la complexité liée à la gestion de plusieurs systèmes. Cela permet le développement et la maintenance efficaces de divers flux de données tout en garantissant la cohérence et la compatibilité entre les différents types de charge de travail.
3. AWS Glue Le streaming convient parfaitement aux scénarios impliquant des volumes de données en streaming extrêmement importants et des transformations complexes, telles que des jointures entre des flux ou des bases de données relationnelles. Il peut traiter et analyser efficacement des flux de données massifs, ce qui vous permet de gérer facilement des charges de travail exigeantes. Qu'il s'agisse d'une ingestion de données à grande vitesse ou de manipulations de données complexes, l'évolutivité et les capacités de traitement avancées du AWS Glue streaming garantissent des performances optimales et des résultats précis.
4. Si vous préférez une approche visuelle pour créer des jobs de streaming, Studio vous AWS Glue propose AWS Glue Studio, avec lequel vous pouvez concevoir et gérer visuellement vos applications de streaming, simplifiant ainsi le processus de développement. Cette interface intuitive

permet aux développeurs de créer, configurer et surveiller les flux de travail de streaming à l’aide d’une interface visuelle, réduisant ainsi la courbe d’apprentissage et augmentant la productivité.

5. AWS Glue Le streaming est un excellent choix pour les cas near-real-time d'utilisation où des accords de niveau de service stricts SLAs (accords de niveau de service) sont supérieurs à 10 secondes.
6. Si vous créez un lac de données transactionnel à l'aide d'Apache Iceberg, Apache Hudi ou Delta Lake, AWS Glue Streaming fournit un support natif pour ces formats de table ouverts. Cette intégration fluide vous permet de traiter les données de streaming directement à partir de ces lacs de données transactionnels, garantissant ainsi la cohérence, l’intégrité et la compatibilité des données.
7. Lorsque vous devez ingérer des données de streaming pour diverses cibles de données : le AWS Glue streaming fournit des cibles natives à diverses cibles de données telles qu'Amazon Redshift, Amazon RDS, Amazon Aurora, Oracle, SQL Server et d'autres cibles.

Sources de données prises en charge

AWS Glue Le streaming prend en charge les sources de données suivantes :

- Amazon Kinesis
- Amazon MSK (Managed Streaming for Apache Kafka)
- Self-managed Apache Kafka

Cibles de données prises en charge

AWS Glue Le streaming prend en charge diverses cibles de données, telles que :

- Cibles de données prises en charge par AWS Glue Data Catalog
- Amazon S3
- Amazon Redshift
- MySQL
- PostgreSQL
- Oracle
- Microsoft SQL Server

- Snowflake
- Toute base de données pouvant être connectée à l'aide de JDBC
- Apache Iceberg, Delta et Apache Hudi
- AWS Glue Connecteurs Marketplace

Tutoriel : Créez votre première charge de travail de streaming à l'aide de AWS Glue Studio

Dans ce didacticiel, vous allez apprendre à créer une tâche de streaming à l'aide de AWS Glue Studio. AWS Glue Studio est une interface visuelle permettant de créer des AWS Glue emplois.

Vous pouvez créer des tâches d'extraction, de transformation et de chargement (ETL) en streaming qui s'exécutent en continu et consomment des données provenant de sources en streaming dans Amazon Kinesis Data Streams, Apache Kafka et Amazon Managed Streaming for Apache Kafka (Amazon MSK).

Conditions préalables

Pour suivre ce didacticiel, vous aurez besoin d'un utilisateur autorisé à utiliser AWS la console AWS Glue, Amazon Kinesis, Amazon S3, Amazon Athena AWS CloudFormation, AWS Lambda et Amazon Cognito.

Utilisation des données de streaming à partir d'Amazon Kinesis

Rubriques

- Génération de données fictives avec Kinesis Data Generator
- Création d'une tâche de AWS Glue streaming avec AWS Glue Studio
- Exécution d'une transformation et stockage du résultat transformé dans Amazon S3

Génération de données fictives avec Kinesis Data Generator

Vous pouvez générer des exemples de données de manière synthétique au format JSON à l'aide de Kinesis Data Generator (KDG). Vous trouverez des instructions complètes et des détails dans la documentation de l'outil.

1. Pour commencer, cliquez

   pour exécuter un AWS CloudFormation modèle dans votre AWS environnement.

   > **Note**
   >
   > Il se peut que le CloudFormation modèle ne fonctionne pas, car certaines ressources, telles que l'utilisateur Amazon Cognito pour Kinesis Data Generator, existent déjà dans votre compte. AWS Cela peut être dû au fait que vous l'avez déjà configuré à partir d'un autre didacticiel ou blog. Pour résoudre ce problème, vous pouvez soit essayer le modèle dans un nouveau AWS compte pour prendre un nouveau départ, soit explorer une autre AWS région. Ces options vous permettent d'exécuter le didacticiel sans entrer en conflit avec les ressources existantes.

   Le modèle vous fournit un flux de données Kinesis et un compte Kinesis Data Generator. Il crée également un compartiment Amazon S3 pour contenir les données et une fonction du service Glue avec les autorisations requises pour ce didacticiel.
2. Saisissez un Nom d'utilisateur et un Mot de passe que le KDG utilisera pour s'authentifier. Notez le nom d'utilisateur et le mot de passe pour une utilisation ultérieure.
3. Sélectionnez Suivant jusqu'à la dernière étape. Reconnaissez la création de ressources IAM. Vérifiez les erreurs en haut de l'écran, par exemple un mot de passe ne répondant pas aux exigences minimales, et déployez le modèle.
4. Accédez à l'onglet Sorties de la pile. Une fois le modèle déployé, il affiche la propriété générée KinesisDataGeneratorUrl. Cliquez sur cette URL.
5. Saisissez le Nom d'utilisateur et le Mot de passe que vous avez notés.
6. Sélectionnez la région que vous utilisez et sélectionnez l'`GlueStreamTest-{AWS::AccountId}` de flux Kinesis.
7. Saisissez le modèle suivant :

```
{
    "ventilatorid": {{random.number(100)}},
    "eventtime": "{{date.now("YYYY-MM-DD HH:mm:ss")}}",
    "serialnumber": "{{random.uuid}}",
    "pressurecontrol": {{random.number(
        {
```

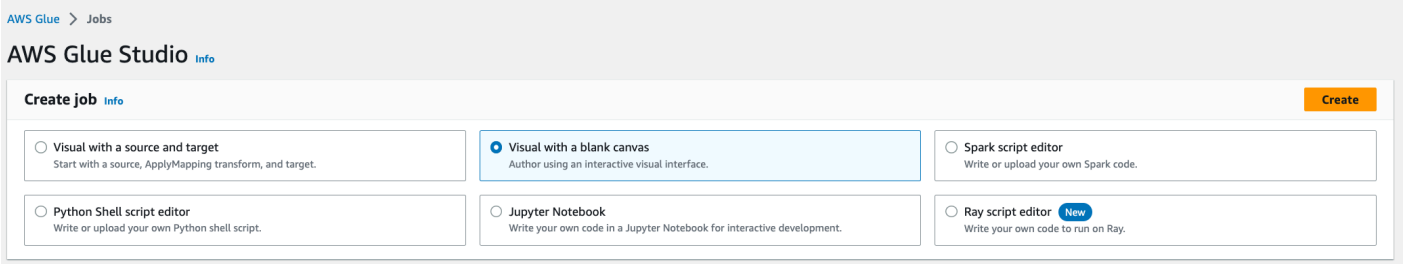
```
            "min":5,
            "max":30
        }
    )}},
    "o2stats": {{random.number(
        {
            "min":92,
            "max":98
        }
    )}},
    "minutevolume": {{random.number(
        {
            "min":5,
            "max":8
        }
    )}},
    "manufacturer": "{{random.arrayElement(
        ["3M", "GE","Vyaire", "Getinge"]
    )}}"
}
```

Vous pouvez désormais afficher des données fictives avec Modèle de test et ingérer les données fictives dans Kinesis avec Envoyer des données.

8. Cliquez sur Envoyer des données et générez 5 000 à 10 000 enregistrements vers Kinesis.

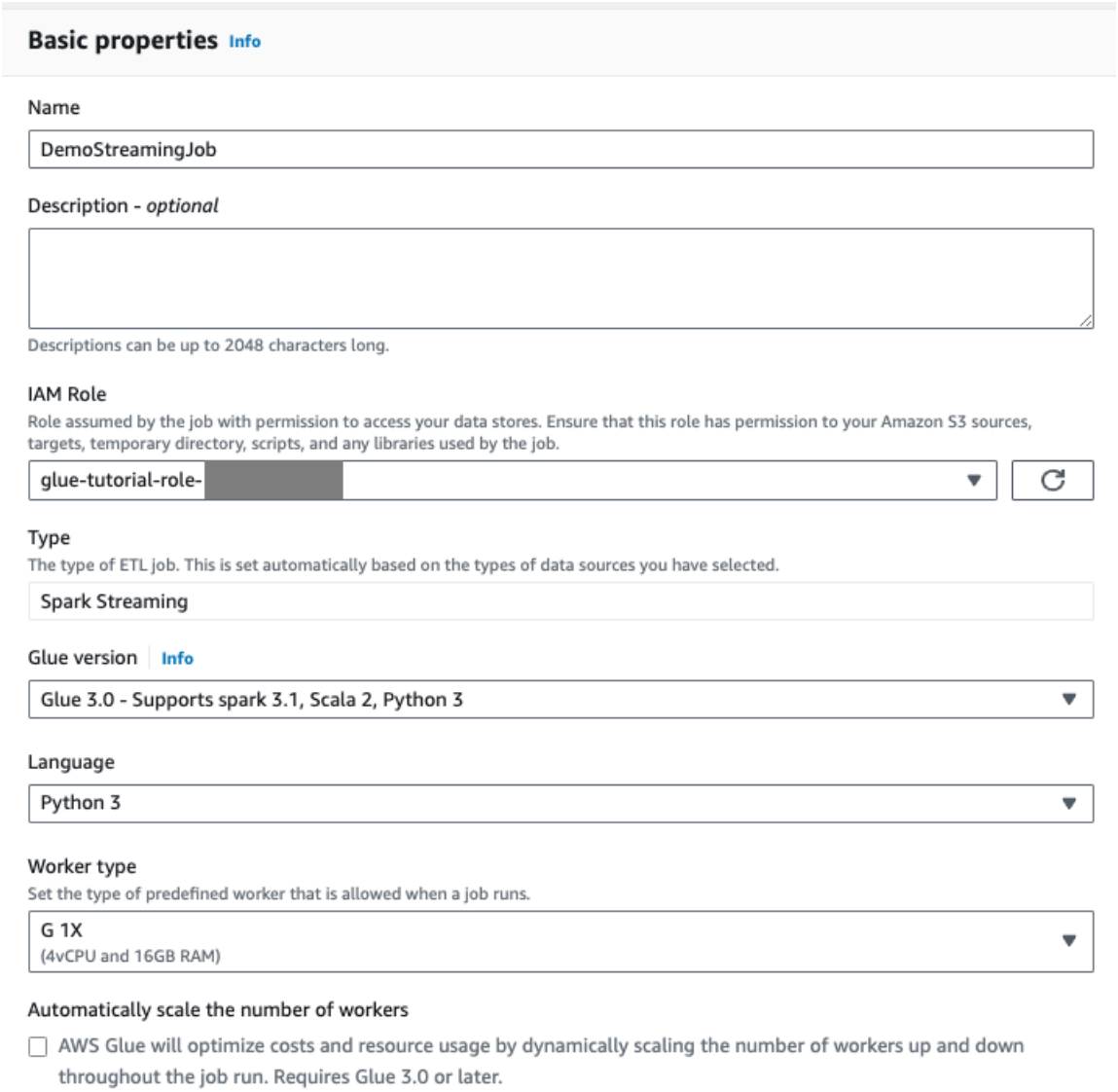
Création d'une tâche de AWS Glue streaming avec AWS Glue Studio

1. Accédez AWS Glue à la console dans la même région.
2. Sélectionnez Tâches ETL dans la barre de navigation de gauche, sous Intégration de données et ETL.
3. Créez un AWS Glue Job via Visual avec un canevas vierge.

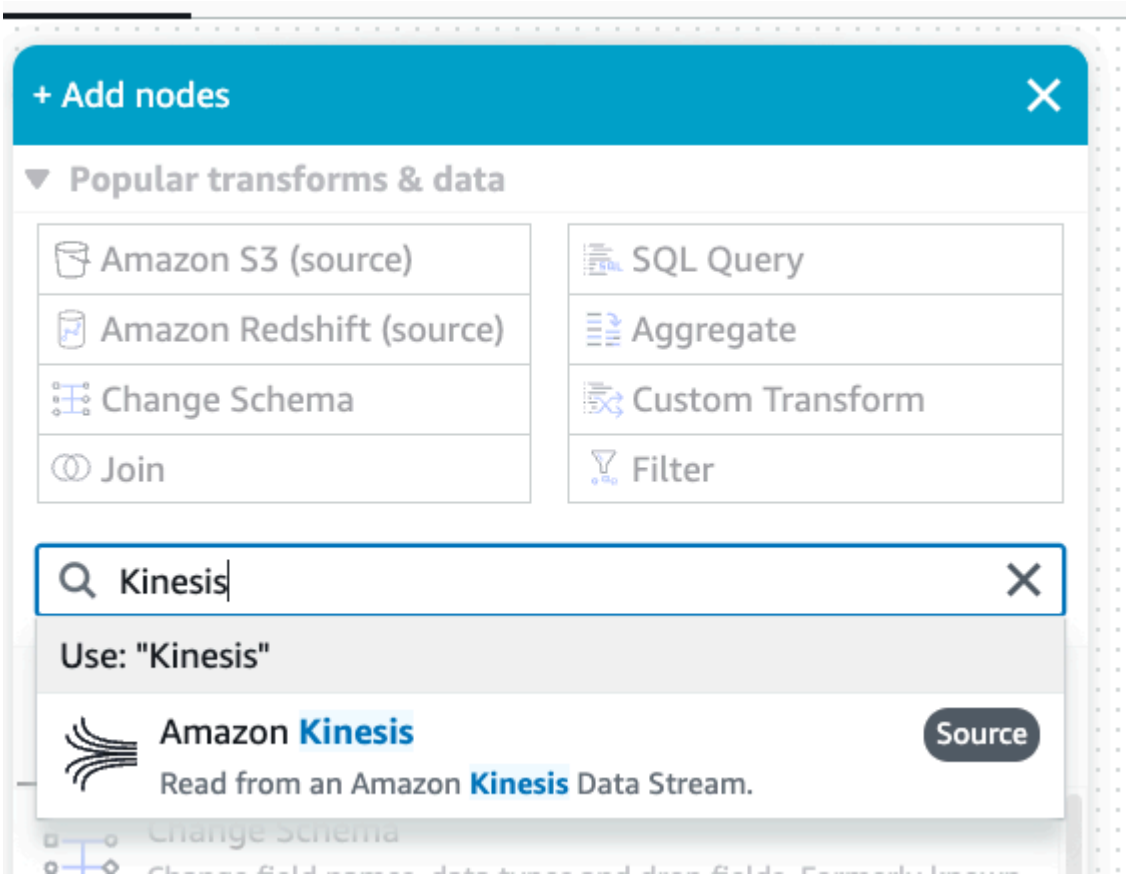


4. Accédez à l'onglet Détails de la tâche.
5. Pour le nom de la AWS Glue tâche, entrez`DemoStreamingJob`.

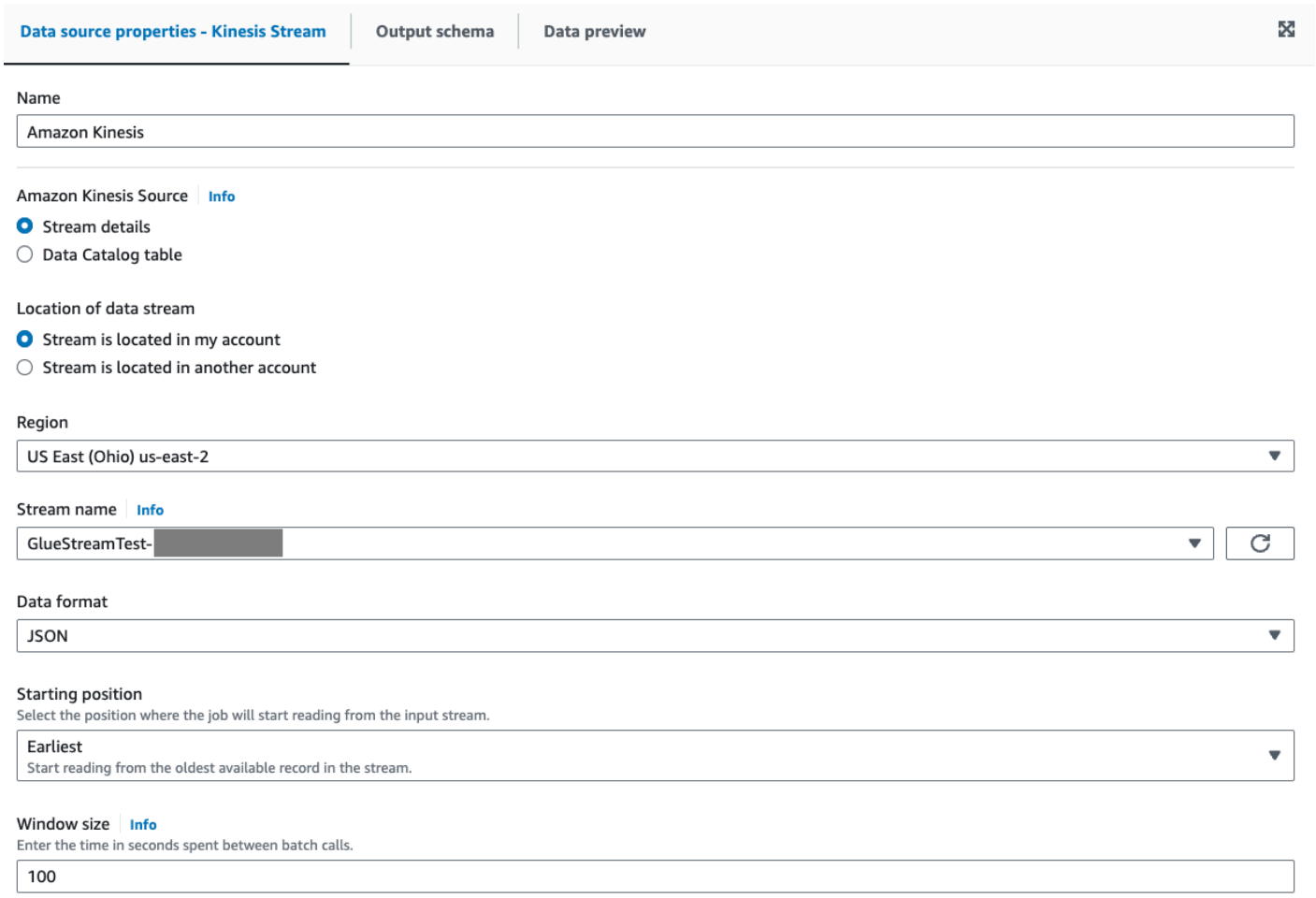
6. Pour le rôle IAM, sélectionnez le rôle fourni par le CloudFormation modèle,. `glue-tutorial-role-${AWS::AccountId}`
7. Pour Version Glue, sélectionnez Glue 3.0. Conservez toutes valeurs par défaut des autres options.



8. Accédez à l'onglet Visuel.
9. Cliquez sur l'icône plus. Saisissez Kinesis dans la barre de recherche. Sélectionnez la source de données Amazon Kinesis.



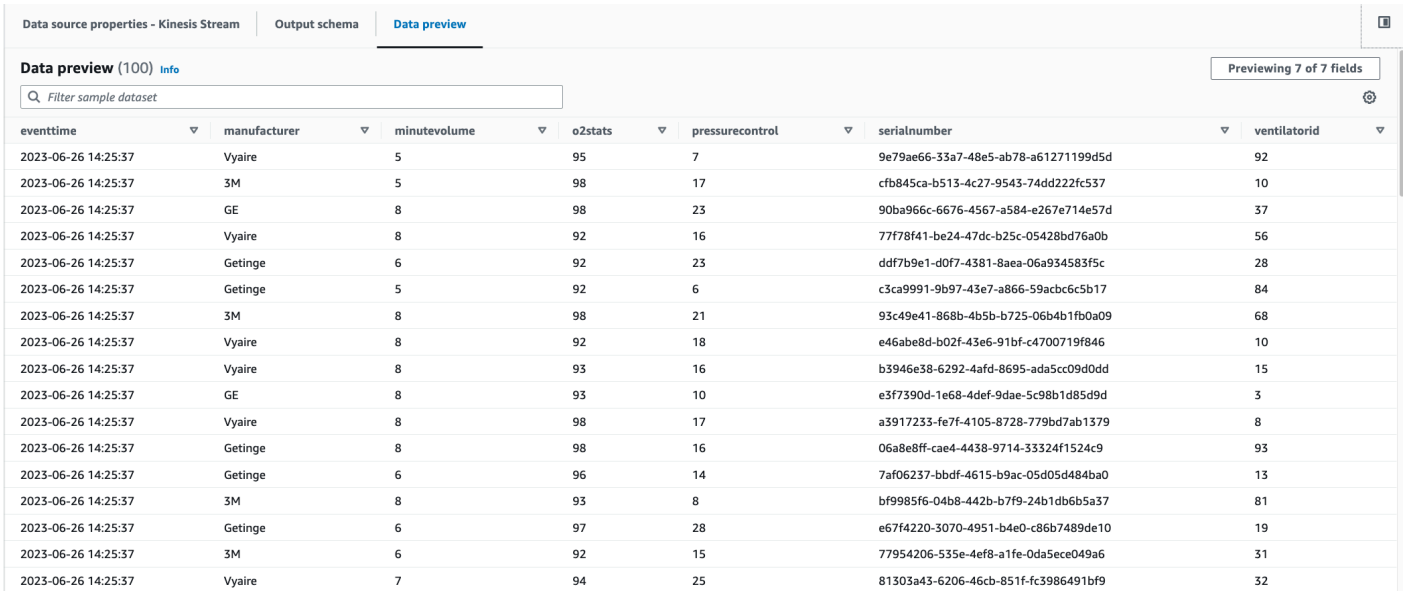
10. Sélectionnez Détails du flux pour Source Amazon Kinesis sous l'onglet Propriétés de la source de données – Flux Kinesis.
11. Sélectionnez Le flux se trouve sur mon compte pour Emplacement du flux de données.
12. Sélectionnez la région que vous utilisez.
13. Sélectionnez le flux `GlueStreamTest-{AWS::AccountId}`.
14. Conservez tous les autres paramètres par défaut.



15. Accédez à l'onglet Aperçu des données.
16. Cliquez sur Démarrer la session d'aperçu des données, qui affiche un aperçu des données fictives générées par KDG. Choisissez le rôle de service Glue que vous avez créé précédemment pour la tâche de AWS Glue streaming.

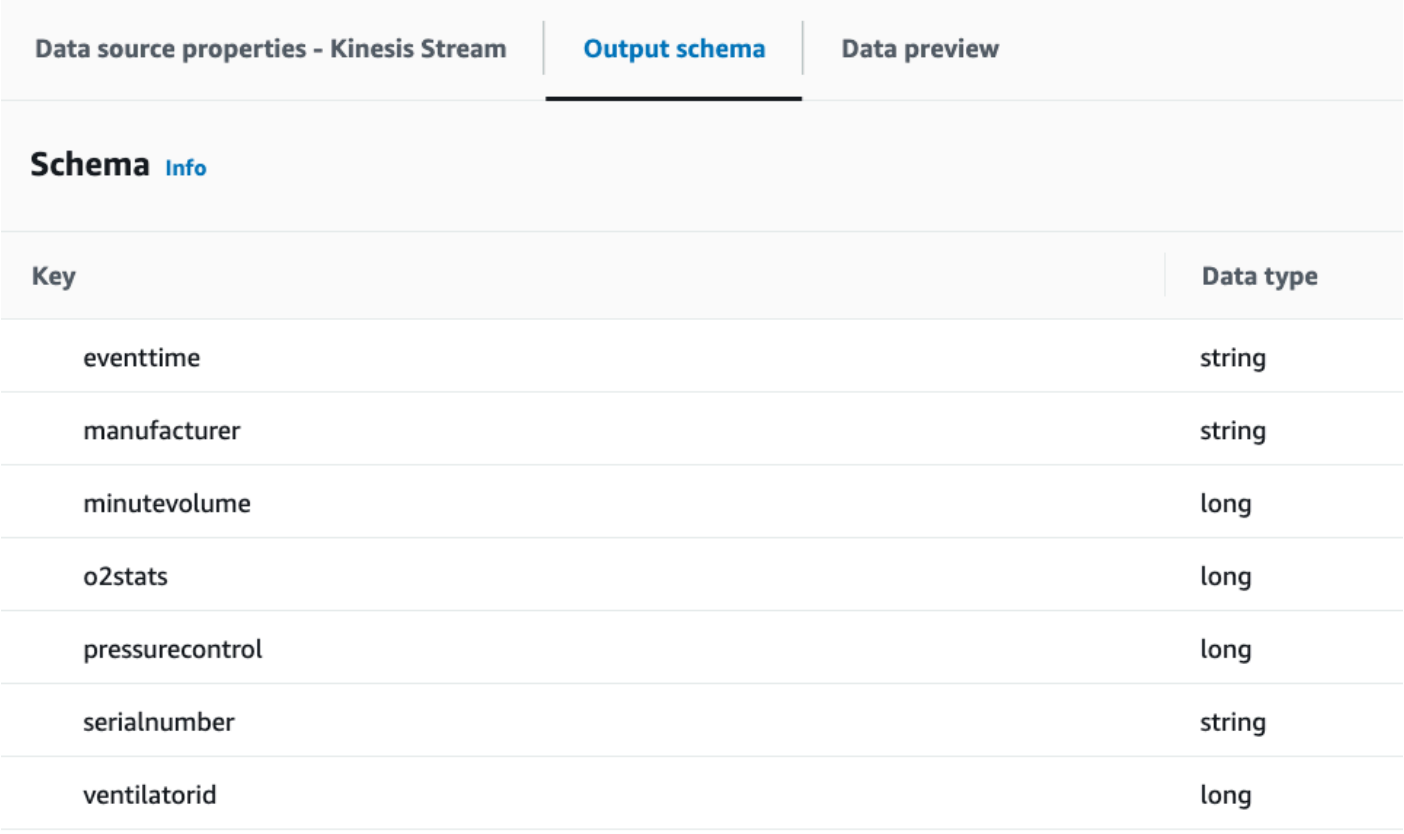
    Il faut 30 à 60 secondes pour que les données d'aperçu s'affichent. Si Aucune donnée à afficher s'affiche, cliquez sur l'icône en forme de rouage et définissez Nombre de lignes à échantillonner sur `100`.

    Vous pouvez voir les exemples de données ci-dessous :



eventtime	manufacturer	minutevolume	o2stats	pressurecontrol	serialnumber	ventilatorid
2023-06-26 14:25:37	Vyaire	5	95	7	9e79ae66-33a7-48e5-ab78-a61271199d5d	92
2023-06-26 14:25:37	3M	5	98	17	cfb845ca-b513-4c27-9543-74dd222fc537	10
2023-06-26 14:25:37	GE	8	98	23	90ba966c-6676-4567-a584-e267e714e57d	37
2023-06-26 14:25:37	Vyaire	8	92	16	77f78f41-be24-47dc-b25c-05428bd76a0b	56
2023-06-26 14:25:37	Getinge	6	92	23	ddf7b9e1-d0f7-4381-8aea-06a934583f5c	28
2023-06-26 14:25:37	Getinge	5	92	6	c3ca9991-9b97-43e7-a866-59acbc6c5b17	84
2023-06-26 14:25:37	3M	8	98	21	93c49e41-868b-4b5b-b725-06b4b1fb0a09	68
2023-06-26 14:25:37	Vyaire	8	92	18	e46abe8d-b02f-43e6-91bf-c4700719f846	10
2023-06-26 14:25:37	Vyaire	8	93	16	b3946e38-6292-4afd-8695-ada5cc09d0dd	15
2023-06-26 14:25:37	GE	8	93	10	e3f7390d-1e68-4def-9dae-5c98b1d85d9d	3
2023-06-26 14:25:37	Vyaire	8	98	17	a3917233-fe7f-4105-8728-779bd7ab1379	8
2023-06-26 14:25:37	Getinge	8	98	16	06a8e8ff-cae4-4438-9714-33324f1524c9	93
2023-06-26 14:25:37	Getinge	6	96	14	7af06237-bbdf-4615-b9ac-05d05d484ba0	13
2023-06-26 14:25:37	3M	8	93	8	bf9985f6-04b8-442b-b7f9-24b1db6b5a37	81
2023-06-26 14:25:37	Getinge	6	97	28	e67f4220-3070-4951-b4e0-c86b7489de10	19
2023-06-26 14:25:37	3M	6	92	15	77954206-535e-4ef8-a1fe-0da5ece049a6	31
2023-06-26 14:25:37	Vyaire	7	94	25	81303a43-6206-46cb-851f-fc3986491bf9	32

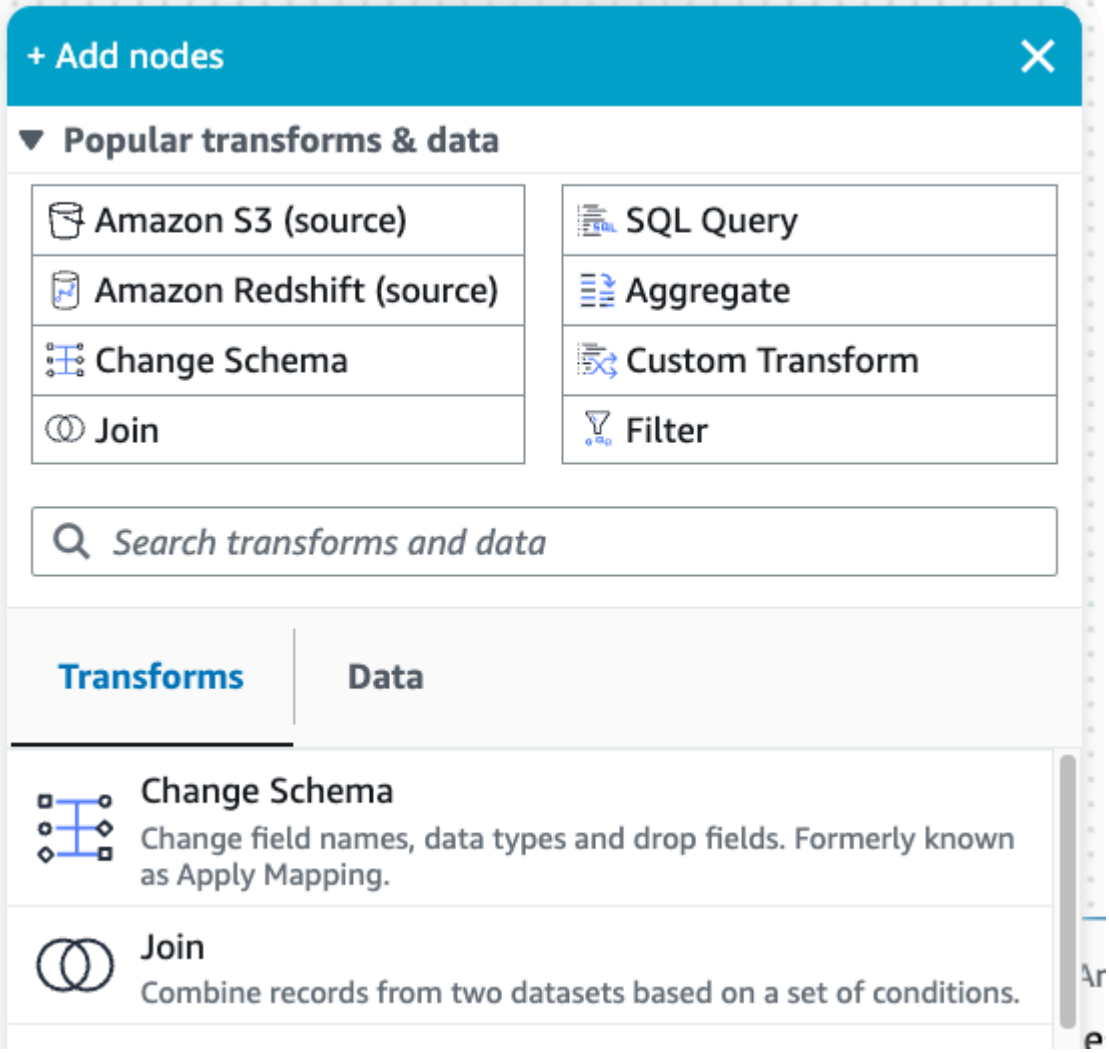
Vous pouvez également voir le schéma déduit dans l'onglet Schéma de sortie.



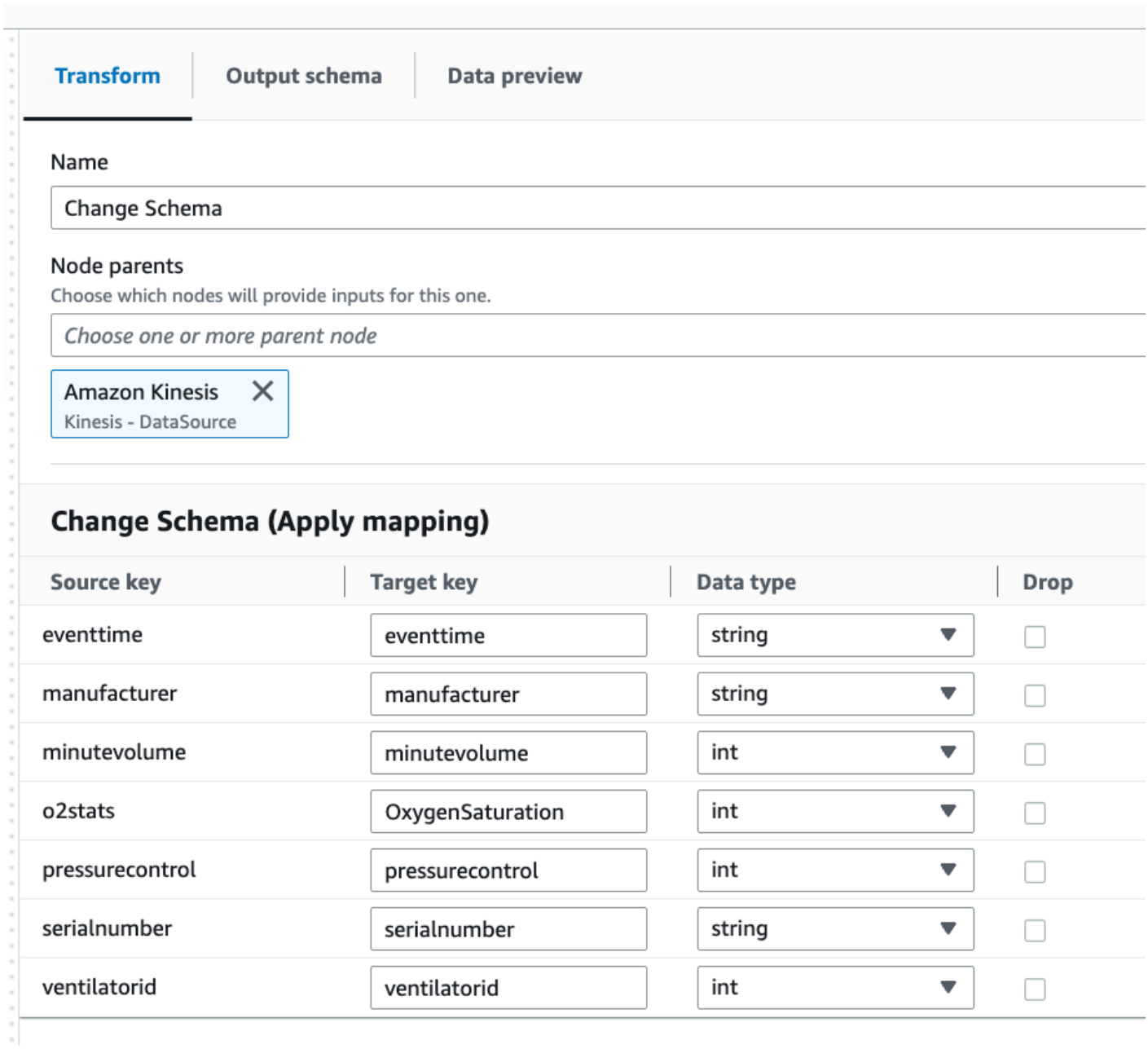
Key	Data type
eventtime	string
manufacturer	string
minutevolume	long
o2stats	long
pressurecontrol	long
serialnumber	string
ventilatorid	long

Exécution d'une transformation et stockage du résultat transformé dans Amazon S3

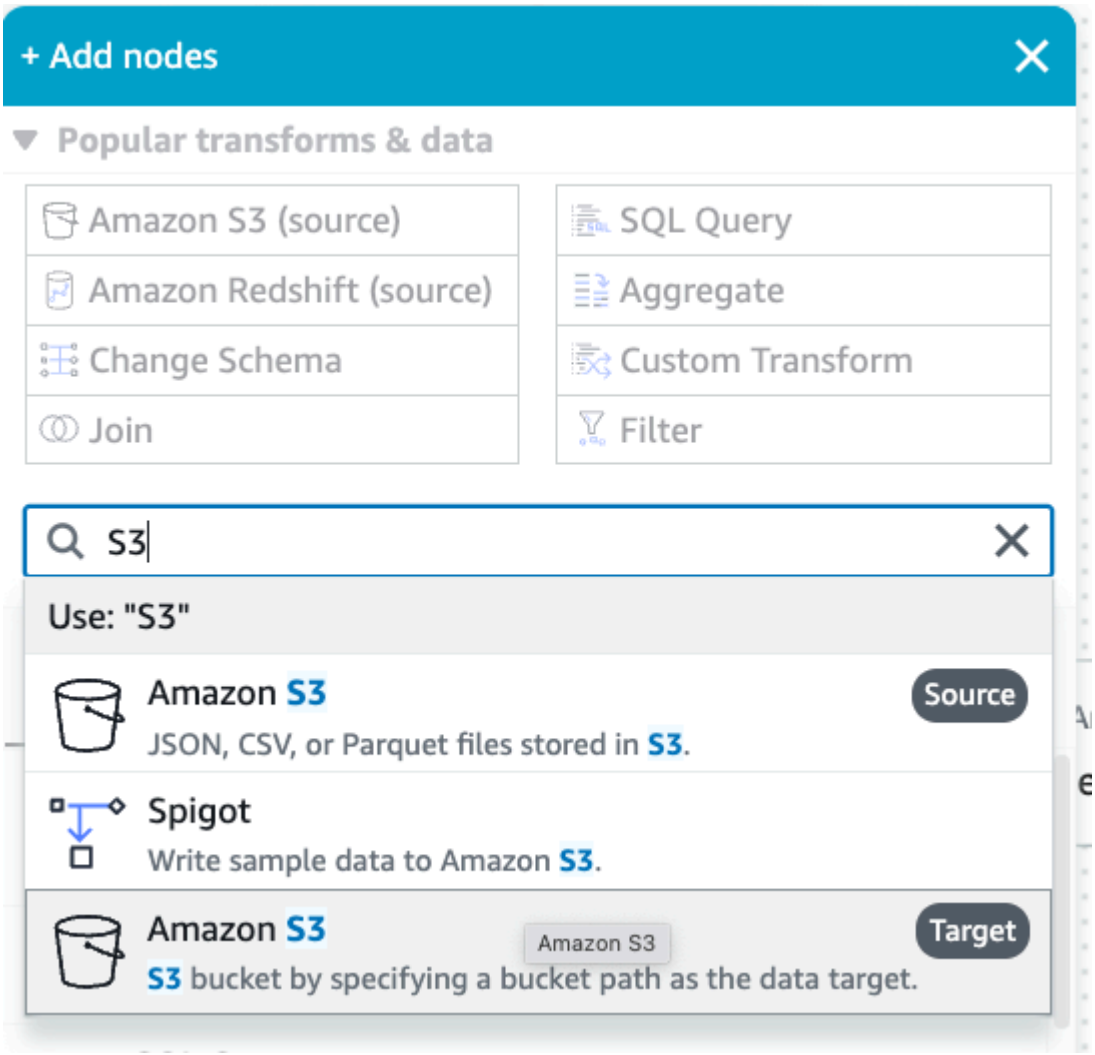
1. Une fois le nœud source sélectionné, cliquez sur l'icône plus en haut à gauche pour ajouter une étape Transformations.
2. Sélectionnez l'étape Modifier le schéma.



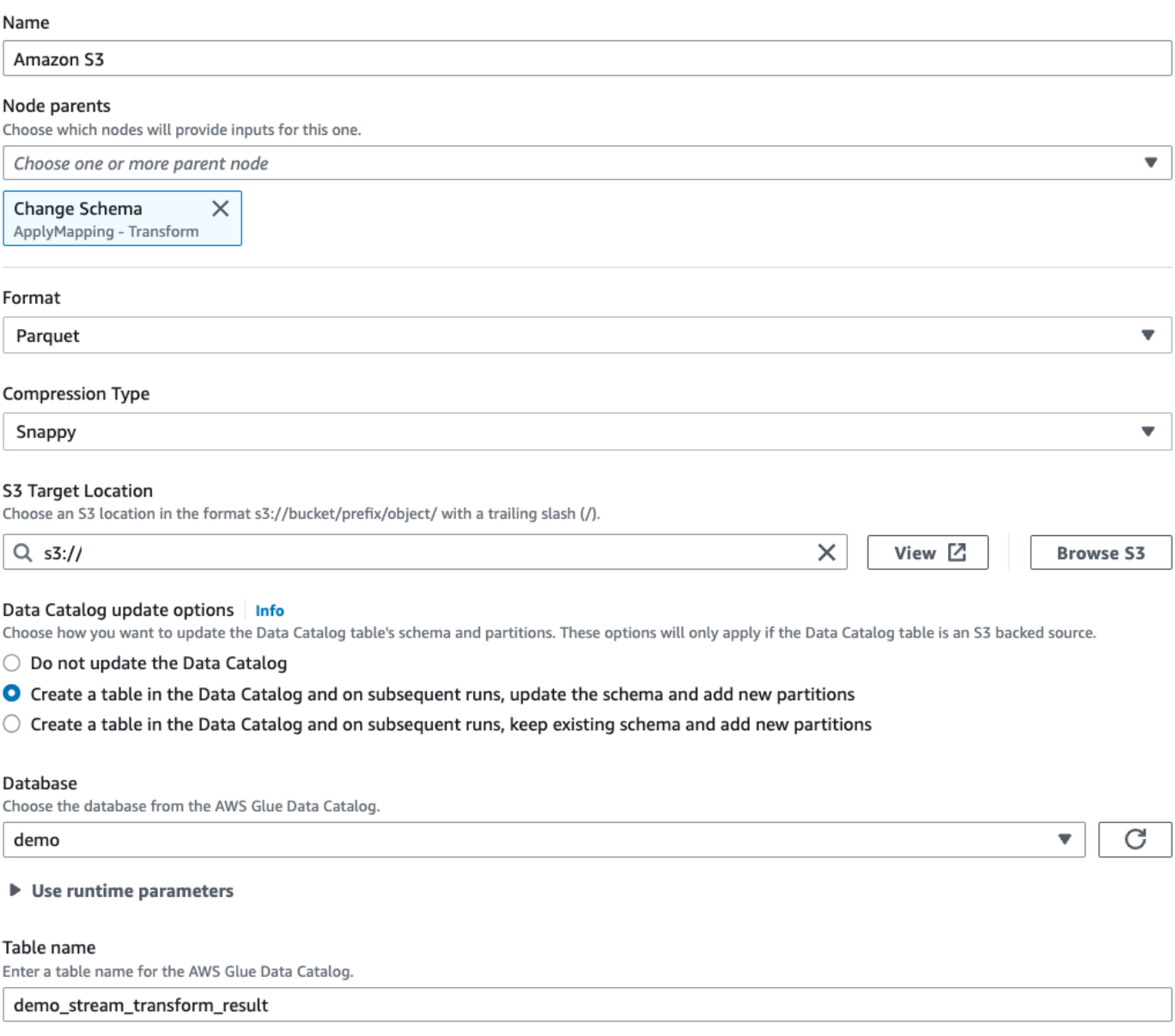
3. Vous pouvez renommer les champs et convertir le type de données des champs au cours de cette étape. Renommez la colonne `o2stats` en `OxygenSaturation` et convertissez tous les types de données `long` en `int`.



4. Cliquez sur l'icône plus pour ajouter une cible Amazon S3. Saisissez S3 dans le champ de recherche et sélectionnez l'étape de transformation Amazon S3 – Cible.

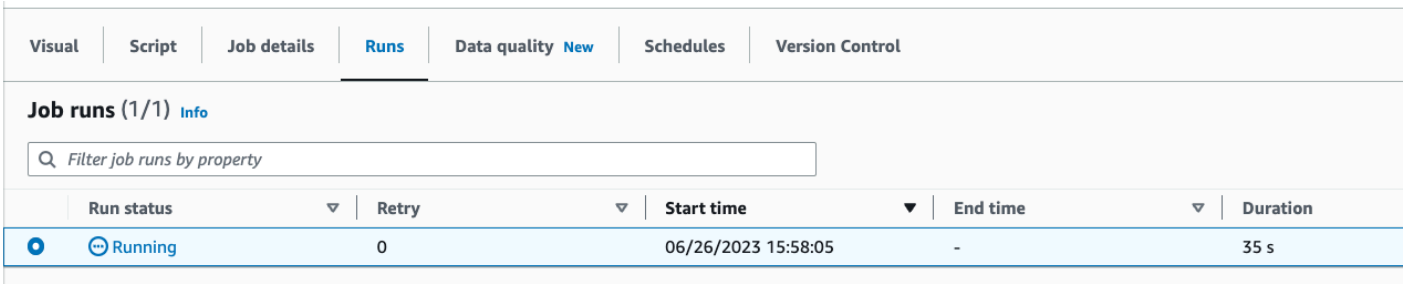


5. Sélectionnez Parquet comme format de fichier cible.
6. Sélectionnez Snappy comme type de compression.
7. Entrez un emplacement cible S3 créé par le CloudFormation modèle,`streaming-tutorial-s3-target-{AWS::AccountId}`.
8. Sélectionnez Créer une table dans le catalogue de données et lors des exécutions suivantes, mettre à jour le schéma et ajouter de nouvelles partitions.
9. Saisissez la Base de données cible et le nom de la Table pour stocker le schéma de la table cible Amazon S3.

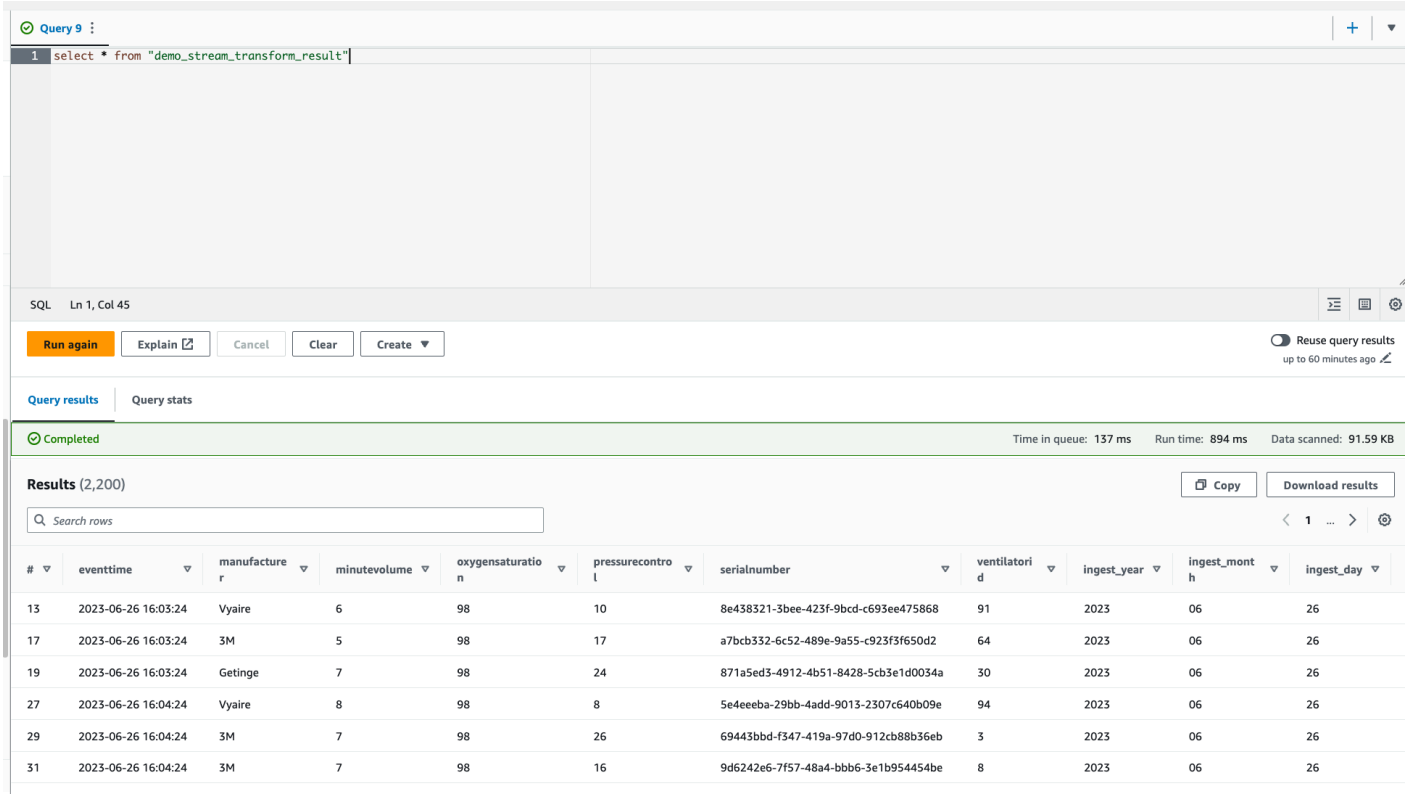


10. Cliquez sur l'onglet Script pour afficher le code généré.
11. Cliquez sur Enregistrer en haut à droite pour enregistrer le code ETL, puis sur Exécuter pour lancer la tâche de AWS Glue streaming.

    Vous pouvez trouver le Statut d'exécution dans l'onglet Exécutions. Laissez la tâche s'exécuter pendant trois à cinq minutes, puis arrêtez-la.



12.Vérifiez la nouvelle table créée dans Amazon Athena.



Tutoriel : créez votre première charge de travail de streaming à l'aide des blocs-notes AWS Glue Studio

Dans ce didacticiel, vous découvrirez comment tirer parti des blocs-notes AWS Glue Studio pour créer et affiner de manière interactive vos tâches ETL pour le traitement des données en temps quasi réel. Que vous débutiez AWS Glue ou que vous souhaitiez améliorer vos compétences, ce guide vous guidera tout au long du processus et vous permettra d'exploiter tout le potentiel des carnets de session AWS Glue interactifs.

Avec AWS Glue Streaming, vous pouvez créer des tâches d'extraction, de transformation et de chargement (ETL) exécutées en continu et consommant des données provenant de sources de streaming telles qu'Amazon Kinesis Data Streams, Apache Kafka et Amazon Managed Streaming for Apache Kafka (Amazon MSK).

Conditions préalables

Pour suivre ce didacticiel, vous aurez besoin d'un utilisateur autorisé à utiliser AWS la console AWS Glue, Amazon Kinesis, Amazon S3, Amazon Athena AWS CloudFormation, AWS Lambda et Amazon Cognito.

Utilisation des données de streaming à partir d'Amazon Kinesis

Rubriques

- Génération de données fictives avec Kinesis Data Generator
- Création d'une tâche de AWS Glue streaming avec AWS Glue Studio
- Nettoyage
- Conclusion

Génération de données fictives avec Kinesis Data Generator

> **Note**
>
> Si vous avez déjà terminé le précédent Tutoriel : Créez votre première charge de travail de streaming à l'aide de AWS Glue Studio, Kinesis Data Generator est déjà installé sur votre compte et vous pouvez ignorer les étapes 1 à 8 ci-dessous et passer à la section Création d'une tâche de AWS Glue streaming avec AWS Glue Studio.

Vous pouvez générer des exemples de données de manière synthétique au format JSON à l'aide de Kinesis Data Generator (KDG). Vous trouverez des instructions complètes et des détails dans la documentation de l'outil.

1. Pour commencer, cliquez

   pour exécuter un AWS CloudFormation modèle dans votre AWS environnement.

> **Note**
>
> Il se peut que le CloudFormation modèle ne fonctionne pas, car certaines ressources, telles que l'utilisateur Amazon Cognito pour Kinesis Data Generator, existent déjà dans votre compte. AWS Cela peut être dû au fait que vous l'avez déjà configuré à partir d'un autre didacticiel ou blog. Pour résoudre ce problème, vous pouvez soit essayer le modèle dans un nouveau AWS compte pour prendre un nouveau départ, soit explorer une autre AWS région. Ces options vous permettent d'exécuter le didacticiel sans entrer en conflit avec les ressources existantes.

Le modèle vous fournit un flux de données Kinesis et un compte Kinesis Data Generator.

2. Saisissez un Nom d'utilisateur et un Mot de passe que le KDG utilisera pour s'authentifier. Notez le nom d'utilisateur et le mot de passe pour une utilisation ultérieure.
3. Sélectionnez Suivant jusqu'à la dernière étape. Reconnaissez la création de ressources IAM. Vérifiez les erreurs en haut de l'écran, par exemple un mot de passe ne répondant pas aux exigences minimales, et déployez le modèle.
4. Accédez à l'onglet Sorties de la pile. Une fois le modèle déployé, il affiche la propriété générée KinesisDataGeneratorUrl. Cliquez sur cette URL.
5. Saisissez le Nom d'utilisateur et le Mot de passe que vous avez notés.
6. Sélectionnez la région que vous utilisez et sélectionnez l'`GlueStreamTest-{AWS::AccountId}` de flux Kinesis.
7. Saisissez le modèle suivant :

```
{
    "ventilatorid": {{random.number(100)}},
    "eventtime": "{{date.now("YYYY-MM-DD HH:mm:ss")}}",
    "serialnumber": "{{random.uuid}}",
    "pressurecontrol": {{random.number(
        {
            "min":5,
            "max":30
        }
    )}},
    "o2stats": {{random.number(
        {
            "min":92,
```

```
            "max":98
        }
    )}},
    "minutevolume": {{random.number(
        {
            "min":5,
            "max":8
        }
    )}},
    "manufacturer": "{{random.arrayElement(
        ["3M", "GE","Vyaire", "Getinge"]
    )}}"
}
```

Vous pouvez désormais afficher des données fictives avec Modèle de test et ingérer les données fictives dans Kinesis avec Envoyer des données.

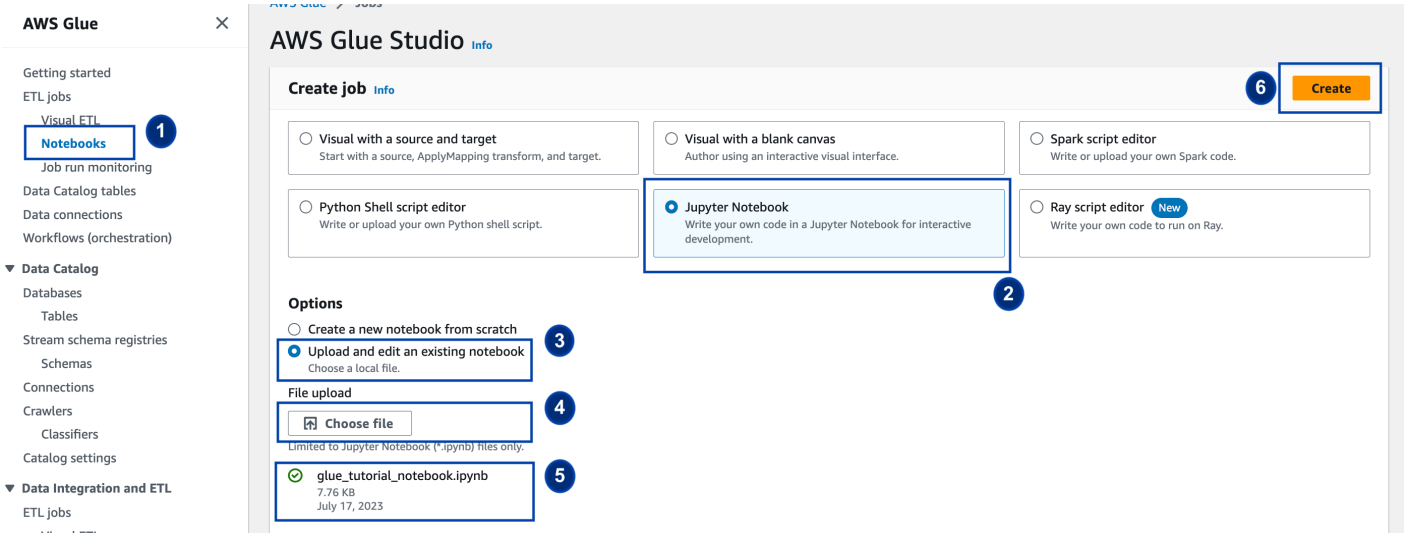
8. Cliquez sur Envoyer des données et générez 5 000 à 10 000 enregistrements vers Kinesis.

Création d'une tâche de AWS Glue streaming avec AWS Glue Studio

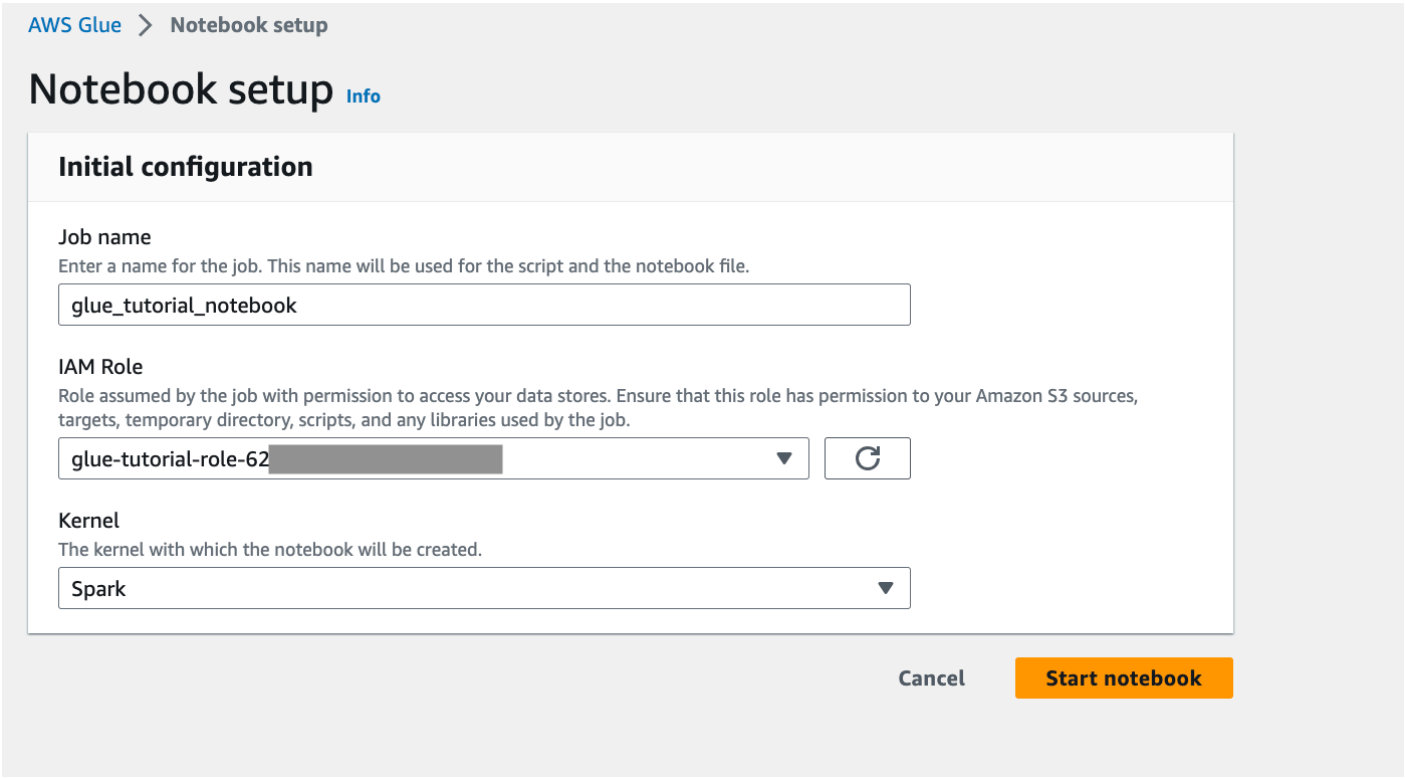
AWS Glue Studio est une interface visuelle qui simplifie le processus de conception, d'orchestration et de surveillance des pipelines d'intégration de données. Il permet aux utilisateurs de créer des pipelines de transformation de données sans écrire de code extensif. Outre l'expérience visuelle de création de tâches, AWS Glue Studio inclut également un bloc-notes Jupyter soutenu par des sessions AWS Glue interactives, que vous utiliserez dans le reste de ce didacticiel.

Configuration de la tâche AWS Glue Streaming interactive sessions

1. Téléchargez le fichier de bloc-notes fourni et enregistrez-le dans un répertoire local.
2. Ouvrez la AWS Glue console et dans le volet de gauche, cliquez sur Carnets de notes > Jupyter Notebook > Télécharger et modifier un bloc-notes existant. Chargez le bloc-notes de l'étape précédente et cliquez sur Créer.



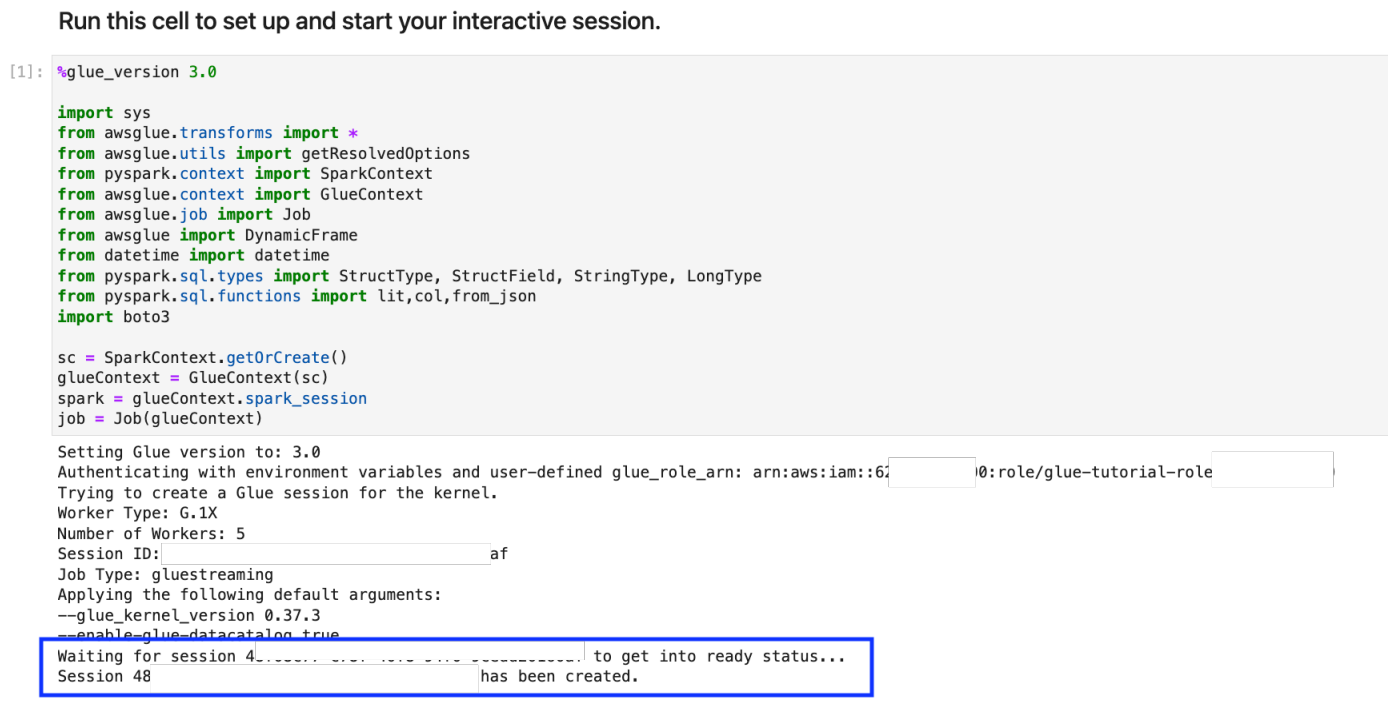
3. Donnez un nom et un rôle à la tâche et sélectionnez le noyau Spark par défaut. Cliquez ensuite sur Démarrer un bloc-notes. Pour le rôle IAM, sélectionnez le rôle fourni par le CloudFormation modèle. Vous pouvez le voir dans l'onglet Sorties de CloudFormation.



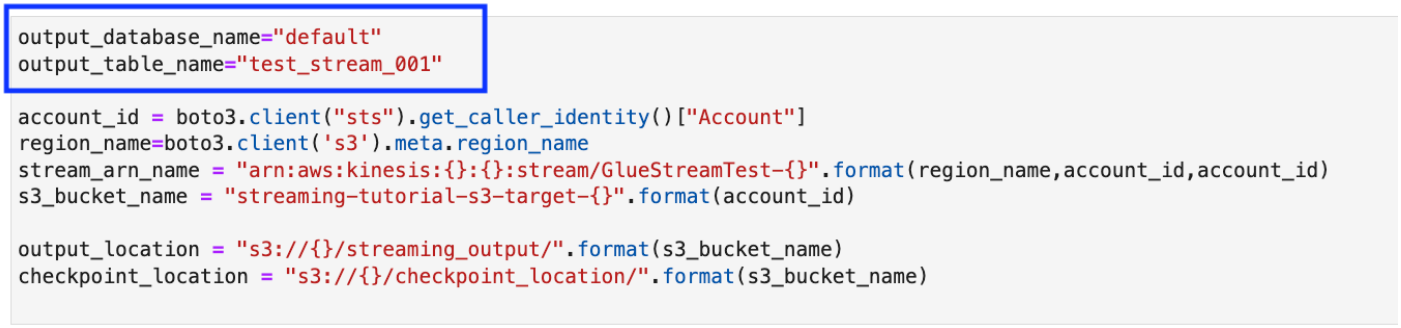
Le bloc-notes contient toutes les instructions nécessaires pour poursuivre le didacticiel. Vous pouvez soit exécuter les instructions sur le bloc-notes, soit suivre ce didacticiel pour poursuivre le développement de la tâche.

Exécution des cellules du bloc-notes

1. (Facultatif) La première cellule de code, `%help`, répertorie toutes les magies de blocs-notes disponibles. Vous pouvez ignorer cette case pour le moment, mais n'hésitez pas à la découvrir.
2. Commencez par le bloc de code suivant `%streaming`. Cette magie définit le type de tâche en streaming, ce qui vous permet de développer, de déboguer et de déployer une tâche ETL AWS Glue en streaming.
3. Exécutez la cellule suivante pour créer une session AWS Glue interactive. La cellule de sortie contient un message qui confirme la création de la session.



4. La cellule suivante définit les variables. Remplacez les valeurs par celles correspondant à votre tâche et exécutez la cellule. Par exemple :



5. Comme les données sont déjà diffusées vers Kinesis Data Streams, votre prochaine cellule utilisera les résultats du flux. Exécutez la cellule suivante. Comme il n'y a aucune instruction print, aucune sortie n'est attendue de cette cellule.

6. Dans la cellule suivante, vous explorez le flux entrant en prenant un ensemble d'échantillons et en imprimant son schéma et les données réelles. Par exemple :

Sample and print the incoming records

the sampling is for debugging purpose. You may comment off the entire code cell below, before deploying the actual code

```
[4]: options = {
        "pollingTimeInMs": "20000",
        "windowSize": "5 seconds"
     }
     sampled_dynamic_frame = glueContext.getSampleStreamingDynamicFrame(data_frame, options, None)

     count_of_sampled_records = sampled_dynamic_frame.count()

     print(count_of_sampled_records)

     sampled_dynamic_frame.printSchema()

     sampled_dynamic_frame.toDF().show(10,False)
```

```
100
root
|-- eventtime: string
|-- manufacturer: string
|-- minutevolume: long
|-- o2stats: long
|-- pressurecontrol: long
|-- serialnumber: string
|-- ventilatorid: long

+-------------------+------------+------------+-------+---------------+------------------------------------+------------+
|eventtime          |manufacturer|minutevolume|o2stats|pressurecontrol|serialnumber                        |ventilatorid|
+-------------------+------------+------------+-------+---------------+------------------------------------+------------+
|2023-07-18 10:20:11|3M          |6           |92     |24             |a3e860ba-24b9-41c4-bc10-91c6b35e1406|6           |
|2023-07-18 10:20:11|Vyaire      |6           |95     |6              |96101dca-3e88-457f-b390-e3291df48a81|26          |
|2023-07-18 10:20:12|Getinge     |8           |96     |24             |18f3d448-1dee-4c80-835b-1a0daa818915|22          |
|2023-07-18 10:20:12|Getinge     |7           |98     |30             |25f425cd-b978-4953-9a03-4d607a639364|91          |
|2023-07-18 10:20:12|GE          |5           |93     |25             |2cd7cdc2-f5f5-4ff2-ae32-45e5a8922d53|93          |
```

7. Ensuite, définissez la logique de transformation réelle des données. La cellule comprend la méthode `processBatch` qui est déclenchée lors de chaque microlot. Exécuter la cellule. À un niveau élevé, nous effectuons les actions suivantes sur le flux entrant :

   a. Sélectionner un sous-ensemble des colonnes d'entrée.

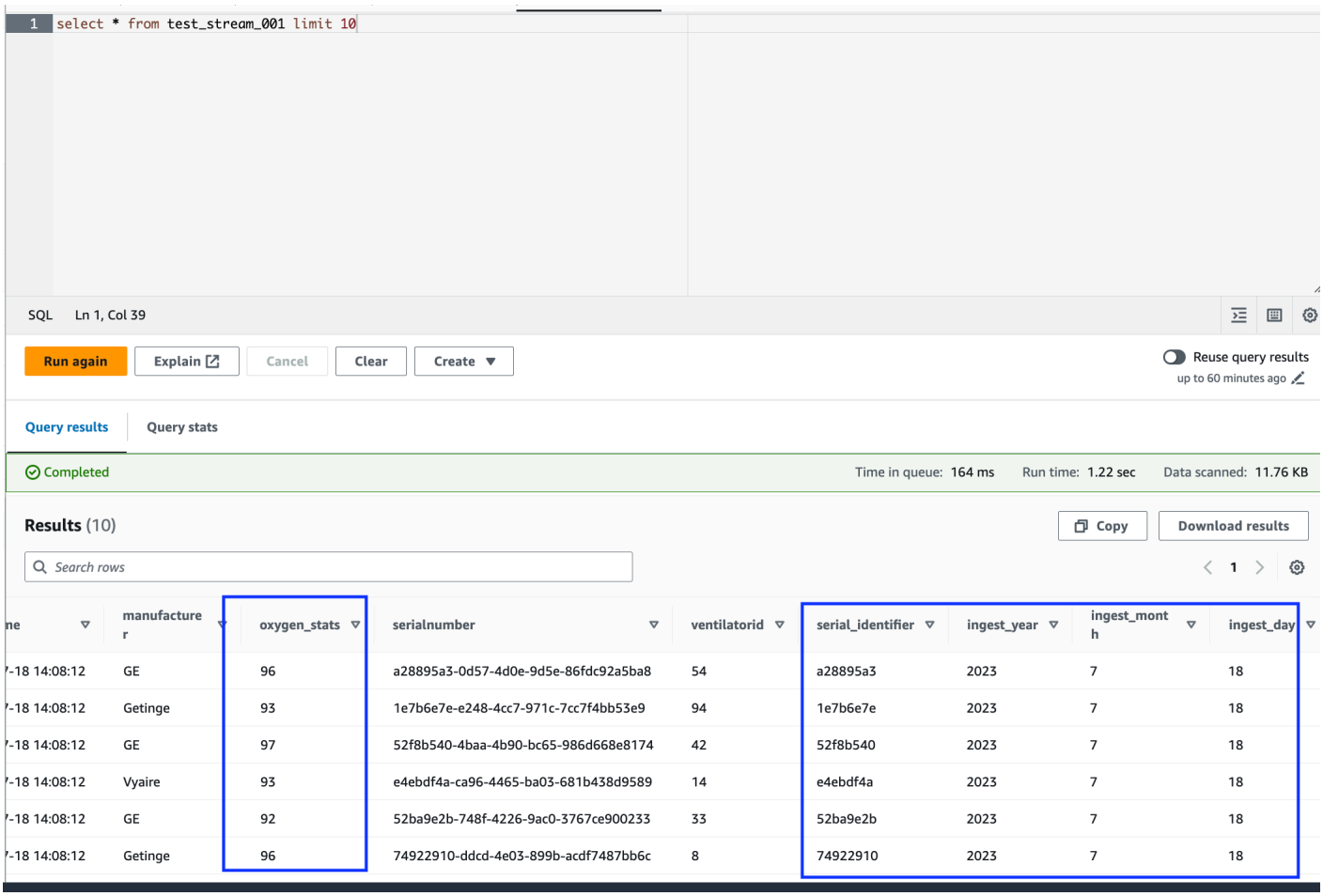
   b. Renommer une colonne (o2stats en oxygen_stats).

   c. Dériver de nouvelles colonnes (serial_identifier, ingest_year, ingest_month and ingest_day).

   d. Stockez les résultats dans un compartiment Amazon S3 et créez également une table de AWS Glue catalogue partitionnée

8. Dans la dernière cellule, vous déclenchez le traitement par lots toutes les 10 secondes. Exécutez la cellule et attendez environ 30 secondes pour qu'elle remplisse le compartiment Amazon S3 et la table du AWS Glue catalogue.

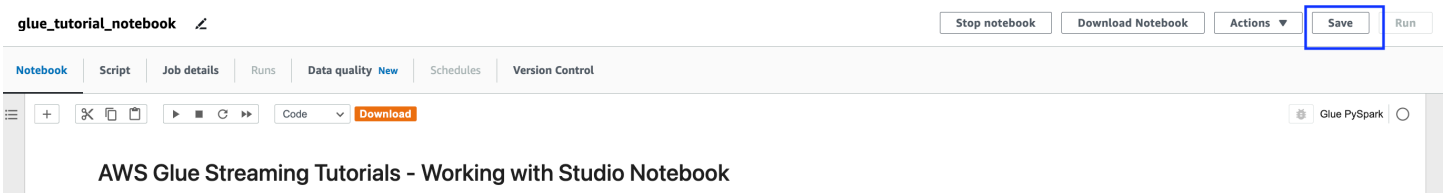
9. Enfin, parcourez les données stockées à l'aide de l'éditeur de requêtes Amazon Athena. Vous pouvez voir la colonne renommée ainsi que les nouvelles partitions.



Le bloc-notes contient toutes les instructions nécessaires pour poursuivre le didacticiel. Vous pouvez soit exécuter les instructions sur le bloc-notes, soit suivre ce didacticiel pour poursuivre le développement de la tâche.

Enregistrez et exécutez le AWS Glue travail

Une fois le développement et le test de votre application terminés à l'aide du bloc-notes de sessions interactives, cliquez sur Enregistrer en haut de l'interface du bloc-notes. Une fois enregistrée, vous pouvez également exécuter l'application en tant que tâche.



Nettoyage

Pour éviter d'entraîner des frais supplémentaires sur votre compte, arrêtez la tâche de streaming que vous avez commencée dans le cadre des instructions. Vous pouvez procéder en arrêtant le bloc-notes, ce qui mettra fin à la session. Videz le compartiment Amazon S3 et supprimez la AWS CloudFormation pile que vous avez provisionnée précédemment.

Conclusion

Dans ce didacticiel, nous avons montré comment effectuer les opérations suivantes à l'aide du bloc-notes AWS Glue Studio

- Création d'une tâche ETL en streaming à l'aide de blocs-notes
- Prévisualisation des flux de données entrants
- Codez et corrigez les problèmes sans avoir à publier des AWS Glue offres d'emploi
- Passez en revue le code de end-to-end travail, supprimez tout débogage et imprimez les instructions ou les cellules du bloc-notes
- Publier le code en tant que AWS Glue tâche

L'objectif de ce didacticiel est de vous donner une expérience pratique du AWS Glue streaming et des sessions interactives. Nous vous encourageons à l'utiliser comme référence pour vos cas d'utilisation individuels du AWS Glue streaming. Pour de plus amples informations, veuillez consulter Démarrage avec séances interactives AWS Glue.

AWS Glue Concepts de diffusion

Les sections suivantes fournissent des informations sur les concepts du AWS Glue streaming.

Rubriques

- Anatomie d'une tâche AWS Glue de streaming

Anatomie d'une tâche AWS Glue de streaming

AWS Glue les jobs de streaming fonctionnent selon le paradigme de streaming Spark et tirent parti du streaming structuré à partir du framework Spark. Les tâches de streaming interrogent en permanence la source de données de streaming, à un intervalle de temps spécifique, pour récupérer

les enregistrements sous forme de microlots. Les sections suivantes examinent les différentes parties d'une tâche de AWS Glue streaming.

forEachBatch

La `forEachBatch` méthode est le point d'entrée d'une tâche de AWS Glue streaming exécutée. AWS Glue les tâches de streaming utilisent la `forEachBatch` méthode d'interrogation des données qui fonctionne comme un itérateur qui reste actif pendant le cycle de vie de la tâche de streaming, interroge régulièrement la source de streaming à la recherche de nouvelles données et traite les données les plus récentes par microlots.

```
glueContext.forEachBatch(
    frame=dataFrame_AmazonKinesis_node1696872487972,
    batch_function=processBatch,
    options={
        "windowSize": "100 seconds",
        "checkpointLocation": args["TempDir"] + "/" + args["JOB_NAME"] + "/
checkpoint/",
    },
)
```

Configurez la propriété `frame` de `forEachBatch` pour spécifier une source de streaming. Dans cet exemple, le nœud source que vous avez créé dans le canevas vierge lors de la création de la tâche est renseigné avec la valeur par défaut DataFrame de la tâche. Définissez la propriété `batch_function` comme la `function` que vous décidez d'invoquer pour chaque opération de microlot. Vous devez définir une fonction pour gérer la transformation par lots des données entrantes.

Source

Dans la première étape de la `processBatch` fonction, le programme vérifie le nombre d'enregistrements de DataFrame ce que vous avez défini comme propriété de cadre de`forEachBatch`. Le programme ajoute un horodatage d'ingestion à un fichier non DataFrame vide. La clause `data_frame.count()>0` détermine si le dernier microlot n'est pas vide et est prêt pour un traitement ultérieur.

```
def processBatch(data_frame, batchId):
    if data_frame.count() >0:
        AmazonKinesis_node1696872487972 = DynamicFrame.fromDF(
            glueContext.add_ingestion_time_columns(data_frame, "hour"),
            glueContext,
            "from_data_frame",
        )
```

Mappage

La section suivante du programme consiste à appliquer le mappage. La `Mapping.apply` méthode sur une étincelle vous DataFrame permet de définir une règle de transformation autour des éléments de données. Vous pouvez généralement renommer, modifier le type de données ou appliquer une fonction personnalisée à la colonne de données source et les faire correspondre aux colonnes cibles.

```
    #Script generated for node ChangeSchema
    ChangeSchema_node16986872679326 = ApplyMapping.apply(
        frame =  AmazonKinesis_node1696872487972,
        mappings = [
            ("eventtime", "string", "eventtime", "string"),
            ("manufacturer", "string", "manufacturer", "string"),
            ("minutevolume", "long", "minutevolume", "int"),
```

```
            ("o2stats", "long", "OxygenSaturation", "int"),
            ("pressurecontrol", "long", "pressurecontrol", "int"),
            ("serialnumber", "string", "serialnumber", "string"),
            ("ventilatorid", "long", "ventilatorid", "long"),
            ("ingest_year", "string", "ingest_year", "string"),
            ("ingest_month", "string", "ingest_month", "string"),
            ("ingest_day", "string", "ingest_day", "string"),
            ("ingest_hour", "string", "ingest_hour", "string"),
        ],
        transformation_ctx="ChangeSchema_node16986872679326",
    )
        )
```

Sink

Dans cette section, le jeu de données entrant provenant de la source de streaming est stocké dans un emplacement cible. Dans cet exemple, nous écrivons les données dans un emplacement Amazon S3. Les détails de la propriété `AmazonS3_node_path` sont préremplis en fonction des paramètres que vous avez utilisés lors de la création de tâches à partir du canevas. Vous pouvez définir le `updateBehavior` en fonction de votre cas d'utilisation et décider soit de ne pas mettre à jour la table du catalogue de données, soit de créer le catalogue de données et de mettre à jour le schéma du catalogue de données lors des exécutions suivantes, soit de créer une table de catalogue sans mettre à jour la définition du schéma lors des exécutions suivantes.

La propriété `partitionKeys` définit l'option de partition de stockage. Le comportement par défaut consiste à partitionner les données conformément aux `ingestion_time_columns` mises à disposition dans la section source. La propriété `compression` vous permet de définir l'algorithme de compression à appliquer lors de l'écriture cible. Vous avez la possibilité de définir Snappy, LZO ou GZIP comme technique de compression. La propriété `enableUpdateCatalog` détermine si la table du catalogue AWS Glue doit être mise à jour. Les options disponibles pour cette propriété sont `True` ou `False`.

```
    #Script generated for node Amazon S3
    AmazonS3_node1696872743449 = glueContext.getSink(
        path =  AmazonS3_node1696872743449_path,
        connection_type = "s3",
        updateBehavior = "UPDATE_IN_DATABASE",
        partitionKeys = ["ingest_year", "ingest_month", "ingest_day", "ingest_hour"],
        compression = "snappy",
```

```
        enableUpdateCatalog = True,
        transformation_ctx = "AmazonS3_node1696872743449",
    )
```

AWS Glue évier de catalogue

Cette section de la tâche contrôle le comportement de mise à jour des tables de AWS Glue catalogue. Définissez `catalogDatabase` et `catalogTableName` propriétés selon le nom de votre base de données AWS Glue Catalog et le nom de table associé à la AWS Glue tâche que vous concevez. Vous pouvez définir le format de fichier des données cibles via la propriété `setFormat`. Dans cet exemple, nous allons stocker les données au format parquet.

Une fois que vous avez configuré et exécuté la tâche de AWS Glue streaming en vous référant à ce didacticiel, les données de streaming produites sur le site Amazon S3 Amazon Kinesis Data Streams seront stockées sur le site Amazon S3 dans un format parquet avec une compression rapide. En cas d'exécution réussie de la tâche de streaming, vous pourrez interroger les données via Amazon Athena.

```
AmazonS3_node1696872743449 = setCatalogInfo(
    catalogDatabase =  "demo", catalogTableName = "demo_stream_transform_result"
)
AmazonS3_node1696872743449.setFormat("glueparquet")
AmazonS3_node1696872743449.writeFormat("ChangeSchema_node16986872679326")
)
```

AWS Glue Connexions de streaming

Les sections suivantes fournissent des informations sur l'utilisation des connexions dans AWS Glue Streaming.

Rubriques

- Utilisation des connexions Kafka
- Utilisation des connexions Kinesis

Utilisation des connexions Kafka

Vous pouvez utiliser une connexion Kafka pour lire et écrire dans des flux de données Kafka à l'aide d'informations stockées dans une table de catalogue de données ou en fournissant des informations permettant d'accéder directement au flux de données. La connexion prend en charge un cluster Kafka ou un cluster Amazon Managed Streaming for Apache Kafka. Vous pouvez lire les informations de Kafka dans un Spark DataFrame, puis les convertir en AWS Glue DynamicFrame. Vous pouvez DynamicFrames écrire dans Kafka au format JSON. Si vous accédez directement au flux de données, utilisez ces options pour fournir des informations sur la façon d'accéder au flux de données.

Si vous utilisez `getCatalogSource` ou `create_data_frame_from_catalog` pour consommer des enregistrements à partir d'une source de streaming Kafka, ou `getCatalogSink` ou `write_dynamic_frame_from_catalog` pour écrire des enregistrements dans Kafka, et que la tâche dispose des informations relatives à la base de données Data Catalog et au nom de la table, et peut les utiliser pour obtenir certains paramètres de base pour la lecture à partir de la source de streaming Kafka. Si vous utilisez `getSource`, `getCatalogSink`, `getSourceWithFormat`, `getSinkWithFormat`, `createDataFrameFromOptions`, `create_data_frame_from_options` ou `write_dynamic_frame_from_catalog`, vous devez spécifier ces paramètres de base à l'aide des options de connexion décrites ici.

Vous pouvez spécifier les options de connexion pour Kafka à l'aide des arguments suivants pour les méthodes spécifiées dans la classe `GlueContext`.

- Scala
  - `connectionOptions` : utiliser avec `getSource`, `createDataFrameFromOptions`, `getSink`
  - `additionalOptions` : utiliser avec `getCatalogSource`, `getCatalogSink`
  - `options` : utiliser avec `getSourceWithFormat`, `getSinkWithFormat`
- Python
  - `connection_options` : utiliser avec `create_data_frame_from_options`, `write_dynamic_frame_from_options`
  - `additional_options` : utiliser avec `create_data_frame_from_catalog`, `write_dynamic_frame_from_catalog`
  - `options` : utiliser avec `getSource`, `getSink`

Pour les remarques et les restrictions concernant les tâches ETL de streaming, consultez the section called "Restrictions et notes sur ETL en streaming".

Rubriques

- Configurer Kafka
- Exemple : lecture à partir de flux Kafka
- Exemple : écriture dans les flux Kafka
- Référence des options de connexion de Kafka

Configurer Kafka

Il n'y a aucune AWS condition préalable pour se connecter aux flux Kafka disponibles sur Internet.

Vous pouvez créer une connexion AWS Glue Kafka pour gérer vos identifiants de connexion. Pour de plus amples informations, veuillez consulter the section called "Création d'une connexion pour un flux de données Kafka". Dans la configuration de votre tâche AWS Glue, fournissez *connectionName* une connexion réseau supplémentaire, puis, dans votre appel de méthode, fournissez *connectionName* le `connectionName` paramètre.

Dans certains cas, vous devrez configurer des prérequis supplémentaires :

- Si vous utilisez Amazon Managed Streaming pour Apache Kafka avec l'authentification IAM, vous aurez besoin d'une configuration IAM appropriée.
- Si vous utilisez Amazon Managed Streaming pour Apache Kafka avec un Amazon VPC, vous aurez besoin d'une configuration Amazon VPC appropriée. Vous devez créer une connexion AWS Glue fournissant les informations de connexion Amazon VPC. Vous aurez besoin de la configuration de votre tâche pour inclure la connexion AWS Glue en tant que connexion réseau supplémentaire.

Pour plus d'informations sur les prérequis de la tâche ETL de streaming, consultez the section called "Tâches ETL en streaming".

Exemple : lecture à partir de flux Kafka

Utilisez conjointement avec the section called "forEachBatch".

Exemple pour la source de streaming Kafka :

```
kafka_options =
    { "connectionName": "ConfluentKafka",
      "topicName": "kafka-auth-topic",
      "startingOffsets": "earliest",
      "inferSchema": "true",
      "classification": "json"
    }
data_frame_datasource0 =
 glueContext.create_data_frame.from_options(connection_type="kafka",
 connection_options=kafka_options)
```

Exemple : écriture dans les flux Kafka

Exemples pour écrire dans Kafka :

Exemple avec la méthode `getSink` :

```
data_frame_datasource0 =
glueContext.getSink(
 connectionType="kafka",
 connectionOptions={
  JsonOptions("""{
   "connectionName": "ConfluentKafka",
   "classification": "json",
   "topic": "kafka-auth-topic",
   "typeOfData": "kafka"}
  """)},
 transformationContext="dataframe_ApacheKafka_node1711729173428")
 .getDataFrame()
```

Exemple avec la méthode `write_dynamic_frame.from_options` :

```
kafka_options =
    { "connectionName": "ConfluentKafka",
      "topicName": "kafka-auth-topic",
      "classification": "json"
    }
data_frame_datasource0 =
 glueContext.write_dynamic_frame.from_options(connection_type="kafka",
 connection_options=kafka_options)
```

Référence des options de connexion de Kafka

Lors de la lecture, utilisez les options de connexion suivantes avec `"connectionType": "kafka"` :

- `"bootstrap.servers"`(Obligatoire) Une liste de serveurs bootstrap URLs, par exemple, commeb-1.vpc-test-2.o4q88o.c6.kafka.us-east-1.amazonaws.com:9094. Cette option doit être spécifiée dans l'appel d'API ou définie dans les métadonnées de la table dans le catalogue de données.
- `"security.protocol"`(Obligatoire) Le protocole utilisé pour communiquer avec les agents. Les valeurs possibles sont `"SSL"` ou `"PLAINTEXT"`.
- `"topicName"` : (requis) liste de rubriques séparées par des virgules auxquelles s'abonner. Vous devez spécifier un seul et unique élément parmi `"topicName"`, `"assign"` ou `"subscribePattern"`.
- `"assign"` : (requis) chaîne JSON indiquant le `TopicPartitions` spécifique à utiliser. Vous devez spécifier un seul et unique élément parmi `"topicName"`, `"assign"` ou `"subscribePattern"`.

  Exemple : '{"topicA":[0,1],"topicB":[2,4]}'
- `"subscribePattern"` : (obligatoire) chaîne d'expression rationnelle Java qui identifie la liste de rubriques à laquelle vous souhaitez vous abonner. Vous devez spécifier un seul et unique élément parmi `"topicName"`, `"assign"` ou `"subscribePattern"`.

  Exemple : 'topic.*'
- `"classification"` : (obligatoire) le format de fichier utilisé par les données de l'enregistrement. Obligatoire, sauf s'il est fourni par le catalogue de données.
- `"delimiter"` : (facultatif) le séparateur de valeurs utilisé lorsque `classification` est CSV. La valeur par défaut est « `,` ».
- `"startingOffsets"` : (facultatif) position de départ dans la rubrique Kafka à partir de laquelle lire les données. Les valeurs possibles sont `"earliest"` ou `"latest"`. La valeur par défaut est `"latest"`.
- `"startingTimestamp"`: (Facultatif, disponible uniquement pour AWS la version 4.0 ou ultérieure de Glue) Horodatage de l'enregistrement dans la rubrique Kafka à partir duquel les données doivent être lues. La valeur possible est une chaîne d'horodatage au format UTC dans le modèle `yyyy-mm-ddTHH:MM:SSZ` (où `Z` représente un décalage de fuseau horaire UTC avec un +/-. Par exemple : « 2023-04-04T08:00:00-04:00 »).

Remarque : seule l'une des propriétés « StartingOffsets » ou « StartingTimestamp » peut figurer dans la liste des options de connexion du script de streaming AWS Glue. L'inclusion de ces deux propriétés entraînera l'échec de la tâche.

- `"endingOffsets"` : (facultatif) point de fin lorsqu'une requête par lots est terminée. Les valeurs possibles sont `"latest"` ou une chaîne JSON qui spécifie un décalage de fin pour chaque `TopicPartition`.

  Pour la chaîne JSON, le format est `{"topicA":{"0":23,"1":-1},"topicB":{"0":-1}}`. La valeur `-1` en tant que décalage représente `"latest"`.

- `"pollTimeoutMs"` : (facultatif) délai d'attente en millisecondes pour interroger les données de Kafka dans les exécuteurs de tâches Spark. La valeur par défaut est `600000`.

- `"numRetries"` : (facultatif) nombre de nouvelles tentatives avant de ne pas récupérer les décalages Kafka. La valeur par défaut est `3`.

- `"retryIntervalMs"` : (facultatif) temps d'attente en millisecondes avant d'essayer de récupérer les décalages Kafka. La valeur par défaut est `10`.

- `"maxOffsetsPerTrigger"` : (facultatif) limite de taux sur le nombre maximal de décalages qui sont traités par intervalle de déclenchement. Le nombre total spécifié de décalages est réparti proportionnellement entre les `topicPartitions` des différents volumes. La valeur par défaut est null, ce qui signifie que le consommateur lit tous les décalages jusqu'au dernier décalage connu.

- `"minPartitions"` : (facultatif) nombre minimum de partitions à lire à partir de Kafka. La valeur par défaut est nulle, ce qui signifie que le nombre de partitions Spark est égal au nombre de partitions Kafka.

- `"includeHeaders"`: (facultatif) indique s'il faut inclure les en-têtes Kafka. Lorsque l'option est définie sur « true » (vrai), la sortie de données contiendra une colonne supplémentaire nommée « glue_streaming_kafka_headers » avec le type `Array[Struct(key: String, value: String)]`. La valeur définie par défaut est « false ». Cette option est disponible dans AWS Glue version 3.0 ou ultérieure.

- `"schema"` : (requis lorsque inferSchema est défini sur false) schéma à utiliser pour traiter la charge utile. Si la classification est `avro`, le schéma fourni doit être au format de schéma Avro. Si la classification n'est pas `avro`, le schéma fourni doit être au format de schéma DDL.

  Voici quelques exemples de schémas.

Example in DDL schema format

```
'column1' INT, 'column2' STRING , 'column3' FLOAT
```

Example in Avro schema format

```
{
"type":"array",
"items":
{
"type":"record",
"name":"test",
"fields":
[
  {
    "name":"_id",
    "type":"string"
  },
  {
    "name":"index",
    "type":
    [
      "int",
      "string",
      "float"
    ]
  }
]
}
}
```

- `"inferSchema"` : (facultatif) la valeur par défaut est « false ». S'il est défini sur « true », le schéma sera détecté lors de l'exécution à partir de la charge utile dans `foreachbatch`.
- `"avroSchema"` : (obsolète) paramètre utilisé pour spécifier un schéma de données Avro lorsque le format Avro est utilisé. Ce paramètre est désormais obsolète. Utilisez le paramètre `schema`.
- `"addRecordTimestamp"` : (facultatif) lorsque cette option est définie sur « true », la sortie de données contient une colonne supplémentaire nommée « __src_timestamp » qui indique l'heure à laquelle l'enregistrement correspondant est reçu par la rubrique. La valeur par défaut est « false ». Cette option est prise en charge dans AWS Glue version 4.0 ou ultérieure.

- `"emitConsumerLagMetrics"`: (Facultatif) Lorsque l'option est définie sur « vrai », pour chaque lot, elle émet les métriques correspondant à la durée comprise entre le plus ancien enregistrement reçu par le sujet et le moment où il arrive CloudWatch. AWS Glue Le nom de la métrique est « glue.driver.streaming ». maxConsumerLagInMs». La valeur par défaut est « false ». Cette option est prise en charge dans AWS Glue version 4.0 ou ultérieure.

Lors de l'écriture, utilisez les options de connexion suivantes avec `"connectionType": "kafka"` :

- `"connectionName"`(Obligatoire) Nom de la connexion AWS Glue utilisée pour se connecter au cluster Kafka (similaire à la source Kafka).
- `"topic"` (Obligatoire) Si une colonne de rubrique existe, sa valeur est utilisée comme rubrique lors de l'écriture de la ligne donnée dans Kafka, sauf si l'option de configuration de la rubrique est définie. C'est-à-dire que l'option de configuration `topic` remplace la colonne de rubrique.
- `"partition"` (Facultatif) Si un numéro de partition valide est spécifié, cette `partition` sera utilisée lors de l'envoi de l'enregistrement.

  Si aucune partition n'est spécifiée, mais qu'une `key` est présente, une partition sera choisie en utilisant le hachage de la clé.

  Si ni une `key`, ni une `partition` ne sont présentes, une partition sera choisie en partitionnant de manière permanente ces modifications lorsqu'au moins des octets batch.size seront produits sur la partition.
- `"key"` (Facultatif) Utilisé pour le partitionnement si `partition` est nul.
- `"classification"` (Facultatif) Format de fichier utilisé par les données de l'enregistrement. Nous prenons uniquement en charge les formats JSON, CSV et Avro.

  Avec le format Avro, nous pouvons fournir une valeur avroSchema personnalisée avec laquelle il est possible de sérialiser, mais notez que cela doit également être fourni sur la source pour la désérialisation. Sinon, par défaut, il utilise Apache AvroSchema pour la sérialisation.

En outre, vous pouvez optimiser le récepteur Kafka selon les besoins en mettant à jour les paramètres de configuration du producteur Kafka. Notez qu'il n'existe aucune liste d'autorisation pour les options de connexion, toutes les paires clé-valeur sont conservées telles quelles sur le récepteur.

Cependant, il existe une petite liste de refus d'options qui ne prendra pas effet. Pour plus d'informations, consultez Kafka specific configurations.

Utilisation des connexions Kinesis

Vous pouvez utiliser une connexion Kinesis pour lire et écrire dans Amazon Kinesis Data Streams à l'aide d'informations stockées dans une table de catalogue de données ou en fournissant des informations permettant d'accéder directement au flux de données. Vous pouvez lire les informations de Kinesis dans un Spark DataFrame, puis les convertir en Glue AWS . DynamicFrame Vous pouvez DynamicFrames écrire dans Kinesis au format JSON. Si vous accédez directement au flux de données, utilisez ces options pour fournir des informations sur la façon d'accéder au flux de données.

Si vous utilisez `getCatalogSource` ou `create_data_frame_from_catalog` pour consommer des enregistrements à partir d'une source de streaming Kinesis, la tâche dispose des informations relatives à la base de données Data Catalog et au nom de la table, et peut les utiliser pour obtenir certains paramètres de base pour la lecture à partir de la source de streaming Kinesis. Si vous utilisez `getSource`, `getSourceWithFormat`, `createDataFrameFromOptions` ou `create_data_frame_from_options`, vous devez spécifier ces paramètres de base à l'aide des options de connexion décrites ici.

Vous pouvez spécifier les options de connexion pour Kinesis à l'aide des arguments suivants pour les méthodes spécifiées dans la classe `GlueContext`.

- Scala
  - `connectionOptions` : utiliser avec `getSource`, `createDataFrameFromOptions`, `getSink`
  - `additionalOptions` : utiliser avec `getCatalogSource`, `getCatalogSink`
  - `options` : utiliser avec `getSourceWithFormat`, `getSinkWithFormat`
- Python
  - `connection_options` : utiliser avec `create_data_frame_from_options`, `write_dynamic_frame_from_options`
  - `additional_options` : utiliser avec `create_data_frame_from_catalog`, `write_dynamic_frame_from_catalog`
  - `options` : utiliser avec `getSource`, `getSink`

Pour les remarques et les restrictions concernant les tâches ETL de streaming, consultez the section called "Restrictions et notes sur ETL en streaming".

Configurer Kinesis

Pour vous connecter à un flux de données Kinesis dans le cadre d'une tâche AWS Glue Spark, vous aurez besoin de certaines conditions préalables :

- En cas de lecture, la tâche AWS Glue doit disposer d'autorisations IAM de niveau d'accès en lecture pour le flux de données Kinesis.
- En cas d'écriture, la tâche AWS Glue doit disposer d'autorisations IAM de niveau d'accès Write au flux de données Kinesis.

Dans certains cas, vous devrez configurer des prérequis supplémentaires :

- Si votre tâche AWS Glue est configurée avec des connexions réseau supplémentaires (généralement pour se connecter à d'autres ensembles de données) et que l'une de ces connexions fournit des options de réseau Amazon VPC, cela indiquera à votre tâche de communiquer via Amazon VPC. Dans ce cas, vous devrez également configurer votre flux de données Kinesis pour qu'il communique via Amazon VPC. Vous pouvez le faire en créant un point de terminaison d'un VPC d'interface entre votre Amazon VPC et votre flux de données Kinesis. Pour plus d'informations, consultez Using Kinesis Data Streams with Interface VPC Endpoints.
- Lorsque vous spécifiez Amazon Kinesis Data Streams dans un autre compte, vous devez configurer les rôles et les stratégies pour autoriser l'accès intercompte. Pour de plus amples informations, veuillez consulter la rubrique Exemple : Lire à partir d'un flux Kinesis dans un autre compte.

Pour plus d'informations sur les prérequis de la tâche ETL de streaming, consultez the section called "Tâches ETL en streaming".

Lecture depuis Kinesis

Exemple : lecture à partir de flux Kinesis

Utilisez conjointement avec the section called "forEachBatch".

Exemple pour la source de streaming Amazon Kinesis :

```
kinesis_options =
   { "streamARN": "arn:aws:kinesis:us-east-2:777788889999:stream/fromOptionsStream",
     "startingPosition": "TRIM_HORIZON",
```

```
    "inferSchema": "true",
    "classification": "json"
  }
data_frame_datasource0 =
 glueContext.create_data_frame.from_options(connection_type="kinesis",
 connection_options=kinesis_options)
```

Écrire à Kinesis

Exemple : écriture dans Kinesis Streams

Utilisez conjointement avec the section called "forEachBatch". Vous DynamicFrame serez écrit dans le flux au format JSON. Si la tâche ne parvient pas à écrire après plusieurs tentatives, elle échoue. Par défaut, chaque DynamicFrame enregistrement est envoyé au flux Kinesis individuellement. Vous pouvez configurer ce comportement à l'aide de la commande `aggregationEnabled` et des paramètres associés.

Exemple d'écriture sur Amazon Kinesis à partir d'une tâche de diffusion :

Python

```
glueContext.write_dynamic_frame.from_options(
    frame=frameToWrite
    connection_type="kinesis",
    connection_options={
        "partitionKey": "part1",
        "streamARN": "arn:aws:kinesis:us-east-1:111122223333:stream/streamName",
    }
)
```

Scala

```
glueContext.getSinkWithFormat(
                connectionType="kinesis",
                options=JsonOptions("""{
                    "streamARN": "arn:aws:kinesis:us-
east-1:111122223333:stream/streamName",
                    "partitionKey": "part1"
                }"""),
            )
            .writeDynamicFrame(frameToWrite)
```

Paramètres de connexion Kinesis

Désigne des options de connexion à Amazon Kinesis Data Streams.

Utilisez les options de connexion suivantes pour les sources de données en streaming Kinesis :

- `"streamARN"` (Obligatoire) Utilisé pour la lecture/l'écriture. ARN du flux de données Kinesis.
- `"classification"` (Obligatoire pour la lecture) Utilisé pour la lecture. Format de fichier utilisé par les données de l'enregistrement. Obligatoire, sauf s'il est fourni par le catalogue de données.
- `"streamName"` : (facultatif) utilisé pour la lecture. Nom du flux de données Kinesis à partir duquel lire. Utilisé avec `endpointUrl`.
- `"endpointUrl"` : (facultatif) utilisé pour la lecture. Par défaut : « https://kinesis.us-east-1.amazonaws.com ». AWS Point de terminaison du flux Kinesis. Vous n'avez pas besoin de modifier ce paramètre, sauf si vous vous connectez à une région spéciale.
- `"partitionKey"` : (facultatif) utilisé pour l'écriture. Clé de partition Kinesis utilisée lors de la production d'enregistrements.
- `"delimiter"` : (facultatif) utilisé pour la lecture. Séparateur de valeurs utilisé lorsque `classification` est CSV. La valeur par défaut est « `,` ».
- `"startingPosition"` : (facultatif) utilisé pour la lecture. La position de départ dans le flux de données Kinesis à partir duquel lire les données. Les valeurs possibles sont `"latest"`, `"trim_horizon"`, `"earliest"`, ou une chaîne d'horodatage au format UTC dans le modèle `yyyy-mm-ddTHH:MM:SSZ` (où `Z` représente un décalage de fuseau horaire UTC avec un +/-. Par exemple : « 2023-04-04T08:00:00-04:00 »). La valeur par défaut est `"latest"`. Remarque : la chaîne d'horodatage au format UTC pour n'`"startingPosition"`est prise en charge que pour la version 4.0 ou ultérieure de AWS Glue.
- `"failOnDataLoss"` : (facultatif) échec de la tâche si une partition active est manquante ou a expiré. La valeur par défaut est `"false"`.
- `"awsSTSRoleARN"` : (facultatif) utilisé pour la lecture/l'écriture. Le nom de ressource Amazon (ARN) du rôle à assumer en utilisant AWS Security Token Service (AWS STS). Ce rôle doit disposer des autorisations nécessaires pour décrire ou lire des registres pour le flux de données Kinesis. Vous devez utiliser ce paramètre lorsque vous accédez à un flux de données dans un autre compte. Utilisez conjointement avec `"awsSTSSessionName"`.
- `"awsSTSSessionName"` : (facultatif) utilisé pour la lecture/l'écriture. Un identifiant de la séance assumant le rôle à l'aide d' AWS STS. Vous devez utiliser ce paramètre lorsque vous accédez à un flux de données dans un autre compte. Utilisez conjointement avec `"awsSTSRoleARN"`.

- `"awsSTSEndpoint"`: (Facultatif) Le AWS STS point de terminaison à utiliser lors de la connexion à Kinesis avec un rôle assumé. Cela permet d'utiliser le point de AWS STS terminaison régional dans un VPC, ce qui n'est pas possible avec le point de terminaison global par défaut.
- `"maxFetchTimeInMs"` : (facultatif) utilisé pour la lecture. Le temps maximal passé dans l'exécuteur de tâches pour lire des enregistrements du lot actuel à partir du flux de données Kinesis, exprimé en millisecondes (ms). Plusieurs appels d'API `GetRecords` peuvent être effectués pendant cette période. La valeur par défaut est `1000`.
- `"maxFetchRecordsPerShard"` : (facultatif) utilisé pour la lecture. Nombre maximal d'enregistrements à récupérer par partition dans le flux de données Kinesis par microlot. Remarque : Le client peut dépasser cette limite si la tâche de streaming a déjà lu des enregistrements supplémentaires provenant de Kinesis (dans le même appel get-records). Si `maxFetchRecordsPerShard` doit être strict, il doit être un multiple de `maxRecordPerRead`. La valeur par défaut est `100000`.
- `"maxRecordPerRead"` : (facultatif) utilisé pour la lecture. Nombre maximal d'enregistrements à extraire du flux de données Kinesis dans chaque opération `getRecords`. La valeur par défaut est `10000`.
- `"addIdleTimeBetweenReads"` : (facultatif) utilisé pour la lecture. Ajoute un délai entre deux opérations `getRecords` consécutives. La valeur par défaut est `"False"`. Cette option n'est configurable que pour Glue version 2.0 et ultérieure.
- `"idleTimeBetweenReadsInMs"` : (facultatif) utilisé pour la lecture. Délai minimum entre deux opérations `getRecords`, en ms. La valeur par défaut est `1000`. Cette option n'est configurable que pour Glue version 2.0 et ultérieure.
- `"describeShardInterval"` : (facultatif) utilisé pour la lecture. Intervalle de temps minimum entre deux appels d'API `ListShards` pour que votre script envisage le repartitionnement. Pour plus d'informations, consultez Politiques de repartitionnement dans le Guide du développeur Amazon Kinesis Data Streams. La valeur par défaut est `1s`.
- `"numRetries"` : (facultatif) utilisé pour la lecture. Le nombre maximal de nouvelles tentatives pour les demandes d'API Kinesis Data Streams. La valeur par défaut est `3`.
- `"retryIntervalMs"` : (facultatif) utilisé pour la lecture. Le délai de réflexion (spécifié en ms) avant de réessayer l'appel d'API Kinesis Data Streams. La valeur par défaut est `1000`.
- `"maxRetryIntervalMs"` : (facultatif) utilisé pour la lecture. Le délai d'attente maximal (spécifié en ms) entre deux tentatives d'appel d'API Kinesis Data Streams. La valeur par défaut est `10000`.

- `"avoidEmptyBatches"` : (facultatif) utilisé pour la lecture. Évite de créer une tâche de micro-lot vide en vérifiant les données non lues dans le flux de données Kinesis avant le démarrage du lot. La valeur par défaut est `"False"`.
- `"schema"` : (obligatoire lorsque inferSchema est défini sur false) utilisé pour la lecture. Schéma à utiliser pour traiter la charge utile. Si la classification est `avro`, le schéma fourni doit être au format de schéma Avro. Si la classification n'est pas `avro`, le schéma fourni doit être au format de schéma DDL.

  Voici quelques exemples de schémas.

  Example in DDL schema format

  ```
  `column1` INT, `column2` STRING , `column3` FLOAT
  ```

  Example in Avro schema format

  ```
  {
    "type":"array",
    "items":
    {
      "type":"record",
      "name":"test",
      "fields":
      [
        {
          "name":"_id",
          "type":"string"
        },
        {
          "name":"index",
          "type":
          [
            "int",
            "string",
            "float"
          ]
        }
      ]
    }
  }
  ```

- `"inferSchema"` : (facultatif) utilisé pour la lecture. La valeur par défaut est « false ». S'il est défini sur « true », le schéma sera détecté lors de l'exécution à partir de la charge utile dans `foreachbatch`.
- `"avroSchema"` : (obsolète) utilisé pour la lecture. Paramètre utilisé pour spécifier un schéma de données Avro lorsque le format Avro est utilisé. Ce paramètre est désormais obsolète. Utilisez le paramètre `schema`.
- `"addRecordTimestamp"` : (facultatif) utilisé pour la lecture. Lorsque cette option est définie sur « true », la sortie de données contient une colonne supplémentaire nommée « __src_timestamp » qui indique l'heure à laquelle l'enregistrement correspondant est reçu par le flux. La valeur par défaut est « false ». Cette option est prise en charge dans AWS Glue version 4.0 ou ultérieure.
- `"emitConsumerLagMetrics"` : (facultatif) utilisé pour la lecture. Lorsque l'option est définie sur « vrai », pour chaque lot, elle émet les métriques correspondant à la durée comprise entre le plus ancien enregistrement reçu par le flux et l'heure AWS Glue à laquelle il arrive CloudWatch. Le nom de la métrique est « glue.driver.streaming ». maxConsumerLagInMs». La valeur par défaut est « false ». Cette option est prise en charge dans AWS Glue version 4.0 ou ultérieure.
- `"fanoutConsumerARN"` : (facultatif) utilisé pour la lecture. ARN d'un consommateur de flux Kinesis pour le flux spécifié dans `streamARN`. Utilisé pour activer le mode diffusion améliorée pour votre connexion Kinesis. Pour plus d'informations sur la consommation d'un flux Kinesis avec diffusion améliorée, consultez the section called "Utilisation de la diffusion améliorée dans les tâches de streaming Kinesis".
- `"recordMaxBufferedTime"` : (facultatif) utilisé pour l'écriture. Par défaut : 1 000 (ms). Durée maximale pendant laquelle un enregistrement est mis en mémoire tampon en attendant d'être écrit.
- `"aggregationEnabled"` : (facultatif) utilisé pour l'écriture. Valeur par défaut : vraie. Spécifie si les enregistrements doivent être agrégés avant de les envoyer à Kinesis.
- `"aggregationMaxSize"` : (facultatif) utilisé pour l'écriture. Par défaut : 51 200 (octets). Si un enregistrement est supérieur à cette limite, il contourne l'agrégateur. Remarque Kinesis impose une limite de 50 Ko à la taille des enregistrements. Si vous définissez ce paramètre au-delà de 50 Ko, les enregistrements surdimensionnés seront rejetés par Kinesis.
- `"aggregationMaxCount"` : (facultatif) utilisé pour l'écriture. Par défaut : 4294967295. Nombre maximum de résultats à regrouper dans un enregistrement agrégé.
- `"producerRateLimit"` : (facultatif) utilisé pour l'écriture. Par défaut : 150 (%). Limite le débit par partition envoyée par un seul producteur (votre tâche, par exemple), sous forme de pourcentage de la limite du backend.

- `"collectionMaxCount"` : (facultatif) utilisé pour l'écriture. Par défaut : 500. Nombre maximum d'articles à inclure dans une PutRecords demande.
- `"collectionMaxSize"` : (facultatif) utilisé pour l'écriture. Par défaut : 5242880 (octets). Quantité maximale de données à envoyer avec une PutRecords demande.

AWS Glue mise à l'échelle automatique du streaming

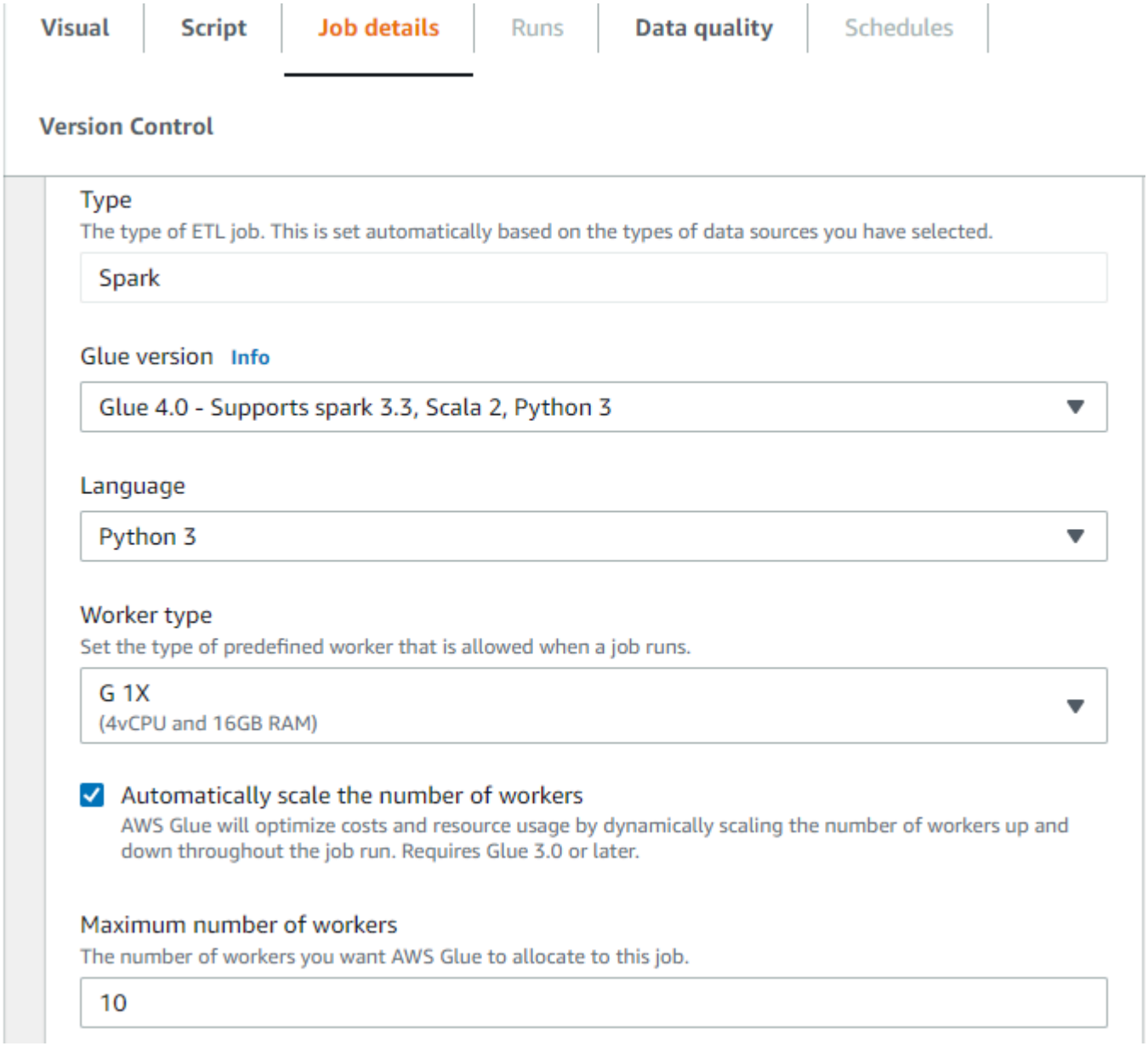
AWS Glue les tâches ETL de streaming consomment en permanence des données provenant de sources de streaming, nettoient et transforment les données en cours de vol, et les mettent à disposition pour analyse. En surveillant chaque étape de l'exécution de la tâche, AWS Glue l'autoscaling peut désactiver les travailleurs lorsqu'ils sont inactifs ou en ajouter si un traitement parallèle supplémentaire est possible.

Les sections suivantes fournissent des informations sur le dimensionnement automatique du AWS Glue streaming.

Activer Auto Scaling dans AWS Glue Studio

Dans l'onglet Détails du Job dans AWS Glue Studio, choisissez le type Spark ou Spark Streaming, et la version Glue **`Glue 3.0`** ou**`Glue 4.0`**. Ensuite, une case à cocher apparaît sous le Worker type (Type d'employé).

- Sélectionnez l'option Automatically scale the number of workers (Mise à l'échelle automatique du nombre d'employés).
- Configurez le Nombre maximal d’employés pour définir le nombre maximal d’employés pouvant être transférés à l’exécution de la tâche.



Activer Auto Scaling avec la AWS CLI ou le SDK

Pour activer Auto Scaling From the AWS CLI pour l'exécution de votre tâche, exécutez `start-job-run` la configuration suivante :

```
{
    "JobName": "<your job name>",
    "Arguments": {
        "--enable-auto-scaling": "true"
    },
```

```
    "WorkerType": "G.2X", // G.1X, G.2X, G.4X, G.8X, G.12X, G.16X, R.1X, R.2X, R.4X,
 and R.8X are supported for Auto Scaling Jobs
    "NumberOfWorkers": 20, // represents Maximum number of workers
    ...other job run configurations...
}
```

Une fois l'exécution de la tâche ETL terminée, vous pouvez également appeler `get-job-run` pour vérifier l'utilisation réelle des ressources de la tâche exécutée en secondes DPU. Remarque : le nouveau champ n'DPUSecondsapparaîtra que pour vos tâches par lots dans la AWS Glue version 3.0 ou ultérieure activée avec Auto Scaling. Ce champ n'est pas pris en charge pour les tâches de streaming.

```
$ aws glue get-job-run --job-name your-job-name --run-id jr_xx --endpoint https://
glue.us-east-1.amazonaws.com --region us-east-1
{
    "JobRun": {
        ...
        "GlueVersion": "3.0",
        "DPUSeconds": 386.0
    }
}
```

Vous pouvez également configurer des exécutions de tâches avec Auto Scaling à l'aide du kit SDK AWS Glue en suivant la même configuration.

Comment ça marche

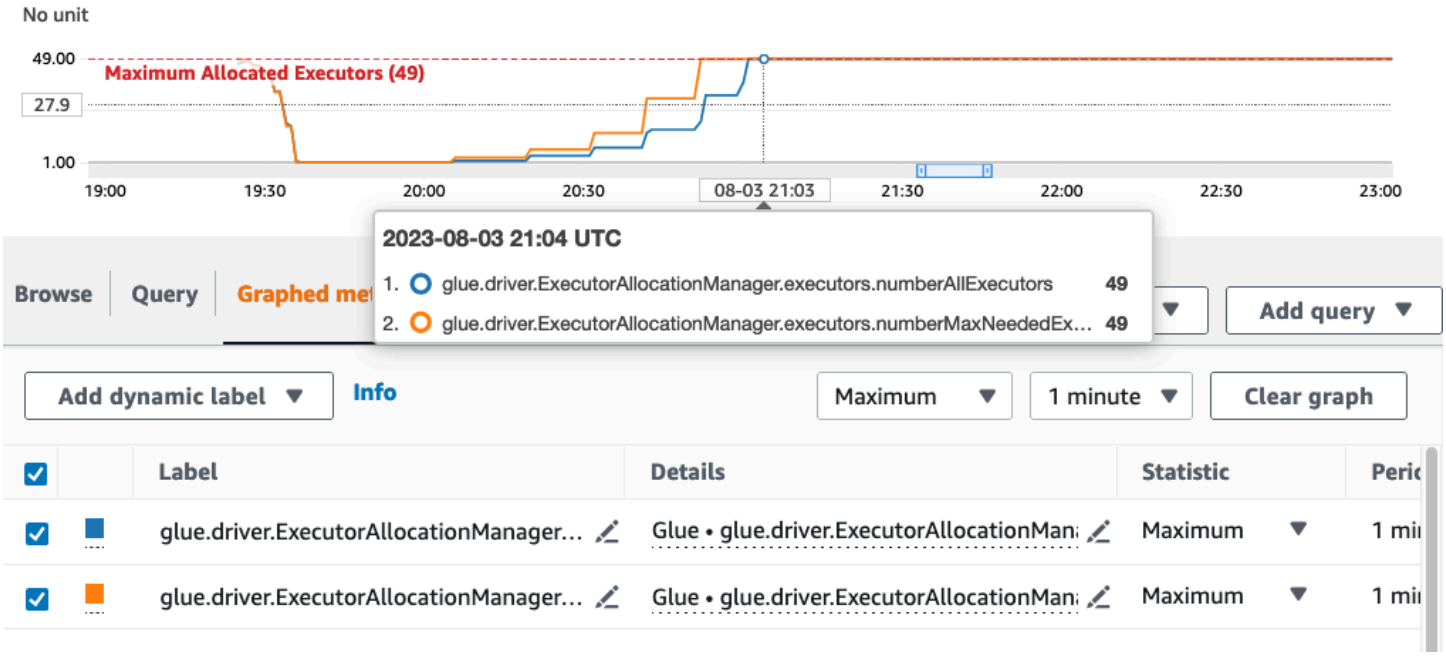
Mise à l'échelle dans les microlots

L'exemple suivant est utilisé pour décrire le fonctionnement de l'autoscaling.

- Vous avez un AWS Glue travail qui commence à 50 ans DPUs.
- L'autoscaling est activé.

Dans cet exemple, AWS Glue examine la métrique batchProcessingTime InMs « » pour quelques microlots et détermine si vos tâches se terminent dans les limites de la fenêtre que vous avez définie. Si vos tâches sont terminées plus tôt et en fonction de la rapidité avec laquelle elles seront terminées, AWS Glue pourra se réduire. Cette métrique, tracée avec numberAllExecutors « », peut être surveillée Amazon CloudWatch pour voir comment fonctionne l'autoscaling.

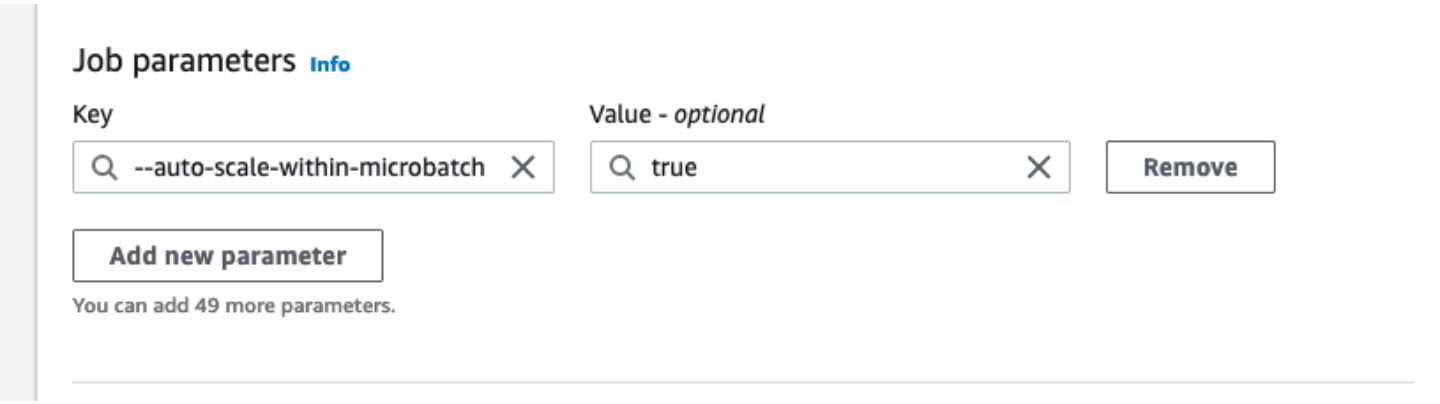
Le nombre de programmes d'exécution augmente ou se réduit de façon exponentielle uniquement après la fin de chaque microlot. Comme vous pouvez le constater dans le journal de Amazon CloudWatch surveillance, AWS Glue il examine le nombre d'exécuteurs nécessaires (ligne orange) et redimensionne les exécuteurs (ligne bleue) pour qu'ils correspondent automatiquement à ce nombre.



Une AWS Glue fois réduit le nombre d'exécuteurs et constaté que les volumes de données augmentent, ce qui augmente le temps de traitement par microlots, AWS Glue on passe à 50 DPUs, soit la limite supérieure spécifiée.

Mise à l'échelle au sein de microlots

Dans l'exemple ci-dessus, le système surveille quelques microlots terminés pour décider de les augmenter ou de les réduire. Les fenêtres plus longues nécessitent un autoscaling pour réagir plus rapidement au sein du microlot, au lieu d'attendre quelques microlots. Dans ces cas, vous pouvez utiliser une configuration supplémentaire `--auto-scaIe-within-microbatch` pour `true`. Vous pouvez l'ajouter aux propriétés de la AWS Glue tâche AWS Glue Studio comme indiqué ci-dessous.



Fenêtres de maintenance pour le AWS Glue streaming

AWS Glue effectue périodiquement des activités de maintenance. Au cours de ces fenêtres de maintenance, vous AWS Glue devrez redémarrer vos tâches de streaming. Vous pouvez contrôler le moment où les tâches sont redémarrées en spécifiant des fenêtres de maintenance. Dans cette section, nous expliquons où vous pouvez configurer la fenêtre de maintenance et les comportements spécifiques à prendre en compte.

Rubriques

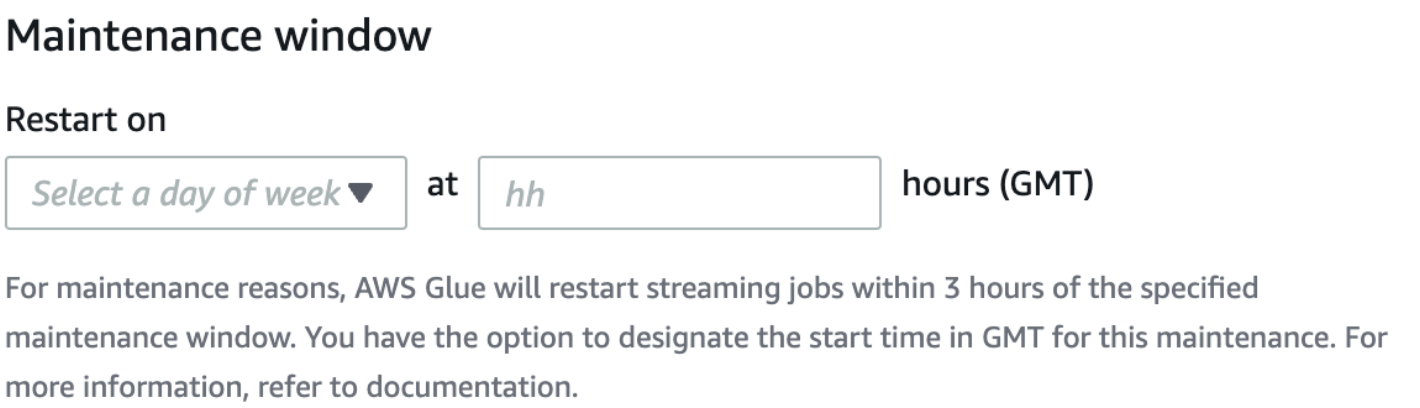
- Configuration d'une fenêtre de maintenance
- Comportement des fenêtres de maintenance
- Surveillance des tâches
- Gestion des pertes de données

Configuration d'une fenêtre de maintenance

Vous pouvez configurer une fenêtre de maintenance à l'aide de AWS Glue Studio ou APIs.

Configuration d'une fenêtre de maintenance dans AWS Glue Studio

Vous pouvez spécifier une fenêtre de maintenance sur la page Détails de la tâche de AWS Glue streaming. Vous pouvez spécifier le jour et l'heure en GMT. AWS Glue redémarrera votre travail dans le délai imparti.



Configuration d'une fenêtre de maintenance dans l'API

Vous pouvez également configurer la fenêtre de maintenance dans l'API de création de tâche. Voici un exemple de configuration des fenêtres de maintenance via l'API.

```
aws glue create-job –name jobName –role roleArnForTheJob –command
 Name=gluestreaming,ScriptLocation=s3-path-to-the-script --maintenance-window="Sun:10"
```

Voici un exemple de commande :

```
aws glue create-job –name testMaintenance –role arn:aws:iam::012345678901:role/
Glue_DefaultRole –command Name=gluestreaming,ScriptLocation=s3://glue-example-test/
example.py –maintenance-window="Sun:10
```

Comportement des fenêtres de maintenance

AWS Glue passe par une série d'étapes pour décider à quel moment il convient de redémarrer une tâche :

1. Lorsqu'une nouvelle tâche de streaming est lancée, vérifiez AWS Glue d'abord si un délai d'expiration est associé à l'exécution de la tâche. Un délai d'attente vous permet de configurer l'heure de fin de la tâche. Si le délai d'expiration est inférieur à sept jours, la tâche ne sera pas redémarrée.
2. Si le délai d'attente est supérieur à 7 jours, AWS Glue vérifie si la fenêtre de maintenance est configurée pour le travail. Si c'est le cas, cette fenêtre est sélectionnée et la fenêtre est affectée à l'exécution du travail. AWS Glue redémarrera le travail dans les 3 heures suivant la période de

maintenance spécifiée. Par exemple, si vous configurez la fenêtre de maintenance pour le lundi à 10h00 GMT, vos tâches seront redémarrées entre 10h00 GMT et 13h00 GMT.

3. Si la fenêtre de maintenance n'est pas configurée, définit AWS Glue automatiquement l'heure de redémarrage sur 7 jours après le début de l'exécution du job. Par exemple, si vous avez lancé votre tâche le 01/07/2024 à 00h00 GMT et que vous n'avez pas spécifié de fenêtres de maintenance, votre tâche sera configurée pour redémarrer le 08/07/2024 à 00h00 GMT.

> **Note**
>
> Si vous exécutez déjà des tâches de streaming, cette modification aura un impact sur vous à compter du 1er juillet 2024. Vous aurez jusqu'au 30 juin pour configurer vos fenêtres de maintenance. Après le 1er juillet, toutes les tâches de streaming que vous lancez seront redémarrées conformément à cette documentation. Si vous avez besoin d'une assistance supplémentaire, vous pouvez contacter le AWS Support.

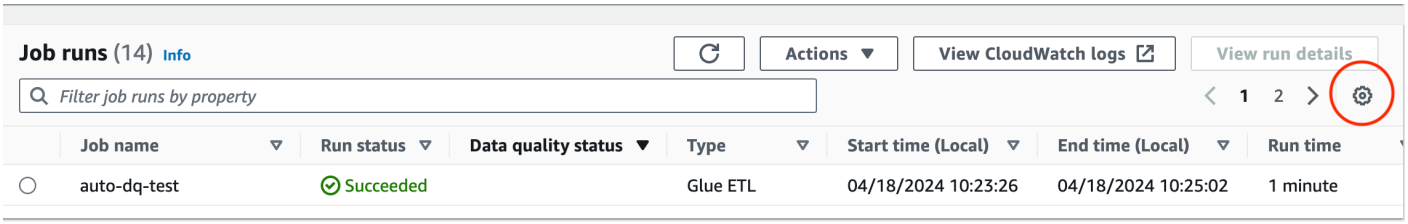
4. Parfois, il AWS Glue se peut que vous ne puissiez pas redémarrer le travail, en particulier lorsque le microlot en cours n'est pas traité. Dans ces cas, la tâche ne sera pas interrompue. Dans ces cas, la tâche AWS Glue redémarrera au bout de 14 jours, et dans ce cas, la fenêtre de maintenance n'est pas respectée.

Surveillance des tâches

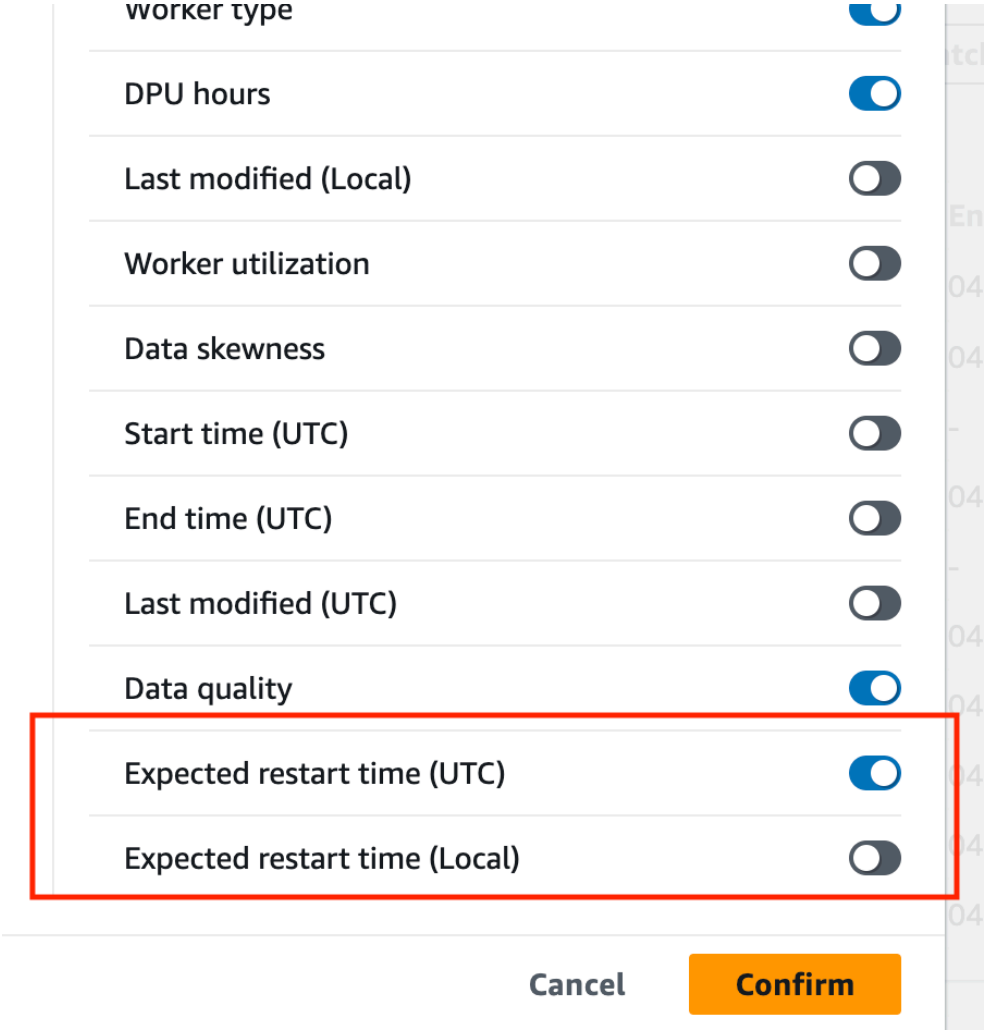
Vous pouvez surveiller les tâches sur la page AWS Glue Studio Monitoring.

Pour connaître l'heure prévue du prochain redémarrage des tâches de streaming, affichez la colonne du tableau des exécutions de tâches sur la page Surveillance.

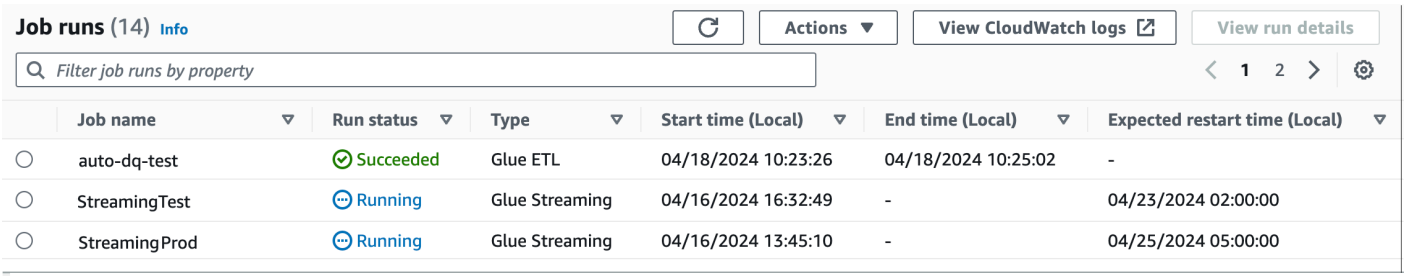
1. Cliquez sur l'icône d'engrenage en haut à droite du tableau.



2. Faites défiler la page vers le bas et activez la colonne Heure de redémarrage prévue. Les options UTC et Heure locale sont disponibles.



3. Vous pouvez ensuite afficher les colonnes du tableau.



La tâche d'origine aura le statut « EXPIRÉ » et la nouvelle instance de tâche aura le statut « EN COURS ». La nouvelle exécution de tâche qui a été redémarrée aura un ID d'exécution sous forme de concaténation de l'ID d'exécution de tâche initial et du préfixe « restart_ » représentant le nombre de redémarrages. Par exemple, si l'ID d'exécution initial de la tâche est `jr_1234`, l'exécution de la tâche redémarrée aura l'ID `jr1234_restart_1` pour le premier redémarrage. Le deuxième redémarrage sera `jr1234_restart_2` pour le deuxième redémarrage, et ainsi de suite.

Votre nouvelle tentative ne sera pas affectée en raison des redémarrages. Si une exécution échoue et qu'une nouvelle exécution est lancée en raison d'une nouvelle tentative automatique, le compteur de redémarrage recommencera à 1. Par exemple, si une exécution échoue à `jr_1234_attempt_3_restart_5`, une nouvelle tentative automatique lancera une nouvelle exécution avec un ID : `jr_id1_attempt_4`, et lorsque cette tentative est redémarrée après sept jours, le nouvel ID d'exécution sera `jr_id1_attempt_4_restart_1`.

Gestion des pertes de données

Lors des redémarrages de maintenance, AWS Glue Streaming suit un processus qui garantit l'intégrité des données et la cohérence entre le travail précédent et le travail redémarré. Notez que cela AWS Glue ne garantit pas l'intégrité et la cohérence des données entre les redémarrages des tâches et nous recommandons de prendre en compte l'architecture pour gérer les données dupliquées dans les tâches de streaming.

1. Détection des conditions de redémarrage pour maintenance : le AWS Glue streaming surveille les conditions qui indiquent quand un redémarrage pour maintenance doit être déclenché, par exemple lorsqu'une fenêtre de maintenance est atteinte après 7 jours ou qu'un redémarrage brutal est nécessaire après 14 jours.
2. Invoquer une résiliation progressive : lorsque les conditions de redémarrage pour maintenance sont remplies, AWS Glue Streaming lance un processus de résiliation progressive pour le travail en cours d'exécution. Ce processus englobe les étapes suivantes :
   a. Arrêt de l'ingestion de nouvelles données : la tâche de streaming arrête de consommer les nouvelles données provenant des sources d'entrée (par exemple, les rubriques Kafka, les flux Kinesis ou les fichiers).
   b. Traitement des données en attente : la tâche continue de traiter toutes les données déjà présentes dans ses mémoires tampons ou ses files d'attente internes.
   c. Validation des décalages et des points de contrôle : la tâche valide les derniers décalages ou points de contrôle vers des systèmes externes (par exemple, Kafka, Kinesis ou Amazon S3) afin de garantir que la tâche redémarrée puisse reprendre là où la tâche précédente s'était arrêtée.
3. Redémarrage de la tâche : une fois le processus d'arrêt progressif terminé, AWS Glue Streaming redémarre la tâche en utilisant l'état préservé et les points de contrôle. La tâche redémarrée reprend le traitement à partir du dernier décalage ou point de contrôle validé, garantissant ainsi qu'aucune donnée n'est perdue ou dupliquée.

4. Reprise du traitement des données : la tâche redémarrée reprend le traitement des données à partir du point où la tâche précédente s'était arrêtée. Elle continue à ingérer les nouvelles données provenant des sources d'entrée, à partir du dernier décalage ou point de contrôle validé, et traite les données conformément à la logique ETL définie.

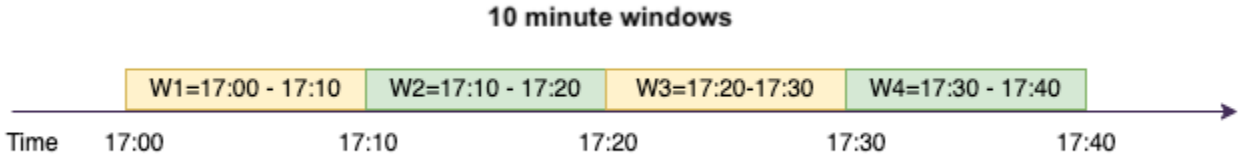
Concepts de streaming AWS Glue avancés

Dans les applications modernes axées sur les données, l'importance des données diminue au fil du temps et leur valeur passe d'une valeur prédictive à une valeur réactive. Par conséquent, les clients souhaitent traiter les données en temps réel pour prendre des décisions plus rapidement. Lorsqu'il s'agit de flux de données en temps réel, tels que ceux provenant de capteurs IoT, les données peuvent arriver de manière désordonnée ou subir des retards de traitement en raison de la latence du réseau ou d'autres défaillances liées à la source lors de l'ingestion. Dans le cadre de la plateforme AWS Glue, le streaming AWS Glue s'appuie sur ces fonctionnalités pour fournir un ETL de streaming évolutif et sans serveur, alimenté par le streaming structuré Apache Spark, permettant aux utilisateurs de traiter les données en temps réel.

Dans cette rubrique, nous allons explorer les concepts de streaming avancés et les fonctionnalités du streaming AWS Glue.

Considérations temporelles lors du traitement des flux

Il existe quatre notions de temps lors du traitement des flux :



- Heure de l'événement : heure de survenue de l'événement. Dans la plupart des cas, ce champ est intégré aux données de l'événement elles-mêmes, à la source.
- E vent-time-window — L'intervalle de temps entre deux événements. Comme le montre le schéma ci-dessus, W1 correspond à un signal event-time-window de 17h00 à 17h10. Chacun event-time-window est un regroupement de plusieurs événements.
- Heure de déclenchement : l'heure de déclenchement contrôle la fréquence du traitement des données et de la mise à jour des résultats. Heure à laquelle le traitement par microlots a commencé.

- Heure d'ingestion : heure à laquelle les données du flux ont été ingérées dans le service de streaming. Si l'heure de l'événement n'est pas intégrée à l'événement lui-même, cette heure peut être utilisée pour le fenêtrage dans certains cas.

Fenêtrage

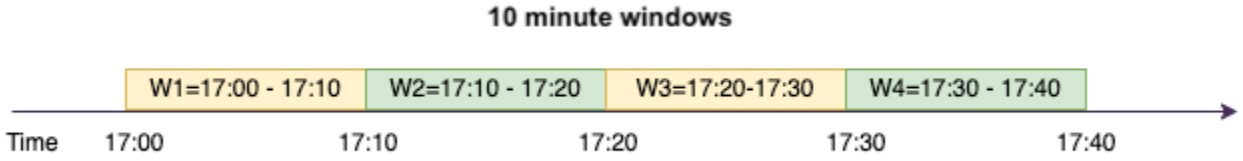
Le fenêtrage est une technique qui permet de regrouper et d'agréger plusieurs événements par event-time-window. Dans les exemples suivants, nous découvrirons les avantages du fenêtrage et les circonstances dans lesquelles vous l'utiliseriez.

Selon le cas d'utilisation métier, Spark prend en charge trois types de fenêtres temporelles.

- Fenêtre pivotante : série de tailles fixes qui ne se chevauchent pas et event-time-windows sur lesquelles vous pouvez agréger.
- Fenêtre défilante : similaire aux fenêtres bascules en ce sens qu'elles sont « de taille fixe », mais les fenêtres peuvent se chevaucher ou défiler tant que la durée du défilement est inférieure à la durée de la fenêtre elle-même.
- Fenêtre de session : commence par un événement de données d'entrée et continue à s'étendre tant qu'elle reçoit des entrées dans un intervalle ou pendant une durée d'inactivité. La longueur d'une fenêtre de session peut être statique ou dynamique, selon les entrées.

Fenêtre bascule

La fenêtre Tumbling est une série de tailles fixes qui ne se chevauchent pas et event-time-windows sur lesquelles vous pouvez agréger. Penchons-nous sur un exemple concret pour comprendre.



La société ABC Auto souhaite réaliser une campagne marketing pour une nouvelle marque de voitures de sport. Elle veut choisir une ville regroupant le plus de fans de voitures de sport. Pour atteindre cet objectif, elle publie une courte publicité de 15 secondes présentant la voiture sur son site Web. Tous les « clics » et la « ville » correspondante sont enregistrés et diffusés. Amazon Kinesis Data Streams Nous voulons compter le nombre de clics dans une fenêtre de 10 minutes et le regrouper par ville pour voir quelle ville est la plus demandée. Voici la sortie de l'agrégation.

window_start_time	window_end_time	city	total_clicks
10/07/2023 17:00:00	10/07/2023 17:10:00	Dallas	75
10/07/2023 17:00:00	10/07/2023 17:10:00	Chicago	10
10/07/2023 17:20:00	10/07/2023 17:30:00	Dallas	20
10/07/2023 17:20:00	10/07/2023 17:30:00	Chicago	50

Comme expliqué ci-dessus, event-time-windows ils sont différents des intervalles de déclenchement. Par exemple, même si votre heure de déclenchement est toutes les minutes, les résultats en sortie n'afficheront que des fenêtres d'agrégation de 10 minutes sans chevauchement. Pour optimiser, il est préférable que l'intervalle de déclenchement soit aligné sur le event-time-window.

Dans le tableau ci-dessus, Dallas a enregistré 75 clics au cours de la fenêtre 17 h 00 – 17 h 10, tandis que Chicago a enregistré 10 clics. De plus, il n'y a aucune donnée pour la fenêtre 17 h 10 – 17 h 20 pour aucune ville, donc cette fenêtre est omise.

Vous pouvez désormais effectuer une analyse plus approfondie de ces données dans l'application d'analytique en aval afin de déterminer la ville la plus privilégiée pour mener la campagne marketing.

Utilisation de fenêtres bascules dans AWS Glue

1. Créez un Amazon Kinesis Data Streams DataFrame et lisez-le. Exemple :

```
parsed_df = kinesis_raw_df \
                .selectExpr('CAST(data AS STRING)') \
                .select(from_json("data", ticker_schema).alias("data")) \
                .select('data.event_time','data.ticker','data.trade','data.volume',
 'data.price')
```

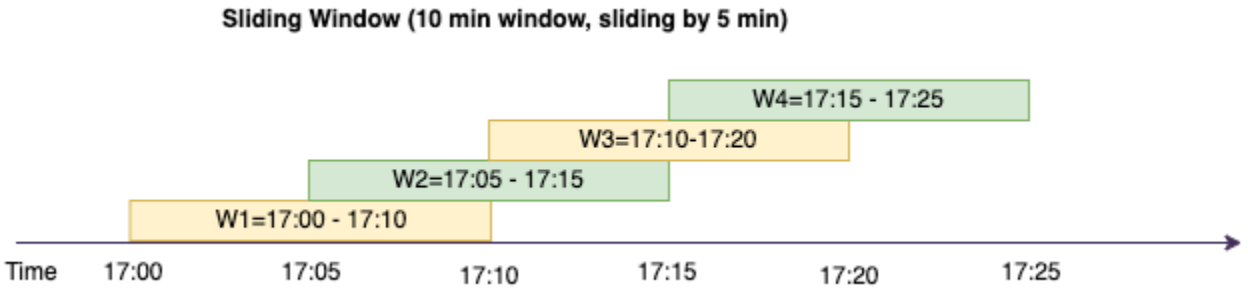
2. Traitez les données dans une fenêtre bascule. Dans l'exemple ci-dessous, les données sont regroupées en fonction du champ de saisie « event_time » dans des fenêtres bascules de 10 minutes et en écrivant le résultat dans un lac de données Amazon S3.

```
grouped_df = parsed_df \
                .groupBy(window("event_time", "10 minutes"), "city") \
                .agg(sum("clicks").alias("total_clicks"))
```

```
summary_df = grouped_df \
                .withColumn("window_start_time", col("window.start")) \
                .withColumn("window_end_time", col("window.end")) \
                .withColumn("year", year("window_start_time")) \
                .withColumn("month", month("window_start_time")) \
                .withColumn("day", dayofmonth("window_start_time")) \
                .withColumn("hour", hour("window_start_time")) \
                .withColumn("minute", minute("window_start_time")) \
                .drop("window")

write_result = summary_df \
                .writeStream \
                .format("parquet") \
                .trigger(processingTime="10 seconds") \
                .option("checkpointLocation", "s3a://bucket-stock-stream/stock-
stream-catalog-job/checkpoint/") \
                .option("path", "s3a://bucket-stock-stream/stock-stream-catalog-
job/summary_output/") \
                .partitionBy("year", "month", "day") \
                .start()
```

Fenêtre défilante

Les fenêtres défilantes sont similaires aux fenêtres bascules en ce sens qu'elles sont « de taille fixe », mais les fenêtres peuvent se chevaucher ou défiler tant que la durée du défilement est inférieure à la durée de la fenêtre elle-même. En raison de la nature du défilement, une entrée peut être liée à plusieurs fenêtres.



Pour mieux comprendre, prenons l'exemple d'une banque qui souhaite détecter une éventuelle fraude par carte de crédit. Une application en streaming pourrait surveiller un flux continu de transactions par carte de crédit. Ces transactions pourraient être agrégées dans des fenêtres

d'une durée de 10 minutes et toutes les 5 minutes, la fenêtre avancerait, éliminant les 5 minutes de données les plus anciennes et ajoutant les 5 dernières minutes de nouvelles données. Dans chaque fenêtre, les transactions pourraient être regroupées par pays pour vérifier l'absence de schémas suspects, comme une transaction aux États-Unis immédiatement suivie d'une autre en Australie. Par souci de simplicité, qualifions ces transactions de fraude lorsque le montant total des transactions est supérieur à 100 USD. Si un tel schéma est détecté, cela indique une fraude potentielle et la carte pourrait être gelée.

Le système de traitement des cartes de crédit envoie un flux d'événements de transaction à Kinesis pour chaque numéro de carte et pour le pays. Une AWS Glue tâche exécute l'analyse et produit le résultat agrégé suivant.

window_start_time	window_end_time	card_last_four	country	total_amount
10/07/2023 17:00:00	10/07/2023 17:10:00	6544	ETATS-UNIS	85
10/07/2023 17:00:00	10/07/2023 17:10:00	6544	Australie	10
10/07/2023 17:05:45	10/07/2023 17:15:45	6544	ETATS-UNIS	50
10/07/2023 17:10:45	10/07/2023 17:20:45	6544	ETATS-UNIS	50
10/07/2023 17:10:45	10/07/2023 17:20:45	6544	Australie	150

Sur la base de l'agrégation ci-dessus, vous pouvez voir la fenêtre de 10 minutes défiler toutes les 5 minutes, résumée au montant de la transaction. L'anomalie est détectée dans la fenêtre de 17 h 10 à 17 h 20 lorsqu'il y a une valeur aberrante, c'est-à-dire une transaction de 150 USD en Australie. AWS Glue peut détecter cette anomalie et envoyer un événement d'alerte avec la clé incriminée vers une rubrique SNS à l'aide de boto3. Par la suite, une fonction Lambda peut s'abonner à cette rubrique et prendre des mesures.

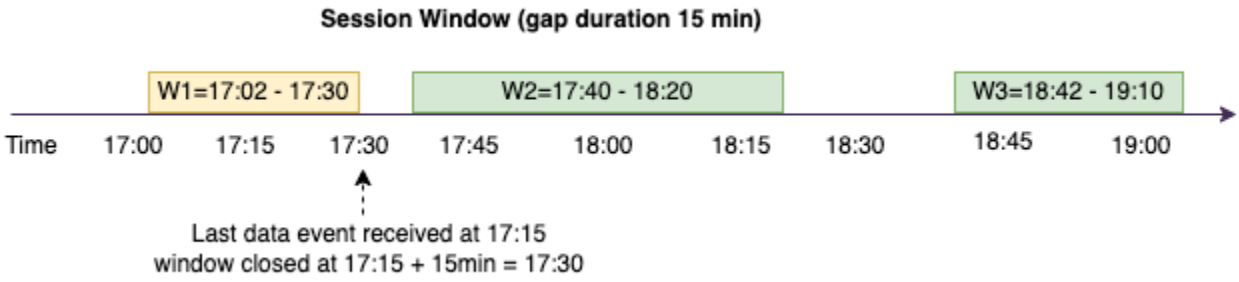
Traitement des données dans une fenêtre défilante

La clause `group-by` et la fonction de fenêtre sont utilisées pour implémenter la fenêtre défilante comme indiqué ci-dessous.

```
grouped_df = parsed_df \
            .groupBy(window(col("event_time"), "10 minute", "5 min"), "country",
 "card_last_four") \
            .agg(sum("tx_amount").alias("total_amount"))
```

Fenêtre de session

Contrairement aux deux fenêtres ci-dessus qui ont une taille fixe, la longueur d'une fenêtre de session peut être statique ou dynamique, selon les entrées. Une fenêtre de session commence par un événement de données d'entrée et continue à s'étendre tant qu'elle reçoit des entrées dans un intervalle ou pendant une durée d'inactivité.



Prenons un exemple. La société ABC Hotel souhaite savoir quelle est la période la plus chargée de la semaine et proposer de meilleures offres à ses clients. Dès qu'un invité s'enregistre, une fenêtre de session est ouverte et Spark maintient un état avec agrégation pour cela. event-time-window Chaque fois qu'un invité s'enregistre, un événement est généré et envoyé à Amazon Kinesis Data Streams. L'hôtel décide que s'il n'y a pas d'enregistrement pendant 15 minutes, il event-time-window peut être fermé. Le prochain event-time-window recommencera lors d'un nouvel enregistrement. La sortie ressemble à ce qui suit.

window_start_time	window_end_time	city	total_checkins
2023-07-10 17:02:00	10/07/2023 17:30:00	Dallas	50
2023-07-10 17:02:00	10/07/2023 17:30:00	Chicago	25

window_start_time	window_end_time	city	total_checkins
10/07/2023 17:40:00	10/07/2023 18:20:00	Dallas	75
10/07/2023 18:50:45	10/07/2023 19:15:45	Dallas	20

Le premier enregistrement a eu lieu à event_time = 17:02 (17 h 02). L'agrégation event-time-window débutera à 17h02. Cette agrégation se poursuivra tant que nous recevrons les événements dans un délai de 15 minutes. Dans l'exemple ci-dessus, le dernier événement que nous avons reçu a eu lieu à 17 h 15, puis il n'y a eu aucun événement pendant les 15 minutes qui ont suivi. Par conséquent, Spark l'a fermé event-time-window à 17 h 15 +15 min = 17 h 30 et l'a défini comme étant de 17 h 02 à 17 h 30. Il a recommencé event-time-window à 17 h 47 lorsqu'il a reçu un nouvel événement relatif aux données d'enregistrement.

Traitement des données dans une fenêtre de session

La clause `group-by` et la fonction de fenêtre sont utilisées pour implémenter la fenêtre défilante.

```
grouped_df = parsed_df \
            .groupBy(session_window(col("event_time"), "10 minute"), "city") \
            .agg(count("check_in").alias("total_checkins"))
```

Modes de sortie

Le mode de sortie est le mode dans lequel les résultats de la table illimitée sont écrits sur le récepteur externe. Trois modes sont disponibles. Dans l'exemple suivant, vous comptez les occurrences d'un mot alors que des lignes de données sont diffusées et traitées dans chaque microlot.

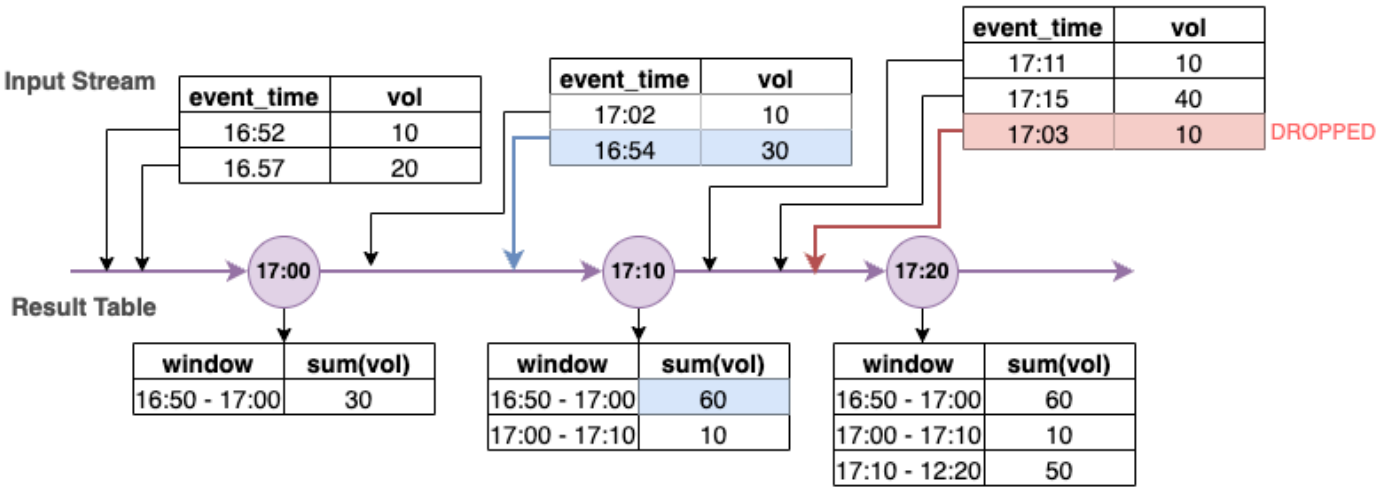
- Mode complet — L'ensemble du tableau des résultats sera écrit sur le récepteur après chaque traitement par microlots, même si le nombre de mots n'a pas été mis à jour dans le courant event-time-window.
- Mode d'ajout : il s'agit du mode par défaut, dans lequel seuls les nouveaux mots et/ou les nouvelles lignes ajoutés à la table de résultats depuis le dernier déclenchement seront écrits dans le récepteur. Ce mode est idéal pour le streaming sans état pour des requêtes telles que map, flatMap, filter, etc.
- Mode de mise à jour : seuls les mots et/ou les lignes de la table de résultats qui ont été mis à jour ou ajoutés depuis le dernier déclenchement seront écrits dans le récepteur.

> **Note**
>
> Mode de sortie = la « mise à jour » n'est pas prise en charge pour les fenêtres de session.

Gestion des données tardives et des filigranes

Lorsque vous travaillez avec des données en temps réel, l'arrivée des données peut être retardée en raison de la latence du réseau et de défaillances en amont. Nous avons besoin d'un mécanisme pour effectuer à nouveau l'agrégation des données manquantes event-time-window. Cependant, pour ce faire, l'état doit être maintenu. Dans le même temps, les anciennes données doivent être nettoyées pour limiter la taille de l'état. La version 2.1 de Spark a ajouté la prise en charge d'une fonctionnalité appelée filigrane qui maintient l'état et permet à l'utilisateur de spécifier le seuil pour les données tardives.

En référence à notre exemple de symbole boursier ci-dessus, considérons que le seuil autorisé pour les données tardives ne doit pas dépasser 10 minutes. Pour faire simple, nous supposerons que la fenêtre est à bascule, que le symbole est AMZ et que la négociation est BUY.



Dans le schéma ci-dessus, nous calculons le volume total sur une fenêtre bascule de 10 minutes. Nous avons le déclenchement à 17 h 00, 17 h 10 et 17 h 20. Au-dessus de la flèche chronologique, nous avons le flux de données d'entrée et ci-dessous se trouve la table des résultats illimités.

Au cours de la première fenêtre bascule de 10 minutes, nous avons agrégé en fonction de event_time et le total_volume a été calculé comme étant 30. Dans le second cas event-time-window, Spark a obtenu le premier événement de données avec event_time= 17:02. Comme il

s'agit de la valeur event_time maximale observée jusqu'à présent par Spark, le seuil du filigrane est défini il y a 10 minutes (c'est-à-dire watermark_event_time = 16:52). Tout événement de données avec un event_time après 16:52 sera pris en compte pour l'agrégation limitée dans le temps et tout événement de données antérieur sera supprimé. Cela permet à Spark de maintenir un état intermédiaire pendant 10 minutes supplémentaires pour tenir compte des données tardives. Vers 17 h 08, durée chronométrée, Spark a reçu un événement avec un event_time = 16:54, ce qui était dans les limites du seuil. Spark a donc recalculé le « 16:50 - 17:00 » event-time-window et le volume total a été mis à jour de 30 à 60.

Cependant, à l'heure de déclenchement 17 h 20, lorsque Spark a reçu un événement avec event_time = 17:15, il a défini le watermark_event_time = 17:05. Par conséquent, l'événement de données tardives avec event_time = 17:03 a été considéré comme « trop tardif » et ignoré.

```
Watermark Boundary = Max(Event Time) - Watermark Threshold
```

Utilisation de filigranes dans AWS Glue

Spark n'émet ni n'écrit les données sur le récepteur externe tant que la limite du filigrane n'est pas dépassée. Pour implémenter un filigrane dans AWS Glue, consultez l'exemple ci-dessous.

```
grouped_df = parsed_df \
                .withWatermark("event_time", "10 minutes") \
                .groupBy(window("event_time", "5 minutes"), "ticker") \
                .agg(sum("volume").alias("total_volume"))
```

À propos de la surveillance des jobs de AWS Glue streaming

La surveillance de votre tâche de streaming est un élément essentiel de la création de votre pipeline ETL. Outre l'interface utilisateur Spark, vous pouvez également utiliser Amazon CloudWatch pour surveiller les métriques. Vous trouverez ci-dessous une liste des métriques de streaming émises par le AWS Glue framework. Pour une liste complète de toutes les AWS Glue métriques, consultez la section Surveillance à AWS Glue l'aide CloudWatch des métriques Amazon.

AWS Glue utilise un framework de streaming structuré pour traiter les événements d'entrée. Vous pouvez soit utiliser l'API Spark directement dans votre code, soit tirer parti du `ForEachBatch` fourni par `GlueContext`, qui publie ces métriques. Pour comprendre ces métriques, nous devons d'abord comprendre `windowSize`.

windowSize : `windowSize` est l'intervalle entre les microlots que vous indiquez. Si vous spécifiez une taille de fenêtre de 60 secondes, la tâche de AWS Glue diffusion attendra 60 secondes (ou plus si le lot précédent n'est pas terminé d'ici là) avant de lire les données d'un lot à partir de la source de diffusion et d'appliquer les transformations fournies dans`ForEachBatch`. Cet intervalle est également appelé intervalle de déclenchement.

Passons en revue les métriques plus en détail pour comprendre les caractéristiques de santé et de performance.

> **Note**
>
> Les métriques sont émises toutes les 30 secondes. Si votre `windowSize` est inférieure à 30 secondes, les métriques rapportées sont une agrégation. Supposons, par exemple, que votre `windowSize` soit de 10 secondes et que vous traitiez régulièrement 20 enregistrements par microlot. Dans ce scénario, la valeur métrique émise pour numRecords serait de 60.
> Aucune métrique n'est émise si aucune donnée n'est disponible pour elle. De plus, dans le cas de la métrique de décalage du consommateur, vous devez activer la fonctionnalité pour obtenir des métriques correspondantes.

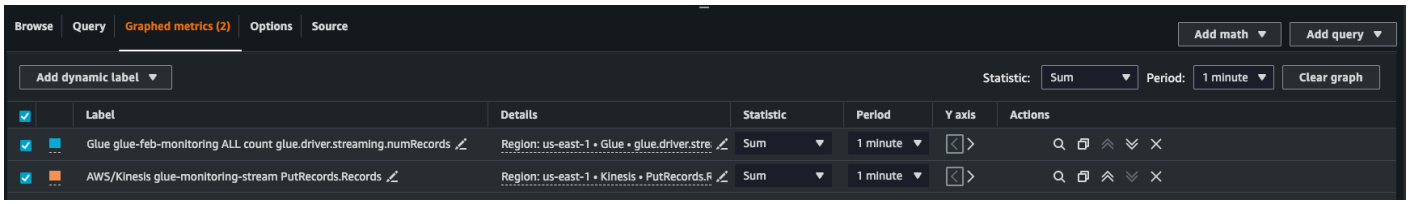
Visualisation des statistiques de AWS Glue streaming

Pour tracer des métriques visuelles :

1. Accédez à Metrics dans la CloudWatch console Amazon, puis choisissez l'onglet Parcourir. Choisissez ensuite Glue sous « Espaces de noms personnalisés ».

2. Choisissez Métriques de tâche pour afficher les métriques de toutes vos tâches.
3. Filtrez les métriques en fonction de votre JobName =, glue-feb-monitoring puis de JobRunId =ALL. Vous pouvez cliquer sur le signe « + » comme indiqué dans la figure ci-dessous pour l'ajouter au filtre de recherche.

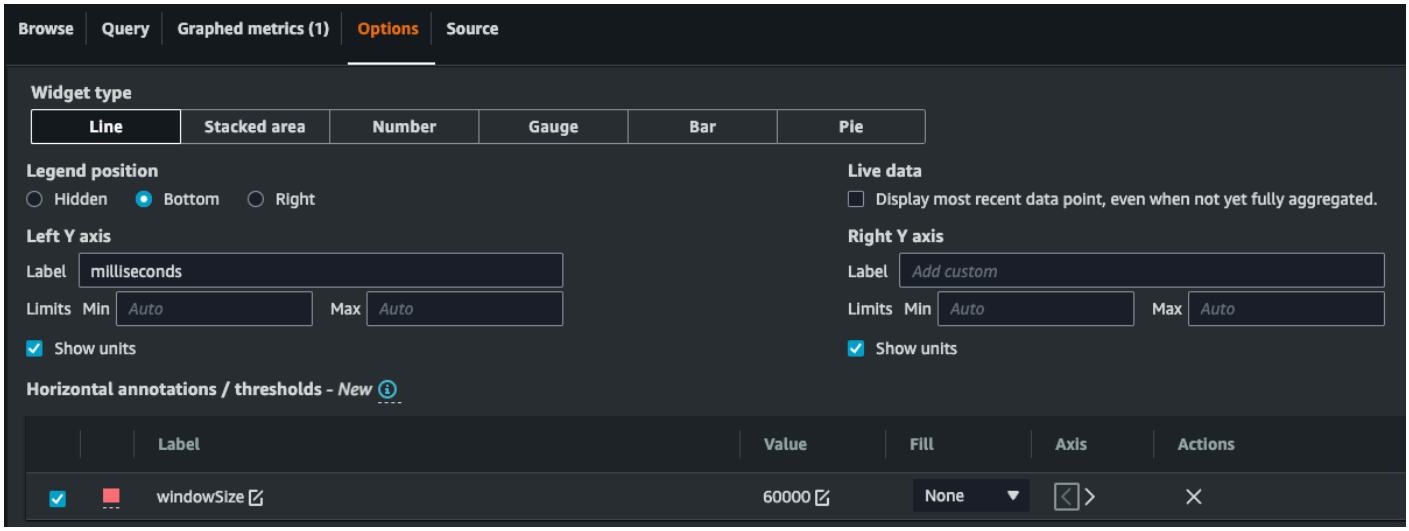
4. Cochez la case correspondant aux métriques qui vous intéressent. Dans la figure ci-dessous, nous avons sélectionné `numberAllExecutors` et `numberMaxNeededExecutors`.



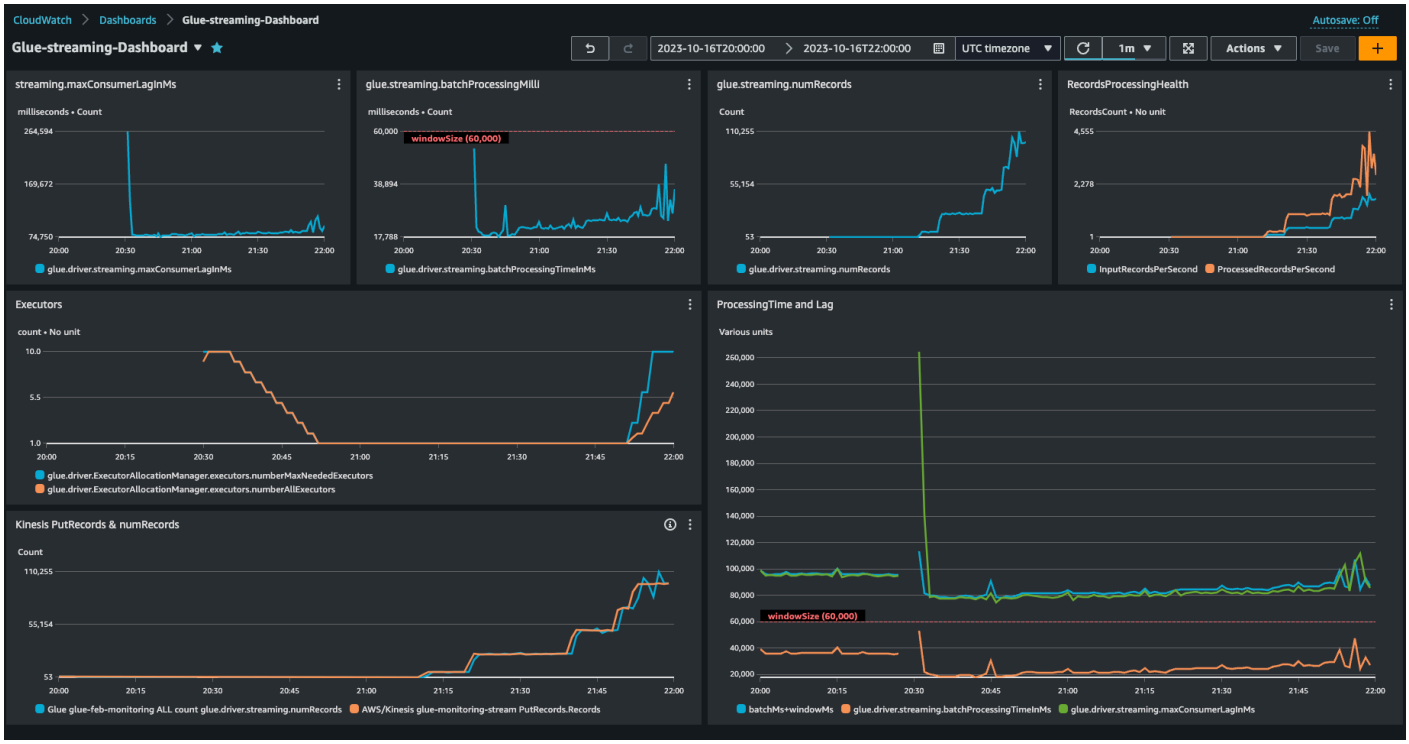
5. Une fois que vous avez sélectionné ces métriques, vous pouvez accéder à l'onglet Graphique des métriques et appliquer vos métriques.

6. Comme les métriques sont émises toutes les minutes, vous pouvez appliquer la « moyenne » sur une minute pour `batchProcessingTimeInMs` et `maxConsumerLagInMs`. Pour `numRecords`, vous pouvez appliquer la « somme » sur chaque minute.

7. Vous pouvez ajouter une annotation `windowSize` horizontale à votre graphique à l'aide de l'onglet Options.



8. Une fois que vous avez sélectionné vos métriques, créez un tableau de bord et ajoutez-le. Voici un exemple de tableau de bord.

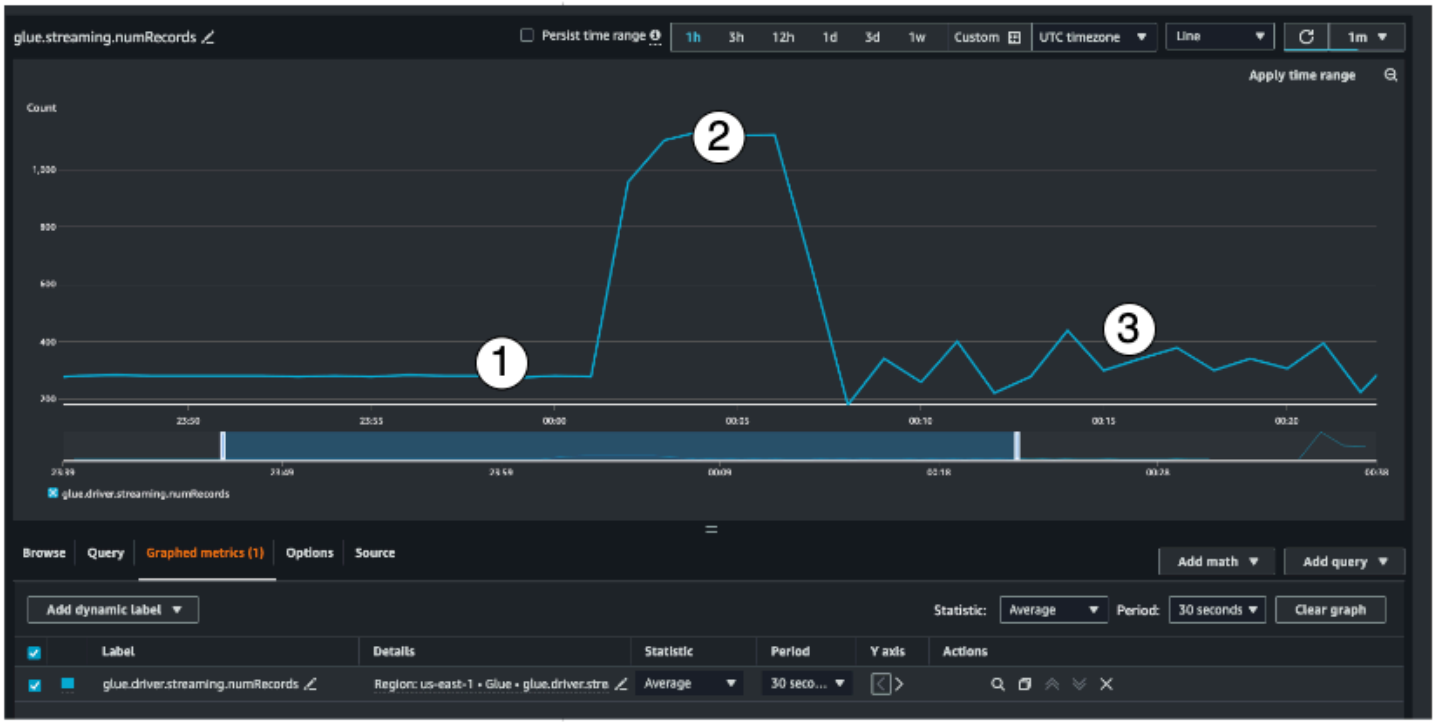


Utilisation des métriques AWS Glue de streaming

Cette section décrit chacune des métriques et la façon dont elles sont corrélées les unes avec les autres.

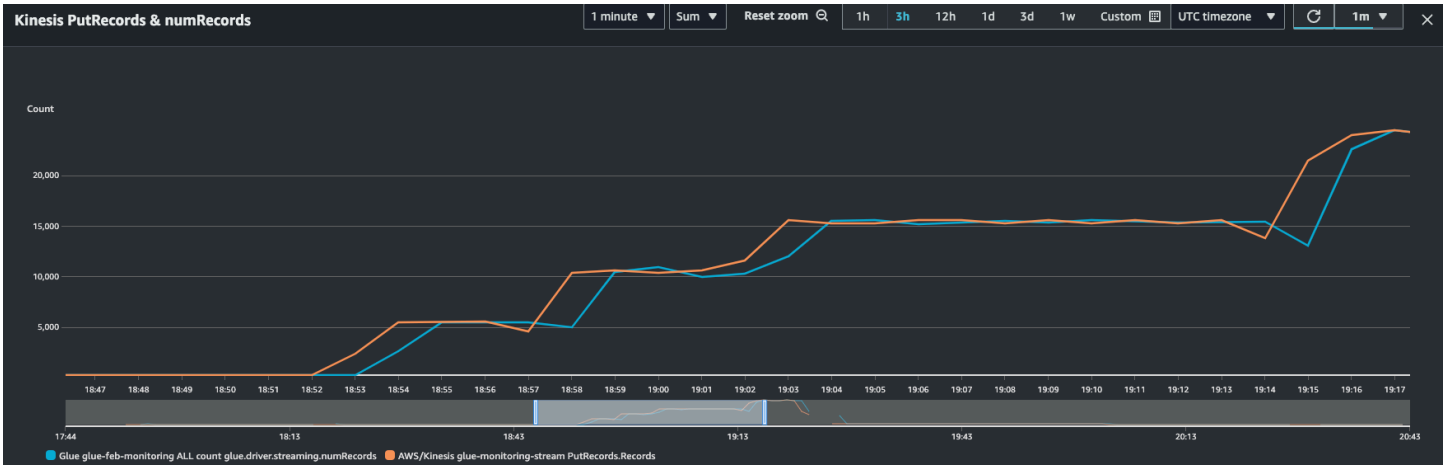
Nombre d'enregistrements (métrique : streaming.numRecords)

Cette métrique indique le nombre d'enregistrements en cours de traitement.



Cette métrique de streaming fournit une visibilité sur le nombre d'enregistrements que vous traitez dans une fenêtre. Outre le nombre d'enregistrements en cours de traitement, cela vous aidera également à comprendre le comportement du trafic d'entrée.

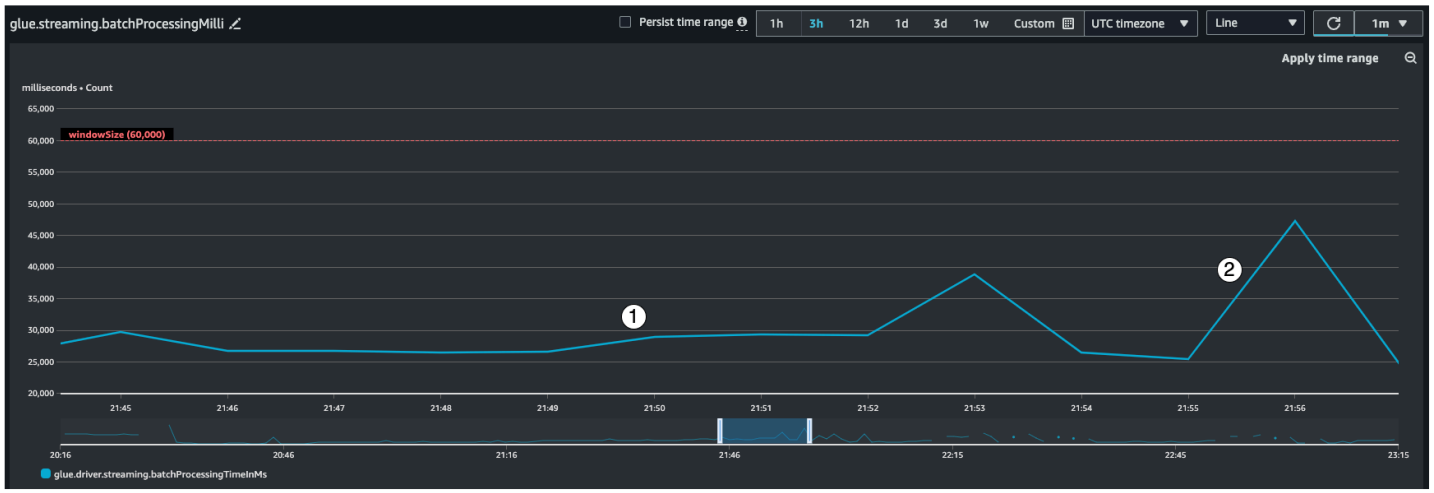
- La métrique nº 1 montre un exemple de trafic stable sans pics. Il s'agit généralement d'applications telles que des capteurs IoT qui collectent des données à intervalles réguliers et les envoient à la source de streaming.
- La métrique nº 2 montre un exemple d'augmentation soudaine du trafic sur une charge par ailleurs stable. Cela peut se produire dans une application de flux de clics lors d'un événement marketing tel que le Black Friday et lorsque le nombre de clics augmente.
- La métrique nº 3 montre un exemple de trafic imprévisible. Un trafic imprévisible ne signifie pas qu'il y a un problème. Il s'agit simplement de la nature des données d'entrée. Pour en revenir à l'exemple des capteurs IoT, vous pouvez imaginer des centaines de capteurs qui envoient des événements liés aux changements météorologiques à la source de streaming. Comme les changements météorologiques ne sont pas prévisibles, les données ne le sont pas non plus. Comprendre le modèle de trafic est essentiel pour dimensionner vos programmes d'exécution. Si l'entrée est sujette à des pics de trafic, vous pouvez envisager d'utiliser l'autoscaling (nous y reviendrons plus tard).



Vous pouvez combiner cette métrique avec la PutRecords métrique Kinesis pour vous assurer que le nombre d'événements ingérés et le nombre d'enregistrements lus sont quasiment identiques. Cette approche est utile lorsque vous essayez de comprendre le décalage. Au fur et à mesure que le taux d'ingestion augmente, le temps de `numRecords` lecture augmente également AWS Glue.

Temps de traitement par lots (métrique : streaming). batchProcessingTimeInMs)

La métrique du temps de traitement par lots vous aide à déterminer si le cluster est sous-provisionné ou surapprovisionné.

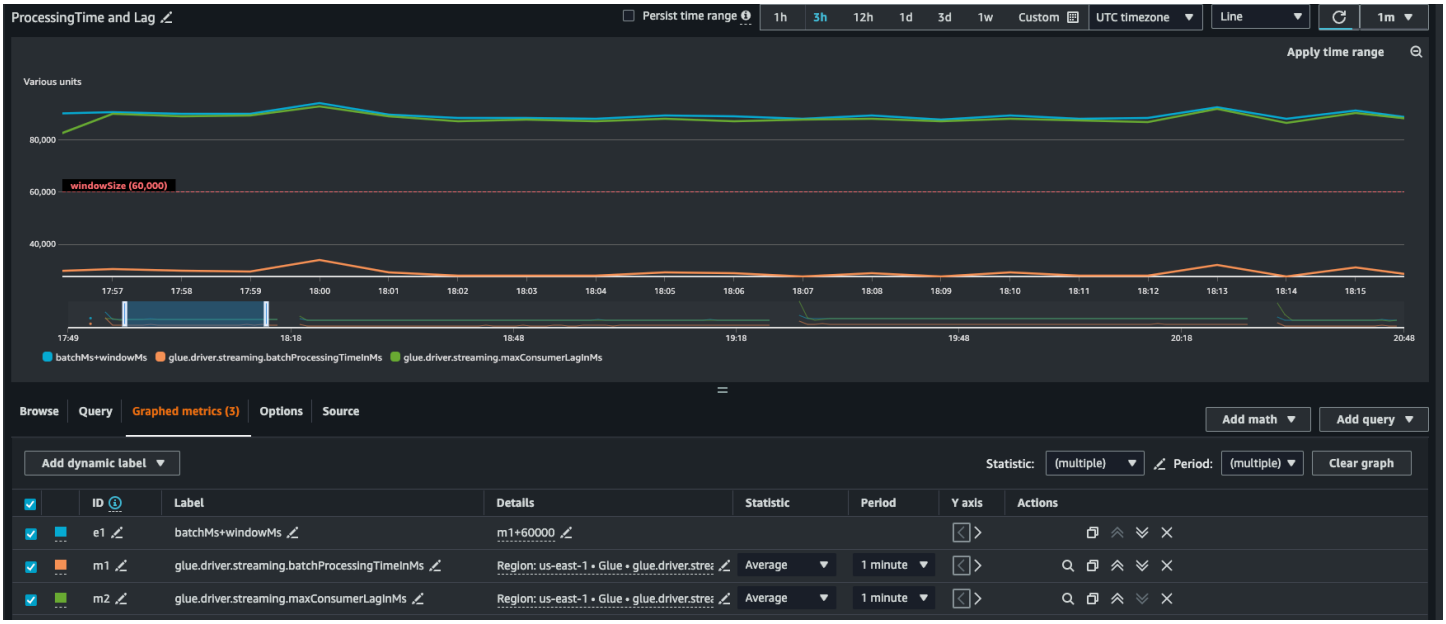


Cette métrique indique le nombre de millisecondes nécessaires au traitement de chaque microlot d'enregistrements. L'objectif principal ici est de surveiller ce temps pour s'assurer qu'il est inférieur à l'intervalle `windowSize`. Ce n'est pas grave si le `batchProcessingTimeInMs` est dépassé temporairement, tant qu'il se rétablit dans l'intervalle de fenêtre suivant. La métrique nº 1 indique un temps plus ou moins stable de traitement de la tâche. Toutefois, si le nombre d'enregistrements d'entrée augmente, le temps nécessaire au traitement de la tâche augmentera, comme le montre

la métrique nº 2. Si le `numRecords` n'augmente pas, mais que le temps de traitement augmente, vous devrez examiner de plus près le traitement des tâches sur les programmes d'exécution. Il est recommandé de définir un seuil et une alerte pour s'assurer que le `batchProcessingTimeInMs` ne dépasse pas 120 % pendant plus de 10 minutes. Pour plus d'informations sur le paramétrage des alarmes, consultez la section Utilisation des CloudWatch alarmes Amazon.

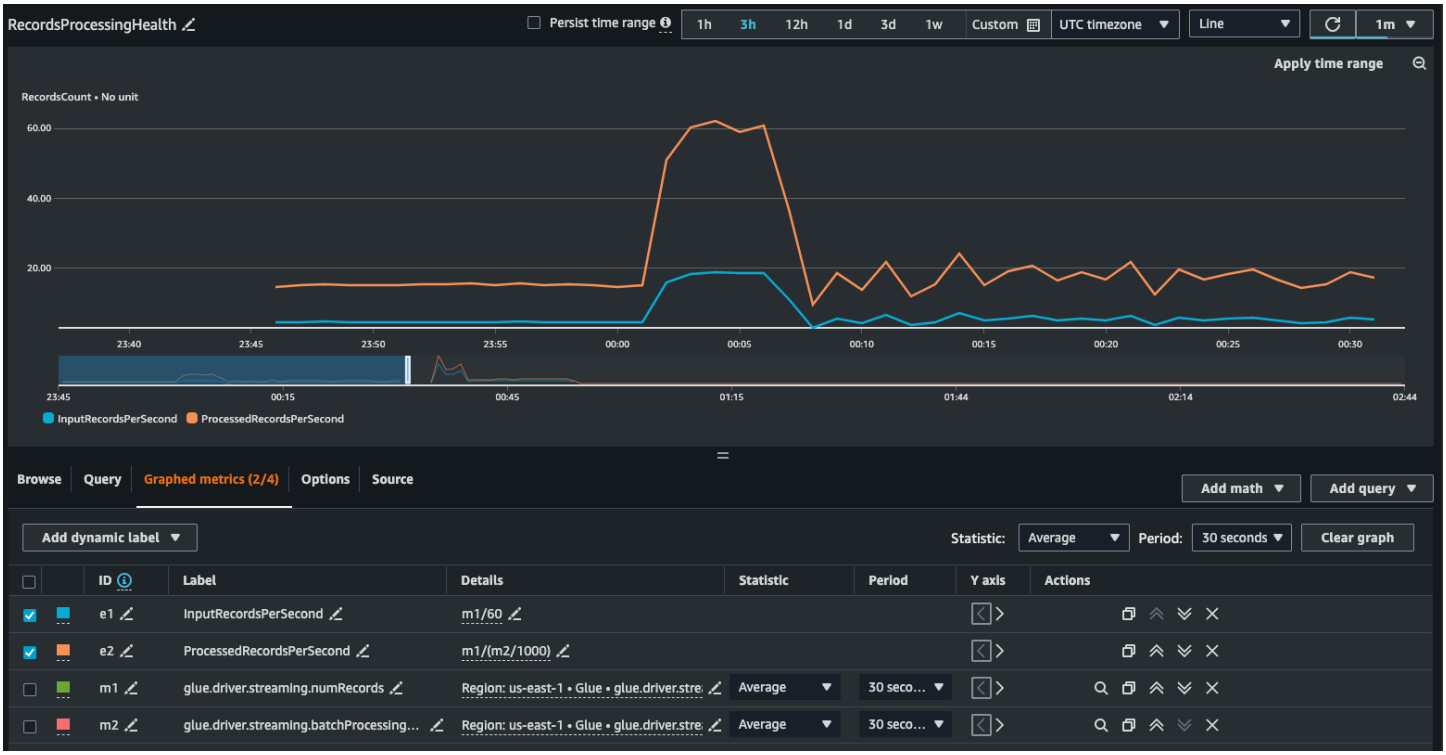
Retard de consommation (métrique : streaming). maxConsumerLagInMs)

La métrique du décalage de consommateur vous aide à comprendre s'il existe un décalage dans le traitement des événements. Si votre décalage est trop important, vous risquez de ne pas respecter le SLA de traitement dont dépend votre activité, même si vous avez une taille de fenêtre windowSize correcte. Vous devez activer explicitement cette métrique à l'aide de l'option de connexion `emitConsumerLagMetrics`. Pour de plus amples informations, veuillez consulter KinesisStreamingSourceOptions.



Métriques dérivées

Pour obtenir des informations plus approfondies, vous pouvez créer des indicateurs dérivés afin d'en savoir plus sur vos jobs de streaming sur Amazon CloudWatch.



Vous pouvez créer un graphique avec des mesures dérivées pour décider si vous devez en utiliser davantage DPUs. Bien que l'autoscaling vous aide à le faire automatiquement, vous pouvez utiliser des métriques dérivées pour déterminer si l'autoscaling fonctionne efficacement.

- InputRecordsPerSecondindique la vitesse à laquelle vous obtenez des enregistrements d'entrée. Il est dérivé comme suit : nombre d'enregistrements d'entrée (Glue.Driver.Streaming.NumRecords)/. WindowSize
- ProcessingRecordsPerSecondindique le rythme auquel vos enregistrements sont traités. Il est dérivé comme suit : nombre d'enregistrements d'entrée (Glue.Driver.Streaming.NumRecords)/. batchProcessingTime InMs

Si le taux d'entrée est supérieur au taux de traitement, vous devrez peut-être augmenter la capacité pour traiter votre tâche ou augmenter le parallélisme.

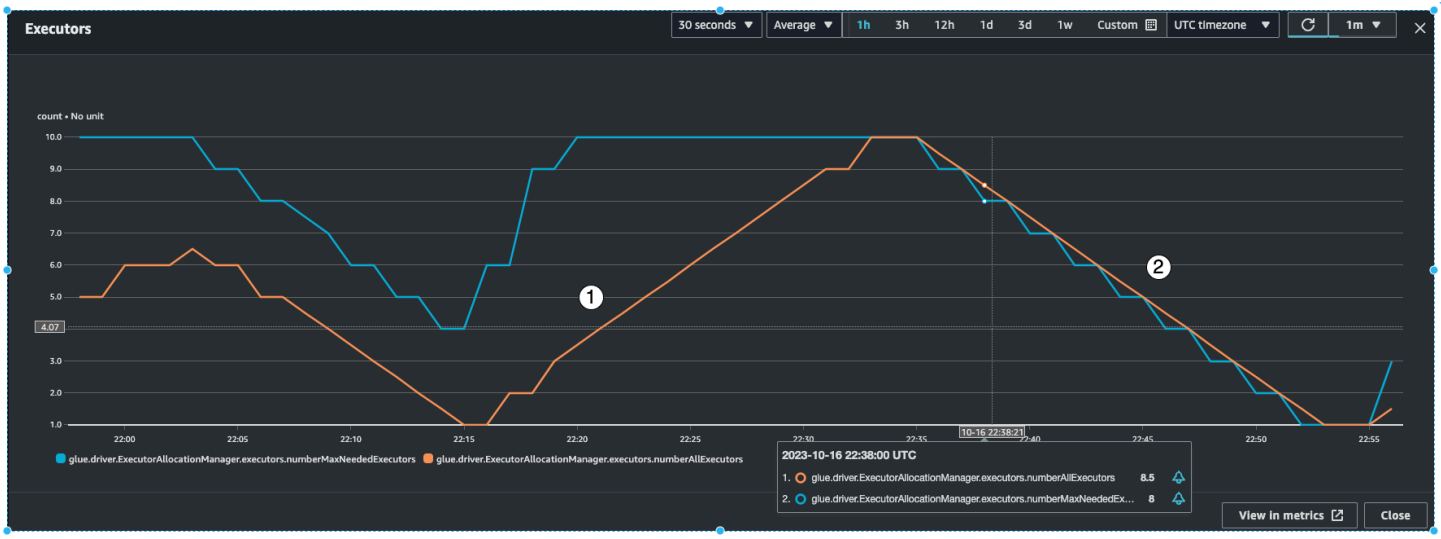
Métriques d'autoscaling

Lorsque votre trafic d'entrée connaît des pics, vous devez envisager d'activer l'autoscaling et de spécifier le nombre maximal de travailleurs. Avec cela, vous obtenez deux métriques supplémentaires, `numberAllExecutors` et `numberMaxNeededExecutors`.

- numberAllExecutorsest le nombre d'exécuteurs de tâches en cours d'exécution active

- numberMaxNeededLes exécuteurs sont le nombre maximum d'exécuteurs de tâches (en cours d'exécution et en attente) nécessaires pour satisfaire la charge actuelle.

Ces deux métriques vous aideront à déterminer si votre autoscaling fonctionne correctement.



AWS Glue surveillera la `batchProcessingTimeInMs` métrique sur quelques microlots et effectuera l'une des deux opérations suivantes. Il fera monter en puissance le nombre de programmes d'exécution, si `batchProcessingTimeInMs` est plus proche de la `windowSize`, ou mettra à l'échelle horizontale les programmes d'exécution, si `batchProcessingTimeInMs` est comparativement inférieur à `windowSize`. En outre, il utilisera un algorithme pour mettre à l'échelle les programmes d'exécution par étapes.

- La métrique nº 1 vous montre comment les programmes d'exécution actifs augmentent pour rattraper le nombre maximal de programmes d'exécution nécessaires afin de traiter la charge.
- La métrique nº 2 vous montre comment le nombre de programmes d'exécution actifs a été mis à l'échelle horizontale comme le `batchProcessingTimeInMs` était faible.

Vous pouvez utiliser ces métriques pour surveiller le parallélisme actuel au niveau des programmes d'exécution et ajuster le nombre maximal de travailleurs dans votre configuration d'autoscaling en conséquence.

Comment obtenir les meilleures performances

Spark essaiera de créer une tâche par partition, à partir de laquelle lire, dans le flux Amazon Kinesis. Les données de chaque partition deviennent une partition. Il répartira ensuite ces tâches entre les

programmes d'exécution/travailleurs, en fonction du nombre de cœurs de chaque travailleur (le nombre de cœurs par travailleur dépend du type de travailleur que vous sélectionnez, `G.025X`, `G.1X`, etc.). Cependant, la façon dont les tâches sont réparties n'est pas déterministe. Toutes les tâches sont exécutées en parallèle sur leurs cœurs respectifs. S'il y a plus de partitions que de cœurs de programme d'exécution disponibles, les tâches sont mises en file d'attente.

Vous pouvez utiliser une combinaison des métriques ci-dessus et du nombre de partitions pour fournir à vos programmes d'exécution une charge stable avec de la place pour les pics. Il est recommandé d'exécuter quelques itérations de votre tâche afin de déterminer le nombre approximatif de travailleurs. Pour une unstable/spiky charge de travail, vous pouvez faire de même en configurant le dimensionnement automatique et le nombre maximal de travailleurs.

Définissez la `windowSize` conformément aux exigences du SLA de votre entreprise. Par exemple, si votre entreprise exige que les données traitées ne soient pas obsolètes de plus de 120 secondes, réglez votre `windowSize` sur au moins 60 secondes afin que le décalage moyen du consommateur soit inférieur à 120 secondes (voir la section sur le décalage du consommateur ci-dessus). À partir de là, en fonction de la quantité `numRecords` et du nombre de fragments, planifiez la capacité en vous DPUs assurant que vous `batchProcessingTimeInMs` êtes inférieure à 70 % de votre capacité la `windowSize` plupart du temps.

> **Note**
>
> Les hot shards peuvent fausser les données, ce qui signifie que certains shards/partitions sont beaucoup plus gros que d'autres. Certaines tâches exécutées en parallèle peuvent donc prendre plus de temps, ce qui peut entraîner des ralentissements. Par conséquent, le lot suivant ne pourra pas démarrer tant que toutes les tâches du précédent ne seront pas terminées, ce qui aura un impact sur le `batchProcessingTimeInMillis` et le décalage maximal.

Intégrations sans ETL

Zero-ETL est un ensemble d'intégrations entièrement gérées AWS qui minimise le besoin de créer des pipelines de données ETL pour les cas d'utilisation courants d'ingestion et de réplication. Il rend les données disponibles dans Amazon SageMaker Lakehouse et Amazon Redshift à partir de multiples sources opérationnelles, transactionnelles et applicatives. Avec l'intégration zéro ETL, vous disposez de données plus récentes pour l'analytique, l'IA/le ML et la génération de rapports. Vous obtenez des informations plus précises et plus opportunes pour des cas d'utilisation tels que les tableaux de bord métier, l'optimisation de l'expérience de jeu, le suivi de la qualité des données et l'analyse du comportement des clients. Vous pouvez établir des prévisions basées sur les données en toute confiance, améliorer l'expérience client et promouvoir des informations basées sur les données à l'échelle de l'entreprise.

Amazon Redshift est un service d'entrepôt de données rapide, entièrement géré et doté d'une capacité de plusieurs pétaoctets. Il permet d'analyser de manière efficace, simple et économique toutes vos données grâce à vos outils d'informatique décisionnelle existants.

Amazon SageMaker Lakehouse unifie toutes vos données entre les lacs de données Amazon Simple Storage Service (S3) et les entrepôts de données Amazon Redshift, vous aidant ainsi à créer de puissantes analyses et AI/ML applications à partir d'une seule copie des données. SageMaker Lakehouse vous donne la flexibilité d'accéder à vos données et de les interroger sur place avec tous les outils et moteurs compatibles avec Apache Iceberg. Avec SageMaker Lakehouse, vous avez également la flexibilité d'accéder à vos données et de les interroger sur place à l'aide d'outils et de moteurs compatibles avec Apache Iceberg. En outre, vous pouvez sécuriser vos données grâce à des contrôles précis des accès intégrés, qui sont appliqués à toutes vos données dans tous les outils et moteurs d'analytique. Définissez les autorisations une seule fois et partagez les données en toute confiance au sein de votre organisation.

Fonctionnalités zéro ETL dans AWS Glue

Les intégrations sans ETL AWS Glue simplifient l'ingestion et la réplication des données depuis le service de AWS données, les applications tierces et les bases de données autogérées vers les destinations. AWS

AWS les services pris en charge en tant que sources zéro ETL AWS Glue incluent :

- Amazon DynamoDB

Les applications tierces prises en charge en tant que sources zéro ETL AWS Glue sont les suivantes :

- Facebook Ads
- Instagram Ads
- Salesforce
- Salesforce Marketing Cloud Account Engagement
- SAP OData
- ServiceNow
- Zendesk
- Zoho CRM

Les bases de données autogérées prises en charge en tant que sources zéro ETL incluent : AWS Glue

- Oracle
- SQL Server
- MySQL
- PostgreSQL

Pour plus d'informations sur les intégrations zéro ETL à partir de sources autogérées, voir Intégration AWS zéro ETL pour les sources de base de données autogérées.

AWS les services pris en charge en tant que cibles zéro ETL AWS Glue incluent :

- Amazon Redshift
- Amazon SageMaker Lakehouse

> **Note**
>
> Lorsque vous créez une intégration zéro ETL avec une AWS Glue source Amazon DynamoDB dans, la cible est prise en charge par Amazon Lakehouse. SageMaker

Conditions préalables à la configuration d'une intégration zéro ETL

La configuration d'une intégration entre la source et la cible nécessite certaines conditions préalables, telles que la configuration des rôles IAM qui AWS Glue permettent d'accéder aux données depuis la source et d'écrire sur la cible, et l'utilisation de clés KMS pour chiffrer les données à l'emplacement intermédiaire ou cible.

Rubriques

- Configuration des ressources source
- Configuration des ressources cible
- Création d'un entrepôt de données Amazon Redshift
- Configuration d'un VPC pour votre intégration zéro ETL
- Configuration d'une intégration entre comptes zéro ETL

Configuration des ressources source

Effectuez les tâches de configuration suivantes selon les besoins de votre source.

Configuration du rôle source

Cette section décrit la manière de transmettre un rôle source pour permettre à l'intégration zéro ETL d'accéder à votre connexion. Cela s'applique également uniquement aux sources SaaS.

> **Note**
>
> Pour limiter l'accès à quelques connexions uniquement, vous pouvez d'abord créer la connexion pour obtenir l'ARN de connexion. Consultez Configuration d'une source pour une intégration zéro ETL.

Créez un rôle disposant des autorisations nécessaires pour que l'intégration puisse accéder à la connexion :

JSON

```
{
```

```
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "GlueConnections",
            "Effect": "Allow",
            "Action": [
                "glue:GetConnections",
                "glue:GetConnection"
            ],
            "Resource": [
       "arn:aws:glue:*:111122223333:catalog",
       "arn:aws:glue:us-east-1:111122223333:connection/*"
            ]
        },
        {
            "Sid": "GlueActionBasedPermissions",
            "Effect": "Allow",
            "Action": [
                "glue:ListEntities",
                "glue:RefreshOAuth2Tokens"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

Stratégie d'approbation :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "glue.amazonaws.com"
        ]
      },
```

```
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Configuration des ressources cible

Effectuez les tâches de configuration suivantes selon les besoins pour AWS Glue la cible d'intégration du catalogue de Amazon Redshift données ou de l'entrepôt de données.

Pour les intégrations avec une cible AWS Glue de base de données :

- Configuration d'une AWS Glue base de données
- Fourniture d'une politique d'accès basée sur les ressources (RBAC) du catalogue
- Création d'un rôle IAM cible

Pour les intégrations avec une cible d' Amazon Redshift entrepôt de données :

- https://docs.aws.amazon.com/glue/latest/dg/zero-etl-prerequisites.html#zero-etl-setup-target-redshift-data-warehouse

Configuration d'une AWS Glue base de données

Pour les intégrations qui utilisent une AWS Glue base de données :

Pour configurer une base de données cible dans le catalogue de AWS Glue données avec un emplacement Amazon S3 :

1. Sur la page d'accueil de la AWS Glue console, sélectionnez Base de données sous Catalogue de données.
2. Choisissez Ajouter une base de données en haut à droite. Si vous avez déjà créé une base de données, assurez-vous que l'emplacement avec l'URI Amazon S3 est défini pour la base de données.
3. Saisissez un nom et un emplacement (URI Amazon S3). Notez que l'emplacement est requis pour l'intégration zéro ETL. Cliquez sur Créer une base de données lorsque vous avez terminé.

> **Note**
>
> Le compartiment Amazon S3 doit se trouver dans la même région que la AWS Glue base de données.

Pour plus d'informations sur la création d'une nouvelle base de données dans AWS Glue, voir Mise en route avec le catalogue de AWS Glue données.

Vous pouvez également utiliser la commande CLI `create-database` pour créer la base de données dans AWS Glue. Notez que le `LocationUri` dans `--database-input` est obligatoire.

Optimisation des tables Iceberg

Une fois qu'une table est créée AWS Glue dans la base de données cible, vous pouvez activer le compactage pour accélérer les requêtes dans Amazon Athena. Pour plus d'informations sur la configuration des ressources (rôle IAM) pour le compactage, consultez Conditions préalables requises pour l'optimisation des tables.

Pour plus d'informations sur la configuration du compactage sur la AWS Glue table créée par l'intégration, consultez Optimisation des tables Iceberg.

Fourniture d'une politique d'accès basée sur les ressources (RBAC) du catalogue

Pour les intégrations qui utilisent une AWS Glue base de données, ajoutez les autorisations suivantes à la politique RBAC du catalogue afin de permettre les intégrations entre la source et la cible.

> **Note**
>
> Pour les intégrations entre comptes, la politique de rôle d'Alice (utilisateur créant l'intégration) et la politique de ressources du catalogue doivent autoriser `glue:CreateInboundIntegration` sur la ressource. Pour un même compte, une politique de ressources ou une politique de rôle autorisant `glue:CreateInboundIntegration` sur la ressource est suffisante. Les deux scénarios doivent toujours autoriser `glue.amazonaws.com` pour `glue:AuthorizeInboundIntegration`.

Vous pouvez accéder aux paramètres du catalogue sous Catalogue de données. Indiquez ensuite les autorisations suivantes et complétez les informations manquantes.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Principal": {
        "AWS": [
         "arn:aws:iam::123456789012:user/Alice"
        ]
      },
      "Effect": "Allow",
      "Action": [
        "glue:CreateInboundIntegration"
      ],
      "Resource": [
       "arn:aws:glue:us-east-1:111122223333:catalog",
    "arn:aws:glue:us-east-1:111122223333:database/DatabaseName"
      ],
      "Condition": {
        "StringLike": {
        "aws:SourceArn": "arn:aws:dynamodb:us-east-1:444455556666:table/<table-
name>"
        }
      }
    },
    {
      "Principal": {
        "Service": [
          "glue.amazonaws.com"
        ]
      },
      "Effect": "Allow",
      "Action": [
        "glue:AuthorizeInboundIntegration"
      ],
      "Resource": [
       "arn:aws:glue:us-east-1:111122223333:catalog",
    "arn:aws:glue:us-east-1:111122223333:database/DatabaseName"
      ],
```

```
      "Condition": {
        "StringEquals": {
        "aws:SourceArn": "arn:aws:dynamodb:us-east-1:444455556666:table/<table-
name>"
        }
      }
    }
  ]
}
```

Création d'un rôle IAM cible

Créez un rôle IAM cible avec les autorisations et les relations d'approbation suivantes :

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
            "Effect": "Allow"
        },
        {
            "Action": [
                "s3:GetObject",
                "s3:PutObject",
                "s3:DeleteObject"
            ],
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/prefix/*",
            "Effect": "Allow"
        },
        {
            "Action": [
                "glue:GetDatabase"
            ],
            "Resource": [
                "arn:aws:glue:us-east-1:111122223333:catalog",
                "arn:aws:glue:us-east-1:111122223333:database/DatabaseName"
            ],
            "Effect": "Allow"
```

```
        },
        {
            "Action": [
                "glue:CreateTable",
                "glue:GetTable",
                "glue:GetTables",
                "glue:DeleteTable",
                "glue:UpdateTable",
                "glue:GetTableVersion",
                "glue:GetTableVersions",
                "glue:GetResourcePolicy"
            ],
            "Resource": [
                "arn:aws:glue:us-east-1:111122223333:catalog",
                "arn:aws:glue:us-east-1:111122223333:database/DatabaseName",
                "arn:aws:glue:us-east-1:111122223333:table/DatabaseName/*"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "cloudwatch:PutMetricData"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "cloudwatch:namespace": "AWS/Glue/ZeroETL"
                }
            },
            "Effect": "Allow"
        },
        {
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": "*",
            "Effect": "Allow"
        }
    ]
}
```

Ajoutez la politique de confiance suivante pour permettre au AWS Glue service d'assumer le rôle :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "glue.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Création d'un entrepôt de données Amazon Redshift

Lorsque votre objectif d'intégration zéro ETL est un entrepôt de données Amazon Redshift, créez-le si vous n'en avez pas déjà un. Pour créer un groupe de travail Amazon Redshift sans serveur, voir Création d'un groupe de travail avec un espace de noms. Pour créer un cluster Amazon Redshift, consultez Création d'un cluster.

Le `enable_case_sensitive_identifier` paramètre doit être activé dans le Amazon Redshift groupe de travail ou le cluster cible pour que l'intégration soit réussie. Pour plus d'informations sur l'activation de la sensibilité à la casse, consultez Activation de la sensibilité à la casse pour votre entrepôt de données dans le Guide de la gestion Amazon Redshift.

Une fois la configuration Amazon Redshift du groupe de travail ou du cluster terminée, vous devez configurer votre entrepôt de données. Pour plus d'informations, reportez-vous à la section Getting started with zero-ETL integrations dans le Guide de Amazon Redshift gestion.

Configuration d'un VPC pour votre intégration zéro ETL

Pour configurer un VPC pour votre intégration zéro ETL :

1. Accédez à VPC > Votre VPCs et choisissez Create VPC.
   a. Sélectionnez VPC et plus encore.

b. Définissez le nom de votre VPC.

c. Définissez le IPv4 CIDR : 10.0.0.0/16.

d. Définissez le nombre de zones de disponibilité sur 1.

e. Définissez le nombre de sous-réseaux publics et privés sur 1.

f. Définissez Passerelles NAT sur Aucune.

g. Définissez Points de terminaison d'un VPC sur Passerelle S3.

h. Activez les noms d'hôte DNS et la résolution DNS.

2. Accédez à Points de terminaison, puis choisissez Créer un point de terminaison.
3. Créez des points de terminaison pour ces services dans le sous-réseau privé de votre VPC (utilisez le groupe de sécurité par défaut) :

   a. com.amazonaws.us-east-1.lambda

   b. com.amazonaws.us-east-1.glue

   c. com.amazonaws.us-east-1.sts

Créez la AWS Glue connexion :

1. Accédez à AWS Glue > Connexions de données, puis choisissez Créer une connexion.
2. Sélectionnez Réseau.
3. Sélectionnez le VPC, le sous-réseau (privé) et le groupe de sécurité par défaut que vous avez créés.

Configuration du rôle cible pour le VPC

Le rôle cible doit disposer des autorisations suivantes (en plus des autres autorisations requises par les ETl intégrations Zéro) :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "CustomerVpc",
      "Effect": "Allow",
```

```
      "Action": [
        "ec2:CreateTags",
        "ec2:DeleteTags",
        "ec2:DescribeRouteTables",
        "ec2:DescribeVpcEndpoints",
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeSubnets",
        "ec2:CreateNetworkInterface",
        "ec2:DeleteNetworkInterface",
        "glue:GetConnection"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
```

Configuration des propriétés des ressources de l'étape cible

Si vous utilisez la CLI, définissez les propriétés des ressources du segment cible sur la AWS Glue base de données cible que vous avez créée. Transmettez l'ARN du rôle cible, ainsi que le nom de la AWS Glue connexion.

```
aws glue create-integration-resource-property \
--resource-arn arn:aws:glue:us-east-1:<account-id>:database/exampletarget \
--target-processing-properties '{"RoleArn" : "arn:aws:iam::<account-id>:role/example-role", "ConnectionName":"example-vpc-3"}' \
--endpoint-url https://example.amazonaws.com --region us-east-1
```

Erreurs client possibles

Les erreurs client possibles associées à une intégration configurée avec un VPC sont les suivantes.

Message d'erreur	Action requise
Le rôle fourni n'est pas autorisé à effectuer du collage : GetConnection lors de la connexion.	Mettre à jour la politique de rôle

Message d’erreur	Action requise
Ajoutez cette autorisation à la politique de rôle, puis attendez que l’intégration soit rétablie.	
Le rôle fourni n'est pas autorisé à exécuter ec2 :DescribeSubnets. Ajoutez cette autorisat ion à la politique de rôle, puis attendez que l’intégration soit rétablie.	Mettre à jour la politique de rôle
Le rôle fourni n'est pas autorisé à exécuter ec2 :DescribeSecurityGroups. Ajoutez cette autorisation à la politique de rôle, puis attendez que l’intégration soit rétablie.	Mettre à jour la politique de rôle
Le rôle fourni n'est pas autorisé à exécuter ec2 :DescribeVpcEndpoints. Ajoutez cette autorisation à la politique de rôle, puis attendez que l’intégration soit rétablie.	Mettre à jour la politique de rôle
Le rôle fourni n'est pas autorisé à exécuter ec2 :DescribeRouteTables. Ajoutez cette autorisation à la politique de rôle, puis attendez que l’intégration soit rétablie.	Mettre à jour la politique de rôle
Le rôle fourni n'est pas autorisé à exécuter ec2 :CreateTags. Ajoutez cette autorisation à la politique de rôle, puis attendez que l’intégration soit rétablie.	Mettre à jour la politique de rôle
Le rôle fourni n'est pas autorisé à exécuter ec2 :CreateNetworkInterface. Ajoutez cette autorisation à la politique de rôle, puis attendez que l’intégration soit rétablie.	Mettre à jour la politique de rôle
Le sous-réseau de connexion fourni ne contient pas de point de terminaison S3 ou de passerell e NAT valide. Mettez à jour le sous-réseau, puis attendez que l’intégration soit rétablie.	Mettre à jour les points de terminaison du sous-réseau VPC

Message d'erreur	Action requise
Sous-réseau de connexion introuvable. Mettez à jour le sous-réseau de connexion, puis attendez que l'intégration soit rétablie.	Mettre à jour AWS Glue la connexion
Groupe de sécurité de connexion introuvable. Mettez à jour le groupe de sécurité de connexion, puis attendez que l'intégration soit rétablie.	Mettre à jour AWS Glue la connexion
Impossible de se connecter à S3 via la connexion VPC fournie. Mettez à jour les configurations de sous-réseau, puis attendez que l'intégration soit rétablie.	Mettre à jour les points de terminaison du sous-réseau VPC
Impossible de se connecter à Lambda via la connexion VPC fournie. Mettez à jour les configurations de sous-réseau, puis attendez que l'intégration soit rétablie.	Mettre à jour les points de terminaison du sous-réseau VPC

Configuration d'une intégration entre comptes zéro ETL

Pour configurer une intégration entre comptes zéro ETL :

1. Configurez une politique de ressources cible comme décrit dans Fourniture d'une politique d'accès basée sur les ressources (RBAC) du catalogue. Assurez-vous que le rôle du compte source est explicitement autorisé sur la ressource cible.
2. Vérifiez que le rôle du compte source (le rôle utilisé pour créer l'intégration) possède les caractéristiques suivantes :

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "Stmt123456789012",
            "Action": [
```

```
                "glue:CreateInboundIntegration"
            ],
            "Effect": "Allow",
            "Resource": [
             "arn:aws:glue:us-east-1:111122223333:catalog",
       "arn:aws:glue:us-east-1:111122223333:database/DatabaseName"
            ]
        }
    ]
}
```

3. Créez l'intégration comme décrit dans Création d'une intégration.

Configuration d'une source pour une intégration zéro ETL

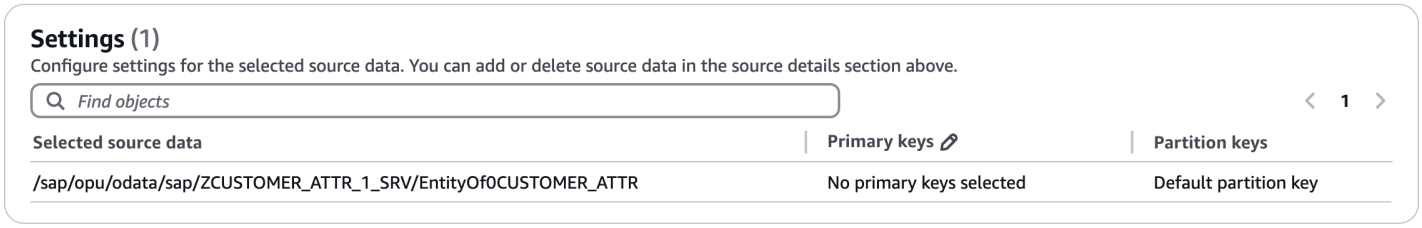
Prise en charge pour les entités SAP spéciales

AWS Glue Zero-ETL prend en charge OData les entités SAP qui utilisent le framework de provisionnement des données opérationnelles (ODP) de SAP ainsi que celles qui n'utilisent pas le framework ODP (entités non ODP). La liste des entités prises en charge inclut : ODP_SAP (Business Warehouse ou extracteurs BW), ODP_CDS (Core Data Services ou CDS Views) et les services non basés sur ODP pour SAP. OData APIs AWS Glue Zero-ETL prend en charge la capture instantanée complète et la capture incrémentielle des données de modification pour les entités SAP ODP et non ODP. Pour les entités ODP, les modifications incrémentielles sont capturées à l'aide de liens delta. Pour les entités non ODP, si un champ interrogeable pouvant être utilisé pour une ingestion basée sur un horodatage est sélectionné, Zero-ETL utilisera ce champ pour une ingestion incrémentielle.

Lors de l'ingestion de données provenant d'entités SAP à l'aide de AWS Glue Zero-ETL, il convient de noter les points suivants :

- Zero-ETL ne peut ingérer que les entités SAP configurées pour la méthode GET_ENTITYSET dans SAP.
- Pour les entités SAP non ODP, si aucun champ d'horodatage n'est sélectionné pour les mises à jour incrémentielles, AWS Glue Zero-ETL prend en charge l'extraction et la réplication complètes des données avec des upserts uniquement (aucune suppression).
- Pour les entités d'extraction ODP, nous déterminons les ensembles de clés primaires valides lors du traitement des données. D'autres entités SAP nécessitent une étape supplémentaire consistant à fournir le jeu de clés primaires valide en entrée, en particulier les entités SAP qui commencent

par`EntityOf`. Lorsqu'une entité `EntityOf` est sélectionnée, vous êtes invité à fournir l'ensemble de clés primaires.



Configuration d'une source Amazon DynamoDB

Pour accéder aux données de votre table Amazon DynamoDB source AWS Glue , vous devez avoir accès à la description de la table et à l'exportation des données depuis celle-ci. Amazon DynamoDB a récemment introduit une fonctionnalité qui permet de configurer une politique d'accès basé sur les ressources (RBAC).

L'exemple de politique d'accès basé sur les ressources (RBAC) suivant utilise un caractère générique (*) pour l'intégration :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "1111",
      "Effect": "Allow",
      "Principal": {
        "Service": "glue.amazonaws.com"
      },
      "Resource": "*",
      "Action": [
        "dynamodb:ExportTableToPointInTime",
        "dynamodb:DescribeTable",
        "dynamodb:DescribeExport"
      ],
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "111122223333"
```

```
        },
        "ArnLike": {
          "aws:SourceArn": "arn:aws:glue:us-east-1:111122223333:integration:*"
        }
      }
    }
  ]
}
```

1. Pour la DynamoDB que vous souhaitez répliquer, collez le modèle de politique RBAC ci-dessus dans Politique basée sur les ressources pour la table et renseignez les champs.
2. Si vous souhaitez rendre la politique restrictive, vous devez la mettre à jour après avoir créé l'intégration, spécifier l'`integrationArn` complet et utiliser la condition `StringEquals` à la place de `StringLike`.
3. Assurez-vous que la Point-in-time restauration (PITR) est activée pour la table DynamoDB.
4. Assurez-vous d'ajouter `Describe Export` à la politique d'accès basé sur les ressources (RBAC).

Vous pouvez également ajouter la politique RBAC à la table à l'aide de la commande suivante :

```
aws dynamodb put-resource-policy \
--resource-arn arn:aws:dynamodb:<region>:<account-id>:table/<ddb-table-name> \
--policy file://resource-policy-with-condition.json \
--region <region>
```

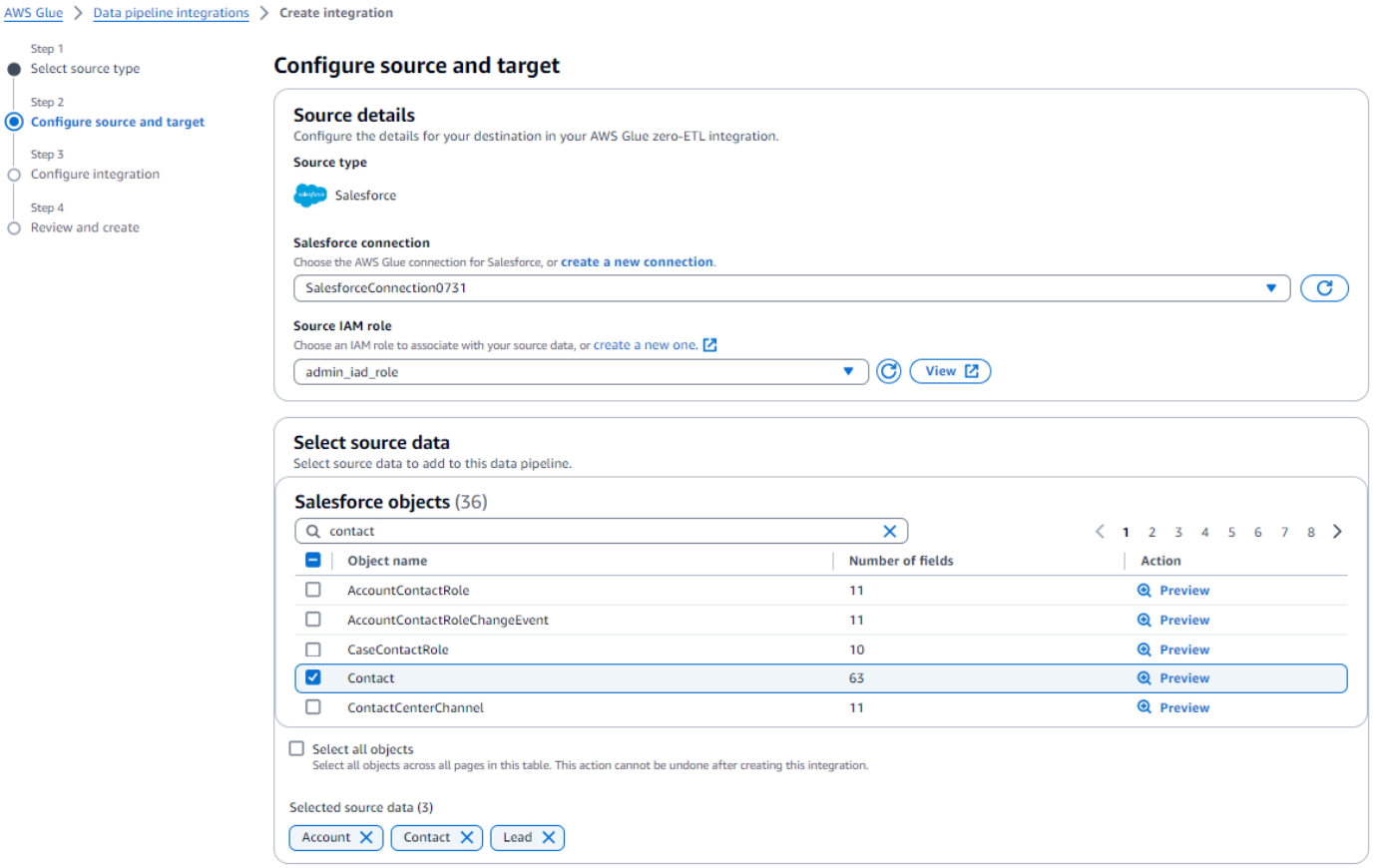
Pour vérifier que la politique est correctement appliquée, utilisez la commande suivante pour obtenir la politique de ressources d'une table :

```
aws dynamodb get-resource-policy \
--resource-arn arn:aws:dynamodb:<region>:<account-id>:table/<ddb-table-name>
--region <region>
```

Configuration d'une source Salesforce

Pour créer une connexion pour une source Salesforce, consultez Connexion à Salesforce.

Après avoir créé la connexion, vous pouvez spécifier les données source à répliquer.



À l'aide de votre intégration zéro ETL, vous pouvez effectuer des opérations DDL pour les entités prises en charge. Pour obtenir la liste des entités non prises en charge, consultez Entités et champs non pris en charge pour Salesforce.

Configuration d'une source Salesforce Marketing Cloud Account Engagement

Pour créer une connexion pour une source Salesforce Marketing Cloud Account Engagement, consultez Connexion à Salesforce Marketing Cloud Account Engagement.

À l'aide de votre intégration zéro ETL, vous pouvez effectuer des opérations DDL pour les entités prises en charge suivantes :

Libellé d'entité	Nom de l'entité
Campaign	campaign
List	list

Libellé d'entité	Nom de l'entité
Contenu dynamique	dynamic-content
List Membership	list-membership
Prospect	prospect
Utilisateur	user
EmailTemplate	email-template
EngagementStudioProgram	engagement-studio-program
Landing Page	landing-page
List Email	list-email

Configuration Salesforce supplémentaire

Salesforce zéro ETL a besoin de l'autorisation Lake Formation sur la base de données Glue, sinon il recevra `IngestionFailed` du journal avec l'erreur suivante :

```
"errorMessage": "Insufficient lake formation permissions on Target Glue database."
```

Configuration d'une OData source SAP

Pour créer une connexion pour une OData source SAP, consultezConnexion à SAP OData.

Les intégrations sans ETL avec une OData source SAP prennent désormais en charge les entités commençant par. `EntityOf` La possibilité de remplacer la clé primaire n'est actuellement prise en charge que pour les SAPOData EntityOf objets. Une fois cette propriété définie, elle ne peut pas être modifiée.

Configuration d'une ServiceNow source

Pour créer une connexion pour une ServiceNow source, consultezConnexion à ServiceNow.

Configuration d'une source Zendesk

Pour créer une connexion pour une source Zendesk, consultez Connexion à Zendesk.

À l'aide de votre intégration zéro ETL, vous pouvez effectuer les opérations DDL suivantes pour les entités prises en charge :

Libellé d'entité	Nom de l'entité	Création prise en charge	Mise à jour prise en charge	Suppression prise en charge
Tickets	billets	Y	Y	Y
Utilisateur	users	Y	Y	Y
Satisfaction Rating	satisfaction-rating	Y	Y	N
Articles	articles	Y	Y	N
Organisation	organisations	Y	Y	Y
Appels	calls	Y	Y	N
Call Legs	legs	Y	Y	N

Configuration d'une source Zoho CRM

Pour créer une connexion pour une source Zoho CRM, consultez Connexion à Zoho CRM.

À l'aide de votre intégration zéro ETL, vous pouvez effectuer les opérations DDL suivantes pour les entités prises en charge :

Libellé d'entité	Nom de l'entité	DML-Insert pris en charge	DML-Modify pris en charge	DML-Delete pris en charge	DDL-Insert pris en charge	DDL-Modify pris en charge	DDL-Delete pris en charge
Leads	lead	Y	Y	Y	Y	Y	Y

Libellé d’entité	Nom de l'entité	DML-Insert pris en charge	DML-Modify pris en charge	DML-Delete pris en charge	DDL-Insert pris en charge	DDL-Modify pris en charge	DDL-Delete pris en charge
Comptes	compte	Y	Y	Y	Y	Y	Y
Contacts	contact	Y	Y	Y	Y	Y	Y
Campagnes	campaign	Y	Y	Y	Y	Y	Y
Tâches	tâche	Y	Y	Y	Y	Y	Y
Événements	event	Y	Y	Y	Y	Y	Y
Appels	appel	Y	Y	Y	Y	Y	Y
Solutions	solution	Y	Y	Y	Y	Y	Y
Produits	product	Y	Y	Y	Y	Y	Y
Vendors	vendor	Y	Y	Y	Y	Y	Y
Quotes	quote	Y	Y	Y	Y	Y	Y
Sales Orders	sales-order	Y	Y	Y	Y	Y	Y
Purchase Orders	purchase-order	Y	Y	Y	Y	Y	Y
Factures	facture	Y	Y	Y	Y	Y	Y
Cas	cas	Y	Y	Y	Y	Y	Y
Price Books	price-book	Y	Y	Y	Y	Y	Y

Configuration d'une source Facebook Ads

Pour créer une connexion pour une source Facebook Ads, consultez Connexion à Facebook Ads.

À l'aide de votre intégration zéro ETL, vous pouvez effectuer les opérations DDL suivantes pour les entités prises en charge :

Libellé d'entité	Nom de l'entité	Création prise en charge	Mise à jour prise en charge	Suppression prise en charge
Adset	*/adsets	Y	Y	Y
Campaign	*/campaigns	Y	Y	Y
Ads	*/ads	Y	Y	Y

Configuration d'une source Instagram Ads

Pour créer une connexion pour une source Instagram Ads, consultez Connexion à Instagram Ads.

À l'aide de votre intégration zéro ETL, vous pouvez effectuer les opérations DDL suivantes pour les entités prises en charge :

Nom de l'entité	Création prise en charge	Mise à jour prise en charge	Suppression prise en charge
*/adsets	Y	Y	Y
*/campaigns	Y	Y	Y
*/ads	Y	Y	Y

Entités et champs non pris en charge pour Salesforce

Les entités ou champs Salesforce suivants ne sont pas pris en charge pour une utilisation dans le cadre d'une intégration zéro ETL avec une source Salesforce.

```
AccountChangeEvent, AccountContactRoleChangeEvent, AccountHistory, AccountShare,
 ActiveFeatureLicenseMetric, ActivePermSetLicenseMetric, ActiveProfileMetric,
 ActivityFieldHistory, amzsec__asi_Telemetry_Data_Store__ChangeEvent,
 amzsec__asi_Telemetry_Data_Store__History, amzsec__asi_Telemetry_Data_Store__Share,
 amzsec__asi_Telemetry_Job_Log__ChangeEvent, amzsec__asi_Telemetry_Job_Log__History,
 amzsec__asi_Telemetry_Job_Log__Share, amzsec__asi_Telemetry_Requirement__ChangeEvent,
 amzsec__asi_Telemetry_Requirement__History, amzsec__asi_Telemetry_Requirement__Share,
 ApexClass, ApexComponent, ApexLog, ApexPage, ApexTestQueueItem, ApexTestResult,
 ApexTrigger, AssetChangeEvent, AssetHistory, AssetRelationshipHistory,
 AssetShare, AssignmentRule, AssociatedLocationHistory, AsyncApexJob,
 AuditTrailFileExportShare, AuthorizationFormConsentChangeEvent,
 AuthorizationFormConsentHistory, AuthorizationFormConsentShare,
 AuthorizationFormDataUseHistory, AuthorizationFormDataUseShare,
 AuthorizationFormHistory, AuthorizationFormShare, AuthorizationFormTextHistory,
 AuthProvider, AuthSession, BatchJobHistory, BatchJobPartFailedRecordHistory,
 BatchJobPartHistory, BatchJobShare, BrandTemplate, BriefcaseAssignmentChangeEvent,
 BriefcaseDefinitionChangeEvent, BusinessBrandShare, BusinessHours,
 BusinessProcess, CalcMatrixColumnRangeHistory, CalcProcStepRelationshipHistory,
 CalculationMatrixColumnHistory, CalculationMatrixHistory, CalculationMatrixRowHistory,
 CalculationMatrixShare, CalculationMatrixVersionHistory, CalculationProcedureHistory,
 CalculationProcedureShare, CalculationProcedureStepHistory,
 CalculationProcedureVariableHistory, CalculationProcedureVersionHistory, Calendar,
 CalendarViewShare, CallCenter, CallCoachConfigModifyEvent, CampaignChangeEvent,
 CampaignHistory, CampaignMemberChangeEvent, CampaignMemberStatusChangeEvent,
 CampaignShare, CaseChangeEvent, CaseHistory, CaseHistory2 CaseHistory2ChangeEvent,
 CaseRelatedIssueChangeEvent, CaseRelatedIssueHistory, CaseShare, CaseStatus,
 CaseTeamMember, CaseTeamRole, CaseTeamTemplate, CaseTeamTemplateMember,
 CaseTeamTemplateRecord, CategoryNode, ChangeRequestChangeEvent, ChangeRequestHistory,
 ChangeRequestRelatedIssueChangeEvent, ChangeRequestRelatedIssueHistory,
 ChangeRequestRelatedItemChangeEvent, ChangeRequestRelatedItemHistory,
 ChangeRequestShare, ChatRetirementRdyMetrics, ChatterActivity, ClientBrowser,
 CollaborationGroup, CollaborationGroupMember, CollaborationGroupMemberRequest,
 CollaborationInvitation, CommSubscriptionChannelTypeHistory,
 CommSubscriptionChannelTypeShare, CommSubscriptionConsentChangeEvent,
 CommSubscriptionConsentHistory, CommSubscriptionConsentShare, CommSubscriptionHistory,
 CommSubscriptionShare, CommSubscriptionTimingHistory, Community, ConnectedApplication,
 ContactChangeEvent, ContactHistory, ContactPointAddressChangeEvent,
 ContactPointAddressHistory, ContactPointAddressShare, ContactPointConsentChangeEvent,
 ContactPointConsentHistory, ContactPointConsentShare, ContactPointEmailChangeEvent,
 ContactPointEmailHistory, ContactPointEmailShare, ContactPointPhoneChangeEvent,
 ContactPointPhoneHistory, ContactPointPhoneShare, ContactPointTypeConsentChangeEvent,
 ContactPointTypeConsentHistory, ContactPointTypeConsentShare, ContactRequestShare,
 ContactShare, ContentDocumentChangeEvent, ContentDocumentHistory,
```

```
ContentDocumentLink, ContentDocumentLinkChangeEvent, ContentDocumentSubscription,
ContentFolderItem, ContentFolderLink, ContentFolderMember, ContentNote,
ContentNotification, ContentTagSubscription, ContentUserSubscription,
ContentVersionChangeEvent, ContentVersionComment, ContentVersionHistory,
ContentVersionRating, ContentWorkspace, ContentWorkspaceMember,
ContentWorkspacePermission, ContentWorkspaceSubscription, ContractChangeEvent,
ContractHistory, ContractLineItemChangeEvent, ContractLineItemHistory,
ContractStatus, Conversation, ConversationParticipant, CronJobDetail,
CronTrigger, CustomBrand, CustomBrandAsset, CustomerShare, CustomHttpHeader,
DashboardComponent, DataUseLegalBasisHistory, DataUseLegalBasisShare,
DataUsePurposeHistory, DataUsePurposeShare, DecisionTableRecordset, DeleteEvent,
DocumentAttachmentMap, Domain, DomainSite, DTRecordsetReplicaShare, EmailBounceEvent,
EmailMessageChangeEvent, EmailServicesAddress, EmailServicesFunction,
EmailTemplate, EmailTemplateChangeEvent, EngagementAttendeeChangeEvent,
EngagementAttendeeHistory, EngagementChannelTypeHistory, EngagementChannelTypeShare,
EngagementInteractionChangeEvent, EngagementInteractionHistory,
EngagementInteractionShare, EngagementInterface, EngagementTopicChangeEvent,
EngagementTopicHistory, EntitlementChangeEvent, EntitlementHistory,
EntitlementTemplate, EntityMilestoneHistory, EntitySubscription, EventChangeEvent,
EventRelationChangeEvent, EventRelayConfigChangeEvent, ExpressionSetHistory,
ExpressionSetShare, ExpressionSetVersionHistory, ExternalEventMappingShare,
FeedAttachment, FeedLike, FeedPollChoice, FeedPollVote, FeedSignal,
FieldPermissions, FieldSecurityClassification, FiscalYearSettings,
FlowInterviewLogShare, FlowInterviewShare, FlowOrchestrationEvent,
FlowOrchestrationInstanceShare, FlowOrchestrationStageInstanceShare,
FlowOrchestrationStepInstanceShare, FlowOrchestrationWorkItemShare,
FlowRecordShare, FlowRecordVersionChangeEvent, FlowTestResultShare, Folder,
Group, GroupMember, Holiday, IdeaComment, IdpEventLog, ImageHistory, ImageShare,
IncidentChangeEvent, IncidentHistory, IncidentRelatedItemChangeEvent,
IncidentRelatedItemHistory, IncidentShare, IndividualChangeEvent,
IndividualHistory, IndividualShare, KnowledgeableUser, LeadChangeEvent, LeadHistory,
LeadShare, LeadStatus, LightningExitByPageMetrics, LightningToggleMetrics,
LightningUsageByAppTypeMetrics, LightningUsageByBrowserMetrics,
LightningUsageByFlexiPageMetrics, LightningUsageByPageMetrics, ListEmailChangeEvent,
ListEmailShare, ListView, LocationChangeEvent, LocationHistory, LocationShare,
LocationTrustMeasureShare, LoginHistory, LoginIp, MacroChangeEvent,
MacroHistory, MacroInstructionChangeEvent, MacroShare, MacroUsageShare,
ManagedContentVariantChangeEvent, MessagingEndUserHistory, MessagingEndUserShare,
MessagingSessionHistory, MessagingSessionShare, MilestoneType, MLEngagementEvent,
ObjectPermissions, OpportunityChangeEvent, OpportunityContactRoleChangeEvent,
OpportunityFieldHistory, OpportunityLineItemChangeEvent, OpportunityShare,
OpportunityStage, OrderChangeEvent, OrderHistory, OrderItemChangeEvent,
OrderItemHistory, OrderShare, OrderStatus, Organization, OrgEmailAddressSecurity,
OrgWideEmailAddress, OutgoingEmail, OutgoingEmailRelation, PackageLicense,
```

```
PartnerRole, PartyConsentChangeEvent, PartyConsentHistory, PartyConsentShare,
Period, PermissionSet, PermissionSetAssignment, PermissionSetTabSetting,
Pricebook2ChangeEvent, Pricebook2History, PricebookEntryChangeEvent,
PricebookEntryHistory, PrivacyJobSessionShare, PrivacyObjectSessionShare,
PrivacyRTBFRequestHistory, PrivacyRTBFRequestShare, PrivacySessionRecordFailureShare,
ProblemChangeEvent, ProblemHistory, ProblemIncidentChangeEvent,
ProblemIncidentHistory, ProblemRelatedItemChangeEvent, ProblemRelatedItemHistory,
ProblemShare, ProcessDefinition, ProcessExceptionEvent, ProcessExceptionShare,
ProcessInstanceChangeEvent, ProcessInstanceStep, ProcessInstanceStepChangeEvent,
ProcessNode, Product2ChangeEvent, Product2History, ProductEntitlementTemplate,
Profile, ProfileSkillEndorsementHistory, ProfileSkillHistory,
ProfileSkillShare, ProfileSkillUserHistory, PromptActionShare, PromptErrorShare,
QueueSobject, QuickTextChangeEvent, QuickTextHistory, QuickTextShare,
QuickTextUsageShare, RecentlyViewed, RecommendationChangeEvent,
RecordActionHistory, RecordAlertHistory, RecordAlertShare, RecordType,
ScorecardShare, SellerHistory, SellerShare, ServiceContractChangeEvent,
ServiceContractHistory, ServiceContractShare, SetupAuditTrail, SetupEntityAccess,
SharingRecordCollectionShare, Site, SiteHistory, SiteRedirectMapping,
SocialPersonaHistory, SocialPostChangeEvent, SocialPostHistory, SocialPostShare,
SolutionHistory, SolutionStatus, StaticResource, StreamingChannelShare,
TableauHostMappingShare, TaskChangeEvent, TaskPriority, TaskStatus,
ThreatDetectionFeedback, TimelineObjectDefinitionChangeEvent, TodayGoalShare,
Topic, TopicUserEvent, Translation, User, UserAppMenuCustomizationShare,
UserChangeEvent, UserDefinedLabelAssignmentShare, UserDefinedLabelShare,
UserEmailPreferredPersonShare, UserLicense, UserLogin, UserPackageLicense,
UserPreference, UserPrioritizedRecordShare, UserProvisioningRequestShare,
UserRole, UserShare, VideoCallChangeEvent, VideoCallParticipantChangeEvent,
VideoCallRecordingChangeEvent, VideoCallShare, VisualforceAccessMetrics,
VoiceCallChangeEvent, VoiceCallRecordingChangeEvent, VoiceCallShare, Vote,
WebLink, WorkAccessShare, WorkBadgeDefinitionHistory, WorkBadgeDefinitionShare,
WorkOrderChangeEvent, WorkOrderHistory, WorkOrderLineItemChangeEvent,
WorkOrderLineItemHistory, WorkOrderLineItemStatus, WorkOrderShare, WorkOrderStatus,
WorkPlanChangeEvent, WorkPlanHistory, WorkPlanShare, WorkPlanTemplateChangeEvent,
WorkPlanTemplateEntryChangeEvent, WorkPlanTemplateEntryHistory,
WorkPlanTemplateHistory, WorkPlanTemplateShare, WorkStepChangeEvent, WorkStepHistory,
WorkStepStatus, WorkStepTemplateChangeEvent, WorkStepTemplateHistory,
WorkStepTemplateShare, WorkThanksShare
```

Types d'entités non pris en charge pour SAP

Les types d'entités SAP suivants ne sont pas pris en charge pour une utilisation dans le cadre d'une intégration zéro ETL avec une source SAP.

```
ODP_HANA (HANA calculation views), ODP_SLT (SLT Replication Server)
```

Entités et champs non pris en charge pour ServiceNow

Les ServiceNow entités ou champs suivants ne sont pas pris en charge pour une utilisation dans le cadre d'une intégration zéro ETL avec une ServiceNow source.

```
ais_acl_overrides, ais_async_genius_result, ais_async_request, ais_connection,
 ais_genius_result_configuration_parameters, ais_partition_health,
 ais_partition_health_response, ais_publish_history, ais_relevancy_training_execution,
 ais_relevancy_training_staging, ais_search_profile_relevancy_model,
 catalog_draft_entities, clone_log, clone_log0000, clone_log0001, clone_log0002,
 clone_log0003, clone_log0004, clone_log0005, clone_log0006, clone_log0007,
 cmdb_ie_context, cmdb_ie_log, cmdb_ie_run, cmdb_ire_partial_payloads_index,
 cmdb_qb_result_base$par1, cmdb$par1, discovery_log, discovery_log0000,
 discovery_log0001, discovery_log0002, discovery_log0003, discovery_log0004,
 discovery_log0005, discovery_log0006, discovery_log0007, ecc_agent_log,
 entitlement_data, entl_subscription_map, gs_entitlement_plugin_mapping,
 ih_transaction_exclusion, import_log, import_log0000, import_log0001, import_log0002,
 import_log0003, import_log0004, import_log0005, import_log0006, import_log0007,
 jrobin_archive, jrobin_database, jrobin_datasource, jrobin_definition, jrobin_graph,
 jrobin_graph_line, jrobin_graph_set, jrobin_graph_set_member, jrobin_shard,
 jrobin_shard_location, license_role_discovery_run, logger_configuration_validation,
 m2m_analytics_event_logger, m2m_user_consent_info, ml_artifact_object_store,
 np$sys_gen_ai_filter_sample, np$sys_ui_element, np$sys_ui_list_element,
 one_api_service_plan_feature_invocation, one_api_service_plan_invocation,
 open_nlu_predict_log, open_nlu_predict_log0000, open_nlu_predict_log0001,
 open_nlu_predict_log0002, open_nlu_predict_log0003, open_nlu_predict_log0004,
 open_nlu_predict_log0005, open_nlu_predict_log0006, open_nlu_predict_log0007,
 pa_diagnostic_log, pa_diagnostic_log0000, pa_diagnostic_log0001,
 pa_diagnostic_log0002, pa_diagnostic_log0003, pa_diagnostic_log0004,
 pa_diagnostic_log0005, pa_diagnostic_log0006, pa_diagnostic_log0007, pa_favorites,
 pa_job_log_rows, pa_job_log_rows0000, pa_job_log_rows0001, pa_job_log_rows0002,
 pa_job_log_rows0003, pa_job_log_rows0004, pa_job_log_rows0005, pa_job_log_rows0006,
 pa_job_log_rows0007, pa_migration_ignored_scores, pa_scores_l1, pa_scores_l2,
 pa_scores_migration_groups, par_dashboard_conversion_backup, promin_log,
 promin_log0000, promin_log0001, promin_log0002, promin_log0003, promin_log0004,
 promin_log0005, promin_log0006, promin_log0007, promin_request_object,
 proposed_change_verification_log, proposed_change_verification_log0000,
 proposed_change_verification_log0001, proposed_change_verification_log0002,
 proposed_change_verification_log0003, proposed_change_verification_log0004,
 proposed_change_verification_log0005, proposed_change_verification_log0006,
```

```
proposed_change_verification_log0007, protected_table_log, protected_table_log0000,
protected_table_log0001, protected_table_log0002, protected_table_log0003,
protected_table_log0004, protected_table_log0005, protected_table_log0006,
protected_table_log0007, pwd_history, qb_query_results, scan_log,
scan_log0000, scan_log0001, scan_log0002, scan_log0003, scan_log0004,
scan_log0005, scan_log0006, scan_log0007, schema_validator_error,
sla_repair_log_entry, sla_repair_log_entry0000, sla_repair_log_entry0001,
sla_repair_log_entry0002, sla_repair_log_entry0003, sla_repair_log_entry0004,
sla_repair_log_entry0005, sla_repair_log_entry0006, sla_repair_log_entry0007,
sla_repair_log_message, sla_repair_log_message0000, sla_repair_log_message0001,
sla_repair_log_message0002, sla_repair_log_message0003, sla_repair_log_message0004,
sla_repair_log_message0005, sla_repair_log_message0006, sla_repair_log_message0007,
sn_bm_client_activity, sn_ci_analytics_st_actionable_notifs,
sn_ci_analytics_st_conv_completion_by_cat, sn_ci_analytics_st_conv_dynamic_property,
sn_ci_analytics_st_conversation, sn_ci_analytics_st_count_by_date,
sn_ci_analytics_st_event_occurrence, sn_ci_analytics_st_event_property_value_trend,
sn_ci_analytics_st_issue_auto_resolution, sn_ci_analytics_st_no_clicks,
sn_ci_analytics_st_no_results, sn_ci_analytics_st_property_summary_by_event,
sn_ci_analytics_st_session_count_per_locale, sn_ci_analytics_st_session_duration,
sn_ci_analytics_st_spokes_usage, sn_ci_analytics_st_topic_execution_stats,
sn_ci_analytics_st_topic_occurrence, sn_ci_analytics_st_trending_content,
sn_ci_analytics_st_trending_queries, sn_ci_analytics_st_users,
sn_cs_plugin_signatures, sn_cs_telemetry_log, sn_dfc_application, sn_dfc_product,
sn_employee_position, sn_entitlement_st_subscription_application_family,
sn_entitlement_st_subscription_application_users, sn_hr_sp_st_relevant_for_you,
sn_instance_clone_log, sn_instance_clone_log0000, sn_instance_clone_log0001,
sn_instance_clone_log0002, sn_instance_clone_log0003, sn_instance_clone_log0004,
sn_instance_clone_log0005, sn_instance_clone_log0006, sn_instance_clone_log0007,
sn_km_mr_st_kb_knowledge, sn_me_st_topic, sn_rf_conditional_definition,
sn_rf_evaluation_type, sn_rf_evaluation_type_input, sn_rf_recommendation_action,
sn_rf_recommendation_experience, sn_rf_recommendation_history,
sn_rf_recommendation_rule, sn_rf_record_display_configuration, sn_rf_trend_definition,
sn_sub_man_st_account_level_entitlement, sn_sub_man_st_gen_ai_metadata,
sn_sub_man_st_instance_used_assist_count, sn_sub_man_st_now_assist_creator_instances,
sn_sub_man_st_now_assists_aggregate, sn_sub_man_st_subscribed_groups,
sn_sub_man_st_subscription_insights, sn_sub_man_st_subscription_license_detail_metric,
sn_sub_man_st_unallocated_group_recommendation, sn_sub_man_st_unconfirmed_user_group,
sn_wn_user_app_activity, sn_wn_user_content_activity, snc_monitorable_item,
snpar_sched_export_v_scheduled_export_visualization, spotlight,
spotlight_audit, spotlight_copy_log_row, spotlight_copy_log_row0000,
spotlight_copy_log_row0001, spotlight_copy_log_row0002, spotlight_copy_log_row0003,
spotlight_copy_log_row0004, spotlight_copy_log_row0005, spotlight_copy_log_row0006,
spotlight_copy_log_row0007, spotlight_job_log_row, spotlight_job_log_row0000,
spotlight_job_log_row0001, spotlight_job_log_row0002, spotlight_job_log_row0003,
```

```
spotlight_job_log_row0004, spotlight_job_log_row0005, spotlight_job_log_row0006,
spotlight_job_log_row0007, st_dfc_performance_metric, st_license_detail_metric,
st_on_call_hour, st_sc_wizard_question, st_sys_catalog_items_and_variable_sets,
st_sys_design_system_icon, subscription_instance_stats, svc_container_config,
svc_environment_config, svc_layer_config, svc_model_assoc_ci,
svc_model_checkpoint_attr, svc_model_obj_cluster, svc_model_obj_constraint,
svc_model_obj_deployable, svc_model_obj_element, svc_model_obj_impact,
svc_model_obj_impactrule, svc_model_obj_package, svc_model_obj_path,
svc_model_obj_relation, svc_model_obj_service, sys_administrative_script_transaction,
sys_amb_message, sys_amb_processor, sys_analytics_batch_state,
sys_analytics_config, sys_analytics_data_points_error, sys_analytics_event,
sys_analytics_logger, sys_analytics_logger_field, sys_app_payload_loader_rule,
sys_app_payload_unloader_rule, sys_app_scan_payload, sys_app_scan_variable,
sys_app_scan_variable_type, sys_archive_destroy_log, sys_archive_destroy_run,
sys_archive_log, sys_archive_run, sys_atf_transaction_log, sys_attachment_doc,
sys_attachment_doc_v2, sys_attachment_soft_deleted, sys_audit, sys_audit_relation,
sys_auth_policy_api_allowed, sys_aw_registered_scripting_modal,
sys_cache_flush, sys_data_egress_source, sys_dm_delete_count,
sys_export_set_log, sys_export_set_log0000, sys_export_set_log0001,
sys_export_set_log0002, sys_export_set_log0003, sys_export_set_log0004,
sys_export_set_log0005, sys_export_set_log0006, sys_export_set_log0007,
sys_flow_compiled_flow, sys_flow_compiled_flow_chunk, sys_flow_context_chunk,
sys_flow_context_chunk_archive, sys_flow_context_inputs_chunk,
sys_flow_execution_history, sys_flow_log, sys_flow_log0000, sys_flow_log0001,
sys_flow_log0002, sys_flow_log0003, sys_flow_plan_context_binding,
sys_flow_report_doc, sys_flow_report_doc_chunk, sys_flow_report_doc_chunk_archive,
sys_flow_runtime_state_chunk, sys_flow_runtime_value_chunk,
sys_flow_subflow_plan_chunk, sys_flow_trigger_plan_chunk, sys_flow_val_listener,
sys_flow_value, sys_flow_value_chunk, sys_gen_ai_config_example,
sys_gen_ai_feature_mapping, sys_gen_ai_strategy_mapping, sys_gen_ai_usage_log,
sys_generative_ai_capability_definition, sys_generative_ai_log,
sys_generative_ai_response_validator, sys_generative_ai_validator,
sys_geo_routing, sys_geo_routing_config, sys_hop_token, sys_hub_action_plan_chunk,
sys_hub_snapshot_chunk, sys_journal_field, sys_journal_field_edit, sys_json_chunk,
sys_kaa_policy, sys_kaa_subidentity_assertion, sys_kaa_user_policy_mapping,
sys_mapplication, sys_mass_encryption_job, sys_notification_execution_log,
sys_notification_execution_log0000, sys_notification_execution_log0001,
sys_notification_execution_log0002, sys_notification_execution_log0003,
sys_notification_execution_log0004, sys_notification_execution_log0005,
sys_notification_execution_log0006, sys_notification_execution_log0007,
sys_nowmq_message, sys_nowmq_provider_param_definition, sys_orchestrator_action,
sys_pd_asset_configuration, sys_pd_context_chunk, sys_pd_context_log,
sys_pd_snapshot_chunk, sys_pd_trigger_license, sys_processing_framework_job,
sys_query_index_hint, sys_query_rewrite, sys_query_string_log,
```

```
sys_replication_queue, sys_replication_queue0, sys_replication_queue1,
sys_replication_queue2, sys_replication_queue3, sys_replication_queue4,
sys_replication_queue5, sys_replication_queue6, sys_replication_queue7,
sys_request_performance, sys_rollback_blacklisted, sys_rollback_conflict,
sys_rollback_incremental, sys_rollback_log, sys_rollback_log0000,
sys_rollback_log0001, sys_rollback_log0002, sys_rollback_log0003,
sys_rollback_log0004, sys_rollback_log0005, sys_rollback_log0006,
sys_rollback_log0007, sys_rollback_run, sys_rollback_schema_change,
sys_rollback_schema_conflict, sys_rollback_sequence, sys_scheduler_assignment,
sys_scheduler_memory_pressure_job_log, sys_script_adapter_rule,
sys_script_batch_adapter_rule, sys_search_source_filter, sys_service_authentication,
sys_signing_job, sys_suggestion_reader, sys_sync_history_review, sys_trend,
sys_unreferenced_preview, sys_unreferenced_record_rule, sys_upgrade_manifest,
sys_upgrade_state, sys_ux_asset_cache_buster, sys_ux_lib_component_prop,
sys_ux_lib_presource, sys_ux_page_action, sys_ux_page_action_binding,
sysevent_queue_runtime, syslog, syslog_app_scope0000, syslog_app_scope0001,
syslog_app_scope0002, syslog_app_scope0003, syslog_app_scope0004,
syslog_app_scope0005, syslog_app_scope0006, syslog_app_scope0007, syslog_email0000,
syslog_email0001, syslog_email0002, syslog_email0003, syslog_email0004,
syslog_email0005, syslog_email0006, syslog_email0007, syslog_transaction,
syslog_transaction0000, syslog_transaction0001, syslog_transaction0002,
syslog_transaction0003, syslog_transaction0004, syslog_transaction0005,
syslog_transaction0006, syslog_transaction0007, syslog0000, syslog0001, syslog0002,
syslog0003, syslog0004, syslog0005, syslog0006, syslog0007, ts_attachment,
ts_c_1_0, ts_c_1_1, ts_c_1_2, ts_c_1_3, ts_c_1_4, ts_c_1_5, ts_c_1_6, ts_c_1_7,
ts_c_1_8, ts_c_1_9, ts_c_10_0, ts_c_10_1, ts_c_10_2, ts_c_10_3, ts_c_10_4,
ts_c_10_5, ts_c_10_6, ts_c_10_7, ts_c_10_8, ts_c_10_9, ts_c_11_0, ts_c_11_1,
ts_c_11_2, ts_c_11_3, ts_c_11_4, ts_c_11_5, ts_c_11_6, ts_c_11_7, ts_c_11_8,
ts_c_11_9, ts_c_2_0, ts_c_2_1, ts_c_2_2, ts_c_2_3, ts_c_2_4, ts_c_2_5, ts_c_2_6,
ts_c_2_7, ts_c_2_8, ts_c_2_9, ts_c_3_0, ts_c_3_1, ts_c_3_2, ts_c_3_3, ts_c_3_4,
ts_c_3_5, ts_c_3_6, ts_c_3_7, ts_c_3_8, ts_c_3_9, ts_c_39_0, ts_c_39_1, ts_c_39_2,
ts_c_39_3, ts_c_39_4, ts_c_39_5, ts_c_39_6, ts_c_39_7, ts_c_39_8, ts_c_39_9,
ts_c_40_0, ts_c_40_1, ts_c_40_2, ts_c_40_3, ts_c_40_4, ts_c_40_5, ts_c_40_6,
ts_c_40_7, ts_c_40_8, ts_c_40_9, ts_c_41_0, ts_c_41_1, ts_c_41_2, ts_c_41_3,
ts_c_41_4, ts_c_41_5, ts_c_41_6, ts_c_41_7, ts_c_41_8, ts_c_41_9, ts_c_42_0,
ts_c_42_1, ts_c_42_2, ts_c_42_3, ts_c_42_4, ts_c_42_5, ts_c_42_6, ts_c_42_7,
ts_c_42_8, ts_c_42_9, ts_c_43_0, ts_c_43_1, ts_c_43_2, ts_c_43_3, ts_c_43_4,
ts_c_43_5, ts_c_43_6, ts_c_43_7, ts_c_43_8, ts_c_43_9, ts_c_44_0, ts_c_44_1,
ts_c_44_2, ts_c_44_3, ts_c_44_4, ts_c_44_5, ts_c_44_6, ts_c_44_7, ts_c_44_8,
ts_c_44_9, ts_c_45_0, ts_c_45_1, ts_c_45_2, ts_c_45_3, ts_c_45_4, ts_c_45_5,
ts_c_45_6, ts_c_45_7, ts_c_45_8, ts_c_45_9, ts_c_6_0, ts_c_6_1, ts_c_6_2,
ts_c_6_3, ts_c_6_4, ts_c_6_5, ts_c_6_6, ts_c_6_7, ts_c_6_8, ts_c_6_9, ts_c_7_0,
ts_c_7_1, ts_c_7_2, ts_c_7_3, ts_c_7_4, ts_c_7_5, ts_c_7_6, ts_c_7_7, ts_c_7_8,
ts_c_7_9, ts_c_8_0, ts_c_8_1, ts_c_8_2, ts_c_8_3, ts_c_8_4, ts_c_8_5, ts_c_8_6,
```

```
ts_c_8_7, ts_c_8_8, ts_c_8_9, ts_c_attachment, ts_chain, ts_deleted_doc,
ts_document, ts_field, ts_index_stats, ts_phrase, ts_search_stats, ts_v4_attachment,
ts_word, ua_app_metadata, ua_audit_stats, ua_extra_page, ua_monitor_property,
ua_monitor_property_audit, ua_shared_service, ua_sn_table_inventory, ua_sp_known_bot,
ua_upload_log, v_ais_result_improvement_rule_condition_builder_values,
v_cluster_nodes, v_cxs_search_resource, v_db_index, v_db_trigger, v_file_load_order,
v_interaction_context, v_iostats, v_mysql_proclist, v_mysql_status, v_mysql_variables,
v_on_call_report_cache, v_pa_par_combined_dashboard, v_pd_activity_condition_to_run,
v_pd_activity_start_rule_with_condition, v_pd_lane_condition_to_run,
v_sql_debug, v_st_kb_category, v_st_kb_most_viewed, v_st_kb_recently_viewed,
v_st_km_genai_mra_similar_task, v_st_popular_item, v_st_recent_item, v_st_sc_cat_item,
v_st_sc_catalog, v_st_sc_category, validator_run_summary, vtb_card_history,
wf_log, wf_log0000, wf_log0001, wf_log0002, wf_log0003, wf_log0004, wf_log0005,
wf_log0006, wf_log0007, spotlight_criteria, pa_snapshots, pa_widget_indicators,
pa_widgets, sn_cim_register, global, gsw_change_log, gsw_content, gsw_content_group,
gsw_content_information, gsw_status_of_content, multi_factor_browser_fingerprint,
multi_factor_criteria, pa_filters, password_policy, plan_execution,
plan_mysql, plan_oracle, plan_postgres, sc_rest_api_without_access_policy,
sn_actsub_activity, sn_actsub_activity_fanout, sn_actsub_activity_stream,
sn_actsub_activity_type, sn_actsub_atype_attributes, sn_actsub_atype_notif_pref,
sn_actsub_module, sn_actsub_notif_object, sn_actsub_subobject_stream,
sn_actsub_subscribable_object, sn_actsub_subscription_notif_pref,
sn_actsub_user_stream, sn_appclient_store_outbound_http_quota,
sn_appcreator_app_template, sn_critical_update, sn_docker_spoke_images,
sn_employee_app, sn_employee_app_access, sn_employee_app_access_criteria,
sn_entitlement_genai_assist_counts, sn_entitlement_genai_creator_user_counts,
sn_entitlement_genai_creator_users, sn_mif_instance, sn_mif_sync_data,
sn_mif_sync_status, sn_mif_table_registration, sn_mif_trust_config,
sn_vsc_best_practice_configurations, sn_vsc_best_practice_goals,
sn_vsc_changed_hardening_settings, sn_vsc_changed_scan_findings,
sn_vsc_check_security_area, sn_vsc_elevation_event, sn_vsc_event,
sn_vsc_export_event, sn_vsc_export_setting, sn_vsc_harc_compliance_status_lookup,
sn_vsc_hardening_compliance_scores, sn_vsc_impersonation_event,
sn_vsc_instance_hardening_settings, sn_vsc_login_event,
sn_vsc_scan_comparisons, sn_vsc_scan_summary, sn_vsc_security_check_categories,
sn_vsc_security_check_configurations, sn_vsc_security_configuration_groups,
sn_vsc_security_privacy_capabilities, sn_vsc_updated_settings,
sn_vsc_user_comparisons, sys_app_hash_inventory, sys_coalesce_strategy_deferred,
sys_flow_secure_data, sys_formula_function, sys_geocoding_request,
sys_global_file_hash, sys_import_set_row, sys_index, sys_index_explain,
sys_installation_schedule, sys_installation_schedule_item, sys_offline_app,
sys_package, sys_package_dependency_item, sys_package_dependency_m2m,
sys_plugins, sys_querystat, sys_reap_package, sys_scoped_plugin,
sys_stage_storage_alias, sys_storage_alias, sys_storage_table_alias, sys_store_app,
```

```
sys_table_partition, sys_upgrade_history_log, sys_user_public_credential,
sys_webauthn_authentication_request, sys_webauthn_registration_request,
syslog_app_scope, syslog_email, syslog_page_timing, ua_instance_state_config,
v_expression_cache, v_private_cache, v_shared_cache, sys_amb_message0002,
sys_amb_message0004, sys_amb_message0005, sys_metadata, v_par_unified_report_viz
```

Configuration d'une cible d'intégration zéro ETL

Plusieurs options sont proposées AWS lors de la configuration d'une cible pour une intégration zéro ETL. La cible peut être un entrepôt de Amazon Redshift données crypté ou un catalogue Amazon SageMaker Lakehouse.

Avant de sélectionner la cible pour l'intégration zéro ETL, vous devez configurer l'une des ressources cible suivantes.

Les options de configuration pour une cible dans une intégration zéro ETL sont les suivantes :

- Catalogue et base de données Amazon SageMaker Lakehouse configurés avec le stockage Amazon S3 standard. Consultez Configuration d'un catalogue Amazon SageMaker Lakehouse avec un stockage S3 standard.
- Un catalogue Amazon SageMaker Lakehouse configuré avec le bucket Amazon S3 Tables. Consultez Configuration de tableaux Amazon S3 en tant que cibles.
- Un catalogue Amazon SageMaker Lakehouse configuré avec le stockage géré par Amazon Redshift. Consultez Configuration d'un catalogue Amazon SageMaker Lakehouse avec stockage géré Amazon Redshift.
- Un entrepôt de données Amazon Redshift identifié par un espace de noms Redshift. Consultez Configuration d'une cible d'entrepôt de Amazon Redshift données.

> **Note**
>
> Vous ne pouvez pas modifier la cible d'une intégration zéro ETL après sa création.

Configuration d'un catalogue Amazon SageMaker Lakehouse avec un stockage S3 standard

Cette section décrit les conditions préalables et les étapes de configuration pour configurer un compartiment Amazon S3 normal comme espace de stockage pour la cible de votre catalogue Amazon SageMaker Lakehouse dans le cadre d'une intégration zéro ETL.

Conditions préalables à la configuration d'une intégration

Avant de créer une intégration zéro ETL avec un catalogue Amazon SageMaker Lakehouse à l'aide d'un stockage S3 standard, vous devez effectuer les tâches de configuration suivantes :

1. Configuration d'une AWS Glue base de données
2. Fournir une politique RBAC du catalogue
3. Créer un rôle IAM cible

Après avoir configuré le catalogue Amazon SageMaker Lakehouse avec le stockage Amazon S3 standard, vous pouvez terminer la configuration Configuration de l'intégration avec votre cible de l'intégration.

Configuration de tableaux Amazon S3 en tant que cibles

Cette section décrit les conditions préalables et les étapes de configuration de tableaux Amazon S3 en tant que cibles pour votre intégration zéro ETL.

Conditions préalables à la configuration d'une intégration

Avant de créer une intégration zéro ETL avec des tableaux Amazon S3 comme cibles, vous devez effectuer les tâches de configuration suivantes :

1. Configuration d'un compartiment de tableaux Amazon S3
2. Fournir une politique RBAC du catalogue
3. Créer un rôle IAM cible

Configuration d'un compartiment de tableaux Amazon S3

1. Créez un compartiment de tableaux S3 dans votre compte en suivant les instructions de la section Getting started with Amazon S3 Tables.

2. Activez les intégrations Analytics avec votre bucket S3-Table en suivant ces instructions : Intégration des AWS services avec Amazon S3 Tables.

Fourniture d'une politique RBAC du catalogue

Les autorisations suivantes doivent être ajoutées à la politique RBAC du catalogue pour permettre les intégrations entre la source et la cible du catalogue des tableaux Amazon S3.

La politique de ressources du AWS Glue catalogue cible doit inclure les autorisations du service Glue pour AuthorizeInboundIntegration. En outre, une CreateInboundIntegration autorisation est requise soit sur le principal source qui crée l'intégration, soit dans la politique de AWS Glue ressource cible.

> **Note**
>
> Dans le cas d'un scénario entre comptes, la politique de ressources du catalogue principal et celle du AWS Glue catalogue cible doivent inclure des CreateInboundIntegration autorisations « glue : » sur la ressource.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Principal": {
        "AWS": [
         "arn:aws:iam::123456789012:user/Alice"
        ]
      },
      "Effect": "Allow",
      "Action": [
        "glue:CreateInboundIntegration"
      ],
      "Resource": [
       "arn:aws:glue:us-east-1:111122223333:catalog/<s3tablescatalog>/*"
      ],
      "Condition": {
        "StringLike": {
```

```
        "aws:SourceArn": "arn:aws:dynamodb:us-east-1:444455556666:table/<table-
name>"
        }
      }
    },
    {
      "Principal": {
        "Service": [
          "glue.amazonaws.com"
        ]
      },
      "Effect": "Allow",
      "Action": [
        "glue:AuthorizeInboundIntegration"
      ],
      "Resource": [
      "arn:aws:glue:us-east-1:111122223333:catalog/<s3tablescatalog>/*"
      ],
      "Condition": {
        "StringEquals": {
        "aws:SourceArn": "arn:aws:dynamodb:us-east-1:444455556666:table/<table-
name>"
        }
      }
    }
  ]
}
```

> **Note**
>
> Remplacez `<s3tablescatalog>` par le nom du catalogue de vos tableaux S3.

Création d'un rôle IAM cible

Créez un rôle IAM cible avec les autorisations et les relations d'approbation suivantes :

Exemple de politique IAM :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Action": [
        "s3tables:ListTableBuckets",
        "s3tables:GetTableBucket",
        "s3tables:GetTableBucketEncryption",
        "s3tables:GetNamespace",
        "s3tables:CreateNamespace",
        "s3tables:ListNamespaces",
        "s3tables:CreateTable",
        "s3tables:GetTable",
        "s3tables:GetTableEncryption",
        "s3tables:ListTables",
        "s3tables:GetTableMetadataLocation",
        "s3tables:UpdateTableMetadataLocation",
        "s3tables:GetTableData",
        "s3tables:PutTableData"
      ],
      "Resource": "arn:aws:s3tables:us-east-1:111122223333:bucket/*",
      "Effect": "Allow"
    },
    {
      "Action": [
        "cloudwatch:PutMetricData"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "cloudwatch:namespace": "AWS/Glue/ZeroETL"
        }
      },
      "Effect": "Allow"
    },
    {
      "Action": [
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents"
      ],
```

```
      "Resource": "*",
      "Effect": "Allow"
    }
  ]
}
```

Ajoutez la politique de confiance suivante dans le rôle Target IAM pour permettre au AWS Glue Service de l'assumer :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "glue.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

> **Note**
>
> Assurez-vous qu'il n'existe aucune instruction DENY explicite pour ce rôle IAM cible dans la politique de ressources du compartiment des tableaux S3. Une instruction DENY explicite remplacerait toutes les autorisations ALLOW et empêcherait l'intégration de fonctionner correctement.

Configuration d'un catalogue Amazon SageMaker Lakehouse avec stockage géré Amazon Redshift

Cette section décrit les conditions préalables et les étapes de configuration pour configurer un catalogue Amazon SageMaker Lakehouse avec un stockage Amazon Redshift géré (RMS) comme cible pour votre intégration zéro ETL.

Conditions préalables à la configuration d'une intégration

Avant de créer une intégration zéro ETL avec un catalogue Amazon SageMaker Lakehouse à l'aide du stockage géré Redshift, vous devez effectuer les tâches de configuration suivantes :

1. Configuration d'un Amazon Redshift cluster ou d'un groupe de travail sans serveur
2. Enregistrez l' Amazon Redshift intégration avec Lake Formation
3. Créer un catalogue géré dans Lake Formation
4. Configuration des autorisations IAM

Configuration du stockage Amazon Redshift géré

Pour configurer le stockage Amazon Redshift géré pour votre intégration Zero-ETL :

1. Créez ou utilisez un cluster Amazon Redshift ou un groupe de travail sans serveur existant. Assurez-vous que le `enable_case_sensitive_identifier` paramètre est activé dans le Amazon Redshift groupe de travail ou le cluster cible pour que l'intégration soit réussie. Pour plus d'informations sur l'activation de la sensibilité à la casse, consultez Activation de la sensibilité à la casse pour votre entrepôt de données dans le Guide de la gestion Amazon Redshift.
2. Enregistrez une intégration de Redshift dans le catalogue dans AWS Lake Formation. Consultez la section Enregistrement de Amazon Redshift clusters et d'espaces de noms dans le catalogue de AWS Glue données.
3. Créez un catalogue fédéré ou géré dans. AWS Lake Formation Pour en savoir plus, consultez :
   - Intégrer Amazon Redshift des données dans le catalogue AWS Glue de données
   - Création d'un catalogue Amazon Redshift géré dans le catalogue AWS Glue de données
4. Configurez les autorisations IAM pour le rôle cible. Le rôle a besoin d'autorisations pour accéder aux ressources Redshift et Lake Formation. Au minimum, le rôle doit disposer des éléments suivants :

- Autorisations d'accès au cluster Redshift ou au groupe de travail
- Autorisations d'accès au catalogue Lake Formation
- Autorisations de création et de gestion de tableaux dans le catalogue
- CloudWatch et CloudWatch enregistre les autorisations à des fins de surveillance

Après avoir configuré le catalogue Amazon SageMaker Lakehouse avec le stockage géré par Amazon Redshift, vous pouvez terminer la configuration Configuration de l'intégration avec votre cible de l'intégration.

Configuration d'une cible d'entrepôt de Amazon Redshift données

Cette section décrit les conditions préalables et les étapes de configuration pour configurer un entrepôt de Amazon Redshift données en tant que cible pour votre intégration zéro ETL.

Conditions préalables à la configuration d'une intégration

Avant de créer une intégration zéro ETL avec une cible d'entrepôt de Amazon Redshift données, vous devez effectuer les tâches de configuration suivantes :

1. Configuration d'un Amazon Redshift cluster ou d'un groupe de travail sans serveur
2. Configurer la sensibilité à la casse
3. Configuration des autorisations IAM

Configuration de l'entrepôt Amazon Redshift de données

Pour configurer un entrepôt de Amazon Redshift données pour votre intégration Zero-ETL :

1. Accédez à la console Amazon Redshift et cliquez sur Créer un cluster ou utilisez un cluster existant. Pour Amazon Redshift sans serveur, cliquez sur Créer un groupe de travail.
2. Si vous créez un cluster, choisissez une taille de cluster appropriée et assurez-vous que votre cluster est chiffré. Pour le sans serveur, configurez les paramètres de groupe de travail en fonction de vos besoins.
3. Assurez-vous que le `enable_case_sensitive_identifier` paramètre est activé dans le Amazon Redshift groupe de travail ou le cluster cible pour que l'intégration soit réussie. Pour plus d'informations sur l'activation de la sensibilité à la casse, consultez Activation de la sensibilité à la casse pour votre entrepôt de données dans le Guide de la gestion Amazon Redshift.

4. Configurez les autorisations IAM pour permettre à l'intégration Zero-ETL d'accéder à votre entrepôt de Amazon Redshift données. Vous devrez créer un rôle IAM avec les autorisations suivantes :
   - Autorisations d'accès au Amazon Redshift cluster ou au groupe de travail
   - Autorisations permettant de créer et de gérer des bases de données et des tables dans Amazon Redshift
   - CloudWatch et Amazon CloudWatch enregistre les autorisations à des fins de surveillance
5. Une fois la configuration du Amazon Redshift groupe de travail ou du cluster terminée, vous devez configurer votre entrepôt de données pour les intégrations sans ETL. Pour plus d'informations, reportez-vous à Démarrer avec les intégrations zéro ETL dans le Guide de gestion Amazon Redshift.

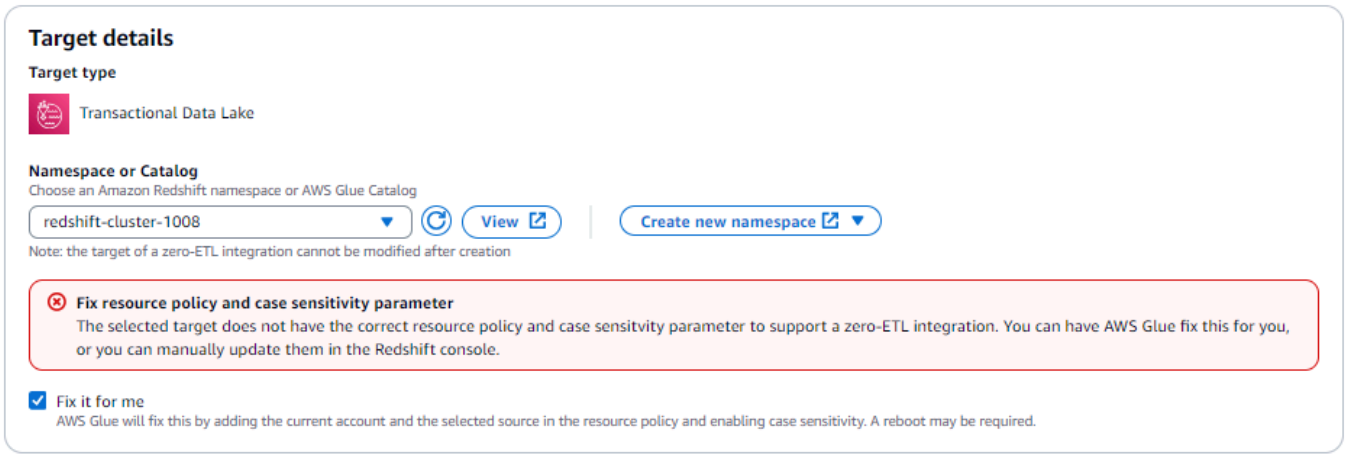
> **Note**
>
> Lorsque vous utilisez un entrepôt de Amazon Redshift données comme cible, l'intégration crée un schéma dans la base de données spécifiée pour stocker les données répliquées. Le nom du schéma provient du nom de l'intégration.

Après avoir configuré l'entrepôt de Amazon Redshift données, vous pouvez Configuration de l'intégration avec votre cible terminer la configuration de l'intégration.

Configuration de l'intégration avec votre cible

Après avoir configuré vos ressources cible, sélectionné votre connexion et spécifié un rôle IAM source, procédez comme suit pour terminer la configuration de l'intégration :

1. Spécifiez la cible que vous avez configurée lors des étapes précédentes.
2. Sélectionnez l'option AWS Glue Corriger le problème pour moi. Pour la Amazon Redshift cible, cela permettra de :
   - Appliquez un principal de service autorisé sur le Amazon Redshift cluster ou le groupe de travail sans serveur.
   - Appliquez un ARN AWS Glue source autorisé au Amazon Redshift cluster ou au groupe de travail sans serveur.
   - Associer un nouveau groupe de paramètres à `enable_case_sensitive_identifier = true`.



3. Indiquez le nom de l'intégration et choisissez Créer et lancer l'intégration.
4. Une fois que votre intégration est active, accédez à la page des détails de l'intégration et choisissez Créer une base de données à partir de l'intégration.
5. Enfin, vous pouvez accéder à l'éditeur de requêtes Redshift et vous connecter à votre base de données pour valider l'instantané et les données incrémentielles.

> **Note**
>
> Vous ne pouvez utiliser que des caractères alphanumériques minuscules et des traits de soulignement dans l'espace de noms ou le nom du catalogue. Cela est différent de ce que permet le catalogue de AWS Glue données pour créer une base de données avec n'importe quel nom (y compris les caractères spéciaux).

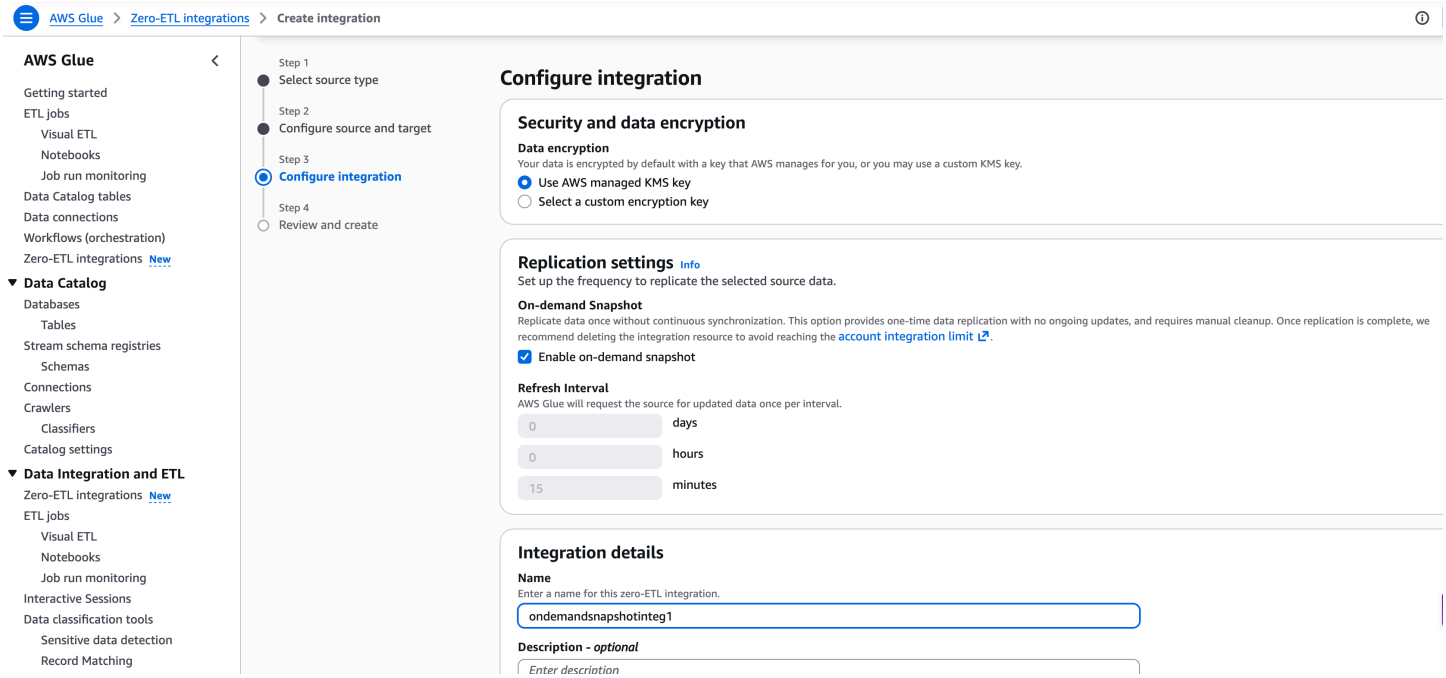
Configuration d'une intégration

Lorsque vous configurez une intégration zéro ETL, vous pouvez configurer différents paramètres pour contrôler la façon dont les données sont synchronisées entre vos systèmes source et cible. Cette section décrit les principales options de configuration qui affectent le processus d'extraction et de chargement des données.

Configuration des instantanés à la demande

Le paramètre On-Demand Snapshot vous permet de choisir de synchroniser en permanence les mises à jour des sources de données avec votre cible de données. Lorsqu'elle est désactivée

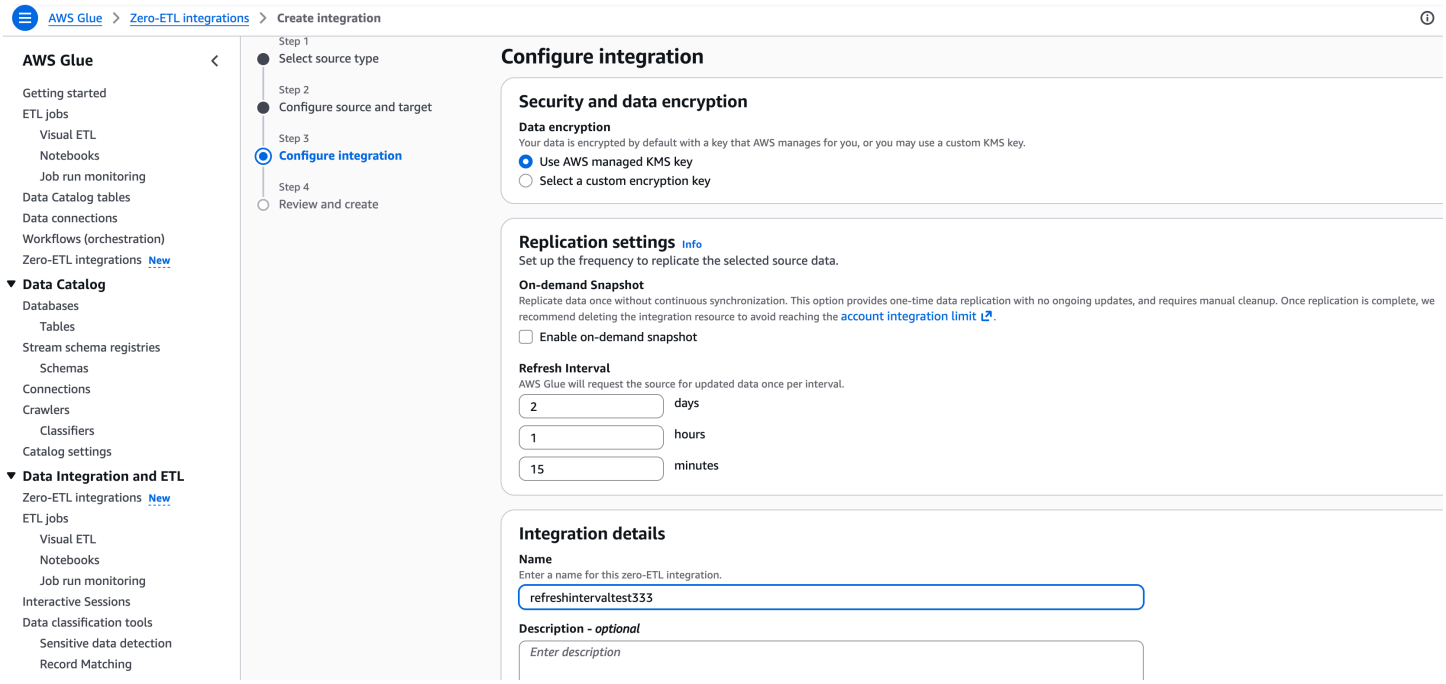
(valeur par défaut), l'intégration assure une synchronisation continue au fur et à mesure que des modifications se produisent dans les systèmes source. Lorsqu'elle est activée, l'intégration effectue une réplication des données unique sans mises à jour continues.



> **Note**
>
> Le paramètre On-Demand Snapshot ne peut pas être modifié une fois l'intégration créée. Choisissez cette option avec soin en fonction de vos besoins en matière de synchronisation des données.

RefreshInterval réglage

Le `RefreshInterval` paramètre spécifie la fréquence à laquelle les extractions de capture des données de modification (CDC) ou les chargements incrémentiels seront déclenchés. Ce paramètre offre la flexibilité nécessaire pour aligner le taux CDC sur vos modèles de mise à jour des données spécifiques, les considérations relatives à la charge du système et les objectifs d'optimisation des performances. L'intervalle d'actualisation ne peut pas être modifié une fois l'intégration créée lorsque la cible est Redshift. Pour les autres cibles, l'intervalle d'actualisation peut être modifié après la création de l'intégration. Pour les sources DynamoDB dont les intervalles d'actualisation sont supérieurs ou égaux à 24 heures, Lots quotidiens séquentiels pour les sources DynamoDB voir pour plus de détails sur le traitement par lots quotidien séquentiel.



L'incrément de temps peut être réglé de 15 minutes à 8 640 minutes (six jours), ce qui vous permet de trouver un équilibre entre la fraîcheur des données et l'utilisation des ressources du système. Actuellement, l'intervalle d'actualisation est personnalisable pour les sources DynamoDB et SaaS :

- Intervalle minimum : 15 minutes
- Intervalle maximal : 8640 minutes (6 jours)
- Valeur par défaut : 15 minutes pour la source DynamoDB et 60 minutes pour la source SaaS

Facteurs à prendre en compte lors du choix d'un intervalle d'actualisation :

- Volatilité des données : fréquence à laquelle vos données sources changent
- Besoins commerciaux : dans quelle mesure vos données analytiques doivent être à jour
- Considérations financières : des mises à jour plus fréquentes peuvent entraîner des coûts de traitement et de stockage plus élevés

> **Note**
>
> RefreshInterval paramètre définit la fréquence de déclenchement du CDC. La fréquence d'actualisation réelle peut être affectée par le volume de modifications apportées à vos données source et par la capacité de traitement du système cible. Surveillez les

performances de votre intégration et ajustez l'intervalle d'actualisation selon les besoins afin de l'optimiser en fonction de votre cas d'utilisation spécifique.

Pour modifier l'intervalle d'actualisation par programmation, vous pouvez utiliser l'ModifyIntegration API avec le IntegrationConfig paramètre.

Lots quotidiens séquentiels pour les sources DynamoDB

Pour les intégrations sans ETL avec une source Amazon DynamoDB, lorsque vous configurez un intervalle d'actualisation de 1 440 minutes (24 heures) ou plus, l'intégration utilise un traitement par lots quotidien séquentiel au lieu d'une seule opération d'exportation. Ce comportement est dû à la limitation de la fenêtre d'exportation DynamoDB, dont la durée d'exportation maximale est de 24 heures.

Lorsque l'intervalle d'actualisation dépasse 24 heures, l'intégration fonctionne comme suit :

1. Le processus CDC attend la durée complète de l'intervalle d'actualisation (par exemple, 6 jours pour un intervalle de 8 640 minutes).
2. Une fois l'intervalle d'actualisation écoulé, l'intégration effectue plusieurs exportations DynamoDB séquentielles, chacune couvrant une fenêtre de 24 heures maximum.
3. Les tâches CDC traitent chaque lot de manière séquentielle afin de capturer toutes les modifications survenues pendant la période d'intervalle d'actualisation.

Par exemple, si vous définissez un intervalle d'actualisation de 8 640 minutes (6 jours), l'intégration attendra 6 jours, puis exécutera 6 ou 7 exportations séquentielles (1 exportation finale couvrant le temps supplémentaire consacré aux opérations d'exportation) et des tâches CDC pour synchroniser toutes les modifications effectuées au cours de cette période.

Guide de spécification des partitions et de désimbrication des schémas

Lorsque vous utilisez des sources de données NoSQL telles que DynamoDB et des applications SaaS, les données présentent souvent des défis uniques en matière d'analytique :

1. Les enregistrements dans une même table peuvent avoir un schéma différent.

2. Les enregistrements imbriqués dans la même table peuvent être représentés différemment.
3. Les structures imbriquées complexes telles que les cartes et les tableaux nécessitent une transformation pour des requêtes efficaces.
4. Une organisation optimale des données est nécessaire pour garantir les performances des requêtes à grande échelle.

AWS Les intégrations Glue Zero-ETL répondent à ces défis grâce à deux puissantes fonctionnalités :

- Désimbrication de schéma : aplatit automatiquement les structures de données imbriquées complexes dans des formats adaptés à l’analytique, avec des niveaux de désimbrication configurables pour trouver un équilibre entre la préservation de la structure des données et l’optimisation pour la simplicité des requêtes.
- Partitionnement des données : organise les données en partitions logiques en fonction de colonnes spécifiées ou de dimensions temporelles, améliorant ainsi les performances des requêtes et réduisant les coûts en permettant l’élagage des partitions lors de l’exécution des requêtes.

Afin d'interroger efficacement ces sources de données, AWS Glue Zero-ETL fournit des out-of-the-box schémas de gestion et de partitionnement pour les données sources répliquées dans la base de données Glue AWS cible. Vous pouvez configurer les paramètres de désimbrication et de partitionnement du schéma pour chaque table via l' CreateIntegrationTableProperty API, ce qui permet de contrôler avec précision la manière dont les données sont structurées et organisées pour les charges de travail d'analyse.

Comportement de désimbrication et de partitionnement par défaut

1. AWS Glue Zero-ETL prend par défaut la valeur FULL Unnest lorsqu'aucune option de désimbrication n'est fournie pour la table cible
2. AWS Glue Zero-ETL utilise par défaut le partitionnement par bucket lorsque aucun élément n'est fourni pour la PartitionSpec table cible

Désimbrication du schéma

Lors de l'intégration à des services d'analyse via Zero-ETL, vous pouvez choisir la manière dont les structures imbriquées sont représentées dans les tables cibles. AWS Glue Zero-ETL propose des options de désimbrication des schémas pour aplatir les structures de données complexes dans des formats plus adaptés aux analyses.

Options de désimbrication

Lorsque vous créez une intégration zéro ETL avec une source, vous pouvez choisir l'une des options de désimbrication suivantes. Ces options correspondent à des valeurs d'énumération spécifiques que vous utiliserez lors de l'appel de l' CreateIntegrationTableProperty API.

Aucune désimbrication (par défaut)

Valeur d'API : **`NO_UNNEST`**

Préserve la structure imbriquée d'origine des éléments DynamoDB. Les cartes et les listes sont stockées sous forme de colonnes structurées dans la cible.

Idéal pour : préserver la structure exacte de vos données DynamoDB lorsque vos outils d'analytique peuvent fonctionner avec des données imbriquées.

Désimbriquer un niveau

Valeur d'API : **`TOP_LEVEL`**

Aplatit le niveau supérieur des cartes imbriquées en colonnes individuelles. Les structures de listes restent imbriquées.

Idéal pour : trouver un équilibre entre la préservation de la structure des données et la simplicité des requêtes lorsque vos éléments DynamoDB ont un schéma cohérent.

Désimbriquer tous les niveaux

Valeur d'API : **`FULL`**

Aplatit de manière récursive toutes les structures imbriquées (cartes et listes) en colonnes individuelles avec notation par points pour la dénomination.

Idéal pour : optimiser la simplicité des requêtes lorsque vous travaillez avec des structures profondément imbriquées et des outils d'analytique privilégiant les schémas plats.

> **Note**
>
> Une désimbrication complète peut donner lieu à des tables très larges comportant de nombreuses colonnes si vos données DynamoDB présentent des structures variables ou profondément imbriquées.

Example Utilisation des options de désimbrication dans l'API

Lors de la configuration de la désimbrication du schéma via l' CreateIntegrationTableProperty API, spécifiez l'option de désimbrication dans le paramètre : UnnestSpec

```
aws glue create-integration-table-property
  --resource-arn "arn:aws:glue:us-east-1:123456789012:integration/my-integration"
  --table-name "my-table"
  --cli-input-json '{
      "TargetTableConfig": {
          "UnnestSpec": "FULL",
          "TargetTableName": "my-target-table",
      }
  }'
```

Exemples de désimbrication

Prenons l'exemple d'un élément DynamoDB dont la structure est la suivante :

```
{
  "ProductId": "P12345",
  "ProductDetails": {
    "Name": "Smartphone",
    "Brand": "TechCo",
    "Specifications": {
      "Color": "Black",
      "Storage": "128GB"
    }
  },
  "Reviews": [
    {
      "Rating": 5,
      "Comment": "Great product!"
    },
    {
      "Rating": 4,
      "Comment": "Good value."
    }
  ]
}
```

Exemple d'absence de désimbrication

En l'absence de désimbrication, la table cible comporterait des colonnes :

- ProductId (chaîne)
- ProductDetails (structure)
- Reviews (tableau de structures)

Les requêtes devraient utiliser des modèles d'accès aux structures et aux tableaux :

```
SELECT
  ProductId,
  ProductDetails.Name,
  ProductDetails.Specifications.Color
FROM product_table;
```

Exemple de désimbrication d'un niveau

Avec la désimbrication d'un niveau, la table cible comporterait des colonnes :

- ProductId (chaîne)
- ProductDetails_Nom (chaîne)
- ProductDetails_Brand (chaîne)
- ProductDetails_Spécifications (structure)
- Reviews (tableau de structures)

Les requêtes seraient simplifiées pour le premier niveau :

```
SELECT
  ProductId,
  ProductDetails_Name,
  ProductDetails_Specifications.Color
FROM product_table;
```

Exemple de désimbrication de tous les niveaux

Avec la désimbrication de tous les niveaux, la table cible comporterait des colonnes et des valeurs :

Nom de colonne (type)	Value
ProductId (chaîne)	P12345
ProductDetails_Nom (chaîne)	Smartphone
ProductDetails_Brand (chaîne)	TechCo
ProductDetails_Specifications_Color (chaîne)	Black
ProductDetails_Specifications_Storage (chaîne)	128 Go
Reviews_0_Rating (nombre)	5
Reviews_0_Comment (chaîne)	Excellent produit !
Reviews_1_Rating (nombre)	4
Reviews_1_Comment (chaîne)	Bon rapport qualité/prix.

Les requêtes seraient entièrement aplaties :

```
SELECT
  ProductId,
  ProductDetails_Name,
  ProductDetails_Specifications_Color
FROM product_table;
```

Partitionnement de données

Qu'est-ce que le partitionnement des données ?

Le partitionnement des données est une technique qui divise les grands jeux de données en segments plus petits et plus faciles à gérer, appelés partitions. Dans le contexte des intégrations de AWS Glue Zero-ETL, le partitionnement organise vos données dans l'emplacement cible en fonction de valeurs de colonne spécifiques ou de transformations de ces valeurs.

Avantages du partitionnement des données

Un partitionnement efficace des données présente plusieurs avantages essentiels pour les charges de travail analytiques :

- Performances de requêtes améliorées : les requêtes peuvent ignorer les partitions non pertinentes (élagage des partitions), réduisant ainsi la quantité de données à analyser.
- Coûts réduits : en scannant moins de données, vous pouvez réduire le calcul et I/O les coûts liés à vos requêtes d'analyse.
- Meilleure capacité de mise à l'échelle : le partitionnement permet le traitement parallèle des segments de données, et donc une mise à l'échelle plus efficace des charges de travail analytiques.
- Gestion simplifiée du cycle de vie des données : vous pouvez gérer les politiques de conservation au niveau de la partition, ce qui facilite l'archivage ou la suppression des données plus anciennes.

Concepts clés du partitionnement

Colonnes de partition

Colonnes dans vos données utilisées pour déterminer la manière dont les enregistrements sont organisés en partitions. Des colonnes de partition efficaces doivent s'aligner sur les modèles de requêtes courants et disposer d'une cardinalité appropriée.

Fonctions de partition

Transformations appliquées aux valeurs des colonnes de partition pour créer les limites réelles de la partition. Les exemples incluent l'identité (en utilisant la valeur brute) et les fonctions basées sur le temps (année, mois, jour, heure).

Élagage des partitions

Processus par lequel le moteur de requête identifie et ignore les partitions qui ne contiennent pas de données pertinentes pour une requête, ce qui améliore considérablement les performances.

Granularité de partition

Niveau de détail selon lequel les données sont partitionnées. Une granularité plus fine (davantage de partitions) peut améliorer les performances des requêtes, mais peut augmenter la surcharge de métadonnées. Une granularité plus grossière (moins de partitions) réduit la surcharge de métadonnées, mais peut entraîner l'analyse d'un plus grand nombre de données que nécessaire.

Partitionnement dans les intégrations AWS Glue Zero-ETL

AWS Les intégrations Glue Zero-ETL utilisent le format de table Apache Iceberg, qui fournit des fonctionnalités de partitionnement avancées. Lorsque vous créez une intégration zéro ETL, vous pouvez effectuer les actions suivantes :

- Utiliser des stratégies de partitionnement par défaut optimisées pour votre source de données
- Définir des spécifications de partitionnement personnalisées adaptées à vos modèles de requêtes
- Appliquer des transformations aux colonnes de partition (ce qui est particulièrement utile pour le partitionnement basé sur l'horodatage)
- Combiner plusieurs stratégies de partition pour un partitionnement à plusieurs niveaux

Les configurations de partitionnement sont spécifiées via l'API `CreateIntegrationTableProperty` lors de la configuration de votre intégration zéro ETL. Une fois configuré, AWS Glue applique automatiquement ces stratégies de partitionnement pour organiser vos données dans l'emplacement cible.

Référence d'API de spécification de partition

Utilisez les paramètres suivants dans l' CreateIntegrationTableProperties API pour configurer le partitionnement :

PartitionSpec

Tableau de spécifications de partition qui définit la manière dont les données sont partitionnées dans l'emplacement cible.

```
{
  "partitionSpec": [
    {
      "fieldName": "timestamp_col",
      "functionSpec": "month",
      "conversionSpec": "epoch_milli"
    },
    {
      "fieldName": "category",
      "functionSpec": "identity"
    }
  ]
```

```
}
```

FieldName

Chaîne UTF-8 (1 à 128 octets) spécifiant le nom de colonne à utiliser pour le partitionnement.

FunctionSpec

Spécifie la fonction de partitionnement. Valeurs valides :

- `identity` : utilise les valeurs source directement sans transformation
- `year` : extrait l'année à partir des valeurs d'horodatage (par exemple, 2023)
- `month` : extrait le mois à partir des valeurs d'horodatage (par exemple, 2023-01)
- `day` : extrait le jour à partir des valeurs d'horodatage (par exemple, 2023-01-15)
- `hour` : extrait l'heure à partir des valeurs d'horodatage (par exemple, 2023-01-15-14)

> **Note**
>
> Les fonctions basées sur le temps (`year`, `month`, `day`, `hour`) ont besoin du paramètre `ConversionSpec` pour spécifier le format d'horodatage source.

ConversionSpec

Chaîne UTF-8 qui spécifie le format d'horodatage des données source. Les valeurs valides sont :

- `epoch_sec` : horodatage d'époque Unix en secondes
- `epoch_milli` : horodatage de l'époque Unix en millisecondes
- `iso` : horodatage au format ISO 8601

Stratégies de partitionnement

Partitionnement par défaut

Lorsqu'aucune colonne de partition n'est spécifiée, AWS Glue Zero-ETL applique des stratégies de partitionnement par défaut optimisées pour votre source de données :

- Partitionnement basé sur les clés primaires : pour les sources dotées de clés primaires (telles que les tables DynamoDB), l'intégration zéro ETL AWS Glue partitionne automatiquement les données à l'aide de la clé primaire avec des compartiments afin d'éviter l'explosion des partitions.

Le partitionnement par défaut est conçu pour fonctionner correctement pour les modèles de requêtes courants sans nécessiter de configuration manuelle. Toutefois, pour des modèles de requêtes ou des exigences de performances spécifiques, vous souhaiterez peut-être définir des stratégies de partitionnement personnalisées.

Stratégies de partitionnement définies par l'utilisateur

AWS Glue Zero-ETL vous permet de définir des stratégies de partitionnement personnalisées à l'aide du `PartitionSpec` paramètre. Vous pouvez spécifier une ou plusieurs colonnes de partition et appliquer différentes fonctions de partitionnement à chaque colonne.

Le partitionnement d'identité utilise les valeurs brutes d'une colonne pour créer des partitions. Cette stratégie est utile pour les colonnes dont la cardinalité est faible à moyenne, telles que les champs de catégorie, de région ou d'état.

Example Exemple de partitionnement d'identité

```
{
  "partitionSpec": [
    {
      "fieldName": "category",
      "functionSpec": "identity"
    }
  ]
}
```

Cette fonction crée des partitions distinctes pour chaque valeur unique dans la colonne « category ».

> **Warning**
>
> Évitez d'utiliser le partitionnement d'identité avec des colonnes à cardinalité élevée (comme clés primaires ou horodatages), car cela peut entraîner une explosion des partitions, ce qui dégrade les performances et augmente la surcharge de métadonnées.

Le partitionnement basé sur le temps organise les données en fonction des valeurs d'horodatage et selon différents niveaux de granularités (année, mois, jour ou heure). Cette stratégie est idéale pour les données de séries temporelles et permet d'effectuer des requêtes temporelles efficaces.

Lors de l'utilisation du partitionnement basé sur le temps, AWS Glue Zero-ETL peut convertir automatiquement différents formats d'horodatage en un format standardisé avant d'appliquer la fonction de partition. Cette conversion est spécifiée à l'aide du paramètre `ConversionSpec`.

Example Exemple de partitionnement basé sur le temps

```
{
  "partitionSpec": [
    {
      "fieldName": "created_at",
      "functionSpec": "month",
      "conversionSpec": "epoch_milli"
    }
  ]
}
```

Cela partitionne les données par mois en fonction de la colonne « created_at », qui contient les horodatages d'époque Unix en millisecondes.

AWS Glue Zero-ETL prend en charge les fonctions de partition temporelles suivantes :

- year : partitionne les données par année (par exemple, 2023, 2024)
- month : partitionne les données par mois (par exemple, 2023-01, 2023-02)
- day : partitionne les données par jour (par exemple, 2023-01-01, 2023-01-02)
- hour : partitionne les données par heure (par exemple, 2023-01-01-01, 2023-01-01-02)

AWS Glue Zero-ETL prend en charge les formats d'horodatage suivants via le paramètre : `ConversionSpec`

- epoch_sec : horodatages d'époque Unix en secondes
- epoch_milli : horodatages d'époque Unix en millisecondes
- iso : horodatages au format ISO 8601

> **Note**
>
> Les valeurs de colonne d'origine restent inchangées dans vos données source. AWS Glue transforme uniquement les valeurs des colonnes de partition en type Timestamp dans la table

de base de données cible. Les transformations s'appliquent uniquement au processus de partitionnement.

Le partitionnement à plusieurs niveaux combine plusieurs stratégies de partition pour créer un schéma de partitionnement hiérarchique. Cela est utile pour optimiser différents types de requêtes sur un même jeu de données.

Example Exemple de partitionnement à plusieurs niveaux

```
{
  "partitionSpec": [
    {
      "fieldName": "created_at",
      "functionSpec": "month",
      "conversionSpec": "iso"
    },
    {
      "fieldName": "region",
      "functionSpec": "identity"
    }
  ]
}
```

Cela crée un schéma de partitionnement à deux niveaux : d'abord par mois (à partir de la colonne « created_at »), puis par région. Cela permet des requêtes efficaces qui filtrent par plages de dates, régions spécifiques ou une combinaison de ces dimensions.

Lorsque vous concevez des schémas de partitionnement à plusieurs niveaux, tenez compte des points suivants :

- Placement des colonnes à haute sélectivité en premier dans la hiérarchie des partitions
- Équilibrage de la granularité des partitions avec le nombre de partitions
- Alignement du schéma de partitionnement sur vos modèles de requêtes les plus courants

Bonnes pratiques

Sélection de la colonne de partition

- N'utilisez pas de colonnes à cardinalité élevée avec la fonction de partition `identity`. L'utilisation de colonnes à haute cardinalité avec partitionnement des identités crée de nombreuses petites partitions, ce qui peut considérablement dégrader les performances d'ingestion. Les colonnes à haute cardinalité peuvent inclure :
  - Clés primaires
  - Champs d'horodatage (tels que `LastModifiedTimestamp` et `CreatedDate`)
  - Horodatages générés par le système
- Ne sélectionnez pas plusieurs partitions d'horodatage sur la même colonne. Par exemple :

```
"partitionSpec": [
      {"fieldName": "col1", "functionSpec": "year", "conversionSpec" :
 "epoch_milli"},
      {"fieldName": "col1", "functionSpec": "month", "conversionSpec" :
 "epoch_milli"},
      {"fieldName": "col1", "functionSpec": "day", "conversionSpec" : "epoch_milli"},
      {"fieldName": "col1", "functionSpec": "hour", "conversionSpec" : "epoch_milli"}
]
```

FunctionSpec/ConversionSpec Sélection de partitions

- Spécifiez le format correct ConversionSpec (epoch_sec | epoch_milli | iso) qui représente le format des valeurs de colonne choisies pour le partitionnement basé sur l'horodatage lors de l'utilisation de fonctions de partition basées sur l'horodatage. AWS Glue Zero-ETL utilise ce paramètre pour transformer correctement les données source au format d'horodatage avant le partitionnement.
- Utilisez la granularité appropriée (year/month/day/hour) en fonction du volume de données.
- Tenez compte des implications du fuseau horaire lorsque vous utilisez des horodatages ISO. AWS Glue Zero-ETL remplit toutes les valeurs d'enregistrement de la colonne d'horodatage choisie avec le fuseau horaire UTC.

Gestion des erreurs

État NEEDS_ATTENTION

Une intégration passe à l’état NEEDS_ATTENTION dans les cas suivants :

- Les colonnes de partition spécifiées n’existent pas dans la source.
- La conversion d’horodatage échoue pour les colonnes de partition.

Limitations

Restrictions du partitionnement

- Les spécifications de partition ne peuvent pas être modifiées une fois qu’une intégration a été créée. Pour pouvoir utiliser une stratégie de partitionnement différente, vous devez créer une intégration.
- Le nombre maximum de colonnes de partition est limité à 10.

Restrictions de l’intégration entre comptes

- Lors de la création d'intégrations entre comptes, AWS la console Glue est limitée par le fait qu'elle n'invoque pas d' CreateIntegrationTableProperty API pour configurer UnnestSpec et PartitionSpec pour cibler les tables AWS Glue hébergées sur le compte où l'intégration n'existe pas.

  Solution : L' CreateIntegrationTableProperty API doit être invoquée par CX depuis le compte sur lequel se trouve la base de données cible.

Restrictions des intégrations multiples

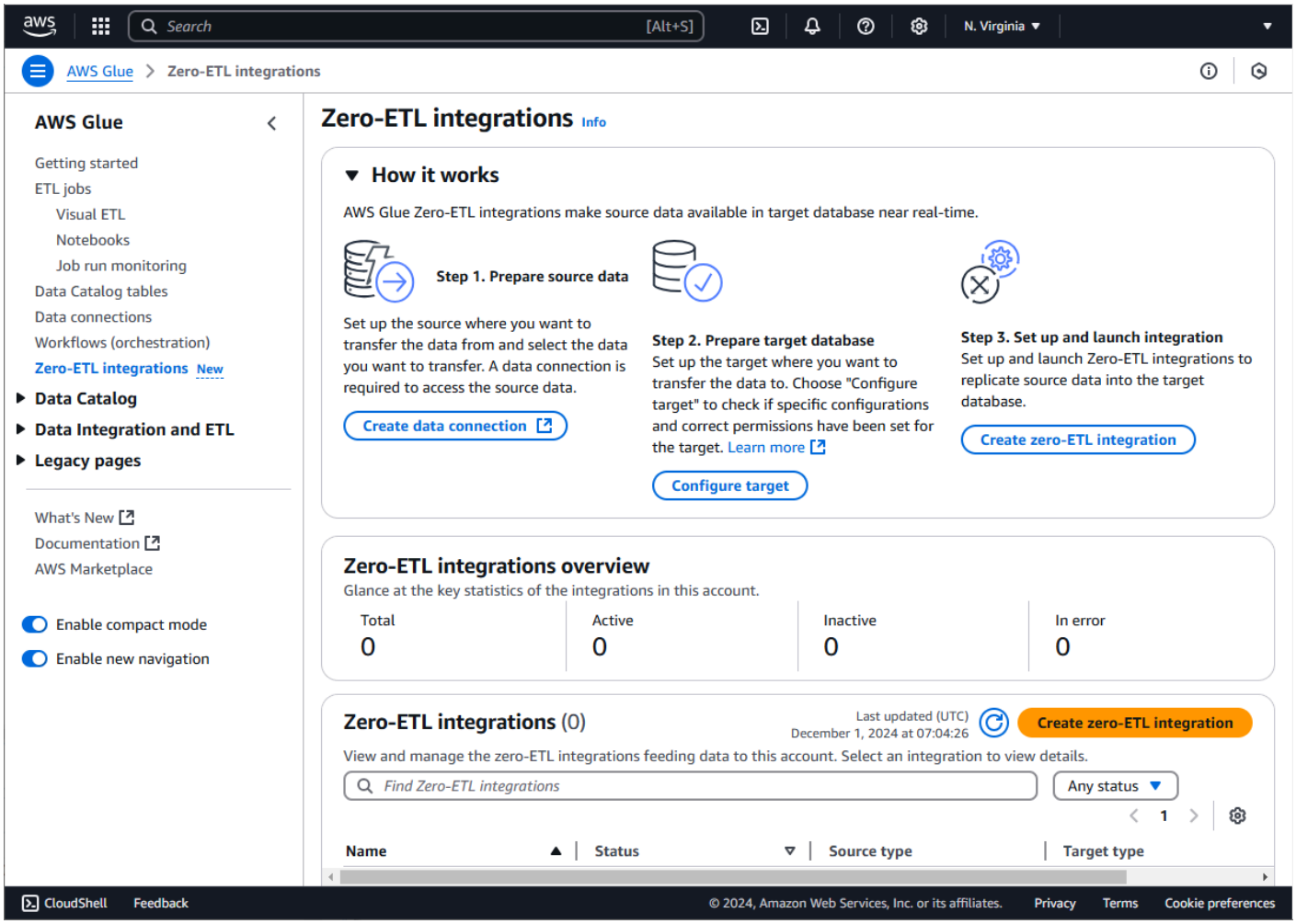
- Si vous devez répliquer la même source avec des unnest/partition configurations de schéma différentes, il est nécessaire de créer une nouvelle base de données AWS Glue séparément pour chaque intégration. Invoquez ensuite CreateIntegrationTableProperty pour chaque table à partir de AWS la base de données Glue individuelle avec les configurations de dénidification et de partitionnement du schéma souhaitées.

Tâches d’intégration courantes

Création d'une intégration

Cette section décrit les étapes générales permettant de créer une intégration. Cet exemple utilise Amazon DynamoDB en tant que source.

1. Sur la page d'accueil de la AWS Glue console, sélectionnez Zero-ETL integrations.
2. Vous pouvez consulter toutes vos intégrations sur la page d'accueil de l'intégration zéro ETL. Pour créer une intégration, sélectionnez Créer une intégration zéro ETL.



3. Vous êtes invité à sélectionner un type de source. Sélectionnez votre source et cliquez sur Suivant. Reportez-vous aux sections de configuration des sources pour les sources d'intégration SaaS.
4. Sur la page Configurer la source et la cible, sélectionnez les tables ou les entités à répliquer. Pour Amazon DynamoDB, assurez-vous que la politique PITR et RBAC est configurée.
5. Spécifiez votre cible d'intégration :

- Pour une cible de catalogue de AWS Glue données, sélectionnez la AWS Glue base de données dans laquelle vous souhaitez répliquer les données.
- Pour une cible d'entrepôt de données Amazon Redshift, sélectionnez l'espace de noms du cluster Redshift ou l'espace de noms du groupe de travail Redshift sans serveur.

Pour de plus amples informations, veuillez consulter Configuration de l'intégration avec votre cible.

6. Indiquez le rôle IAM cible que vous avez créé dans les conditions préalables.
7. Si vous souhaitez configurer une clé KMS cible facultative pour les données stockées dans la cible, indiquez une clé KMS activée. De même, si vous souhaitez configurer une connexion réseau cible, sélectionnez-en une AWS Glue .
8. Le bouton Corriger la cible configure certaines des étapes de la section Prérequis de cette documentation. À savoir, 1) il fournira une politique RBAC du catalogue et 2) si aucun URI Amazon S3 n'est fourni, il en générera un pour vous, sinon il utilisera l'URI fourni.
9. Pour les intégrations avec une cible d'entrepôt de données Redshift :
10. Dans la section Paramètre de sortie de la page Configurer la source et la cible, sélectionnez l'option de désimbrication du schéma que vous souhaitez pour vos données dans la cible. Si vous souhaitez utiliser des clés de partition client pour vos données, sélectionnez Spécifier des clés de partition personnalisées et fournissez jusqu'à 10 clés. Sinon, vous pouvez simplement utiliser les clés de partition attribuées à votre table DynamoDB en cours de réplication.
11. Dans la section Sécurité et chiffrement des données, vous pouvez fournir une clé KMS qui sera utilisée dans le processus intermédiaire de réplication de vos données vers la cible. Dans le cas contraire, une clé KMS AWS gérée sera utilisée. Actuellement, nous ne prenons en charge qu'un paramètre de réplication de 15 minutes. Saisissez un nom pour l'intégration zéro ETL dans Détails de l'intégration.
12. Vérifiez et assurez-vous que tous les détails fournis sont corrects. Cliquez sur Créer et lancer l'intégration une fois que tout a été confirmé.
13. Sur la page d'accueil de zéro ETL, vous pouvez sélectionner l'intégration que vous avez créée et les détails de vos intégrations apparaîtront. « État » indique l'état de votre intégration.

Modification d'une intégration

Vous pouvez modifier une intégration existante.

1. Choisissez Modifier dans le coin supérieur droit de la page des détails de l'intégration.

2. Sur la page Modifier la source et la cible, vous pouvez modifier le rôle IAM Cible et la connexion réseau Cible. Les autres champs ne sont pas modifiables après la création de l'intégration. Cliquez sur Suivant.
3. Vous pouvez également modifier le nom et la description de l'intégration sur la page Modifier l'intégration et la configuration. Cliquez sur Suivant.
4. Vérifiez vos modifications et une fois confirmées, cliquez sur Mettre à jour l'intégration.

Suppression d'une intégration

La suppression est un état terminal d'une intégration. Une fois supprimée, l'intégration ne peut pas être rétablie. La suppression d'une intégration efface toutes les métadonnées internes et toutes les données stockées intermédiaires.

Au cours de ce processus, toutes les tâches en cours d'exécution qui écrivent des données dans une table cible sont interrompues. AWS Glue ne supprimera ni ne nettoiera la AWS Glue base de données cible (dans le catalogue de données) ni les données associées dans le compartiment Amazon S3 de votre compte. Vous devez les supprimer explicitement si nécessaire.

Pour supprimer une intégration :

1. Sur la page des détails de l'intégration, cliquez sur Supprimer.
2. Saisissez « Supprimer » et cliquez sur Supprimer. Remarque : Cette action est irréversible.
3. Sur la page des détails de l'intégration, l'état indique « Suppression ». Une fois l'intégration supprimée, elle n'apparaîtra plus sur la page d'accueil de l'intégration zéro ETL.

Création d'intégrations à l'aide du APIs

Vous pouvez utiliser ce qui suit APIs pour créer et gérer des intégrations sans ETL dans : AWS Glue

- CreateIntegration
- CreateIntegrationTableProperties
- CreateIntegrationResourceProperty
- UpdateIntegrationTableProperties
- UpdateIntegrationResourceProperty
- ModifyingIntegration

- DeleteIntegration
- DeleteIntegrationTableProperties
- DescribeIntegrations
- DescribeInboundIntegrations
- GetIntegrationTableProperties
- GetIntegrationResourceProperty

Pour de plus amples informations, veuillez consulter Intégration APIs dans AWS Glue.

Surveillance d'une intégration

États d'intégration

Les états d'intégration suivants décrivent l'intégration :

- `Creating` : l'intégration est en cours de création.
- `Active` : l'intégration envoie des données transactionnelles à la cible.
- `Modifying` : l'intégration est en cours de modification.
- `Syncing` : l'intégration a rencontré une erreur récupérable et réensemence les données.
- `Needs attention` : l'intégration a rencontré un événement ou une erreur nécessitant une intervention manuelle pour être résolu. Pour corriger le problème, suivez les instructions du message d'erreur dans les détails relatifs à l'intégration.
- `Failed` : l'intégration a rencontré un événement ou une erreur irrécupérable. Vous devez supprimer et recréer l'intégration.
- `Deleting` : l'intégration est en cours de suppression.

Consulter CloudWatch les journaux Amazon pour une intégration

AWS Glue Les intégrations sans ETL génèrent des CloudWatch journaux Amazon pour avoir une meilleure visibilité sur le mouvement de vos données. Les événements du journal relatifs à chaque ingestion réussie ou à tout échec dû à des enregistrements de données problématiques au niveau de la source, ou à des erreurs d'écriture de données dues à des modifications de schéma ou à des autorisations insuffisantes sont transmis à un groupe de journaux par défaut créé dans un compte client.

Pour chaque intégration créée, les événements du journal correspondant à cette intégration seront collectés sous `/aws-glue/zeroETL-integrations/logs/` dans Amazon Cloudwatch. Au sein du groupe de journaux, les messages du journal seront divisés en flux de journaux. Chaque intégration créée possède un flux de journal dédié dans lequel sont écrits tous les journaux pour cette intégration. Par exemple, les journaux d'une intégration avec se trouvent sous /aws- IntegrationArn `arn:aws:glue:us-east-1:123456789012:integration:03cabe77-79e7-4b7a-b3da-8c160bea6bbf` /03cabe77-79e7-4b7a-b3da-8c160bea6bbf. glue/zeroETL-integrations/ logs {IntegrationId} peut être référencé à partir du {integrationArn} généré lors de la création d'une intégration.

> **Note**
>
> Dans un scénario entre comptes, les journaux de traitement source sont émis dans le compte source où existe l'intégration et les journaux de traitement cible sont émis dans le compte cible où se trouve la base de données cible.

Autorisations IAM requises pour activer la journalisation

Lors de la création de votre intégration, les rôles source et cible ont besoin des autorisations IAM suivantes pour activer la CloudWatch journalisation d'une intégration. AWS Glue Les intégrations Zero-ETL utilisent ces autorisations fournies dans les rôles source et cible pour émettre des CloudWatch journaux sur les comptes clients.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents"
      ],
      "Resource": [
        "*"
      ]
```

```
    }
  ]
}
```

Messages de journaux

Format de journal : les intégrations zéro ETL émettent quatre types de messages de journal :

```
// Ingestion started
{
"integrationArn": "arn:aws:glue:us-east-2:123456789012:integration/1a012bba-123a-1bba-
ab1c-173de3b12345",
...
    "messageType": "IngestionStarted",
    "details": {
        "tableName": "testDDBTable",
        "message": "Ingestion Job started"
    }
}
// Data processing stats on successful table ingestion
{
...
    "messageType": "IngestionProcessingStats",
    "details": {
        "tableName": "testDDBTable",
        "insert_count": 100,
        "update_count": 10,
        "delete_count": 10
    }
}
// Ingestion failure logs for failed table-processing
{
...
    "messageType": "IngestionFailed",
    "details": {
        "tableName": "testDDBTable",
        "errorMessage": "Failed to ingest data with error: Target Glue database not
 found.",
        "error_code" : "client_error"
    }
}
// Ingestion completed notification with lastSyncedTimestamp
```

```
{
...
    "messageType": "IngestionCompleted",
    "details": {
        "tableName": "testDDBTable",
        "message": "Ingestion Job completed"
        "lastSyncedTimestamp": "1132344255745"
    }
}
```

Afficher CloudWatch les statistiques Amazon pour une intégration

Une fois l'intégration terminée, vous pouvez voir les statistiques Amazon Cloudwatch générées dans votre compte pour chaque AWS Glue tâche exécutée :

CloudWatch espace de noms des métriques : "AWS/glue/ZeroEtl »

Dimensions des métriques :

- `integrationArn`
- `loadType`
- `tableName`

Noms des métriques :

- `InsertCount` : nombre d'enregistrements insérés dans la table Iceberg cible.
- `UpdateCount` : nombre d'enregistrements mis à jour dans la table Iceberg cible.
- `DeleteCount` : nombre d'enregistrements supprimés de la table Iceberg cible.
- `IngestionSucceeded` : compte 1, si l'ingestion a réussi pour l'intégration.
- `IngestionFailed` : compte 1, si l'ingestion a échoué pour l'intégration.
- `LastSyncTimestamp` : horodatage jusqu'à la source qui a été synchronisée avec la cible.

Gérer les notifications d'événements avec Amazon EventBridge

Les intégrations Zero-ETL utilisent Amazon EventBridge pour gérer les notifications d'événements afin de vous tenir au courant up-to-date des modifications apportées à vos intégrations. Amazon

EventBridge est un service de bus d'événements sans serveur que vous pouvez utiliser pour connecter vos applications à des données provenant de diverses sources. Dans ce cas, la source de l'événement est AWS Glue. Les événements, qui sont des modifications surveillées dans un environnement, sont EventBridge envoyés AWS Glue automatiquement vers Les événements sont diffusés en temps quasi réel.

EventBridge fournit un environnement dans lequel vous pouvez écrire des règles d'événements, qui peuvent spécifier les actions à entreprendre pour des événements spécifiques. Vous pouvez également définir des cibles, c'est-à-dire des ressources auxquelles un événement EventBridge peut être envoyé. Une cible peut inclure une destination d'API, un groupe de CloudWatch journaux Amazon, etc. Pour plus d'informations sur les règles, consultez les EventBridge règles d'Amazon. Pour plus d'informations sur les cibles, consultez la section Amazon EventBridge Targets.

Pour capturer toutes les notifications zéro ETL, créez une règle EventBridge correspondant aux critères suivants :

```
{
  "source": [{
    "prefix": "aws.glue-zero-etl“
  }],
  "detail-type": [{
    "prefix": "Glue Zero ETL“
  }]
}
```

Le tableau suivant inclut des événements d'intégration zéro ETL, avec des métadonnées supplémentaires :

Type de détail destiné au client	Explication
AWS Glue Aucune ingestion d'ETL terminée	L'exécution individuelle d'une entité s'est achevée avec succès.
AWS Glue Aucune ingestion d'ETL n'a échoué	L'exécution individuelle d'une entité a échoué (en raison d'une erreur du client ou du système).
AWS Glue Aucune intégration ETL resynchro nisée	L'intégration a été RESYNCHRONISÉE.

Type de détail destiné au client	Explication
AWS Glue Aucune intégration ETL n'a échoué	L'état de l'intégration est passé à FAILED en raison d'une erreur.
AWS Glue Aucune intégration ETL n'a besoin d'attention	L'état de l'intégration est passé à NEEDS_ATTENTION en raison d'une erreur.
AWS Glue Aucune ingestion d'ETL en cours	L'exécution individuelle d'une entité a partiellement progressé vers son achèvement.

Limitations

Les restrictions ou considérations générales concernant les intégrations zéro ETL sont les suivantes :

- Les propriétés des ressources ont une one-to-one relation avec la ressource correspondante. Par conséquent, toutes les intégrations créées à l'aide de cette ressource doivent respecter la propriété de ressource unique. La modification d'une propriété de ressource aura donc un impact sur toutes les intégrations associées à cette ressource.
- Les propriétés de table ont une one-to-one relation avec la table ou l'objet correspondant au sein d'une ressource. Par conséquent, toutes les intégrations qui traitent la même table vers ou depuis la même ressource doivent respecter la propriété de table unique.
- Vous ne pouvez pas renommer une colonne au niveau de la source. Si une colonne est renommée, il n'y a aucune garantie que la détection du schéma sera effectuée avec précision AWS Glue et les répercussions sur l'intégration ne sont pas définies.
- La considération suivante s'applique au fonctionnement de l'intégration avec les tables AWS Lake Formation gérées : Par défaut, vous utilisez la AWS Glue politique IAM/ pour gérer votre table et vos bases de données.
- Si vous souhaitez utiliser AWS Lake Formation pour gérer la création de tables dans cette base de données, vous devez vous assurer que le rôle dispose des autorisations Lake Formation suffisantes pour créer, modifier et supprimer la table et la base de données.
- La page récapitulative zéro ETL ne contient aucune métrique pour le moment.

Les restrictions suivantes sont spécifiques à la source pour les intégrations zéro ETL :

- Les intégrations sans ETL avec une OData source SAP prennent en charge les entités commençant par. `Entity0f` La possibilité de remplacer la clé primaire n'est actuellement prise en charge que pour les SAPOData `Entity0f` objets. Une fois cette propriété définie, elle ne peut pas être modifiée.
- Les intégrations sans ETL entre Amazon DynamoDB et SageMaker Amazon Lakehouse (via S3) prennent en charge une taille de table DynamoDB maximale de 50 To.
- La table DynamoDB source doit être chiffrée à l'aide d'une clé KMS appartenant à Amazon ou gérée par le client. AWS Le chiffrement géré par Amazon n'est pas pris en charge pour la table DynamoDB source.
- SAP OData utilise un jeton delta, où la combinaison d'un OAuth client et d'une entité, ou d'une authentification de base associée à une entité, ne peut avoir qu'un seul jeton delta. Évitez d'utiliser la même entité dans deux intégrations différentes avec le même client.
- Les entités ou champs Salesforce suivants ne sont pas pris en charge pour une utilisation dans le cadre d'une intégration zéro ETL avec une source Salesforce.

```
AccountChangeEvent, AccountContactRoleChangeEvent, AccountHistory, AccountShare,
 ActiveFeatureLicenseMetric, ActivePermSetLicenseMetric, ActiveProfileMetric,
 ActivityFieldHistory, amzsec__asi_Telemetry_Data_Store__ChangeEvent,
 amzsec__asi_Telemetry_Data_Store__History, amzsec__asi_Telemetry_Data_Store__Share,
 amzsec__asi_Telemetry_Job_Log__ChangeEvent, amzsec__asi_Telemetry_Job_Log__History,
 amzsec__asi_Telemetry_Job_Log__Share,
 amzsec__asi_Telemetry_Requirement__ChangeEvent,
 amzsec__asi_Telemetry_Requirement__History,
 amzsec__asi_Telemetry_Requirement__Share, ApexClass, ApexComponent, ApexLog,
 ApexPage, ApexTestQueueItem, ApexTestResult, ApexTrigger, AssetChangeEvent,
 AssetHistory, AssetRelationshipHistory, AssetShare, AssignmentRule,
 AssociatedLocationHistory, AsyncApexJob, AuditTrailFileExportShare,
 AuthorizationFormConsentChangeEvent, AuthorizationFormConsentHistory,
 AuthorizationFormConsentShare, AuthorizationFormDataUseHistory,
 AuthorizationFormDataUseShare, AuthorizationFormHistory, AuthorizationFormShare,
 AuthorizationFormTextHistory, AuthProvider, AuthSession, BatchJobHistory,
 BatchJobPartFailedRecordHistory, BatchJobPartHistory, BatchJobShare,
 BrandTemplate, BriefcaseAssignmentChangeEvent, BriefcaseDefinitionChangeEvent,
 BusinessBrandShare, BusinessHours, BusinessProcess, CalcMatrixColumnRangeHistory,
 CalcProcStepRelationshipHistory, CalculationMatrixColumnHistory,
 CalculationMatrixHistory, CalculationMatrixRowHistory, CalculationMatrixShare,
 CalculationMatrixVersionHistory, CalculationProcedureHistory,
 CalculationProcedureShare, CalculationProcedureStepHistory,
 CalculationProcedureVariableHistory, CalculationProcedureVersionHistory, Calendar,
 CalendarViewShare, CallCenter, CallCoachConfigModifyEvent, CampaignChangeEvent,
```

```
CampaignHistory, CampaignMemberChangeEvent, CampaignMemberStatusChangeEvent,
CampaignShare, CaseChangeEvent, CaseHistory, CaseHistory2 CaseHistory2ChangeEvent,
CaseRelatedIssueChangeEvent, CaseRelatedIssueHistory, CaseShare, CaseStatus,
CaseTeamMember, CaseTeamRole, CaseTeamTemplate, CaseTeamTemplateMember,
CaseTeamTemplateRecord, CategoryNode, ChangeRequestChangeEvent,
ChangeRequestHistory, ChangeRequestRelatedIssueChangeEvent,
ChangeRequestRelatedIssueHistory, ChangeRequestRelatedItemChangeEvent,
ChangeRequestRelatedItemHistory, ChangeRequestShare, ChatRetirementRdyMetrics,
ChatterActivity, ClientBrowser, CollaborationGroup, CollaborationGroupMember,
CollaborationGroupMemberRequest, CollaborationInvitation,
CommSubscriptionChannelTypeHistory, CommSubscriptionChannelTypeShare,
CommSubscriptionConsentChangeEvent, CommSubscriptionConsentHistory,
CommSubscriptionConsentShare, CommSubscriptionHistory, CommSubscriptionShare,
CommSubscriptionTimingHistory, Community, ConnectedApplication, ContactChangeEvent,
ContactHistory, ContactPointAddressChangeEvent, ContactPointAddressHistory,
ContactPointAddressShare, ContactPointConsentChangeEvent,
ContactPointConsentHistory, ContactPointConsentShare,
ContactPointEmailChangeEvent, ContactPointEmailHistory, ContactPointEmailShare,
ContactPointPhoneChangeEvent, ContactPointPhoneHistory, ContactPointPhoneShare,
ContactPointTypeConsentChangeEvent, ContactPointTypeConsentHistory,
ContactPointTypeConsentShare, ContactRequestShare, ContactShare,
ContentDocumentChangeEvent, ContentDocumentHistory, ContentDocumentLink,
ContentDocumentLinkChangeEvent, ContentDocumentSubscription, ContentFolderItem,
ContentFolderLink, ContentFolderMember, ContentNote, ContentNotification,
ContentTagSubscription, ContentUserSubscription, ContentVersionChangeEvent,
ContentVersionComment, ContentVersionHistory, ContentVersionRating,
ContentWorkspace, ContentWorkspaceMember, ContentWorkspacePermission,
ContentWorkspaceSubscription, ContractChangeEvent, ContractHistory,
ContractLineItemChangeEvent, ContractLineItemHistory, ContractStatus,
Conversation, ConversationParticipant, CronJobDetail, CronTrigger, CustomBrand,
CustomBrandAsset, CustomerShare, CustomHttpHeader, DashboardComponent,
DataUseLegalBasisHistory, DataUseLegalBasisShare, DataUsePurposeHistory,
DataUsePurposeShare, DecisionTableRecordset, DeleteEvent, DocumentAttachmentMap,
Domain, DomainSite, DTRecordsetReplicaShare, EmailBounceEvent,
EmailMessageChangeEvent, EmailServicesAddress, EmailServicesFunction,
EmailTemplate, EmailTemplateChangeEvent, EngagementAttendeeChangeEvent,
EngagementAttendeeHistory, EngagementChannelTypeHistory, EngagementChannelTypeShare,
EngagementInteractionChangeEvent, EngagementInteractionHistory,
EngagementInteractionShare, EngagementInterface, EngagementTopicChangeEvent,
EngagementTopicHistory, EntitlementChangeEvent, EntitlementHistory,
EntitlementTemplate, EntityMilestoneHistory, EntitySubscription, EventChangeEvent,
EventRelationChangeEvent, EventRelayConfigChangeEvent, ExpressionSetHistory,
ExpressionSetShare, ExpressionSetVersionHistory, ExternalEventMappingShare,
FeedAttachment, FeedLike, FeedPollChoice, FeedPollVote, FeedSignal,
```

```
FieldPermissions, FieldSecurityClassification, FiscalYearSettings,
FlowInterviewLogShare, FlowInterviewShare, FlowOrchestrationEvent,
FlowOrchestrationInstanceShare, FlowOrchestrationStageInstanceShare,
FlowOrchestrationStepInstanceShare, FlowOrchestrationWorkItemShare,
FlowRecordShare, FlowRecordVersionChangeEvent, FlowTestResultShare, Folder,
Group, GroupMember, Holiday, IdeaComment, IdpEventLog, ImageHistory, ImageShare,
IncidentChangeEvent, IncidentHistory, IncidentRelatedItemChangeEvent,
IncidentRelatedItemHistory, IncidentShare, IndividualChangeEvent,
IndividualHistory, IndividualShare, KnowledgeableUser, LeadChangeEvent, LeadHistory,
LeadShare, LeadStatus, LightningExitByPageMetrics, LightningToggleMetrics,
LightningUsageByAppTypeMetrics, LightningUsageByBrowserMetrics,
LightningUsageByFlexiPageMetrics, LightningUsageByPageMetrics, ListEmailChangeEvent,
ListEmailShare, ListView, LocationChangeEvent, LocationHistory, LocationShare,
LocationTrustMeasureShare, LoginHistory, LoginIp, MacroChangeEvent,
MacroHistory, MacroInstructionChangeEvent, MacroShare, MacroUsageShare,
ManagedContentVariantChangeEvent, MessagingEndUserHistory, MessagingEndUserShare,
MessagingSessionHistory, MessagingSessionShare, MilestoneType, MLEngagementEvent,
ObjectPermissions, OpportunityChangeEvent, OpportunityContactRoleChangeEvent,
OpportunityFieldHistory, OpportunityLineItemChangeEvent, OpportunityShare,
OpportunityStage, OrderChangeEvent, OrderHistory, OrderItemChangeEvent,
OrderItemHistory, OrderShare, OrderStatus, Organization, OrgEmailAddressSecurity,
OrgWideEmailAddress, OutgoingEmail, OutgoingEmailRelation,
PackageLicense, PartnerRole, PartyConsentChangeEvent, PartyConsentHistory,
PartyConsentShare, Period, PermissionSet, PermissionSetAssignment,
PermissionSetTabSetting, Pricebook2ChangeEvent, Pricebook2History,
PricebookEntryChangeEvent, PricebookEntryHistory, PrivacyJobSessionShare,
PrivacyObjectSessionShare, PrivacyRTBFRequestHistory, PrivacyRTBFRequestShare,
PrivacySessionRecordFailureShare, ProblemChangeEvent, ProblemHistory,
ProblemIncidentChangeEvent, ProblemIncidentHistory, ProblemRelatedItemChangeEvent,
ProblemRelatedItemHistory, ProblemShare, ProcessDefinition, ProcessExceptionEvent,
ProcessExceptionShare, ProcessInstanceChangeEvent, ProcessInstanceStep,
ProcessInstanceStepChangeEvent, ProcessNode, Product2ChangeEvent, Product2History,
ProductEntitlementTemplate, Profile, ProfileSkillEndorsementHistory,
ProfileSkillHistory, ProfileSkillShare, ProfileSkillUserHistory, PromptActionShare,
PromptErrorShare, QueueSobject, QuickTextChangeEvent, QuickTextHistory,
QuickTextShare, QuickTextUsageShare, RecentlyViewed, RecommendationChangeEvent,
RecordActionHistory, RecordAlertHistory, RecordAlertShare, RecordType,
ScorecardShare, SellerHistory, SellerShare, ServiceContractChangeEvent,
ServiceContractHistory, ServiceContractShare, SetupAuditTrail, SetupEntityAccess,
SharingRecordCollectionShare, Site, SiteHistory, SiteRedirectMapping,
SocialPersonaHistory, SocialPostChangeEvent, SocialPostHistory, SocialPostShare,
SolutionHistory, SolutionStatus, StaticResource, StreamingChannelShare,
TableauHostMappingShare, TaskChangeEvent, TaskPriority, TaskStatus,
ThreatDetectionFeedback, TimelineObjectDefinitionChangeEvent, TodayGoalShare,
```

```
 Topic, TopicUserEvent, Translation, User, UserAppMenuCustomizationShare,
 UserChangeEvent, UserDefinedLabelAssignmentShare, UserDefinedLabelShare,
 UserEmailPreferredPersonShare, UserLicense, UserLogin, UserPackageLicense,
 UserPreference, UserPrioritizedRecordShare, UserProvisioningRequestShare,
 UserRole, UserShare, VideoCallChangeEvent, VideoCallParticipantChangeEvent,
 VideoCallRecordingChangeEvent, VideoCallShare, VisualforceAccessMetrics,
 VoiceCallChangeEvent, VoiceCallRecordingChangeEvent, VoiceCallShare, Vote,
 WebLink, WorkAccessShare, WorkBadgeDefinitionHistory, WorkBadgeDefinitionShare,
 WorkOrderChangeEvent, WorkOrderHistory, WorkOrderLineItemChangeEvent,
 WorkOrderLineItemHistory, WorkOrderLineItemStatus, WorkOrderShare,
 WorkOrderStatus, WorkPlanChangeEvent, WorkPlanHistory, WorkPlanShare,
 WorkPlanTemplateChangeEvent, WorkPlanTemplateEntryChangeEvent,
 WorkPlanTemplateEntryHistory, WorkPlanTemplateHistory, WorkPlanTemplateShare,
 WorkStepChangeEvent, WorkStepHistory, WorkStepStatus, WorkStepTemplateChangeEvent,
 WorkStepTemplateHistory, WorkStepTemplateShare, WorkThanksShare
```

- Les ServiceNow entités ou champs suivants ne sont pas pris en charge pour une utilisation dans le cadre d'une intégration zéro ETL avec une ServiceNow source.

```
ais_acl_overrides, ais_async_genius_result, ais_async_request,
 ais_connection, ais_genius_result_configuration_parameters,
 ais_partition_health, ais_partition_health_response, ais_publish_history,
 ais_relevancy_training_execution, ais_relevancy_training_staging,
 ais_search_profile_relevancy_model, catalog_draft_entities, clone_log,
 clone_log0000, clone_log0001, clone_log0002, clone_log0003, clone_log0004,
 clone_log0005, clone_log0006, clone_log0007, cmdb_ie_context, cmdb_ie_log,
 cmdb_ie_run, cmdb_ire_partial_payloads_index, cmdb_qb_result_base
$par1, cmdb$par1, discovery_log, discovery_log0000, discovery_log0001,
 discovery_log0002, discovery_log0003, discovery_log0004, discovery_log0005,
 discovery_log0006, discovery_log0007, ecc_agent_log, entitlement_data,
 entl_subscription_map, gs_entitlement_plugin_mapping, ih_transaction_exclusion,
 import_log, import_log0000, import_log0001, import_log0002, import_log0003,
 import_log0004, import_log0005, import_log0006, import_log0007, jrobin_archive,
 jrobin_database, jrobin_datasource, jrobin_definition, jrobin_graph,
 jrobin_graph_line, jrobin_graph_set, jrobin_graph_set_member, jrobin_shard,
 jrobin_shard_location, license_role_discovery_run, logger_configuration_validation,
 m2m_analytics_event_logger, m2m_user_consent_info, ml_artifact_object_store,
 np$sys_gen_ai_filter_sample, np$sys_ui_element, np$sys_ui_list_element,
 one_api_service_plan_feature_invocation, one_api_service_plan_invocation,
 open_nlu_predict_log, open_nlu_predict_log0000, open_nlu_predict_log0001,
 open_nlu_predict_log0002, open_nlu_predict_log0003, open_nlu_predict_log0004,
 open_nlu_predict_log0005, open_nlu_predict_log0006, open_nlu_predict_log0007,
 pa_diagnostic_log, pa_diagnostic_log0000, pa_diagnostic_log0001,
```

```
pa_diagnostic_log0002, pa_diagnostic_log0003, pa_diagnostic_log0004,
pa_diagnostic_log0005, pa_diagnostic_log0006, pa_diagnostic_log0007, pa_favorites,
pa_job_log_rows, pa_job_log_rows0000, pa_job_log_rows0001, pa_job_log_rows0002,
pa_job_log_rows0003, pa_job_log_rows0004, pa_job_log_rows0005, pa_job_log_rows0006,
pa_job_log_rows0007, pa_migration_ignored_scores, pa_scores_l1, pa_scores_l2,
pa_scores_migration_groups, par_dashboard_conversion_backup, promin_log,
promin_log0000, promin_log0001, promin_log0002, promin_log0003, promin_log0004,
promin_log0005, promin_log0006, promin_log0007, promin_request_object,
proposed_change_verification_log, proposed_change_verification_log0000,
proposed_change_verification_log0001, proposed_change_verification_log0002,
proposed_change_verification_log0003, proposed_change_verification_log0004,
proposed_change_verification_log0005, proposed_change_verification_log0006,
proposed_change_verification_log0007, protected_table_log, protected_table_log0000,
protected_table_log0001, protected_table_log0002, protected_table_log0003,
protected_table_log0004, protected_table_log0005, protected_table_log0006,
protected_table_log0007, pwd_history, qb_query_results, scan_log,
scan_log0000, scan_log0001, scan_log0002, scan_log0003, scan_log0004,
scan_log0005, scan_log0006, scan_log0007, schema_validator_error,
sla_repair_log_entry, sla_repair_log_entry0000, sla_repair_log_entry0001,
sla_repair_log_entry0002, sla_repair_log_entry0003, sla_repair_log_entry0004,
sla_repair_log_entry0005, sla_repair_log_entry0006, sla_repair_log_entry0007,
sla_repair_log_message, sla_repair_log_message0000, sla_repair_log_message0001,
sla_repair_log_message0002, sla_repair_log_message0003, sla_repair_log_message0004,
sla_repair_log_message0005, sla_repair_log_message0006, sla_repair_log_message0007,
sn_bm_client_activity, sn_ci_analytics_st_actionable_notifs,
sn_ci_analytics_st_conv_completion_by_cat, sn_ci_analytics_st_conv_dynamic_property,
sn_ci_analytics_st_conversation, sn_ci_analytics_st_count_by_date,
sn_ci_analytics_st_event_occurrence, sn_ci_analytics_st_event_property_value_trend,
sn_ci_analytics_st_issue_auto_resolution, sn_ci_analytics_st_no_clicks,
sn_ci_analytics_st_no_results, sn_ci_analytics_st_property_summary_by_event,
sn_ci_analytics_st_session_count_per_locale, sn_ci_analytics_st_session_duration,
sn_ci_analytics_st_spokes_usage, sn_ci_analytics_st_topic_execution_stats,
sn_ci_analytics_st_topic_occurrence, sn_ci_analytics_st_trending_content,
sn_ci_analytics_st_trending_queries, sn_ci_analytics_st_users,
sn_cs_plugin_signatures, sn_cs_telemetry_log, sn_dfc_application, sn_dfc_product,
sn_employee_position, sn_entitlement_st_subscription_application_family,
sn_entitlement_st_subscription_application_users, sn_hr_sp_st_relevant_for_you,
sn_instance_clone_log, sn_instance_clone_log0000, sn_instance_clone_log0001,
sn_instance_clone_log0002, sn_instance_clone_log0003, sn_instance_clone_log0004,
sn_instance_clone_log0005, sn_instance_clone_log0006, sn_instance_clone_log0007,
sn_km_mr_st_kb_knowledge, sn_me_st_topic, sn_rf_conditional_definition,
sn_rf_evaluation_type, sn_rf_evaluation_type_input, sn_rf_recommendation_action,
sn_rf_recommendation_experience, sn_rf_recommendation_history,
sn_rf_recommendation_rule, sn_rf_record_display_configuration,
```

```
sn_rf_trend_definition, sn_sub_man_st_account_level_entitlement,
sn_sub_man_st_gen_ai_metadata, sn_sub_man_st_instance_used_assist_count,
sn_sub_man_st_now_assist_creator_instances, sn_sub_man_st_now_assists_aggregate,
sn_sub_man_st_subscribed_groups, sn_sub_man_st_subscription_insights,
sn_sub_man_st_subscription_license_detail_metric,
sn_sub_man_st_unallocated_group_recommendation,
sn_sub_man_st_unconfirmed_user_group, sn_wn_user_app_activity,
sn_wn_user_content_activity, snc_monitorable_item,
snpar_sched_export_v_scheduled_export_visualization, spotlight,
spotlight_audit, spotlight_copy_log_row, spotlight_copy_log_row0000,
spotlight_copy_log_row0001, spotlight_copy_log_row0002, spotlight_copy_log_row0003,
spotlight_copy_log_row0004, spotlight_copy_log_row0005, spotlight_copy_log_row0006,
spotlight_copy_log_row0007, spotlight_job_log_row, spotlight_job_log_row0000,
spotlight_job_log_row0001, spotlight_job_log_row0002, spotlight_job_log_row0003,
spotlight_job_log_row0004, spotlight_job_log_row0005, spotlight_job_log_row0006,
spotlight_job_log_row0007, st_dfc_performance_metric, st_license_detail_metric,
st_on_call_hour, st_sc_wizard_question, st_sys_catalog_items_and_variable_sets,
st_sys_design_system_icon, subscription_instance_stats,
svc_container_config, svc_environment_config, svc_layer_config,
svc_model_assoc_ci, svc_model_checkpoint_attr, svc_model_obj_cluster,
svc_model_obj_constraint, svc_model_obj_deployable, svc_model_obj_element,
svc_model_obj_impact, svc_model_obj_impactrule, svc_model_obj_package,
svc_model_obj_path, svc_model_obj_relation, svc_model_obj_service,
sys_administrative_script_transaction, sys_amb_message, sys_amb_processor,
sys_analytics_batch_state, sys_analytics_config, sys_analytics_data_points_error,
sys_analytics_event, sys_analytics_logger, sys_analytics_logger_field,
sys_app_payload_loader_rule, sys_app_payload_unloader_rule, sys_app_scan_payload,
sys_app_scan_variable, sys_app_scan_variable_type, sys_archive_destroy_log,
sys_archive_destroy_run, sys_archive_log, sys_archive_run, sys_atf_transaction_log,
sys_attachment_doc, sys_attachment_doc_v2, sys_attachment_soft_deleted, sys_audit,
sys_audit_relation, sys_auth_policy_api_allowed, sys_aw_registered_scripting_modal,
sys_cache_flush, sys_data_egress_source, sys_dm_delete_count,
sys_export_set_log, sys_export_set_log0000, sys_export_set_log0001,
sys_export_set_log0002, sys_export_set_log0003, sys_export_set_log0004,
sys_export_set_log0005, sys_export_set_log0006, sys_export_set_log0007,
sys_flow_compiled_flow, sys_flow_compiled_flow_chunk, sys_flow_context_chunk,
sys_flow_context_chunk_archive, sys_flow_context_inputs_chunk,
sys_flow_execution_history, sys_flow_log, sys_flow_log0000, sys_flow_log0001,
sys_flow_log0002, sys_flow_log0003, sys_flow_plan_context_binding,
sys_flow_report_doc, sys_flow_report_doc_chunk, sys_flow_report_doc_chunk_archive,
sys_flow_runtime_state_chunk, sys_flow_runtime_value_chunk,
sys_flow_subflow_plan_chunk, sys_flow_trigger_plan_chunk, sys_flow_val_listener,
sys_flow_value, sys_flow_value_chunk, sys_gen_ai_config_example,
sys_gen_ai_feature_mapping, sys_gen_ai_strategy_mapping, sys_gen_ai_usage_log,
```

```
sys_generative_ai_capability_definition, sys_generative_ai_log,
sys_generative_ai_response_validator, sys_generative_ai_validator,
sys_geo_routing, sys_geo_routing_config, sys_hop_token, sys_hub_action_plan_chunk,
sys_hub_snapshot_chunk, sys_journal_field, sys_journal_field_edit, sys_json_chunk,
sys_kaa_policy, sys_kaa_subidentity_assertion, sys_kaa_user_policy_mapping,
sys_mapplication, sys_mass_encryption_job, sys_notification_execution_log,
sys_notification_execution_log0000, sys_notification_execution_log0001,
sys_notification_execution_log0002, sys_notification_execution_log0003,
sys_notification_execution_log0004, sys_notification_execution_log0005,
sys_notification_execution_log0006, sys_notification_execution_log0007,
sys_nowmq_message, sys_nowmq_provider_param_definition, sys_orchestrator_action,
sys_pd_asset_configuration, sys_pd_context_chunk, sys_pd_context_log,
sys_pd_snapshot_chunk, sys_pd_trigger_license, sys_processing_framework_job,
sys_query_index_hint, sys_query_rewrite, sys_query_string_log,
sys_replication_queue, sys_replication_queue0, sys_replication_queue1,
sys_replication_queue2, sys_replication_queue3, sys_replication_queue4,
sys_replication_queue5, sys_replication_queue6, sys_replication_queue7,
sys_request_performance, sys_rollback_blacklisted, sys_rollback_conflict,
sys_rollback_incremental, sys_rollback_log, sys_rollback_log0000,
sys_rollback_log0001, sys_rollback_log0002, sys_rollback_log0003,
sys_rollback_log0004, sys_rollback_log0005, sys_rollback_log0006,
sys_rollback_log0007, sys_rollback_run, sys_rollback_schema_change,
sys_rollback_schema_conflict, sys_rollback_sequence, sys_scheduler_assignment,
sys_scheduler_memory_pressure_job_log, sys_script_adapter_rule,
sys_script_batch_adapter_rule, sys_search_source_filter, sys_service_authentication,
sys_signing_job, sys_suggestion_reader, sys_sync_history_review, sys_trend,
sys_unreferenced_preview, sys_unreferenced_record_rule, sys_upgrade_manifest,
sys_upgrade_state, sys_ux_asset_cache_buster, sys_ux_lib_component_prop,
sys_ux_lib_presource, sys_ux_page_action, sys_ux_page_action_binding,
sysevent_queue_runtime, syslog, syslog_app_scope0000, syslog_app_scope0001,
syslog_app_scope0002, syslog_app_scope0003, syslog_app_scope0004,
syslog_app_scope0005, syslog_app_scope0006, syslog_app_scope0007, syslog_email0000,
syslog_email0001, syslog_email0002, syslog_email0003, syslog_email0004,
syslog_email0005, syslog_email0006, syslog_email0007, syslog_transaction,
syslog_transaction0000, syslog_transaction0001, syslog_transaction0002,
syslog_transaction0003, syslog_transaction0004, syslog_transaction0005,
syslog_transaction0006, syslog_transaction0007, syslog0000, syslog0001,
syslog0002, syslog0003, syslog0004, syslog0005, syslog0006, syslog0007,
ts_attachment, ts_c_1_0, ts_c_1_1, ts_c_1_2, ts_c_1_3, ts_c_1_4, ts_c_1_5,
ts_c_1_6, ts_c_1_7, ts_c_1_8, ts_c_1_9, ts_c_10_0, ts_c_10_1, ts_c_10_2,
ts_c_10_3, ts_c_10_4, ts_c_10_5, ts_c_10_6, ts_c_10_7, ts_c_10_8, ts_c_10_9,
ts_c_11_0, ts_c_11_1, ts_c_11_2, ts_c_11_3, ts_c_11_4, ts_c_11_5, ts_c_11_6,
ts_c_11_7, ts_c_11_8, ts_c_11_9, ts_c_2_0, ts_c_2_1, ts_c_2_2, ts_c_2_3,
ts_c_2_4, ts_c_2_5, ts_c_2_6, ts_c_2_7, ts_c_2_8, ts_c_2_9, ts_c_3_0,
```

```
ts_c_3_1, ts_c_3_2, ts_c_3_3, ts_c_3_4, ts_c_3_5, ts_c_3_6, ts_c_3_7, ts_c_3_8,
ts_c_3_9, ts_c_39_0, ts_c_39_1, ts_c_39_2, ts_c_39_3, ts_c_39_4, ts_c_39_5,
ts_c_39_6, ts_c_39_7, ts_c_39_8, ts_c_39_9, ts_c_40_0, ts_c_40_1, ts_c_40_2,
ts_c_40_3, ts_c_40_4, ts_c_40_5, ts_c_40_6, ts_c_40_7, ts_c_40_8, ts_c_40_9,
ts_c_41_0, ts_c_41_1, ts_c_41_2, ts_c_41_3, ts_c_41_4, ts_c_41_5, ts_c_41_6,
ts_c_41_7, ts_c_41_8, ts_c_41_9, ts_c_42_0, ts_c_42_1, ts_c_42_2, ts_c_42_3,
ts_c_42_4, ts_c_42_5, ts_c_42_6, ts_c_42_7, ts_c_42_8, ts_c_42_9, ts_c_43_0,
ts_c_43_1, ts_c_43_2, ts_c_43_3, ts_c_43_4, ts_c_43_5, ts_c_43_6, ts_c_43_7,
ts_c_43_8, ts_c_43_9, ts_c_44_0, ts_c_44_1, ts_c_44_2, ts_c_44_3, ts_c_44_4,
ts_c_44_5, ts_c_44_6, ts_c_44_7, ts_c_44_8, ts_c_44_9, ts_c_45_0, ts_c_45_1,
ts_c_45_2, ts_c_45_3, ts_c_45_4, ts_c_45_5, ts_c_45_6, ts_c_45_7, ts_c_45_8,
ts_c_45_9, ts_c_6_0, ts_c_6_1, ts_c_6_2, ts_c_6_3, ts_c_6_4, ts_c_6_5,
ts_c_6_6, ts_c_6_7, ts_c_6_8, ts_c_6_9, ts_c_7_0, ts_c_7_1, ts_c_7_2, ts_c_7_3,
ts_c_7_4, ts_c_7_5, ts_c_7_6, ts_c_7_7, ts_c_7_8, ts_c_7_9, ts_c_8_0, ts_c_8_1,
ts_c_8_2, ts_c_8_3, ts_c_8_4, ts_c_8_5, ts_c_8_6, ts_c_8_7, ts_c_8_8, ts_c_8_9,
ts_c_attachment, ts_chain, ts_deleted_doc, ts_document, ts_field, ts_index_stats,
ts_phrase, ts_search_stats, ts_v4_attachment, ts_word, ua_app_metadata,
ua_audit_stats, ua_extra_page, ua_monitor_property, ua_monitor_property_audit,
ua_shared_service, ua_sn_table_inventory, ua_sp_known_bot, ua_upload_log,
v_ais_result_improvement_rule_condition_builder_values, v_cluster_nodes,
v_cxs_search_resource, v_db_index, v_db_trigger, v_file_load_order,
v_interaction_context, v_iostats, v_mysql_proclist, v_mysql_status,
v_mysql_variables, v_on_call_report_cache, v_pa_par_combined_dashboard,
v_pd_activity_condition_to_run, v_pd_activity_start_rule_with_condition,
v_pd_lane_condition_to_run, v_sql_debug, v_st_kb_category, v_st_kb_most_viewed,
v_st_kb_recently_viewed, v_st_km_genai_mra_similar_task, v_st_popular_item,
v_st_recent_item, v_st_sc_cat_item, v_st_sc_catalog, v_st_sc_category,
validator_run_summary, vtb_card_history, wf_log, wf_log0000, wf_log0001,
wf_log0002, wf_log0003, wf_log0004, wf_log0005, wf_log0006, wf_log0007,
spotlight_criteria, pa_snapshots, pa_widget_indicators, pa_widgets,
sn_cim_register, global, gsw_change_log, gsw_content, gsw_content_group,
gsw_content_information, gsw_status_of_content, multi_factor_browser_fingerprint,
multi_factor_criteria, pa_filters, password_policy, plan_execution,
plan_mysql, plan_oracle, plan_postgres, sc_rest_api_without_access_policy,
sn_actsub_activity, sn_actsub_activity_fanout, sn_actsub_activity_stream,
sn_actsub_activity_type, sn_actsub_atype_attributes, sn_actsub_atype_notif_pref,
sn_actsub_module, sn_actsub_notif_object, sn_actsub_subobject_stream,
sn_actsub_subscribable_object, sn_actsub_subscription_notif_pref,
sn_actsub_user_stream, sn_appclient_store_outbound_http_quota,
sn_appcreator_app_template, sn_critical_update, sn_docker_spoke_images,
sn_employee_app, sn_employee_app_access, sn_employee_app_access_criteria,
sn_entitlement_genai_assist_counts, sn_entitlement_genai_creator_user_counts,
sn_entitlement_genai_creator_users, sn_mif_instance, sn_mif_sync_data,
sn_mif_sync_status, sn_mif_table_registration, sn_mif_trust_config,
```

```
sn_vsc_best_practice_configurations, sn_vsc_best_practice_goals,
sn_vsc_changed_hardening_settings, sn_vsc_changed_scan_findings,
sn_vsc_check_security_area, sn_vsc_elevation_event, sn_vsc_event,
sn_vsc_export_event, sn_vsc_export_setting, sn_vsc_harc_compliance_status_lookup,
sn_vsc_hardening_compliance_scores, sn_vsc_impersonation_event,
sn_vsc_instance_hardening_settings, sn_vsc_login_event,
sn_vsc_scan_comparisons, sn_vsc_scan_summary, sn_vsc_security_check_categories,
sn_vsc_security_check_configurations, sn_vsc_security_configuration_groups,
sn_vsc_security_privacy_capabilities, sn_vsc_updated_settings,
sn_vsc_user_comparisons, sys_app_hash_inventory, sys_coalesce_strategy_deferred,
sys_flow_secure_data, sys_formula_function, sys_geocoding_request,
sys_global_file_hash, sys_import_set_row, sys_index, sys_index_explain,
sys_installation_schedule, sys_installation_schedule_item, sys_offline_app,
sys_package, sys_package_dependency_item, sys_package_dependency_m2m,
sys_plugins, sys_querystat, sys_reap_package, sys_scoped_plugin,
sys_stage_storage_alias, sys_storage_alias, sys_storage_table_alias, sys_store_app,
sys_table_partition, sys_upgrade_history_log, sys_user_public_credential,
sys_webauthn_authentication_request, sys_webauthn_registration_request,
syslog_app_scope, syslog_email, syslog_page_timing, ua_instance_state_config,
v_expression_cache, v_private_cache, v_shared_cache, sys_amb_message0002,
sys_amb_message0004, sys_amb_message0005, sys_metadata, v_par_unified_report_viz
```

AWS Glue Qualité des données

AWS Glue La qualité des données vous permet de mesurer et de surveiller la qualité de vos données afin de prendre de bonnes décisions commerciales. Construit sur le DeeQu framework open source, AWS Glue Data Quality fournit une expérience gérée et sans serveur. AWS Glue Data Quality fonctionne avec le langage DQDL (Data Quality Definition Language), qui est un langage spécifique au domaine que vous utilisez pour définir des règles de qualité des données. Pour en savoir plus sur le langage DQDL et les types de règles pris en charge, consultez Référence DQDL (Data Quality Definition Language).

Pour plus d'informations sur les produits et les tarifs, consultez la page de service relative à AWS Glue Data Quality.

Avantages et fonctionnalités clés

Les avantages et les principales caractéristiques de la qualité AWS Glue des données sont les suivants :

- Sans serveur : il n'y a pas d'installation, de correctifs ou de maintenance.
- Démarrez rapidement — AWS Glue Data Quality analyse rapidement vos données et crée des règles de qualité pour vous. Vous pouvez commencer en deux clics : « Créer des règles de qualité des données → Recommander des règles ».
- Détectez les problèmes de qualité des données : utilisez l'apprentissage automatique (ML) pour détecter les anomalies et les problèmes de qualité des hard-to-detect données.
- Improvisez vos règles : avec plus de 25 règles out-of-the-box DQ à partir de laquelle vous pouvez commencer, vous pouvez créer des règles adaptées à vos besoins spécifiques.
- Évaluer la qualité et prendre des décisions métier en toute confiance : une fois les règles évaluées, vous obtenez un score de qualité des données qui vous donne une vue d'ensemble de l'état de vos données. Utilisez le score de qualité des données pour prendre des décisions métier en toute confiance.
- Concentrez-vous sur les données erronées : la qualité AWS Glue des données vous aide à identifier les enregistrements exacts qui ont entraîné une baisse de vos scores de qualité. Identifiez-les facilement, mettez-les en quarantaine et corrigez-les.
- Payez au fur et à mesure : aucune licence annuelle n'est nécessaire pour utiliser AWS Glue Data Quality.

- Pas de blocage : AWS Glue Data Quality repose sur l'open source DeeQu, ce qui vous permet de conserver les règles que vous créez dans un langage ouvert.
- Contrôles de qualité des données : vous pouvez appliquer des contrôles de qualité des données sur Data Catalog et sur les pipelines ETL AWS Glue, ce qui vous permet de gérer la qualité des données au repos et en transit.
- Détection de la qualité des données basée sur le ML : utilisez l'apprentissage automatique (ML) pour détecter les anomalies et les problèmes de qualité hard-to-detect des données.
- Langage ouvert pour exprimer les règles : garantit que les règles de qualité des données sont créées de manière cohérente et simple. Les utilisateurs professionnels peuvent facilement exprimer les règles de qualité des données dans un langage clair qu’ils peuvent comprendre. Pour les ingénieurs, ce langage offre la flexibilité nécessaire pour générer du code, mettre en œuvre un contrôle de version cohérent et automatiser les déploiements.

Comment ça marche

Il existe deux points d'entrée pour la qualité AWS Glue des données : les jobs AWS Glue ETL AWS Glue Data Catalog et les jobs. Cette section fournit un aperçu des cas d'utilisation et des AWS Glue fonctionnalités pris en charge par chaque point d'entrée.

Qualité des données pour AWS Glue Data Catalog

AWS Glue Data Quality évalue les objets stockés dans le. AWS Glue Data Catalog Cela permet aux non-codeurs de configurer facilement des règles de qualité des données. Ces personas incluent les gestionnaires de données et des analystes métier.

Vous pouvez choisir cette option pour les cas d'utilisation suivants :

- Vous souhaitez effectuer des tâches de qualité des données sur des jeux de données que vous avez déjà catalogués dans AWS Glue Data Catalog.
- Vous travaillez sur la gouvernance des données et avez besoin d'identifier ou d'évaluer en permanence les problèmes de qualité des données dans votre lac de données.

Vous pouvez gérer la qualité des données du catalogue de données à l'aide des interfaces suivantes :

- La console AWS Glue de gestion
- AWS Glue APIs

Pour commencer à utiliser AWS Glue Data Quality for the AWS Glue Data Catalog seePremiers pas avec AWS Glue Data Quality pour le Data Catalog.

Qualité des données pour les tâches AWS Glue ETL

AWS Glue La qualité des données pour les tâches AWS Glue ETL vous permet d'effectuer des tâches proactives de qualité des données. Les tâches proactives vous aident à identifier et à filtrer les données défectueuses avant de charger un jeu de données dans votre lac de données.

Vidéo : Présentation de la qualité AWS Glue des données pour les pipelines ETL

Vous pouvez choisir la qualité des données pour les tâches ETL dans les cas d'utilisation suivants :

- Vous souhaitez intégrer des tâches de qualité des données dans vos tâches ETL
- Vous souhaitez écrire du code qui définit les tâches de qualité des données dans les scripts ETL
- Vous souhaitez gérer la qualité des données qui circulent dans vos pipelines de données visuels

Vous pouvez gérer la qualité des données pour les tâches ETL à l'aide des interfaces suivantes :

- AWS Glue Studio, AWS Glue Studio carnets de notes et sessions AWS Glue interactives
- AWS Glue bibliothèques pour les scripts ETL
- AWS Glue APIs

Pour commencer à utiliser la qualité des données pour les tâches ETL, consultez Tutorial: Getting started with Data Quality dans le Guide de l'utilisateur AWS Glue Studio .

Comparaison entre la qualité des données du catalogue de données et la qualité des données des tâches ETL

Ce tableau fournit un aperçu des fonctionnalités prises en charge par chaque point d'entrée pour AWS Glue Data Quality.

Fonctionnalité	Qualité des données pour le catalogue de données	Qualité des données pour les tâches ETL
Sources de données	Amazon S3 Amazon Redshift, sources JDBC compatibles avec le catalogue de données et formats de lacs de données transactionnels tels qu'Apache Iceberg, Apache Hudi et Delta Lake. AWS Lake Formation les formats OTF gérés sont également pris en charge avec certaines limitations. Amazon Athena les vues cataloguées dans le catalogue de AWS Glue données ne sont pas prises en charge. Veuillez consulter Types de sources pris en charge.	Toutes les sources de données sont prises en charge par AWS Glue, y compris les connecteurs personnalisés et les connecteurs tiers.
Recommandations règles de la qualité des données	Pris en charge	Non pris en charge
Rédiger et appliquer les règles DQDL	Pris en charge	Pris en charge
Auto scaling (Mise à l'échelle automatique)	Non pris en charge	Pris en charge
AWS Glue Support flexible	Non pris en charge	Pris en charge
Planification	Pris en charge lors de l'évaluation des règles de la qualité des données et via Step Functions.	Pris en charge lors de l'utilisation des Step Functions et des flux de travail.
Identification des enregistrements ayant échoué aux	Non pris en charge	Pris en charge

Fonctionnalité	Qualité des données pour le catalogue de données	Qualité des données pour les tâches ETL
contrôles de qualité des données		
Intégration à Amazon EventBridge	Pris en charge	Pris en charge
Intégration à AWS Cloudwatch	Pris en charge	Pris en charge
Écrire les résultats de la qualité des données dans Amazon S3	Pris en charge	Pris en charge
Qualité de données incrément ielle	Pris en charge par le pushdown de prédicats	Pris en charge par les AWS Glue signets
AWS CloudFormation soutien	Pris en charge	Pris en charge
Détection des anomalies basée sur le ML	Non pris en charge	Pris en charge
Règles dynamiques	Non pris en charge	Pris en charge

Considérations

Tenez compte des éléments suivants avant d'utiliser AWS Glue Data Quality :

- Les règles de qualité des données ne peuvent pas évaluer les sources de données imbriquées ou de type liste. Consultez Aplatissement de structs imbriqués.

Terminologie

La liste suivante définit les termes relatifs à la qualité AWS Glue des données.

DQDL (Data Quality Definition Language)

Langage spécifique à un domaine que vous pouvez utiliser pour rédiger des règles de qualité AWS Glue des données.

Pour en savoir plus sur DQDL, consultez le guide Référence DQDL (Data Quality Definition Language).

qualité des données

Décrit dans quelle mesure un ensemble de données répond à son objectif spécifique. AWS Glue La qualité des données évalue les règles par rapport à un ensemble de données afin de mesurer la qualité des données. Chaque règle vérifie des caractéristiques particulières comme l'actualisation ou l'intégrité des données. Pour quantifier la qualité des données, vous pouvez utiliser un score de qualité des données.

score de qualité des données

Pourcentage de règles de qualité des données qui sont satisfaites (aboutissent à un résultat vrai) lorsque vous évaluez un ensemble de règles avec AWS Glue Data Quality.

règle

Expression DQDL qui recherche une caractéristique spécifique dans vos données et renvoie une valeur booléenne. Pour de plus amples informations, veuillez consulter Structure des règles.

analyseur

Expression DQDL qui rassemble des statistiques de données. Un analyseur collecte des statistiques de données qui peuvent être utilisées par les algorithmes de machine learning pour détecter les anomalies et les problèmes de qualité hard-to-detect des données au fil du temps.

jeu de règles

AWS Glue Ressource qui comprend un ensemble de règles de qualité des données. Un jeu de règles doit être associé à une table dans AWS Glue Data Catalog. Lorsque vous enregistrez un jeu de règles, AWS Glue lui attribue un Amazon Resource Name (ARN).

score de qualité des données

Pourcentage de règles de qualité des données qui sont respectées (génèrent « true ») lorsque vous évaluez un jeu de règles avec AWS Glue Data Quality.

observation

Un aperçu non confirmé généré par AWS Glue en analysant les statistiques de données rassemblées à partir de règles et d’analyseurs au fil du temps.

Restrictions

AWS Glue Limites du service de qualité des données :

- Un ensemble de règles peut contenir 2 000 règles. Si vos ensembles de règles sont plus importants, nous vous recommandons de les diviser en plusieurs ensembles de règles.
- La taille de l’ensemble de règles est de 65 Ko. Si vos ensembles de règles sont plus importants, nous vous recommandons de les diviser en plusieurs ensembles de règles.
- AWS Glue Data Quality collecte des statistiques lorsque vous créez une règle ou un analyseur. Le stockage de ces statistiques est gratuit. Cependant, il existe une limite de 100 000 statistiques par compte, et ces statistiques seront conservées pendant un maximum de deux ans.

Notes de mise à jour relatives à la qualité AWS Glue des données

Cette rubrique décrit les fonctionnalités introduites dans AWS Glue Data Quality.

Disponibilité générale : nouvelles fonctionnalités

Les nouvelles fonctionnalités suivantes sont disponibles avec la disponibilité générale de AWS Glue Data Quality :

- La capacité d'identifier les enregistrements ayant échoué aux contrôles de qualité des données est désormais prise en charge dans AWS Glue Studio
- Nouveaux types de règles de qualité des données tels que la validation de l'intégrité référentielle des données entre deux jeux de données, la comparaison des données entre deux jeux de données et les vérifications du type de données.
- Expérience utilisateur améliorée dans AWS Glue Data Catalog
- Prise en charge d'Apache Iceberg, d'Apache Hudi et de Delta Lake
- Prise en charge d'Amazon Redshift
- Notification simplifiée avec Amazon EventBridge

- AWS CloudFormation support pour la création d'ensembles de règles
- Améliorations des performances : option de mise en cache dans ETL et AWS Glue Studio pour des performances plus rapides lors de l'évaluation de la qualité des données

27 novembre 2023 (aperçu)

- Les fonctionnalités de détection d'anomalies basées sur le ML sont désormais disponibles dans AWS Glue ETL et AWS Glue Studio. Grâce à cela, vous pouvez désormais détecter les anomalies et les problèmes de qualité des hard-to-detect données
- Les règles dynamiques vous permettent de fournir des seuils dynamiques (ex :`RowCount > avg(last(10))`)

12 mars 2024

- Améliorations DQDL
  - Prise en charge de mots clés tels que NULL, BLANKS, WHITESPACES_ONLY
  - Options permettant de spécifier la manière dont la qualité AWS Glue des données doit gérer les règles composites
  - ColumnValues le type de règle n'autorisera pas le transfert de valeurs NULL lors des comparaisons
  - Prise en charge de l'opérateur NOT dans DQDL

26 juin 2024

- Améliorations DQDL
  - DQDL prend désormais en charge la clause Where afin que vous puissiez filtrer les données avant d'appliquer les règles DQ.

7 août 2024

- La détection des anomalies et les règles dynamiques sont désormais généralement disponibles

22 novembre 2024

- Les règles composites complexes vous permettent de créer des règles métier plus complexes avec une prise en charge imbriquée.
- Nouveaux types de règles pour gérer la qualité des données de vos fichiers
  - FileFreshness
  - FileSize
  - FileUniqueness
  - FileMatch
- Contrôles de qualité des données par défaut dans les tâches ETL visuel

6 déc. 2024

- AWS Glue Data Quality prend désormais en charge Amazon SageMaker AI LakeHouse les tables et les tables Iceberg, Delta et HUDI AWS Lake Formation gérées dans AWS Glue ETL 5.0.

7 juillet 2025

- AWS Glue Qualité des données ; prend désormais en charge les tables Amazon S3, RMS, Lakehouse et les tables Iceberg AWS Lake Formation gérées dans Data Catalog. AWS Glue

21 novembre 2025

- AWS Glue Data Quality prend désormais en charge l'étiquetage des règles pour améliorer les rapports. Vous pouvez organiser et analyser les résultats de qualité des données de manière plus efficace en interrogeant les résultats à l'aide d'étiquettes spécifiques afin d'identifier les règles défaillantes au sein de catégories particulières, de compter les résultats des règles par équipe ou par domaine et de créer des rapports ciblés pour les différentes parties prenantes. Pour de plus amples informations, veuillez consulter Étiquettes.
- AWS Glue Data Quality prend désormais en charge les constantes dans le DQDL, ce qui vous permet de définir des valeurs constantes et de les référencer tout au long de votre script. Cela permet d'éviter les problèmes liés aux limites de taille des requêtes lors de l'utilisation d'instructions SQL volumineuses. Pour de plus amples informations, veuillez consulter Constantes.

Détection d'anomalies dans Qualité des données d'AWS Glue

Arguments en faveur du machine learning pour la qualité des données

Les ingénieurs gèrent des centaines de pipelines de données simultanément. Chaque pipeline peut extraire des données de différentes sources et les charger dans le lac de données ou d'autres référentiels de données. Pour garantir la fourniture de données de haute qualité pour la prise de décision, ils établissent des règles de qualité des données. Ces règles évaluent les données sur la base de critères fixes reflétant l'état actuel de l'activité. Cependant, lorsque l'environnement métier change, les propriétés des données changent, ce qui rend ces critères fixes obsolètes et entraîne une mauvaise qualité des données.

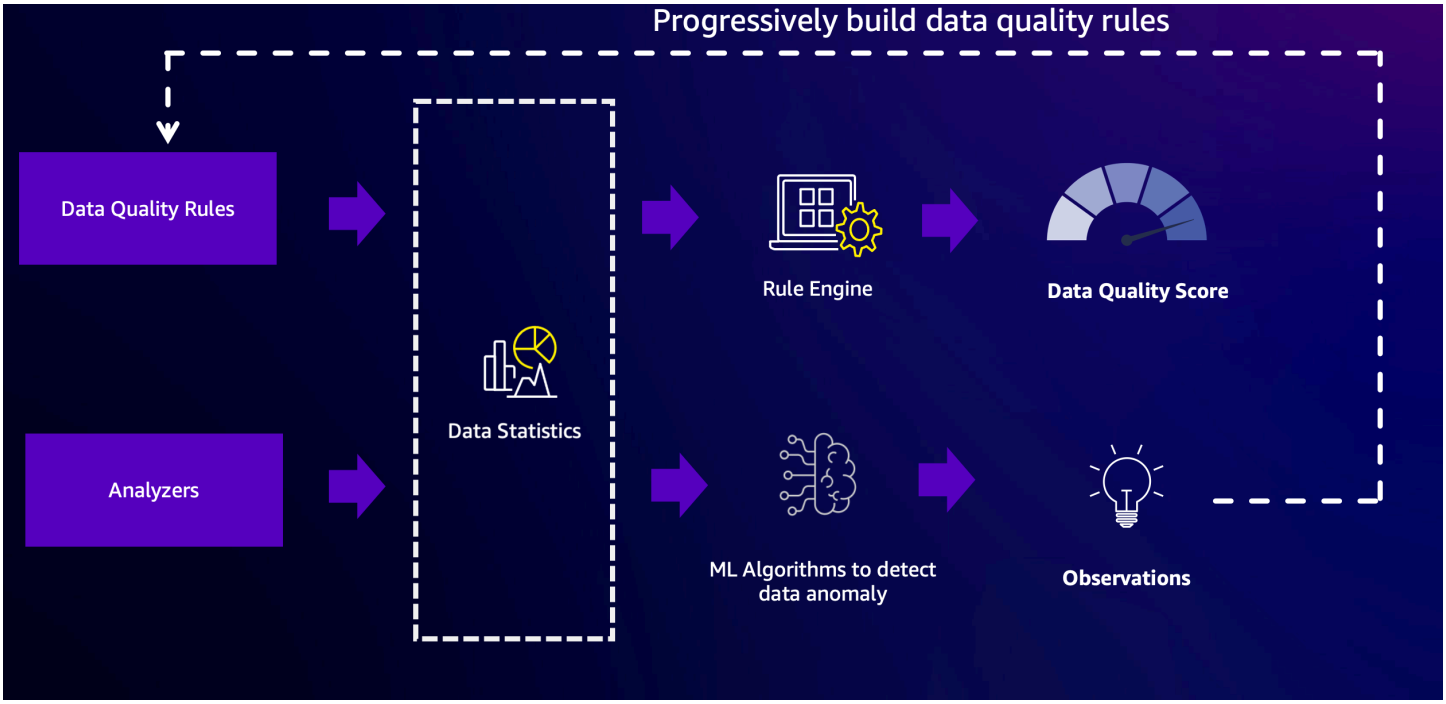
Par exemple, un ingénieur des données d'une entreprise de vente au détail a établi une règle qui valide que les ventes quotidiennes doivent dépasser un one-million-dollar certain seuil. Après quelques mois, les ventes quotidiennes ont dépassé les deux millions de dollars, rendant ainsi le seuil obsolète. L'ingénieur des données n'a pas pu mettre à jour les règles pour refléter les derniers seuils en raison de l'absence de notification et des efforts nécessaires pour analyser et mettre à jour manuellement la règle. Plus tard dans le mois, les utilisateurs métier ont remarqué une baisse de 25 % de leurs ventes. Après des heures d'enquête, les ingénieurs de données ont découvert qu'un pipeline ETL chargé d'extraire les données de certains magasins avait échoué sans générer d'erreurs. La règle dont les seuils étaient obsolètes a continué de fonctionner correctement sans que ce problème ne soit détecté.

Par ailleurs, des alertes proactives capables de détecter ces anomalies auraient pu permettre aux utilisateurs de repérer ce problème. En outre, le suivi de la saisonnalité dans l'activité peut mettre en évidence d'importants problèmes de qualité des données. Par exemple, les ventes au détail peuvent être les plus élevées le week-end et pendant les fêtes de fin d'année, alors qu'elles sont relativement faibles en semaine. Toute divergence par rapport à ce schéma peut indiquer des problèmes de qualité des données ou des changements dans le contexte métier. Les règles de qualité des données ne peuvent pas détecter les modèles saisonniers, car cela nécessite des algorithmes avancés capables de tirer des leçons des modèles précédents pour saisir la saisonnalité afin de détecter les écarts.

Enfin, les utilisateurs estiment qu'il est compliqué de créer et de maintenir des règles en raison de la nature technique du processus de création de règles et du temps nécessaire à leur création. Par conséquent, ils préfèrent d'abord explorer les informations relatives aux données avant de définir des règles. Les clients doivent pouvoir identifier facilement les anomalies, ce qui leur permet de détecter de manière proactive les problèmes de qualité et de prendre des décisions métier en toute confiance.

Comment ça marche

> **Note**
>
> La détection des anomalies n'est prise en charge que dans AWS Glue ETL. Elle n'est pas prise en charge dans la qualité des données basée sur le catalogue de données.



AWS Glue Data Quality combine la puissance de la qualité des données basée sur des règles et des fonctionnalités de détection des anomalies pour fournir des données de haute qualité. Pour commencer, configurez d'abord les règles et les analyseurs, puis activez la détection des anomalies.

Rules

Règles : les règles expriment les attentes à l'égard de vos données dans un langage ouvert appelé DQDL (Data Quality Definition Language). Un exemple de règle est illustré ci-dessous. Cette règle sera efficace s'il n'y a aucune valeur vide ou NULL dans la colonne `passenger_count` :

```
Rules = [
    IsComplete "passenger_count"
]
```

Analyseurs

Dans les situations où vous connaissez les colonnes critiques, mais que vous ne connaissez pas suffisamment les données pour écrire des règles spécifiques, vous pouvez surveiller ces colonnes à l'aide d'analyseurs. Les analyseurs permettent de recueillir des statistiques de données sans définir de règles explicites. Un exemple de configuration des analyseurs est illustré ci-dessous :

```
Analyzers = [
AllStatistics "fare_amount",
DistinctValuesCount "pulocationid",
RowCount
]
```

Dans cet exemple, trois analyseurs sont configurés :

1. Le premier analyseur, `AllStatistics « fare_amount »`, capturera toutes les statistiques disponibles pour le champ `fare_amount`.
2. Le second analyseur, DistinctValuesCount « pulocationid », capturera le nombre de valeurs distinctes dans la colonne « pulocationid ».
3. Le troisième analyseur, `RowCount`, capturera le nombre total d'enregistrements de l'ensemble de données.

Les analyseurs constituent un moyen simple de recueillir des statistiques de données pertinentes sans définir de règles complexes. En surveillant ces statistiques, vous pouvez obtenir des informations sur la qualité des données et identifier les problèmes ou les anomalies potentiels qui peuvent nécessiter une enquête plus approfondie ou la création de règles spécifiques.

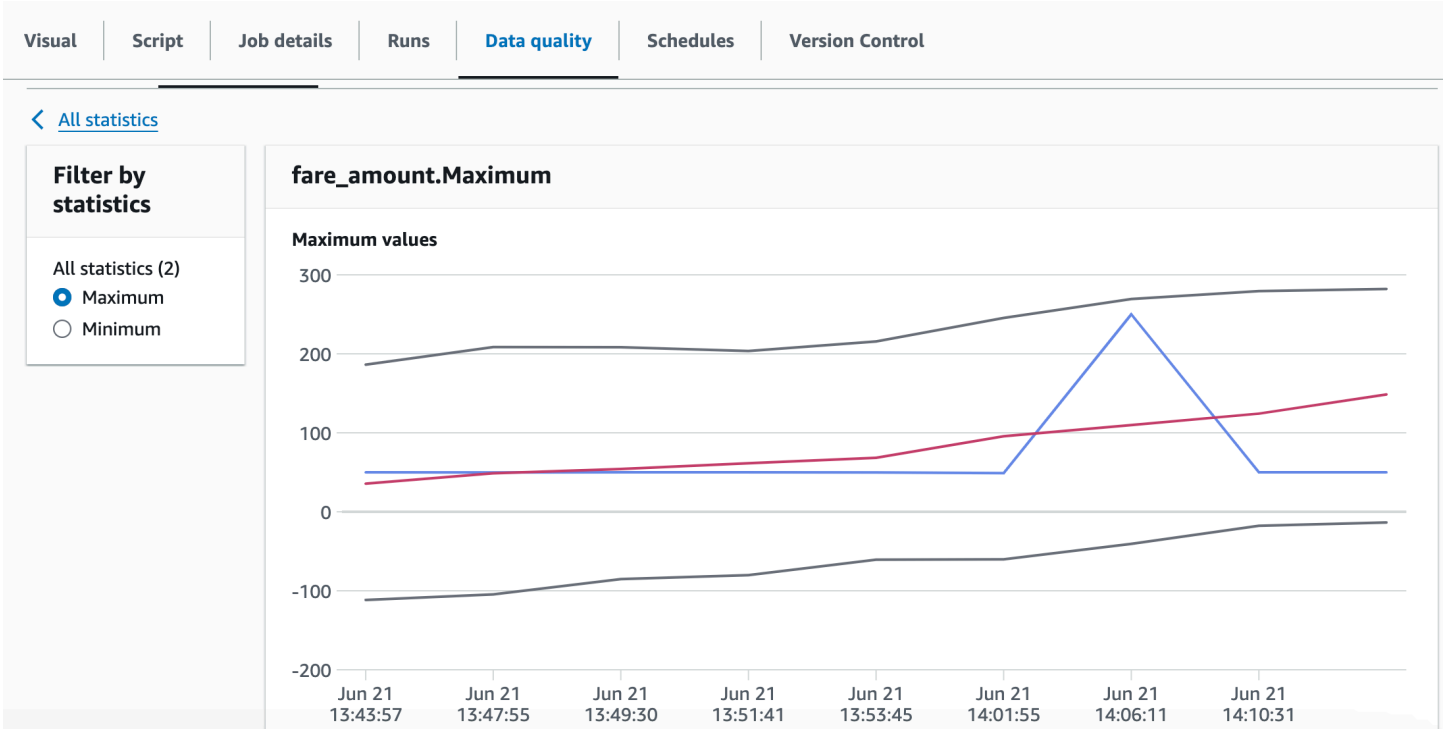
Statistiques des données

Analyzers et Rules in AWS Glue Data Quality collectent des statistiques de données, également appelées profils de données. Ces statistiques fournissent des informations sur les caractéristiques et la qualité de vos données. Les statistiques collectées sont stockées au fil du temps au sein du service AWS Glue, ce qui vous permet de suivre et d'analyser les modifications de vos profils de données.

Vous pouvez facilement récupérer ces statistiques et les écrire sur Amazon S3 pour une analyse plus approfondie ou un stockage à long terme en invoquant le code approprié APIs. Cette fonctionnalité vous permet d'intégrer le profilage des données dans vos flux de travail de traitement des données et d'exploiter les statistiques collectées à diverses fins, telles que le suivi de la qualité des données et la détection des anomalies.

En stockant les profils de données dans Amazon S3, vous pouvez tirer parti de la capacité de mise à l'échelle, de la durabilité et de la rentabilité du service de stockage d'objets d'Amazon. En outre, vous pouvez tirer parti d'autres AWS services ou d'outils tiers pour analyser et visualiser les profils de données, ce qui vous permet de mieux comprendre la qualité de vos données et de prendre des décisions éclairées en matière de gestion et de gouvernance des données.

Voici un exemple de statistiques de données stockées au fil du temps.

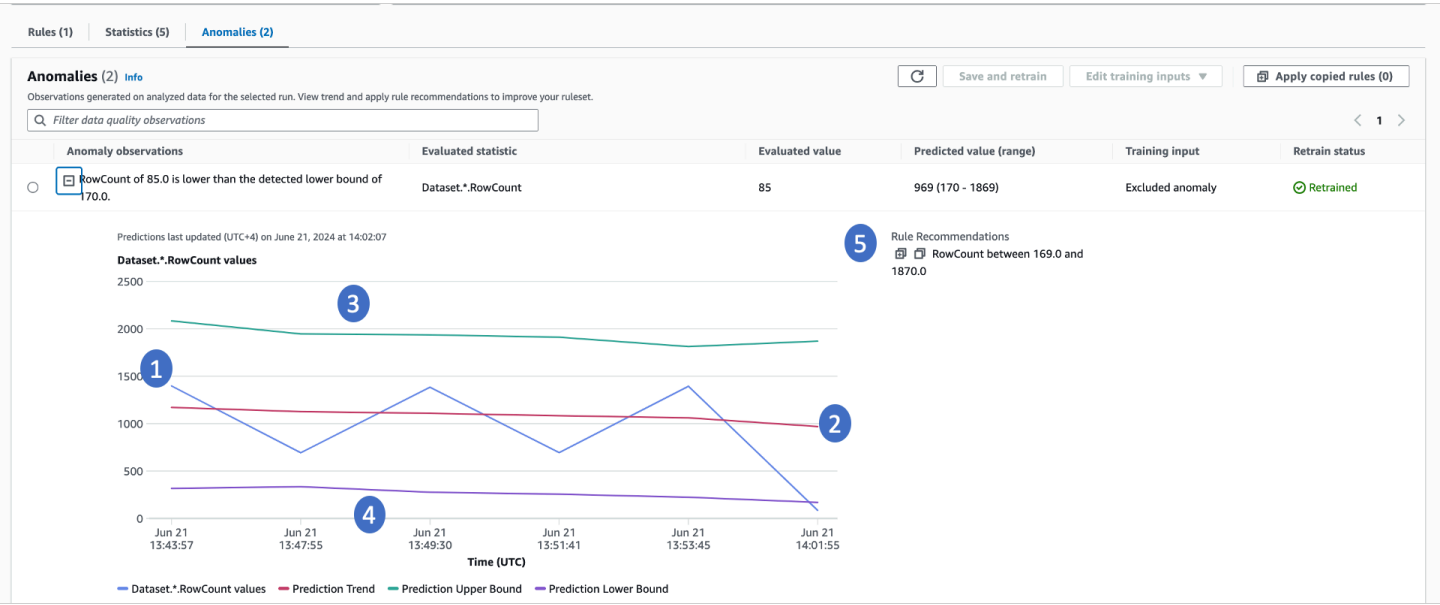


> **Note**
>
> AWS Glue Data Quality ne collectera les statistiques qu'une seule fois, même si vous disposez à la fois de Rule et d'Analyzer pour les mêmes colonnes, ce qui rend le processus de génération de statistiques efficace.

Détection des anomalies

AWS Glue Data Quality nécessite un minimum de trois points de données pour détecter les anomalies. Le service utilise un algorithme de machine learning pour tirer des leçons des tendances passées, puis prévoir les valeurs futures. Lorsque la valeur réelle ne se situe pas dans la plage prévue, AWS Glue Data Quality crée une observation d'anomalie. Elle offre une représentation

visuelle de la valeur réelle et des tendances. Quatre valeurs sont affichées sur le graphique ci-dessous.



1. La statistique réelle et son évolution dans le temps.
2. Une tendance dérivée en tirant des leçons de la tendance réelle. Cette donnée est utile pour comprendre l'orientation de la tendance.
3. Limite supérieure possible pour la statistique.
4. Limite inférieure possible pour la statistique.
5. Règles de qualité des données recommandées permettant de détecter ces problèmes à l'avenir.

Quelques points importants sont à noter concernant les anomalies :

- Lorsque des anomalies sont générées, les scores de qualité des données ne sont pas affectés.
- Lorsqu'une anomalie est détectée, elle est considérée comme normale pour les exécutions suivantes. L'algorithme de machine learning considérera cette valeur anormale comme entrée, sauf si elle est explicitement exclue.

Réentraînement

Il est essentiel de réentraîner le modèle de détection des anomalies pour détecter les bonnes anomalies. Lorsque des anomalies sont détectées, AWS Glue Data Quality inclut l'anomalie dans le modèle en tant que valeur normale. Pour garantir le bon fonctionnement de la détection des anomalies, il est important de fournir un feedback en reconnaissant ou en rejetant l'anomalie. AWS

Glue Data Quality fournit des mécanismes à la fois dans AWS Glue Studio et APIs pour fournir des commentaires au modèle. Pour en savoir plus, consultez la documentation sur la configuration de la détection des anomalies dans les pipelines AWS Glue ETL.

Détails de l'algorithme de détection des anomalies

- L'algorithme de détection des anomalies examine les statistiques des données au fil du temps. L'algorithme prend en compte tous les points de données disponibles et ignore les statistiques explicitement exclues.
- Ces statistiques de données sont stockées dans le service AWS Glue et vous pouvez fournir des AWS KMS clés pour les chiffrer. Reportez-vous au guide de sécurité pour savoir comment fournir des AWS KMS clés pour chiffrer les statistiques de AWS Glue Data Quality.
- La composante temporelle est cruciale pour l'algorithme de détection des anomalies. Sur la base des valeurs passées, AWS Glue Data Quality détermine les limites supérieure et inférieure. Au cours de cette détermination, elle prend en compte la composante temporelle. Les limites seront différentes pour les mêmes valeurs sur un intervalle d'une minute, un intervalle horaire ou un intervalle quotidien.

Capture la saisonnalité

AWS L'algorithme de détection des anomalies de Glue Data Quality peut capturer les tendances saisonnières. Par exemple, il peut comprendre que les schémas des jours de semaine diffèrent des schémas du week-end. C'est ce que montre l'exemple ci-dessous, où AWS Glue Data Quality détecte une tendance saisonnière dans les valeurs des données. Vous n'avez pas besoin de faire quoi que ce soit pour activer cette fonctionnalité. Au fil du temps, AWS Glue Data Quality apprend les tendances saisonnières et détecte les anomalies lorsque ces tendances se cassent.

Cost

Vous serez facturé en fonction du temps nécessaire pour détecter les anomalies. Chaque statistique est facturée 1 DPU pour le temps nécessaire à la détection des anomalies. Reportez-vous à Tarification AWS Glue pour des exemples détaillés.

Considérations clés

Le stockage des statistiques est gratuit. Cependant, il existe une limite de 100 000 statistiques par compte. Ces statistiques seront conservées pendant deux ans au maximum.

Configurer les autorisations IAM pour AWS Glue Data Quality

Cette rubrique fournit des informations pour vous aider à comprendre les actions et les ressources qu'un administrateur IAM peut utiliser dans le cadre d'une politique Gestion des identités et des accès AWS (IAM) pour AWS Glue Data Quality. Il inclut également des exemples de politiques IAM avec les autorisations minimales dont vous avez besoin pour utiliser AWS Glue Data Quality with the AWS Glue Data Catalog.

Pour plus d'informations sur la sécurité dans AWS Glue, consultezSécurité dans AWS Glue.

Autorisations IAM pour AWS Glue Data Quality

Le tableau suivant répertorie les autorisations dont les utilisateurs ont besoin pour effectuer des opérations spécifiques avec la qualité des données d' AWS Glue. Pour définir une autorisation précise pour AWS Glue Data Quality, vous pouvez spécifier ces actions dans l'`Action`élément d'une déclaration de politique IAM.

AWS Actions de Glue Data Quality

Action	Description	Types de ressources
`glue:CreateDataQualityRuleset`	Accorde l'autorisation de créer un jeu de règles de qualité des données.	`::dataQualityRuleset/<name>`
`glue:DeleteDataQualityRuleset`	Accorde l'autorisation de supprimer un jeu de règles de qualité des données.	`::dataQualityRuleset/<name>`
`glue:GetDataQualityRuleset`	Accorde l'autorisation de récupérer un jeu de règles de qualité des données.	`::dataQualityRuleset/<name>`
`glue:ListDataQualityRulesets`	Accorde l'autorisation de récupérer tous les jeux de règles de qualité des données.	`::dataQualityRuleset/*`
`glue:UpdateDataQualityRuleset`	Accorde l'autorisation de mettre à jour un jeu de règles de qualité des données.	`::dataQualityRuleset/<name>`
`glue:GetDataQualityResult`	Accorde l'autorisation de récupérer le résultat d'une exécution de tâche liée à la qualité des données. Cette action IAM fournit également des autorisations aux API suivantes :	`::dataQualityRuleset/<name>`

Action	Description	Types de ressources
	• BatchGetDataQualityQualityResult • ListDataQualityStatistics • ListDataQualityStatisticAnnotations	
glue:ListDataQualityResults	Accorde l'autorisation de récupérer tous les résultats d'exécution de tâches liées à la qualité des données.	::dataQualityRuleset/*
glue:CancelDataQualityRuleRecommendationRun	Accorde l'autorisation d'arrêter l'exécution d'une tâche de recommandation sur la qualité des données en cours.	::dataQualityRuleset/*
glue:GetDataQualityRuleRecommendationRun	Accorde l'autorisation de récupérer l'exécution d'une tâche de recommandation sur la qualité des données.	::dataQualityRuleset/*
glue:ListDataQualityRuleRecommendationRuns	Accorde l'autorisation de récupérer toutes les exécutions de tâches de recommandation sur la qualité des données.	::dataQualityRuleset/*
glue:StartDataQualityRuleRecommendationRun	Accorde l'autorisation de démarrer l'exécution d'une tâche de recommandation sur la qualité des données.	::dataQualityRuleset/*
glue:CancelDataQualityRulesetEvaluationRun	Accorde l'autorisation d'arrêter l'exécution d'une tâche sur la qualité des données en cours.	::dataQualityRuleset/*

Action	Description	Types de ressources
glue:GetDataQualityRulesetEvaluationRun	Accorde l'autorisation de récupérer l'exécution de tâches liées à la qualité des données.	::dataQualityRuleset/*
glue:ListDataQualityRulesetEvaluationRuns	Accorde l'autorisation de récupérer toutes les exécutions de tâches liées à la qualité des données.	::dataQualityRuleset/*
glue:StartDataQualityRulesetEvaluationRun	Accorde l'autorisation de démarrer l'exécution d'une tâche liée à la qualité des données.	::dataQualityRuleset/<name>
glue:PublishDataQuality	Accorde l'autorisation de publier des résultats sur la qualité des données.	::dataQualityRuleset/<name>
glue:GetDataQualityModel	Accorde l'autorisation de récupérer un modèle de qualité des données.	::dataQualityRuleset/<name>, ::job/<name>
glue:GetDataQualityModelResult	Accorde l'autorisation de récupérer les résultats d'un modèle de qualité des données.	::dataQualityRuleset/<name>, ::job/<name>

Action	Description	Types de ressources
`glue:PutDataQualityStatisticAnnotation`	Accorde l'autorisation d'ajouter des annotations aux statistiques. Cette action IAM fournit également des autorisations aux API suivantes : • `BatchPutDataQualityStatisticAnnotation`	`::dataQualityRuleset/<name>, ::job/<name>`
`glue:PutDataQualityProfileAnnotation`	Accorde l'autorisation d'ajouter des annotations à toutes les statistiques d'un profil.	`::dataQualityRuleset/<name>, ::job/<name>`

Configuration IAM requise pour planifier les exécutions d'évaluation

Autorisations IAM

Pour exécuter des exécutions d'évaluation de la qualité des données planifiées, vous devez ajouter l'action `IAM:PassRole` à la politique d'autorisation.

AWS EventBridge Autorisations requises par le planificateur

Action	Description	Types de ressources
`iam:PassRole`	Permet à IAM d'autoriser l'utilisateur à transmettre les rôles approuvés.	ARN du rôle utilisé pour appeler `StartDataQualityRulesetEvaluationRun`

Sans ces autorisations, l'erreur suivante se produit :

```
"errorCode": "AccessDenied"
"errorMessage": "User: arn:aws:sts::account_id:assumed-role/AWSGlueServiceRole is not authorized to perform: iam:PassRole on resource: arn:aws:iam::account_id:role/service-role/AWSGlueServiceRole
```

```
because no identity-based policy allows the iam:PassRole action"
```

Entités de confiance IAM

Les services AWS Glue et AWS EventBridge Scheduler doivent être répertoriés dans les entités de confiance afin de créer et d'exécuter un planning. `StartDataQualityEvaluationRun`

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "glue.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    },
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "scheduler.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Exemple de politiques IAM

Un rôle IAM pour AWS Glue Data Quality nécessite les types d'autorisations suivants :

- Autorisations pour les opérations de qualité des données de AWS Glue afin que vous puissiez obtenir les règles de qualité des données recommandées et exécuter une tâche de qualité des données par rapport à une table du catalogue de données AWS Glue. Les exemples de politiques IAM présentés dans cette section incluent les autorisations minimales requises pour les opérations de AWS Glue Data Quality.

- Autorisations qui donnent accès à la table de votre catalogue de données et aux données sous-jacentes. Ces autorisations varient en fonction de votre cas d'utilisation. Par exemple, pour les données que vous cataloguez dans Amazon S3, les autorisations doivent inclure l'accès à Amazon S3.

> **Note**
>
> Vous devez configurer ces autorisations Amazon S3 en plus des autorisations décrites dans cette section.

Autorisations minimales pour obtenir les règles de qualité des données recommandées

Cet exemple de politique inclut les autorisations dont vous avez besoin pour générer des règles de qualité des données recommandées.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AllowGlueRuleRecommendationRunActions",
      "Effect": "Allow",
      "Action": [
        "glue:GetDataQualityRuleRecommendationRun",
        "glue:PublishDataQuality",
        "glue:CreateDataQualityRuleset"
      ],
      "Resource": "arn:aws:glue:us-east-1:111122223333:dataQualityRuleset/*"
    },
 {
     "Sid": "AllowCatalogPermissions",
     "Effect": "Allow",
     "Action": [
        "glue:GetPartitions",
        "glue:GetTable"
      ],
      "Resource": [
        "*"
      ]
```

```
    },
    {
      "Sid": "AllowS3GetObjectToRunRuleRecommendationTask",
      "Effect": "Allow",
      "Action": [
        "s3:GetObject"
      ],
      "Resource": "arn:aws:s3:::aws-glue-*"
    },
  {
   "Sid": "AllowPublishingCloudwatchLogs",
   "Effect": "Allow",
   "Action": [
     "logs:CreateLogStream",
     "logs:CreateLogGroup",
     "logs:PutLogEvents"
   ],
   "Resource": "*"
    }
  ]
}
```

Autorisations minimales pour exécuter une tâche liée à la qualité des données

Cet exemple de politique inclut les autorisations dont vous avez besoin pour exécuter une tâche d'évaluation de la qualité des données.

Les instructions de politique suivantes sont facultatives et dépendent de votre cas d'utilisation :

- `AllowCloudWatchPutMetricDataToPublishTaskMetrics` : obligatoire si vous souhaitez publier des métriques d'exécution sur la qualité des données dans Amazon CloudWatch.
- `AllowS3PutObjectToWriteTaskResults` : obligatoire si vous souhaitez écrire des résultats d'exécution sur la qualité des données dans Amazon S3.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
```

```
      "Sid": "AllowGlueGetDataQualityRuleset",
      "Effect": "Allow",
      "Action": [
        "glue:GetDataQualityRuleset"
      ],
      "Resource": "arn:aws:glue:us-east-1:111122223333:dataQualityRuleset/YOUR-
RULESET-NAME"
    },
    {
      "Sid": "AllowGlueRulesetEvaluationRunActions",
      "Effect": "Allow",
      "Action": [
        "glue:GetDataQualityRulesetEvaluationRun",
        "glue:PublishDataQuality"
      ],
      "Resource": "arn:aws:glue:us-east-1:111122223333:dataQualityRuleset/*"
    },
    {
      "Sid": "AllowCatalogPermissions",
      "Effect": "Allow",
      "Action": [
        "glue:GetPartitions",
        "glue:GetTable"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Sid": "AllowS3GetObjectForRulesetEvaluationRun",
      "Effect": "Allow",
      "Action": [
        "s3:GetObject"
      ],
      "Resource": "arn:aws:s3:::aws-glue-*"
    },
    {
      "Sid": "AllowCloudWatchPutMetricDataToPublishTaskMetrics",
      "Effect": "Allow",
      "Action": [
        "cloudwatch:PutMetricData"
      ],
      "Resource": "*",
      "Condition": {
```

```
        "StringEquals": {
          "cloudwatch:namespace": "Glue Data Quality"
        }
      }
    },
    {
      "Sid": "AllowS3PutObjectToWriteTaskResults",
      "Effect": "Allow",
      "Action": [
        "s3:PutObject*"
      ],
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
    }
  ]
}
```

Autorisations minimales pour exécuter une tâche ETL liée à la qualité des données

Cet exemple de politique inclut les autorisations dont vous avez besoin pour exécuter une tâche ETL liée à la qualité des données.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AllowGluePublishDataQualityResult",
      "Effect": "Allow",
      "Action": [
        "glue:PublishDataQuality"
      ],
      "Resource": "arn:aws:glue:us-east-1:111122223333:dataQualityRuleset/*"
    },
    {
      "Sid": "AllowGlueGetDataQualityResult",
      "Effect": "Allow",
      "Action": [
        "glue:GetDataQualityResult"
      ],
      "Resource": "arn:aws:glue:us-east-1:111122223333:dataQualityRuleset/*"
```

```
    },
    {
      "Sid": "AllowGlueDataQualityStatisticAnnotation",
      "Effect": "Allow",
      "Action": [
        "glue:PutDataQualityStatisticAnnotation"
      ],
      "Resource": [
        "arn:aws:glue:us-east-1:111122223333:dataQualityRuleset/*",
        "arn:aws:glue:us-east-1:111122223333::job/{JobName}"
      ]
    },
    {
      "Sid": "AllowGlueDataQualityProfileAnnotation",
      "Effect": "Allow",
      "Action": [
        "glue:PutDataQualityProfileAnnotation"
      ],
      "Resource": [
        "arn:aws:glue:us-east-1:111122223333:dataQualityRuleset/*",
        "arn:aws:glue:us-east-1:111122223333::job/{JobName}"
      ]
    }
  ]
}
```

Premiers pas avec AWS Glue Data Quality pour le Data Catalog

Cette section de mise en route fournit des instructions qui vous aideront à faire vos premiers pas avec AWS Glue Data Quality sur la console AWS Glue. Vous apprendrez à accomplir des tâches essentielles telles que la génération de recommandations de règles portant sur la qualité des données et l'évaluation d'un jeu de règles par rapport à vos données.

Rubriques

- Conditions préalables
- Step-by-step exemple
- Génération de recommandations de règles
- Recommandations sur les règles de surveillance
- Modification des ensembles de règles recommandés

- Création d'un ensemble de règles
- Exécution d'un jeu de règles pour évaluer la qualité des données
- Affichage du score de qualité des données et des résultats
- Utilisation de requêtes de prétraitement
- Types de sources pris en charge
- Rubriques en relation

Conditions préalables

Avant d'utiliser AWS Glue Data Quality, vous devez vous familiariser avec l'utilisation du Data Catalog et des Crawlers dans AWS Glue. Avec AWS Glue Data Quality, vous pouvez évaluer la qualité des tables d'une base de données Data Catalog. Vous avez également besoin des éléments suivants :

- Une table dans le Data Catalog pour évaluer votre ensemble de règles de qualité des données.
- Un rôle IAM pour AWS Glue, à fournir lors de la génération de recommandations de règles ou de l'exécution d'une tâche liée à la qualité des données. Ce rôle doit avoir l'autorisation d'accéder aux ressources dont les différents processus AWS Glue Data Quality ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3 et CloudWatch. Pour consulter des exemples de politiques qui incluent les autorisations minimales pour AWS Glue Data Quality, consultez Exemple de politiques IAM.

  Pour en savoir plus sur les rôles IAM pour AWS Glue Glue, consultez Create an IAM policy for the AWS Glue service et Create an IAM role for the AWS Glue service. Vous pouvez également consulter la liste de toutes les autorisations AWS Glue spécifiques à la qualité des données sur la page Authorization for AWS Glue Data Quality actions.
- Une base de données avec au moins une table qui contient une variété de données. La table utilisée dans ce didacticiel est nommée `yyz-tickets`, avec la table `tickets`. Ces données sont une collection d'informations accessibles au public provenant de la ville de Toronto concernant les contraventions de stationnement. Si vous créez votre propre table, assurez-vous qu'elle est alimentée par une variété de données valides afin d'obtenir le meilleur ensemble de règles recommandées.

Step-by-step exemple

Pour un step-by-step exemple avec des exemples de jeux de données, consultez le billet de blog AWS Glue Data Quality.

Génération de recommandations de règles

Les recommandations de règles facilitent l'amélioration de la qualité des données sans écrire de code. Avec la qualité des données d' AWS , vous pouvez analyser vos données, identifier les règles et créer un ensemble de règles que vous pouvez évaluer dans le cadre d'une tâche liée à la qualité des données. Les exécutions de recommandations sont automatiquement supprimées après 90 jours.

Pour générer des recommandations de règles de qualité des données

1. Ouvrez la console AWS Glue à l'adresse https://console.aws.amazon.com/glue/.
2. Sélectionnez Tables dans le volet de navigation. Choisissez ensuite la table pour laquelle vous souhaitez générer des recommandations de règles de qualité des données.
3. Sur la page des détails du tableau, sélectionnez l'onglet Data quality pour accéder aux règles et aux paramètres de AWS Glue Data Quality pour votre tableau.
4. Dans l'onglet Qualité des données, choisissez Ajouter des règles et surveiller la qualité des données.
5. Sur la page Générateur d'ensemble règles, une alerte en haut de la page vous invite à démarrer une tâche de recommandation s'il n'y a pas d'exécution de recommandation de règles.
6. Choisissez Recommander des règles pour ouvrir la fenêtre modale et saisir vos paramètres pour la tâche de recommandation.
7. Choisissez un rôle IAM avec accès à AWS Glue. Ce rôle doit être autorisé à accéder aux ressources dont les différents processus de AWS Glue Data Quality ont besoin pour être exécutés en votre nom.
8. Une fois les champs remplis conformément à vos préférences, sélectionnez Recommander des règles pour démarrer l'exécution de la tâche de recommandation. Si des exécutions de recommandation sont en cours ou terminées, vous pouvez gérer vos exécutions dans cette alerte. Il se peut que vous deviez actualiser l'alerte pour afficher le changement du statut. Les exécutions de tâches de recommandation terminées et en cours apparaissent sur la page Historique des exécutions qui répertorie toutes les exécutions de recommandation des 90 derniers jours.

Signification des règles recommandées

AWS Glue Data Quality génère des règles basées sur les données de chaque colonne de la table en entrée. Il utilise les règles pour identifier les limites potentielles dans lesquelles les données peuvent être filtrées afin de respecter les exigences de qualité. La liste suivante de règles générées inclut des exemples utiles pour comprendre la signification des règles et ce qu'elles peuvent faire lorsqu'elles sont appliquées à vos données.

Pour obtenir la liste complète des types de règles DQDL (Data Quality Definition Language) générés, veuillez consulter Référence des types de règles DQDL.

- `IsComplete "SET_FINE_AMOUNT"` : la règle `IsComplete` vérifie que la colonne est remplie pour une ligne donnée. Utilisez cette règle pour marquer les colonnes comme non facultatives dans les données.
- `Uniqueness "TICKET_NUMBER" > 0.95` : la règle `Uniqueness` vérifie que les données de la colonne atteignent un certain seuil d'unicité. Dans cet exemple, il a été déterminé que les données qui remplissent une ligne donnée pour `"TICKET_NUMBER"` étaient identiques à 95 % au maximum à toutes les autres lignes, ce qui suggère cette règle.
- `ColumnValues "PROVINCE" in ["ON", "QC", "AB", "NY",...]` : la règle `ColumnValues` définit des valeurs valides pour la colonne, en fonction du contenu existant de la colonne. Dans cet exemple, les données de chaque ligne correspondent à un code d'immatriculation à deux lettres pour un état ou une province.
- `ColumnLength "INFRACTION_DESCRIPTION" between 15 and 31` : la règle `ColumnLength` impose une restriction de longueur aux données d'une colonne. Cette règle est générée à partir des échantillons de données en fonction des longueurs minimale et maximale enregistrées pour une colonne de chaînes.

Recommandations sur les règles de surveillance

Lorsque les recommandations de règles de qualité des données sont en cours d'exécution, la page Ajouter des règles et surveiller la qualité des données affiche des informations et des actions supplémentaires que vous pouvez effectuer dans la barre supérieure.

Lorsque les recommandations de règles sont en cours, vous pouvez choisir Arrêter l'exécution avant que la tâche de recommandation ne soit terminée. Pendant que la tâche est en cours, vous pouvez voir le statut En cours, ainsi que la date et l'heure du début de l'exécution.

Lorsque les recommandations de règles sont terminées, la barre de recommandations de règles affiche le nombre de règles recommandées, le statut de la dernière recommandation exécutée, ainsi que la date et l'horodatage de fin.

Vous pouvez ajouter les règles recommandées en choisissant Insérer une recommandation de règle. Pour consulter les règles précédemment recommandées, sélectionnez une date précise. Pour exécuter une nouvelle recommandation, sélectionnez Plus d'actions, puis sélectionnez Règles recommandées.

Définissez les paramètres par défaut en choisissant Gérer les paramètres utilisateur. Vous pouvez définir le chemin par défaut sur Amazon S3 pour stocker les ensembles de règles ou pour configurer un rôle par défaut pour exécuter le catalogue de données.

Modification des ensembles de règles recommandés

Comme la qualité des données d' AWS Glue génère des règles basées sur les données existantes dont vous disposez, il est possible que certaines règles inattendues ou indésirables apparaissent dans les suggestions automatisées. Afin de tirer le meilleur parti des ensembles de règles recommandés, vous devez les évaluer et les modifier. Pour cette étape du didacticiel, vous reprenez les règles générées à l'étape précédente et les ajustez pour appliquer des qualités plus restrictives à certaines données. Vous assouplissez également d'autres règles afin de garantir que des données correctes et uniques puissent être ajoutées ultérieurement.

Modifier un ensemble de règles suggéré

1. Dans la console AWS Glue, choisissez Data Catalog, puis choisissez Databases de données dans le volet de navigation. Choisissez la table `tickets`.
2. Sur la page des détails du tableau, choisissez l'onglet Data quality pour accéder aux règles et aux paramètres de AWS Glue Data Quality pour le tableau.
3. Dans la section Ensembles de règles, sélectionnez l'ensemble de règles généré dans Génération de recommandations de règles.
4. Choisissez Actions, puis sélectionnez Modifier dans la fenêtre de la console. L'éditeur d'ensembles de règles se charge dans la console. Il inclut un volet d'édition pour vos règles et une référence rapide pour DQDL.
5. Supprimer la ligne 2 du script. Cela permet d'assouplir l'exigence selon laquelle la taille de la base de données doit être limitée à un certain nombre de lignes. Après la modification, votre fichier doit contenir les éléments suivants sur les lignes 1 à 3 :

```
Rules = [
    IsComplete "TAG_NUMBER_MASKED",
    ColumnLength "TAG_NUMBER_MASKED" between 6 and 9,
```

6. Supprimer la ligne 25 du script. Cela permet d'assouplir l'exigence selon laquelle 96 % des provinces enregistrées sont ON. Après la modification, votre fichier doit contenir les éléments suivants, de la ligne 24 à la fin de l'ensemble de règles :

```
ColumnValues "PROVINCE" in ["ON", "QC", "AB", "NY", "AZ", "NS", "BC", "MI", "PQ",
 "MB", "PA", "FL", "SK", "NJ", "OH", "NB", "IL", "MA", "CA",
    "VA", "TX", "NF", "MD", "PE", "CT", "NC", "GA", "IN", "OR", "MN", "TN", "WI",
 "KY", "MO", "WA", "NH", "SC", "CO", "OK", "VT", "RI", "ME", "AL",
    "YT", "IA", "DE", "AR", "LA", "XX", "WV", "MT", "KS", "NT", "DC", "NV", "NE",
 "UT", "MS", "NM", "ID", "SD", "ND", "AK", "NU", "GO", "WY", "HI"],
ColumnLength "PROVINCE" = 2
]
```

7. Changez la ligne 14 comme suit :

```
IsComplete "TIME_OF_INFRACTION",
```

   Cela renforce l'exigence relative à la colonne en limitant la base de données aux seules contraventions qui contiennent une heure d'infraction enregistrée. Vous devez toujours considérer les contraventions sans heure d'infraction enregistrée comme des données non valides dans ce jeu de données. Cela diffère des situations dans lesquelles le partitionnement ou la transformation peuvent être plus appropriés pour une utilisation ou une inspection plus poussée des données afin de déterminer une règle de qualité.

8. Choisissez Mettre à jour l'ensemble de règles en bas de la page de la console.

Création d'un ensemble de règles

Un jeu de règles est un groupe de règles de qualité des données que vous évaluez par rapport à vos données. Dans la console AWS Glue, vous pouvez créer des ensembles de règles personnalisés à l'aide du langage DQDL (Data Quality Definition Language).

Pour créer un jeu de règles de qualité des données

1. Dans la console AWS Glue, choisissez Data Catalog, Databases, puis Tables dans le volet de navigation. Sélectionnez la table `tickets`.
2. Ouvrez l'onglet Data quality (Qualité des données).
3. Dans la section Ensembles de règles choisissez Créer un ensemble de règles. L'éditeur DQDL s'ouvre dans la console. Il dispose d'une zone de texte pour l'édition directe et d'une référence rapide pour les règles DQDL et le schéma de la table.
4. Commencez à ajouter des règles dans la zone de texte de l'éditeur DQDL. Vous pouvez soit écrire des règles directement à partir de ce didacticiel, soit utiliser la fonctionnalité du générateur de règles DQDL de l'éditeur de règles de qualité des données.

   > **Note**
   >
   > Comment utiliser le générateur de règles DQDL
   >
   > 1. Sélectionnez un type de règle dans la liste, puis choisissez le signe plus pour insérer un exemple de syntaxe dans le volet d'édition.
   > 2. Remplacez les noms des colonnes par vos propres noms de colonnes. Les noms des colonnes de la table sont disponibles dans l'onglet Schéma.
   > 3. Mettez à jour le paramètre d'expression selon vos besoins. Pour obtenir la liste complète des expressions prises en charge par DQDL, consultez Expressions.

   Par exemple, les règles suivantes sont des contraintes pour la validation des données de la colonne `ticket_number` dans la table `tickets`. Pour ajouter les règles suivantes, utilisez le générateur de règles DQDL ou modifiez directement votre ensemble de règles :

   ```
   IsComplete "ticket_number",
   IsUnique "ticket_number",
   ColumnValues "ticket_number" > 9000000000
   ```

5. Fournissez un nom pour votre exécution dans le champ Nom de l'ensemble de règles.
6. Choisissez Enregistrer l'ensemble de règles.

Évaluation de la qualité des données sur plusieurs jeux de données

Vous pouvez définir des règles de qualité des données pour plusieurs ensembles de données à l'aide d'ensembles ReferentialIntegrity de DatasetMatch règles. ReferentialIntegrityvérifie si les données du jeu de données principal sont présentes dans d'autres ensembles de données.

Pour ajouter un jeu de données de référence, cliquez sur l'onglet Schéma, puis sur Mettre à jour les tables de référence. Vous serez invité à sélectionner une base de données et une table. Vous pouvez ajouter la table, puis définir des règles de qualité des données. Les types de règles tels que AggregateMatch RowCountMatch ReferentialIntegrity, SchemaMatch, et permettent DatasetMatch d'effectuer des contrôles de qualité des données sur plusieurs ensembles de données.

Exécution d'un jeu de règles pour évaluer la qualité des données

Lorsque vous exécutez une tâche de qualité des données, AWS Glue Data Quality évalue un ensemble de règles par rapport à vos données et calcule un score de qualité des données. Ce score représente le pourcentage de règles de qualité des données qui ont été acceptées pour l'entrée.

Pour exécuter une tâche liée à la qualité des données

1. Dans la console AWS Glue, choisissez Data Catalog, Databases, puis Tables dans le volet de navigation. Sélectionnez le tableau `tickets`.
2. Choisissez l'onglet Qualité des données.
3. Dans la liste Ensembles de règles, sélectionnez l'ensemble de règles que vous souhaitez évaluer par rapport à la table. Pour cette étape, nous vous recommandons d'utiliser un ensemble de règles que vous avez déjà écrit ou modifié plutôt que des règles générées. Cliquez sur Exécuter.
4. Dans la fenêtre modale, choisissez votre rôle IAM. Ce rôle doit être autorisé à accéder aux ressources dont les différents processus de AWS Glue Data Quality ont besoin pour être exécutés en votre nom. Vous pouvez enregistrer le rôle IAM comme rôle par défaut ou le modifier en accédant à la page Paramètres par défaut.
5. Sous Actions relatives à la qualité des données, indiquez si vous souhaitez publier des statistiques sur Amazon CloudWatch. Lorsque cette option est sélectionnée, AWS Glue Data Quality publie des statistiques qui indiquent le nombre de règles passées et le nombre de règles qui ont échoué. Pour agir sur les métriques stockées de cette manière, vous pouvez utiliser des CloudWatch alarmes. Les indicateurs clés sont également publiés Amazon EventBridge pour vous permettre de configurer des alertes. Pour plus d'informations, consultez Setting up alerts, deployments, and scheduling.

6. Sous Fréquence d'exécution, choisissez d'exécuter à la demande ou de planifier l'ensemble de règles. Lorsque vous planifiez un ensemble de règles, le nom de la tâche vous est demandé. Le calendrier sera créé dans Amazon EventBridge. Vous pouvez modifier votre calendrier dans Amazon EventBridge.
7. Pour enregistrer les résultats dans Amazon S3, choisissez un emplacement pour les résultats de la qualité des données. Le rôle IAM que vous avez précédemment sélectionné pour cette tâche doit disposer d'un accès à cet emplacement.
8. Sous Configurations supplémentaires, entrez le nombre de travailleurs demandé que AWS Glue doit affecter à votre tâche de qualité des données.
9. Vous pouvez éventuellement configurer un filtre au niveau de la source de données. Cela vous permet de réduire le nombre de données que vous lisez. Vous pouvez également utiliser un filtre pour exécuter des validations incrémentielles en sélectionnant des informations de partition et en les transmettant en tant que paramètres via des appels d'API. Pour améliorer les performances, vous pouvez fournir un prédicat de partition.
10. Cliquez sur Exécuter. Votre nouvelle tâche doit apparaître dans la liste Data quality task runs (Exécutions de tâches liées à la qualité des données). Lorsque la colonne Statut d'exécution de la tâche indique Terminé, vous pouvez consulter les résultats du score de qualité. Il se peut que vous deviez rafraîchir votre fenêtre de console pour que le statut soit correctement mis à jour.
11. Pour afficher la colonne contenant les détails des résultats de la qualité des données, cliquez sur l'icône « + » pour développer l'ensemble de règles. Les résultats vous indiquent les règles qui ont réussi et échoué lors de l'évaluation, ainsi que les causes de l'échec des règles.

Affichage du score de qualité des données et des résultats

Pour voir la dernière exécution sur tous les ensembles de règles créés

1. Dans la console AWS Glue, choisissez Tables dans le volet de navigation. Choisissez ensuite la table pour laquelle vous souhaitez exécuter une tâche liée à la qualité des données.
2. Choisissez l'onglet Qualité des données.
3. L'instantané de la qualité des données montre une tendance générale des exécutions au fil du temps. Les 10 dernières exécutions de tous les ensembles de règles sont affichées par défaut. Pour filtrer par ensemble de règles, sélectionnez celui que vous souhaitez dans la liste déroulante. S'il y a moins de 10 exécutions, toutes les exécutions terminées disponibles sont affichées.

4. Dans la table Qualité des données, chaque ensemble de règles avec sa dernière exécution (le cas échéant) est affiché, ainsi que le score. En développant l'ensemble de règles, vous affichez les règles qu'il contient, ainsi que les résultats de l'exécution.

Pour voir la dernière exécution sur un ensemble de règles en particulier

1. Dans la console AWS Glue, choisissez Tables dans le volet de navigation. Choisissez ensuite la table pour laquelle vous souhaitez exécuter une tâche liée à la qualité des données.
2. Choisissez l'onglet Qualité des données.
3. Dans le tableau Qualité des données, choisissez un ensemble de règles spécifique.
4. Sur la page Détails de l'ensemble de règles, choisissez l'onglet Historique des exécutions.

   Toutes les exécutions d'évaluation pour cet ensemble de règles particulier sont répertoriées dans la table de cet onglet. Vous pouvez consulter l'historique des scores et le statut des exécutions.
5. Pour obtenir plus d'informations sur une exécution en particulier, choisissez l'ID d'exécution pour accéder à la page Détails de l'exécution d'évaluation. Sur cette page, vous pouvez consulter des informations sur l'exécution et plus de détails sur le statut des résultats des règles individuelles.

Utilisation de requêtes de prétraitement

AWS Glue Data Quality prend en charge les requêtes de prétraitement qui vous permettent de transformer vos données avant d'exécuter des contrôles de qualité des données. Cette fonctionnalité vous permet de :

- Créez des colonnes dérivées pour la validation de la qualité des données.
- Filtrez les données en fonction de conditions spécifiques.
- Effectuez des calculs ou des transformations pour les contrôles de qualité.
- Validez les relations entre les colonnes.

> **Note**
>
> Cette fonctionnalité n'est prise en charge que dans la console APIs et n'est pas prise en charge via celle-ci.

Utilisation de requêtes de prétraitement avec la CLI et le SDK

Les cycles de recommandation

Les exemples suivants montrent comment utiliser des requêtes de prétraitement avec des séries de recommandations.

AWS CLI :

```
aws glue start-data-quality-rule-recommendation-run \
  --data-source '{"DataQualityGlueTable": { \
    "DatabaseName": "mydatabase", \
    "TableName": "mytable", \
    "PreProcessingQuery": "SELECT sepal_length, sepal_width, petal_length, petal_width,
 class, (sepal_length + sepal_width) as sepal_total FROM `mydatabase.mytable`" \
  }}' \
  --role "arn:aws:iam::123456789012:role/GlueDataQualityRole" \
  --created-ruleset-name "my-ruleset-with-preprocessing"
```

SDK Java :

```
StartDataQualityRuleRecommendationRunRequest request = new
 StartDataQualityRuleRecommendationRunRequest()
    .withDataSource(new DataSource()
        .withDataQualityGlueTable(new DataQualityGlueTable()
            .withDatabaseName("mydatabase")
            .withTableName("mytable")
            .withPreProcessingQuery("SELECT sepal_length, sepal_width, " +
                "(sepal_length + sepal_width) as sepal_total " +
                "FROM `mydatabase.mytable`")))
    .withRole("arn:aws:iam::123456789012:role/GlueDataQualityRole")
    .withCreatedRulesetName("my-ruleset-with-preprocessing");

glueClient.startDataQualityRuleRecommendationRun(request);
```

Exécutions d'évaluation des ensembles de règles

Les exemples suivants montrent comment utiliser des requêtes de prétraitement avec des séries d'évaluation d'ensembles de règles.

AWS CLI :

```
aws glue start-data-quality-ruleset-evaluation-run \
```

```
  --data-source '{"DataQualityGlueTable": { \
    "DatabaseName": "mydatabase", \
    "TableName": "mytable", \
    "PreProcessingQuery": "SELECT order_id, amount, (tax + shipping) as total_fees FROM
 `mydatabase.mytable`" \
  }}' \
  --role "arn:aws:iam::123456789012:role/GlueDataQualityRole" \
  --ruleset-names '["my-ruleset"]'
```

SDK Java :

```
StartDataQualityRulesetEvaluationRunRequest request = new
 StartDataQualityRulesetEvaluationRunRequest()
    .withDataSource(new DataSource()
        .withDataQualityGlueTable(new DataQualityGlueTable()
            .withDatabaseName("mydatabase")
            .withTableName("mytable")
            .withPreProcessingQuery("SELECT order_id, amount, " +
                "(tax + shipping) as total_fees " +
                "FROM `mydatabase.mytable`")))
    .withRole("arn:aws:iam::123456789012:role/GlueDataQualityRole")
    .withRulesetNames(Arrays.asList("my-ruleset"));

glueClient.startDataQualityRulesetEvaluationRun(request);
```

Considérations relatives à la création de requêtes de prétraitement

Lorsque vous rédigez une requête de prétraitement :

- La référence au tableau doit être mise en forme de manière à `databaseName.tableName` utiliser des backticks.
- La requête doit être une instruction SELECT valide.
- Les noms de colonnes figurant dans la sortie de la requête seront utilisés pour les règles de qualité des données.

L'exemple suivant montre une requête de prétraitement :

```
SELECT
    sepal_length,
    sepal_width,
```

```
    petal_length,
    petal_width,
    class,
    (sepal_length + sepal_width) as sepal_total
FROM `mydatabase.mytable`
```

Limitations

- La requête doit référencer le tableau à l'aide de backticks et du `` `databaseName.tableName` `` format complet.
- La longueur maximale des requêtes est de 51 200 caractères.
- La requête doit renvoyer au moins une ligne de données.
- Toutes les colonnes référencées dans votre ensemble de règles doivent être présentes dans le résultat de la requête.

Types de sources pris en charge

Support des types de tables par AWS Lake Formation configuration

Type de table	AWS Lake Formation - Accès à toutes les tables	AWS Lake Formation Activé avec des colonnes	AWS Lake Formation Activé avec des filtres de données	AWS Lake Formation Support multi-comptes - Accès à toutes les tables	AWS Lake Formation Désactivé
Parquet	Pris en charge	Non pris en charge	Non pris en charge	Pris en charge	Pris en charge
ORC	Pris en charge	Non pris en charge	Non pris en charge	Pris en charge	Pris en charge
CSV, JSON, TSV	Pris en charge	Non pris en charge	Non pris en charge	Pris en charge	Pris en charge
Avro	Pris en charge	Non pris en charge	Non pris en charge	Pris en charge	Pris en charge

Type de table	AWS Lake Formation - Accès à toutes les tables	AWS Lake Formation Activé avec des colonnes	AWS Lake Formation Activé avec des filtres de données	AWS Lake Formation Support multi-comptes - Accès à toutes les tables	AWS Lake Formation Désactivé
JSON	Pris en charge	Non pris en charge	Non pris en charge	Pris en charge	Pris en charge
Iceberg	Pris en charge	Non pris en charge	Non pris en charge	Pris en charge	Pris en charge
HUDI	Non pris en charge	Non pris en charge	Non pris en charge	Non pris en charge	Pris en charge
Delta	Non pris en charge	Non pris en charge	Non pris en charge	Non pris en charge	Pris en charge
RMS	Pris en charge*	Pris en charge*	Pris en charge*	Non pris en charge	Non pris en charge
Tables Amazon S3	Pris en charge*	Non pris en charge	Non pris en charge	Non applicable	Pris en charge
Amazon RDS et Amazon Aurora	Non applicable	Ne s'applique pas	Ne s'applique pas	Non applicable	Non pris en charge
JDBC	Non applicable	Ne s'applique pas	Ne s'applique pas	Non applicable	Pris en charge

* La prise en charge des tables Amazon S3 et de SageMaker Lakehouse dans AWS Glue Console n'est pas prise en charge. Actuellement, les exécutions de recommandations de table Amazon S3 et de catalogue de données SageMaker Lakehouse et les exécutions d'évaluation de la qualité des données du catalogue de données ne sont prises en charge que via la CLI.

Autres restrictions connues

- Tables de liens symboliques de Delta Lake : non prises en charge pour les cycles de recommandation de AWS Glue Data Quality ou les cycles d'évaluation de la qualité des données du catalogue de données.
- Publication de tables Amazon S3 dans SageMaker Unified Studio : Actuellement, la publication de tables Amazon S3 sous forme d'actifs dans SageMaker Unified Studio n'est pas disponible ; la visualisation des exécutions de qualité des données de table Amazon S3 n'est donc pas disponible dans SageMaker Unified Studio.

Rubriques en relation

- Référence du type de règle DQDL
- Référence DQDL (Data Quality Definition Language)

Évaluation de la qualité des données avec AWS Glue Studio

AWS Glue Data Quality évalue et contrôle la qualité de vos données en fonction de règles que vous définissez. Ainsi, les données qui nécessitent une action sont identifiables facilement. Dans AWS Glue Studio, vous pouvez ajouter des nœuds de qualité des données à votre tâche visuelle afin d'appliquer des règles de qualité des données aux tableaux de votre catalogue de données. Vous pouvez ensuite contrôler et évaluer les modifications apportées à vos jeux de données tout au long de leur évolution. Pour un aperçu de la manière d'utiliser la qualité des données de AWS Glue dans AWS Glue Studio, regardez la vidéo suivante.

Voici les étapes générales décrivant l'utilisation d'AWS Glue Data Quality :

1. Création de règles de qualité des données : créez des règles de qualité des données à l'aide du générateur de règles DQDL en choisissant les ensembles de règles intégrés que vous configurez.
2. Configuration d'une tâche de qualité des données : définissez des actions en fonction des résultats relatifs à la qualité des données et des options de sortie.
3. Enregistrement et exécution d'une tâche de qualité des données : créez et exécutez une tâche. L'enregistrement de la tâche inclut les ensembles de règles que vous avez créés pour la tâche.
4. Contrôle et vérification des résultats relatifs à la qualité des données : vérifiez les résultats une fois la tâche terminée. Vous pouvez également planifier l'exécution de la tâche à une date future.

Avantages

Les analystes, les ingénieurs et les spécialistes des données peuvent utiliser le nœud Évaluer la qualité des données dans AWS Glue Studio pour analyser, configurer, surveiller et améliorer la qualité des données dans l'éditeur de tâches visuel. L'utilisation d'un nœud de qualité des données présente les avantages suivants :

- Détection des problèmes de qualité des données : la création de règles qui vérifient les caractéristiques de vos jeux de données permet de détecter les problèmes.
- Démarrage simplifié : vous pouvez commencer par utiliser des règles et des actions prédéfinies.
- Intégration étroite : vous pouvez utiliser des nœuds de qualité des données dans AWS Glue Studio, car la qualité des données de AWS Glue s'exécute au-dessus de AWS Glue Data Catalog.

Évaluation de la qualité des données pour les tâches ETL dans AWS Glue Studio

Dans ce didacticiel, vous ferez vos premiers pas avec la qualité des données de AWS Glue dans AWS Glue Studio. Vous allez apprendre comment :

- Créer des règles à l'aide du générateur de règles DQDL (Data Quality Definition Language).
- Spécifier les actions relatives à la qualité des données, les données de sortie et l'emplacement de sortie des résultats en matière de qualité des données.
- Vérifier les résultats en matière de qualité des données.

Pour vous entraîner avec un exemple, consultez le billet de blog Getting started with AWS Glue Data Quality for ETL pipelines.

Étape 1 : Ajout du nœud de transformation Évaluer la qualité des données à la tâche visuelle

À ce stade, vous ajoutez le nœud d'évaluation de la qualité des données à l'éditeur de tâches visuel.

Pour ajouter le nœud de qualité des données, procédez comme suit :

1. Dans la console AWS Glue Studio, choisissez Visual avec une source et une cible dans la section Create job, puis choisissez Create.

2. Choisissez le nœud auquel vous souhaitez appliquer la transformation de qualité des données. Il s'agit généralement d'un nœud de transformation ou d'une source de données.
3. Ouvrez le panneau de ressources sur la gauche en cliquant sur l'icône « + ». Recherchez ensuite Évaluer la qualité des données dans la barre de recherche, puis sélectionner Évaluer la qualité des données dans les résultats de la recherche.
4. L'éditeur de tâches visuel affiche la ramification du nœud de transformation Évaluer la qualité des données à partir du nœud que vous avez sélectionné. Sur le côté droit de la console, l'onglet Transform (Transformer) s'ouvre automatiquement. Si vous devez modifier le nœud parent, choisissez l'onglet Propriétés du nœud, puis choisissez le nœud parent dans le menu déroulant.

   Lorsque vous choisissez un nouveau nœud parent, une nouvelle connexion est établie entre le nœud parent et le nœud Évaluer la qualité des données (Évaluer la qualité des données). Supprimez tous les nœuds parents indésirables. Un seul nœud parent peut être connecté à un nœud Évaluer la qualité des données.
5. La transformation Évaluer la qualité des données prend en charge plusieurs parents afin que vous puissiez valider les règles de qualité des données sur plusieurs jeux de données. Les règles qui prennent en charge plusieurs ensembles de données incluent ReferentialIntegrity DatasetMatch, SchemaMatch, RowCountMatch, et AggregateMatch.

   Lorsque vous ajoutez plusieurs entrées à la transformation Évaluer la qualité des données, vous devez sélectionner votre entrée « principale ». Votre entrée principale est le jeu de données dont vous souhaitez valider la qualité des données. Tous les autres nœuds ou entrées sont traités comme des références.

   Vous pouvez utiliser la transformation Évaluer la qualité des données pour identifier les enregistrements spécifiques qui ont échoué aux contrôles de qualité des données. Nous vous recommandons de choisir votre jeu de données principal, car de nouvelles colonnes qui signalent les enregistrements défectueux sont ajoutées au jeu de données principal.
6. Vous pouvez spécifier des alias pour les sources de données d'entrée. Les alias constituent un autre moyen de référencer la source d'entrée lorsque vous utilisez la ReferentialIntegrity règle. Comme une seule source de données peut être désignée comme source principale, chaque source de données supplémentaire que vous ajoutez nécessite un alias.

   Dans l'exemple suivant, la ReferentialIntegrity règle spécifie la source de données d'entrée par le nom d'alias et effectue une one-to-one comparaison avec la source de données principale.

```
Rules = [
```

```
 ReferentialIntegrity “Aliasname.name” = 1
]
```

Étape 2 : Création d'une règle à l'aide de DQDL

Cette étape permet de créer une règle avec DQDL. Dans le cadre de ce didacticiel, vous créez une règle unique à l'aide du type de règle Exhaustivité. Ce type de règle compare le pourcentage de valeurs complètes (non nulles) d'une colonne à une expression donnée. Pour plus d'informations sur l'utilisation de DQDL, consultez DQDL.

1. Dans l'onglet Transformer, ajoutez un Type de règle en cliquant sur le bouton Insérer. Le type de règle est alors ajouté à l'éditeur de règles dans lequel vous pouvez saisir les paramètres de la règle.

   > **Note**
   >
   > Lorsque vous modifiez des règles, assurez-vous qu'elles se trouvent entre crochets et qu'elles sont séparées par des virgules. Par exemple, une expression de règle complète a la syntaxe suivante :
   >
   > ```
   > Rules= [
   >     Completeness "year">0.8, Completeness "month">0.8
   > ]
   > ```
   >
   > Cet exemple spécifie le paramètre d'exhaustivité pour les colonnes « année » et « mois ». Pour que la règle soit transmise, ces colonnes doivent être « complètes » à plus de 80 % ou contenir des données dans plus de 80 % des instances pour chaque colonne respective.

   Dans cet exemple, recherchez et insérez le type de règle Completeness (Exhaustivité). Le type de règle est alors ajouté à l'éditeur de règles. La syntaxe de ce type de règle est la suivante : `Completeness <COL_NAME> <EXPRESSION>`.

Une expression doit être fournie comme paramètre pour la plupart des types de règles afin de créer une réponse booléenne. Pour plus d'informations sur les expressions DQDL prises en charge, consultez DQDL expressions. Vous ajoutez ensuite le nom de la colonne.

2. Dans le générateur de règles DQDL, choisissez l'onglet Schéma. Trouvez le nom de la colonne dans le schéma d'entrée à l'aide de la barre de recherche. Le schéma d'entrée affiche le nom de la colonne et le type de données.
3. Dans l'éditeur de règles, cliquez à droite du type de règle pour insérer le curseur au point d'insertion de la colonne. Vous pouvez également saisir le nom de la colonne dans la règle.

   Par exemple, dans la liste des colonnes de la liste des schémas d'entrée, cliquez sur le bouton Insérer à côté de la colonne (ici, il s'agit de année). La colonne est alors ajoutée à la règle.
4. Ajoutez ensuite une expression pour évaluer la règle dans l'éditeur de règles. Comme le type de règle Exhaustivité vérifie le pourcentage de valeurs complètes (non nulles) dans une colonne par rapport à une expression donnée, saisissez une expression telle que `> 0.8`. Cette règle vérifie si la colonne contient plus de 80 % de valeurs complètes (non nulles).

Étape 3 : Configuration des résultats de la qualité des données

Une fois les règles de qualité des données créées, vous pouvez sélectionner des options supplémentaires pour spécifier la sortie du nœud de qualité des données.

1. Dans Data quality transform output (Sortie de transformation de la qualité des données), choisissez l'une des options suivantes :

   - Données d'origine : permet de fournir les données d'entrée d'origine. Lorsque vous choisissez cette option, un nouveau nœud enfant « rowLevelOutcomes » est ajouté à la tâche. Le schéma correspond au schéma du jeu de données principal qui a été transmis en tant qu'entrée à la transformation. Cette option est utile si vous souhaitez simplement transmettre les données et faire échouer la tâche en cas de problèmes de qualité.

     Un autre cas d'utilisation est celui où vous souhaitez détecter des enregistrements défectueux qui ont échoué aux contrôles de qualité des données. Pour détecter les enregistrements défectueux, choisissez l'option Ajouter de nouvelles colonnes pour indiquer les erreurs de qualité des données. Cette action ajoute quatre nouvelles colonnes au schéma de la transformation « rowLevelOutcomes ».

- DataQualityRulesPass(tableau de chaînes) — Fournit un tableau de règles ayant passé avec succès les contrôles de qualité des données.
- DataQualityRulesFail(tableau de chaînes) — Fournit un tableau de règles ayant échoué aux contrôles de qualité des données.
- DataQualityRulesSkip(tableau de chaînes) — Fournit un tableau de règles qui ont été ignorées. Les règles suivantes ne peuvent pas identifier les enregistrements d'erreur, car elles sont appliquées au niveau du jeu de données.
  - AggregateMatch
  - ColumnCount
  - ColumnExists
  - ColumnNamesMatchPattern
  - CustomSql
  - RowCount
  - RowCountMatch
  - StandardDeviation
  - Mean
  - ColumnCorrelation
- DataQualityEvaluationResult— Indique le statut « Réussi » ou « Échec » au niveau de la ligne. Notez que vos résultats globaux peuvent être un ÉCHEC, mais que certains enregistrements peuvent avoir réussis. Par exemple, il se peut que la RowCount règle ait échoué, mais que toutes les autres règles aient réussi. Dans de tels cas, l'état de ce champ est « Réussi ».

2. Résultats relatifs à la qualité des données : permet de fournir les règles configurées et leur état de réussite ou d'échec. Cette option est utile si vous souhaitez écrire vos résultats dans Amazon S3 ou dans d'autres bases de données.
3. Paramètres de sortie de qualité des données (facultatif) : choisissez Paramètres de sortie de qualité des données pour afficher le champ Emplacement des résultats relatifs à la qualité des données. Choisissez ensuite Parcourir pour rechercher un emplacement Amazon S3 à définir comme cible de sortie relative à la qualité des données.

Étape 4 : Configuration des actions relatives à la qualité des données

Vous pouvez utiliser des actions pour publier des métriques CloudWatch ou pour arrêter des tâches en fonction de critères spécifiques. Les actions ne sont disponibles qu'une fois la règle correspondante créée. Lorsque vous choisissez cette option, les mêmes métriques sont également publiées dans Amazon EventBridge. Vous pouvez utiliser ces options pour créer des alertes de notification.

- En cas d'échec d'un ensemble de règles : vous pouvez choisir la procédure à suivre en cas d'échec d'un ensemble de règles pendant l'exécution de la tâche. Si vous souhaitez que la tâche échoue en cas d'échec de la qualité des données, choisissez le moment où la tâche doit échouer en sélectionnant l'une des options suivantes. Par défaut, cette action n'est pas sélectionnée et la tâche termine son exécution même si les règles de qualité des données échouent.
  - Aucun : si vous choisissez Aucun (par défaut), la tâche n'échoue pas et continue de s'exécuter malgré les échecs de l'ensemble de règles.
  - Échec de la tâche après le chargement des données sur la cible : la tâche échoue et aucune donnée n'est enregistrée. Pour enregistrer les résultats, choisissez un emplacement Amazon S3 où les résultats relatifs à la qualité des données seront enregistrés.
  - Échec de la tâche sans chargement vers les données cibles : cette option échoue immédiatement en cas d'erreur de qualité des données. Aucune cible de données n'est chargée, y compris les résultats de la transformation de la qualité des données.

Étape 5 : Consultation des résultats relatifs à la qualité des données

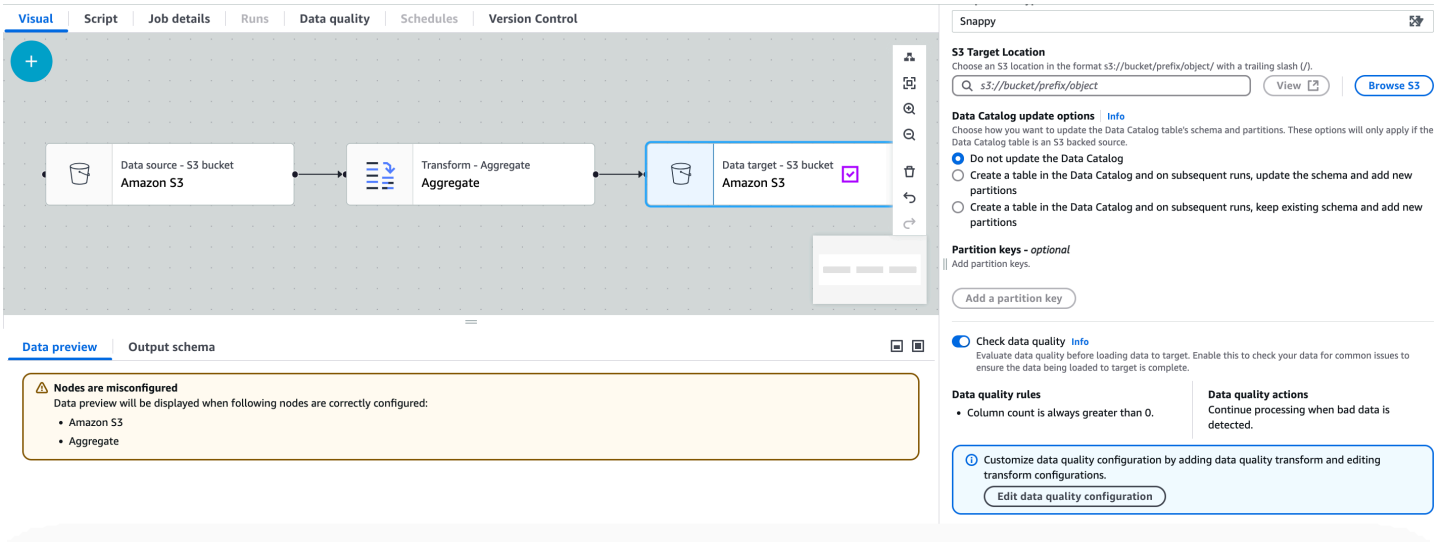
Une fois la tâche exécutée, choisissez l'onglet Qualité des données pour consulter les résultats relatifs à la qualité des données.

1. Pour chaque tâche exécutée, vous pouvez consulter les résultats relatifs à la qualité des données. Chaque nœud affiche un état de qualité des données et des informations détaillées. Choisissez un nœud pour afficher toutes les règles et l'état de chaque règle.
2. Choisissez Télécharger les résultats pour télécharger un fichier CSV contenant des informations sur l'exécution de la tâche et les résultats relatifs à la qualité des données.
3. Si plusieurs tâches exécutées sont associées à des résultats relatifs à la qualité des données, vous pouvez filtrer les résultats par date et par plage horaire. Choisissez Filtrer par plage de dates et d'heures pour agrandir la fenêtre de filtre.

4. Choisissez une plage relative ou une plage absolue. Pour les plages absolues, sélectionnez une date et entrez des valeurs pour l'heure de début et l'heure de fin à l'aide du calendrier. Lorsque vous avez terminé, choisissez Appliquer.

Qualité automatique des données

Lorsque vous créez une tâche AWS Glue ETL avec Amazon S3 comme cible, AWS Glue ETL active automatiquement une règle de qualité des données qui vérifie si les données chargées comportent au moins une colonne. Cette règle est conçue pour garantir que les données chargées ne sont pas vides ou endommagées. Toutefois, si cette règle échoue, la tâche n'échouera pas. Au contraire, vous remarquerez une réduction de votre score de qualité des données. En outre, la détection des anomalies est activée par défaut, ce qui permet de surveiller le nombre de colonnes des données. En cas de variations ou d'anomalies dans le nombre de colonnes, AWS Glue ETL vous informera de ces anomalies. Cette fonctionnalité vous aide à identifier les problèmes potentiels liés aux données et à prendre les mesures appropriées. Pour consulter la règle de qualité des données et sa configuration, vous pouvez cliquer sur la cible Amazon S3 dans votre tâche AWS Glue ETL. La configuration de la règle s'affiche, comme illustré dans la capture d'écran fournie.



Vous pouvez ajouter des règles de qualité des données supplémentaires en sélectionnant Modifier la configuration de la qualité des données.

Métriques agrégées

Vous pouvez avoir besoin de métriques agrégées, telles que le nombre d'enregistrements réussis, échoués ou ignorés au niveau des règles ou des ensembles de règles pour créer des tableaux de bord. Afin d'obtenir les métriques agrégées et les métriques de règles pour chaque règle, activez

d'abord les métriques agrégées en ajoutant l'option `publishAggregatedMetrics` à votre fonction `EvaluateDataQuality`.

Les options possibles pour `additional_options` `publishAggregatedMetrics` sont `ENABLED` et `DISABLED`. À titre d'exemple :

```
EvaluateDataQualityMultiframe = EvaluateDataQuality().process_rows(
    frame=medicare_dyf,
    ruleset=EvaluateDataQuality_ruleset,
    publishing_options={
        "dataQualityEvaluationContext": "EvaluateDataQualityMultiframe",
        "enableDataQualityCloudWatchMetrics": False,
        "enableDataQualityResultsPublishing": False,
    },
    additional_options={"publishAggregatedMetrics.status": "ENABLED"},
)
```

Si elle n'est pas spécifiée, la valeur `publishAggregatedMetrics.status` est `DISABLED` par défaut, tandis que les métriques de règles et les métriques agrégées seront désormais calculées. Cette fonctionnalité est actuellement prise en charge dans les sessions AWS Glue Interactive et dans les jobs Glue ETL. Cela n'est pas pris en charge dans Glue Catalog Data Quality APIs.

Récupération des résultats des métriques agrégées

Quand `additionalOptions` `"publishAggregatedMetrics.status": "ENABLED"`, vous pouvez obtenir les résultats à deux endroits :

1. `AggregatedMetrics` et `RuleMetrics` sont renvoyés par le biais de `GetDataQualityResult()` lorsque vous indiquez le `resultId` où `AggregatedMetrics` et `RuleMetrics` incluent :

   Métriques agrégées :

   - Nombre total de lignes traitées
   - Nombre total de lignes transmises
   - Nombre total de lignes ayant échoué
   - Nombre total de règles traitées
   - Nombre total de règles transmises
   - Nombre total de règles ayant échoué

```
[redacted] % aws --region us-east-1 glue get-data-quality-result --result-id "dqresult-09655f0234b3e3d129381bd162d0347ed25d1c05"
{
    "ResultId": "dqresult-09655f0234b3e3d129381bd162d0347ed25d1c05",
    "ProfileId": "dqprofile-[redacted]",
    "Score": 0.6666666666666666,
    "RulesetName": "EvaluateDataQuality_node1741974822533",
    "EvaluationContext": "EvaluateDataQuality_node1741974822533",
    "StartedOn": 1750687515.374,
    "CompletedOn": 1750687537.485,
    "JobName": "[redacted]_aggregatedmetrics_test",
    "JobRunId": "jr_[redacted]",
    "RuleResults": [
        {
            "Name": "Rule_1",
            "Description": "IsUnique \"customer_identifier\"",
            "Result": "PASS",
            "EvaluatedMetrics": {
                "Column.customer_identifier.Uniqueness": 1.0
            },
            "EvaluatedRule": "IsUnique \"customer_identifier\"",
            "RuleMetrics": {
                "rows_passed": 10.0,
                "rows_failed": 0.0,
                "rows_skipped": 0.0,
                "total_rows": 10.0
            }
        },
        {
            "Name": "Rule_2",
            "Description": "RowCount > 10",
            "EvaluationMessage": "Value: 10 does not meet the constraint requirement!",
            "Result": "FAIL",
            "EvaluatedMetrics": {
                "Dataset.*.RowCount": 10.0
            },
            "EvaluatedRule": "RowCount > 10",
            "RuleMetrics": {
                "rows_passed": 0.0,
                "rows_failed": 0.0,
                "rows_skipped": 10.0,
                "total_rows": 10.0
            }
        },
        {
            "Name": "Rule_3",
            "Description": "Completeness \"customer_identifier\" > 0.5",
            "Result": "PASS",
            "EvaluatedMetrics": {
                "Column.customer_identifier.Completeness": 1.0
            },
            "EvaluatedRule": "Completeness \"customer_identifier\" > 0.5",
            "RuleMetrics": {
                "rows_passed": 10.0,
                "rows_failed": 0.0,
                "rows_skipped": 0.0,
                "total_rows": 10.0
            }
        }
    ],
    "AggregatedMetrics": {
        "TotalRowsProcessed": 10.0,
        "TotalRowsPassed": 10.0,
        "TotalRowsFailed": 0.0,
        "TotalRulesProcessed": 3.0,
        "TotalRulesPassed": 2.0,
        "TotalRulesFailed": 1.0
    }
}
```

En outre, au niveau d'une règle, les métriques suivantes sont fournies :

Métriques de règles :

- Lignes transmises
- Lignes ayant échoué
- Ligne ignorée
- Nombre total de lignes traitées

2. `AggregatedMetrics` est renvoyé sous forme de trame de données supplémentaire et la trame de données `RuleOutcomes` est augmentée pour inclure `RuleMetrics`.

Exemples d'implémentation

L'exemple suivant montre comment implémenter les métriques agrégées dans Scala :

```
// Script generated for node Evaluate Data Quality
val EvaluateDataQuality_node1741974822533_ruleset = """
  # Example rules: Completeness "colA" between 0.4 and 0.8, ColumnCount > 10
  Rules = [
      IsUnique "customer_identifier",
      RowCount > 10,
      Completeness "customer_identifier" > 0.5
  ]
"""

val EvaluateDataQuality_node1741974822533 =
 EvaluateDataQuality.processRows(frame=ChangeSchema_node1742850392012,
 ruleset=EvaluateDataQuality_node1741974822533_ruleset,
 publishingOptions=JsonOptions("""{"dataQualityEvaluationContext":
 "EvaluateDataQuality_node1741974822533", "enableDataQualityCloudWatchMetrics":
 "true", "enableDataQualityResultsPublishing": "true"}"""),
 additionalOptions=JsonOptions("""{"compositeRuleEvaluation.method":"ROW","observations.scope":
 "publishAggregatedMetrics.status": "ENABLED"}""" ))

println("---------------------------------ROW LEVEL
 OUTCOMES---------------------------------")
val rowLevelOutcomes_node = EvaluateDataQuality_node1741974822533("rowLevelOutcomes")

rowLevelOutcomes_node.show(10)

 println("---------------------------------RULE LEVEL
 OUTCOMES---------------------------------")

val ruleOutcomes_node = EvaluateDataQuality_node1741974822533("ruleOutcomes")

ruleOutcomes_node.show()

 println("---------------------------------AGGREGATED
 METRICS---------------------------------")

val aggregatedMetrics_node = EvaluateDataQuality_node1741974822533("aggregatedMetrics")
```

```
aggregatedMetrics_node.show()
```

Exemple de résultats

Les résultats sont renvoyés comme suit :


```
{
    "Rule": "IsUnique \"customer_identifier\"",
    "Outcome": "Passed",
    "FailureReason": null,
    "EvaluatedMetrics": {
        "Column.customer_identifier.Uniqueness": 1
    },
    "EvaluatedRule": "IsUnique \"customer_identifier\"",
    "PassedCount": 10,
    "FailedCount": 0,
    "SkippedCount": 0,
    "TotalCount": 10
}
{
    "Rule": "RowCount > 10",
    "Outcome": "Failed",
    "FailureReason": "Value: 10 does not meet the constraint requirement!",
    "EvaluatedMetrics": {
        "Dataset.*.RowCount": 10
    },
    "EvaluatedRule": "RowCount > 10",
    "PassedCount": 0,
    "FailedCount": 0,
    "SkippedCount": 10,
    "TotalCount": 10
}
{
    "Rule": "Completeness \"customer_identifier\" > 0.5",
    "Outcome": "Passed",
    "FailureReason": null,
    "EvaluatedMetrics": {
        "Column.customer_identifier.Completeness": 1
    },
    "EvaluatedRule": "Completeness \"customer_identifier\" > 0.5",
    "PassedCount": 10,
    "FailedCount": 0,
    "SkippedCount": 0,
```

```
    "TotalCount": 10
}
```

Les métriques agrégées sont les suivantes :

```
{ "TotalRowsProcessed": 10, "PassedRows": 10, "FailedRows": 0, "TotalRulesProcessed":
 3, "RulesPassed": 2, "RulesFailed": 1 }
```

Générateur de règles de qualité des données

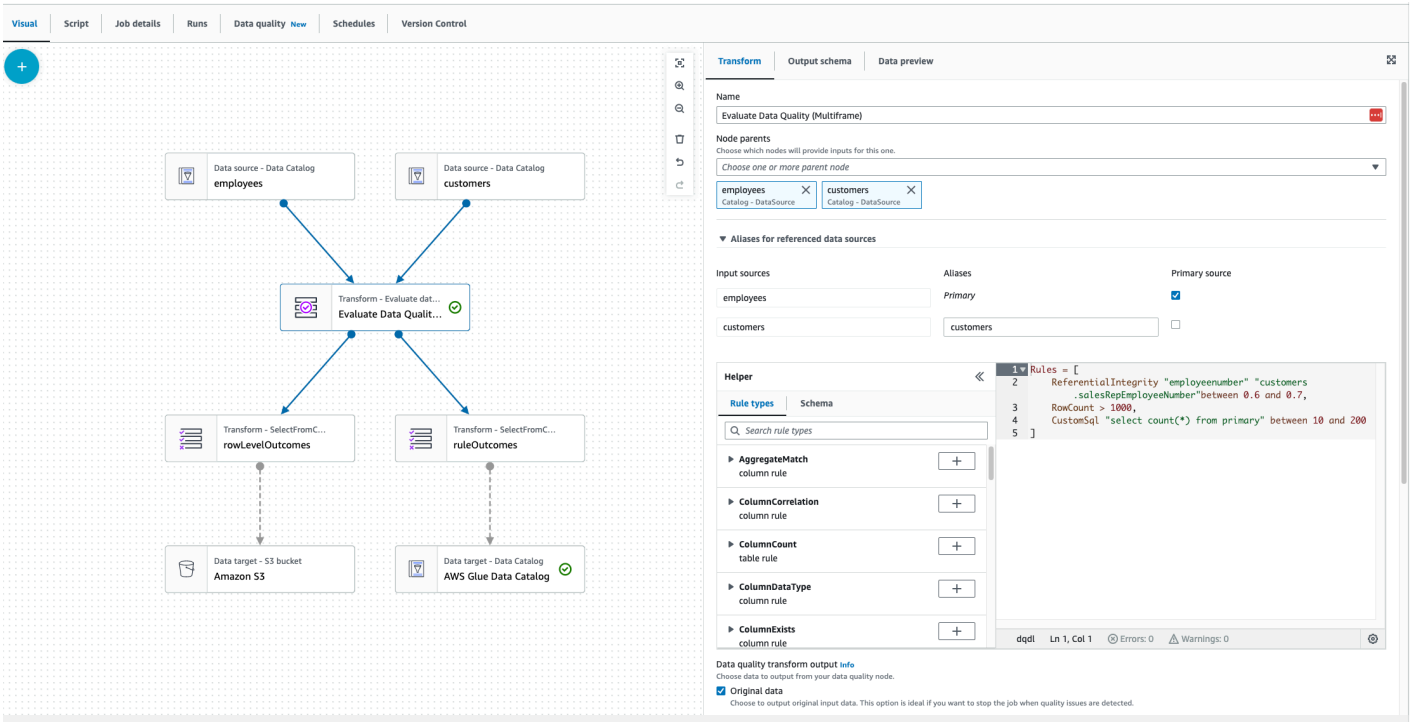
Avec le générateur de règles DQDL (Data Quality Definition Language), vous pouvez créer des règles de qualité pour évaluer vos données. Commencez par sélectionner un type de règle, puis spécifiez les paramètres dans l'éditeur de règles. L'éditeur de règles affiche également les erreurs et les avertissements éventuels lors de la création de règles.

Le guide DQDL documente de manière exhaustive la création de règles à l'aide de la syntaxe DQDL, les types de règles intégrées et des exemples.

Nœud Évaluer la qualité des données

En utilisant le nœud de transformation Évaluer la qualité des données et le générateur de règles DQDL, vous pouvez étendre l'espace de travail.

- Pour afficher l'onglet Transformer en mode plein écran, cliquez sur l'icône d'expansion dans le coin supérieur droit du panneau de détails du nœud.
- Pour développer l'éditeur de règles DQDL, cliquez sur l'icône << pour développer l'éditeur de règles et réduire les onglets Types de règles et Schéma.



Éléments

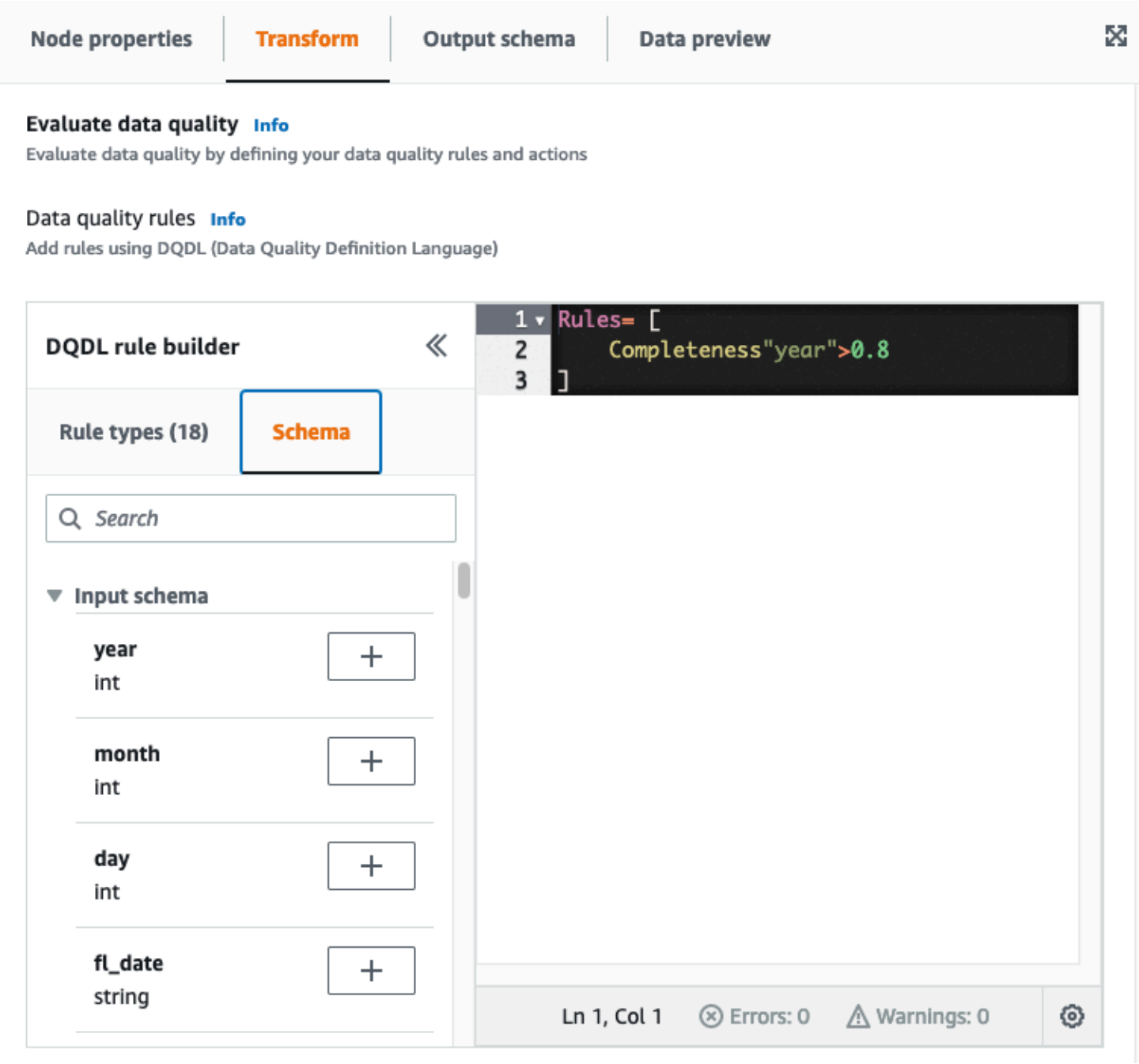
26 types de règles sont intégrés à AWS Glue Studio. Chaque type de règle est associé à une description et à des exemples d'utilisation.

Types de règles de qualité des données

AWS Glue Studio fournit des types de règles intégrés qui facilitent la création de règles. Pour plus d'informations sur les types de règles, consultez la Référence du type de règle DQDL.

Schema

L'onglet Schema (Schéma) affiche les noms des colonnes et le type de données du nœud parent. Les schémas de plusieurs nœuds sont affichés. Vous pouvez consulter le schéma d'entrée, effectuer une recherche par nom de colonne et insérer la colonne dans l'éditeur de règles.



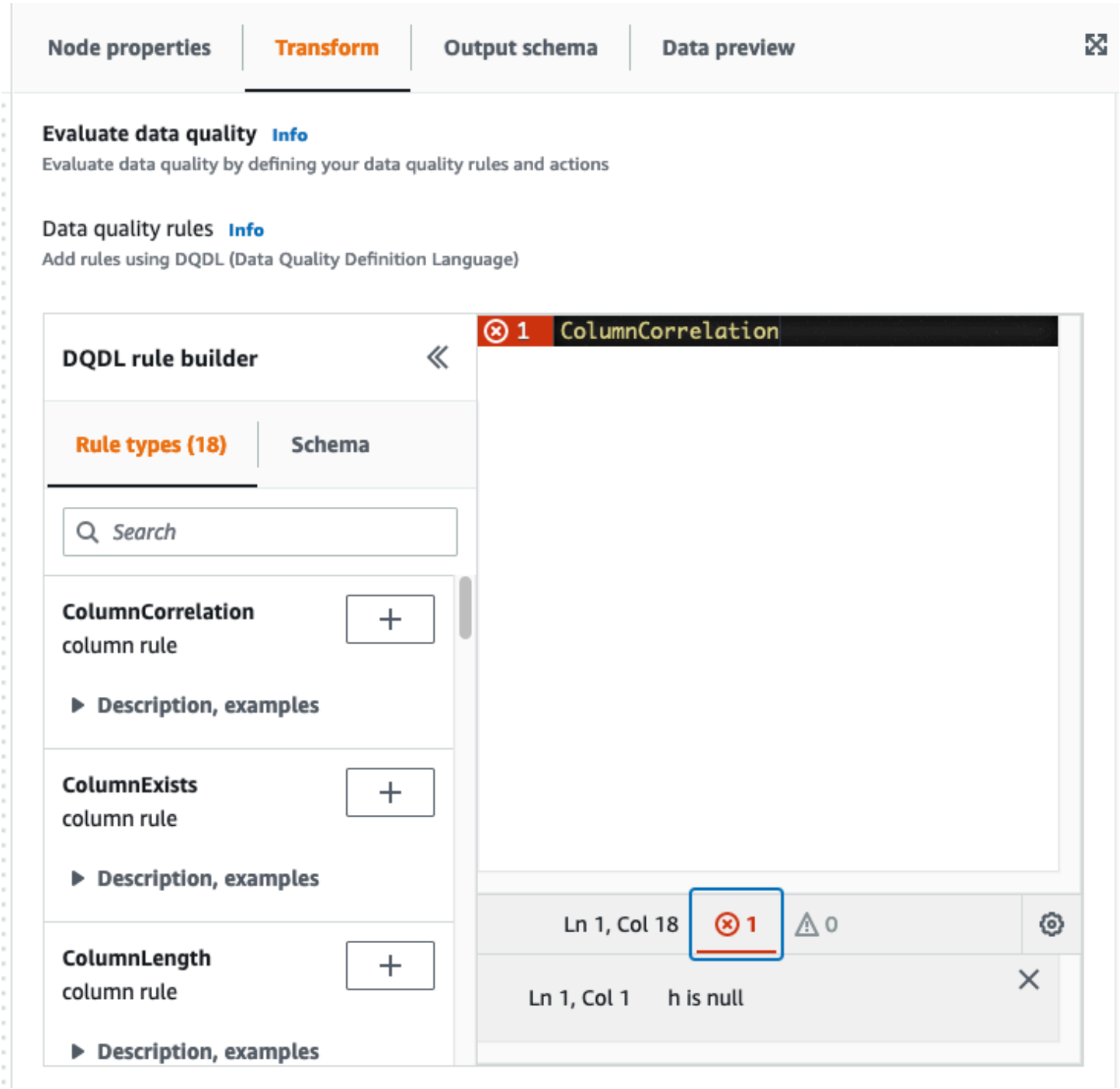
Éditeur de règles

L'éditeur de règles est un éditeur de texte permettant d'écrire et de modifier des règles. Si vous sélectionnez un type de règle dans le générateur de règles DQDL, il est ajouté à l'éditeur de règles. Vous pouvez ensuite spécifier des paramètres, ajouter des règles ou en modifier selon vos besoins en modifiant le texte. AWS Glue Studio valide les règles dans l'éditeur de règles et affiche les erreurs et les avertissements, le cas échéant.

Erreurs et avertissements

Si une règle n'est pas conforme à la syntaxe des règles DQDL, l'éditeur de règles affiche plusieurs indicateurs d'erreur visuels :

- L'éditeur de règles affiche une icône d'erreur de couleur rouge sur la ligne contenant l'erreur.
- L'éditeur de règles affiche le nombre d'erreurs à côté de l'icône d'erreur rouge.
- Lorsque vous cliquez sur la ligne contenant l'erreur, des descriptions et l'emplacement de l'erreur (ligne et colonne) s'affichent en bas de l'éditeur de règles.



Actions relatives à la qualité des données

Par défaut, cette action n'est pas sélectionnée et la tâche s'exécute même si les règles de qualité des données ne sont pas respectées.

Choisissez l'une des actions suivantes. Vous pouvez utiliser des actions pour publier les résultats des tâches CloudWatch ou les arrêter en fonction de critères spécifiques. Les actions ne sont disponibles qu'une fois la règle correspondante créée.

- Publier les résultats sur CloudWatch : lorsque vous exécutez une tâche, ajoutez-les à CloudWatch.
- Échec d'une tâche en cas de problème de qualité des données : si les règles de qualité des données ne sont pas respectées, la tâche échoue.

Sortie de transformation de la qualité des données

- Données d'origine : permet de fournir les données d'entrée d'origine. Cette option est particulièrement utile pour interrompre la tâche en cas de détection de problèmes de qualité.
- Métriques de qualité des données : permet de générer les règles configurées et leur état de réussite ou d'échec. Cette option est utile si vous souhaitez effectuer une action personnalisée.

Paramètres de sortie de qualité des données

Définissez l'emplacement des résultats relatifs à la qualité des données en spécifiant l'emplacement Amazon S3 comme cible de sortie.

Configuration de la détection des anomalies dans les jobs AWS Glue ETL

Pour commencer à détecter les anomalies dans AWS Glue Studio, ouvrez une tâche AWS Glue Studio et cliquez sur Evaluate Data Quality Transform.

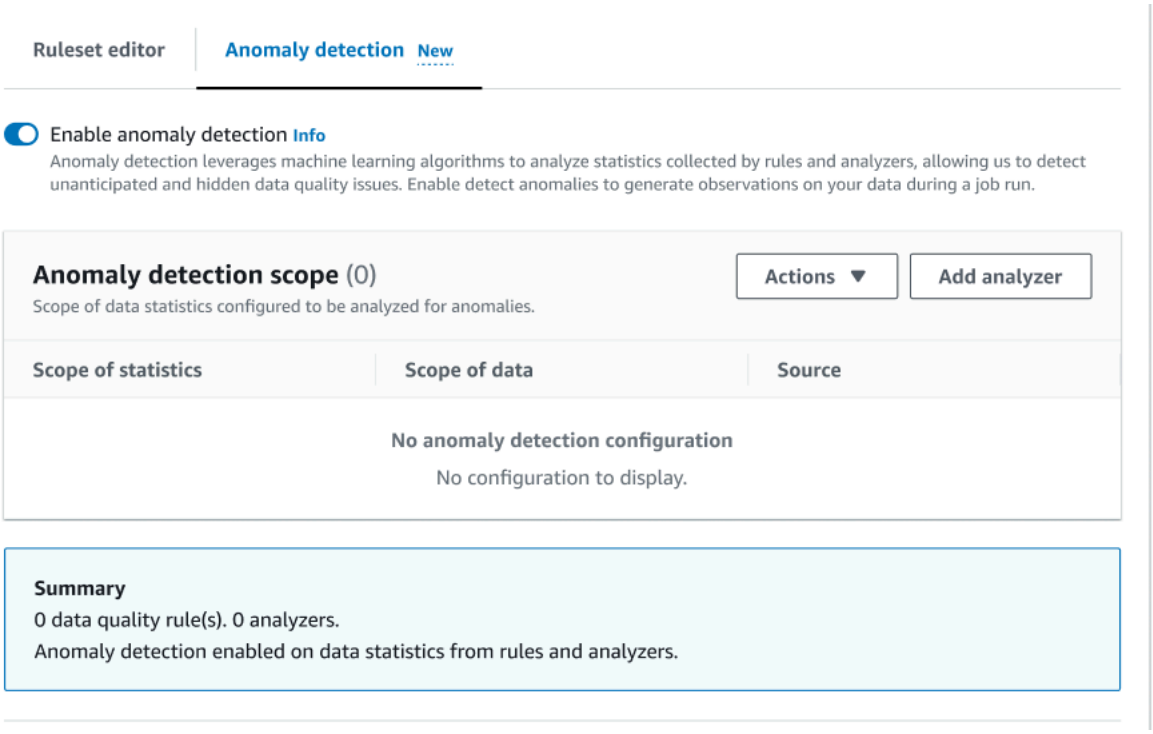
En activant cette fonctionnalité, AWS Glue Data Quality analysera vos données au fil du temps afin de détecter les anomalies. Elle fournit des statistiques et des observations précieuses sur vos données, vous permettant de prendre des mesures en cas d'anomalie identifiée.

Consultez la documentation sur la détection des anomalies pour comprendre le fonctionnement interne de cette fonctionnalité.

Activation de la détection des anomalies

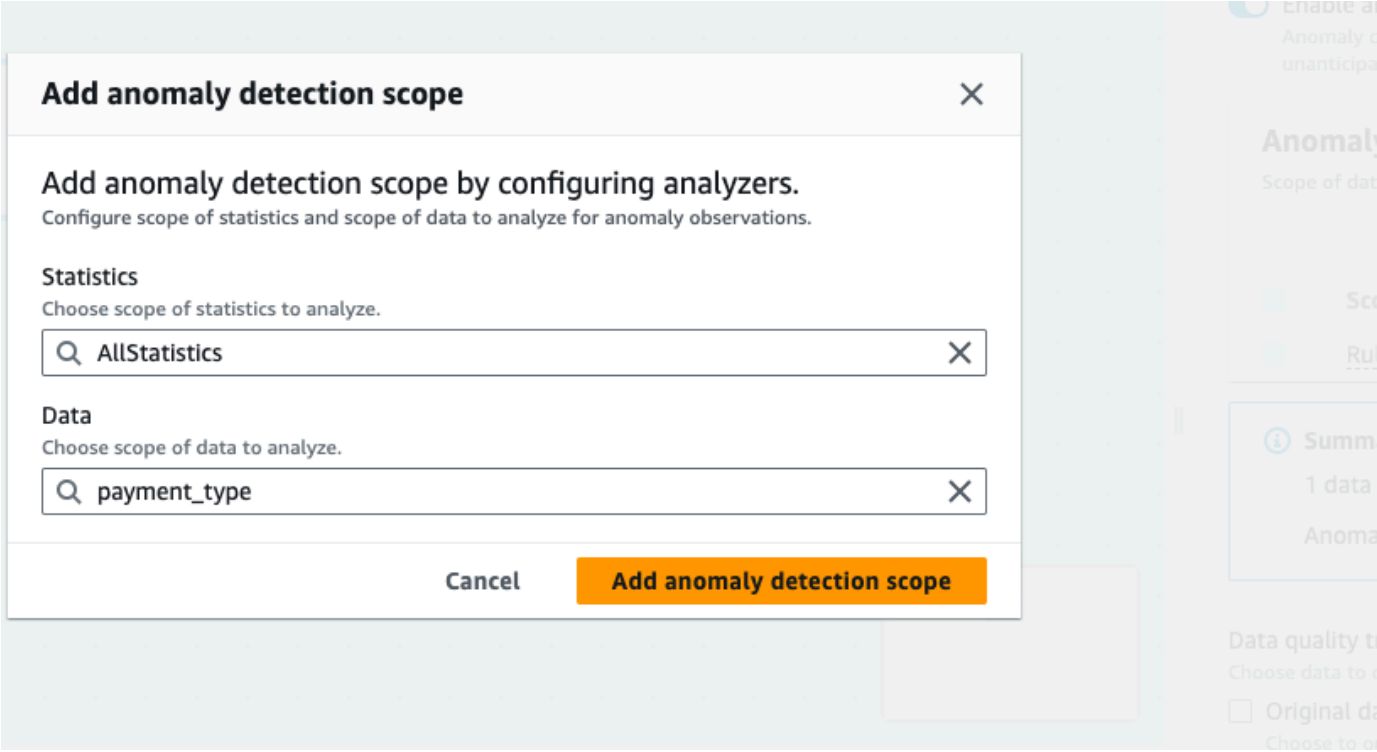
Pour activer la détection d'anomalies dans AWS Glue Studio :

1. Choisissez le nœud Qualité des données dans votre tâche, puis cliquez sur l'onglet Détection des anomalies. Activez l'option Activer la détection des anomalies.



2. Définissez les données à surveiller pour détecter les anomalies en choisissant Ajouter un analyseur. Vous pouvez remplir deux champs : statistiques et données.

   - Les statistiques sont des informations sur la forme et les autres propriétés de vos données. Vous pouvez sélectionner une ou plusieurs statistiques à la fois ou sélectionner Toutes les statistiques. Les statistiques incluent : l'exhaustivité, l'unicité, la moyenne, la somme StandardDeviation, l'entropie DistinctValuesCount, UniqueValueRatio etc. Pour plus de détails, consultez la documentation sur les analyseurs.
   - Les données sont les colonnes de votre jeu de données. Vous pouvez choisir toutes les colonnes ou des colonnes individuelles.



3. Choisissez Ajouter une zone de détection d'anomalies pour enregistrer vos modifications. Lorsque vous avez créé des analyseurs, vous pouvez les consulter dans la section Portée de la détection d'anomalies.

   Vous pouvez également utiliser le menu Actions pour modifier vos analyseurs, ou choisir l'onglet Éditeur de jeu de règles et modifier l'analyseur directement dans le bloc-notes de l'éditeur de jeu de règles. Vous verrez les analyseurs que vous avez enregistrés juste en dessous de toutes les règles que vous avez créées.

```
Rules = [

]

Analyzers = [
    Completeness “id”
]
```

Une fois l'ensemble de règles et les analyseurs mis à jour configurés, AWS Glue Data Quality surveille en permanence les flux de données entrants. Le service peut signaler des anomalies

potentielles par le biais d'alertes ou d'arrêts de tâches, en fonction de vos paramètres. Cette surveillance proactive permet de garantir la qualité et l'intégrité des données dans l'ensemble de vos pipelines de données.

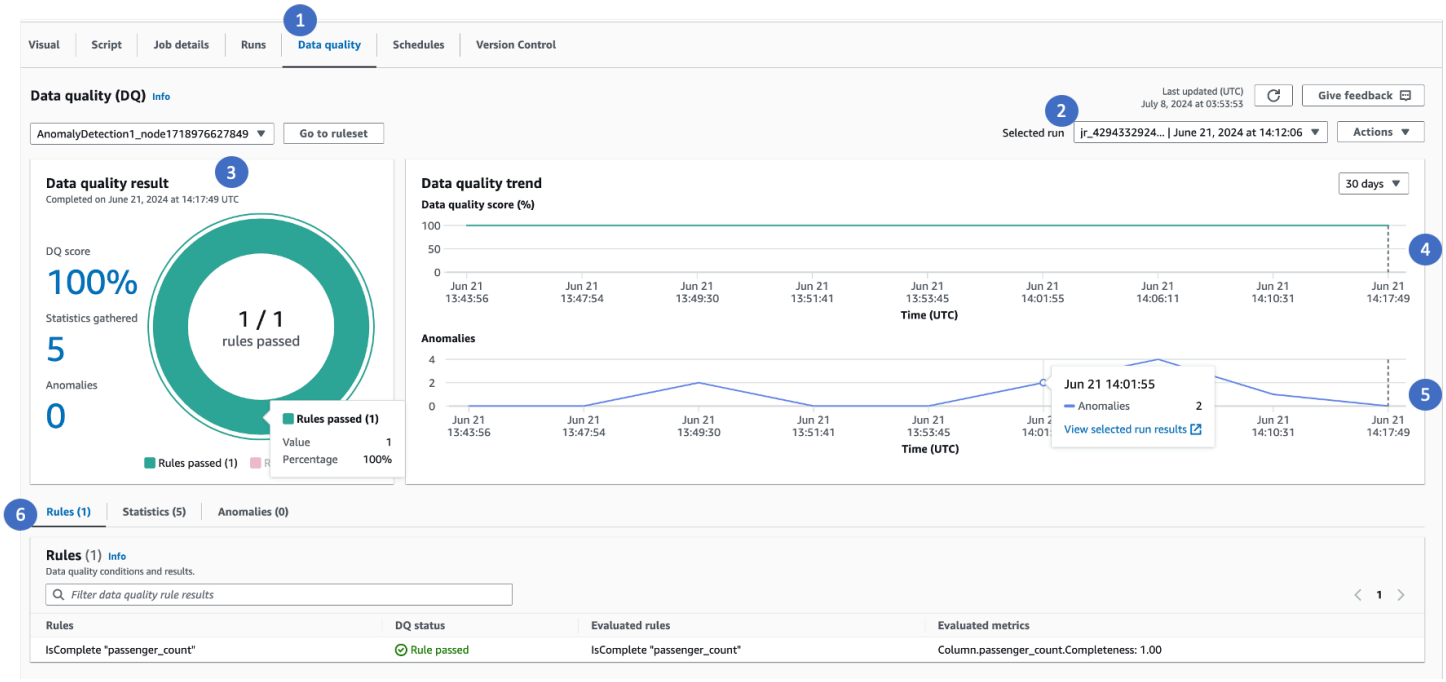
Dans la section suivante, vous apprendrez comment surveiller efficacement les anomalies identifiées par le système. Vous apprendrez également à consulter et à analyser les statistiques de données recueillies par AWS Glue Data Quality. En outre, vous apprendrez à fournir des commentaires au modèle de machine learning qui alimente la fonctionnalité de détection des anomalies. Cette boucle de rétroaction est essentielle pour améliorer la précision du modèle et garantir qu'il peut détecter efficacement les anomalies conformes aux exigences spécifiques de votre entreprise et à vos modèles de données.

Affichage des scores de qualité des données et des anomalies

Dans cette section, nous allons explorer le tableau de bord de qualité des données et les différentes fonctionnalités qu'il fournit.

Visualisation et compréhension des métriques de qualité des données de haut niveau et des tendances

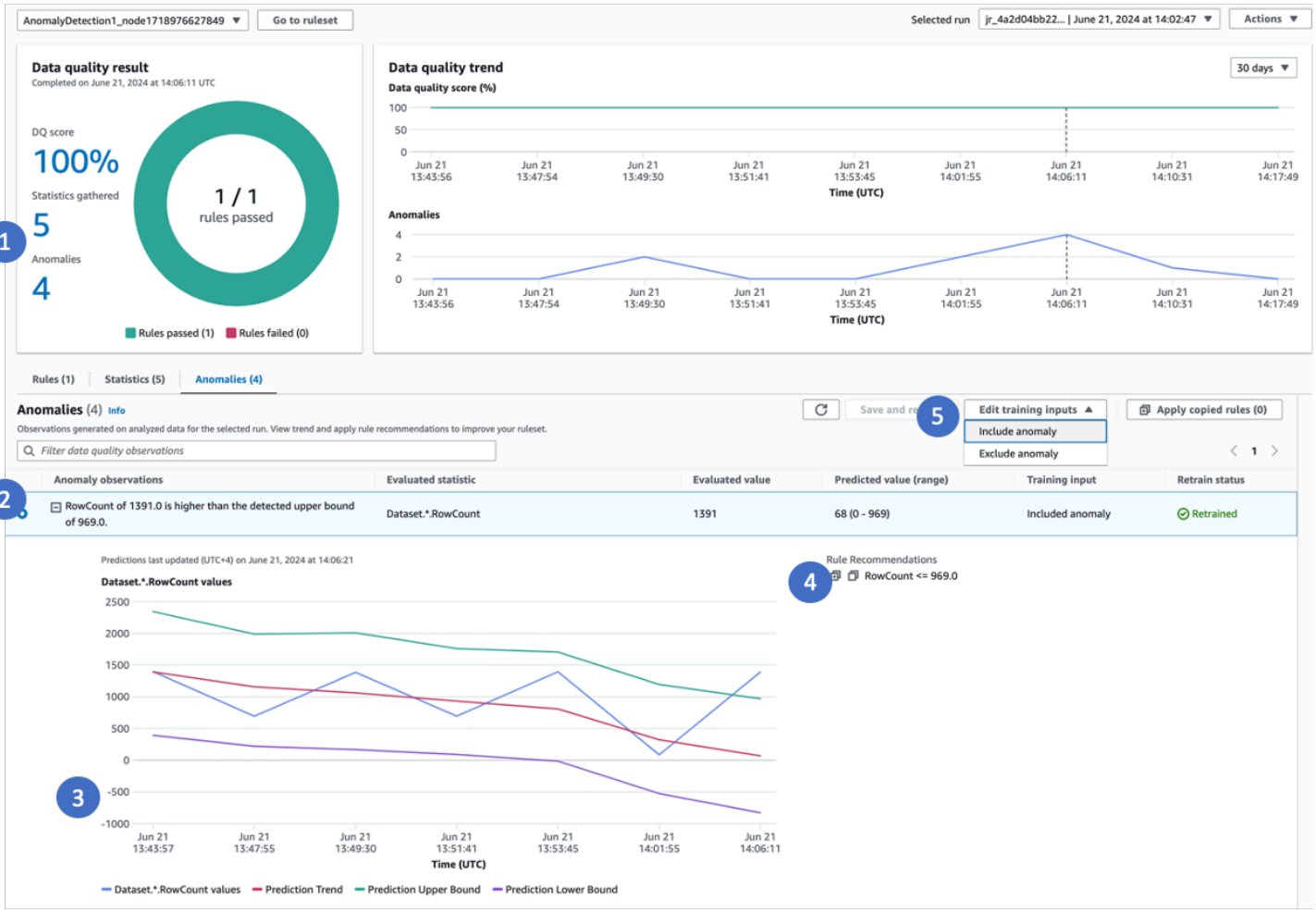
Une fois que votre tâche a abouti, cliquez sur l'onglet Qualité des données pour afficher les scores de qualité des données et les anomalies.



Les composants suivants de l'onglet Qualité des données fournissent des informations utiles.

1. Cliquez sur l'onglet Qualité des données pour afficher les métriques de qualité des données.
2. Sélectionnez un ID d'exécution de tâche spécifique pour afficher le score de qualité des données.
3. Ce volet affiche trois informations importantes. Vous pouvez choisir chacune d'elles pour accéder à des tables spécifiques afin de visualiser les anomalies, les statistiques de données ou les règles.
   - Score de qualité des données lorsque les règles sont configurées.
   - Nombre de statistiques recueillies par Règles et Analyseurs.
   - Nombre total d'anomalies détectées.
4. Ce graphique de tendance montre l'évolution de la qualité des données au fil du temps. Vous pouvez survoler la tendance et atteindre un moment précis où les scores de qualité des données se sont détériorés.
5. L'évolution des anomalies au fil du temps vous indiquera le nombre d'anomalies détectées au fil du temps.
6. Onglets :
   - L'onglet Règles est l'onglet par défaut qui affiche la liste de toutes les règles et leur état. Les règles évaluées sont utiles dans le cas de règles dynamiques pour afficher la valeur réelle à laquelle la règle a été évaluée.
   - L'onglet Statistiques répertorie toutes les statistiques, ce qui vous permet de visualiser les métriques et les tendances au fil du temps.
   - L'onglet Anomalies affiche la liste des anomalies détectées.

Affichage des anomalies et entraînement de l'algorithme de détection des anomalies



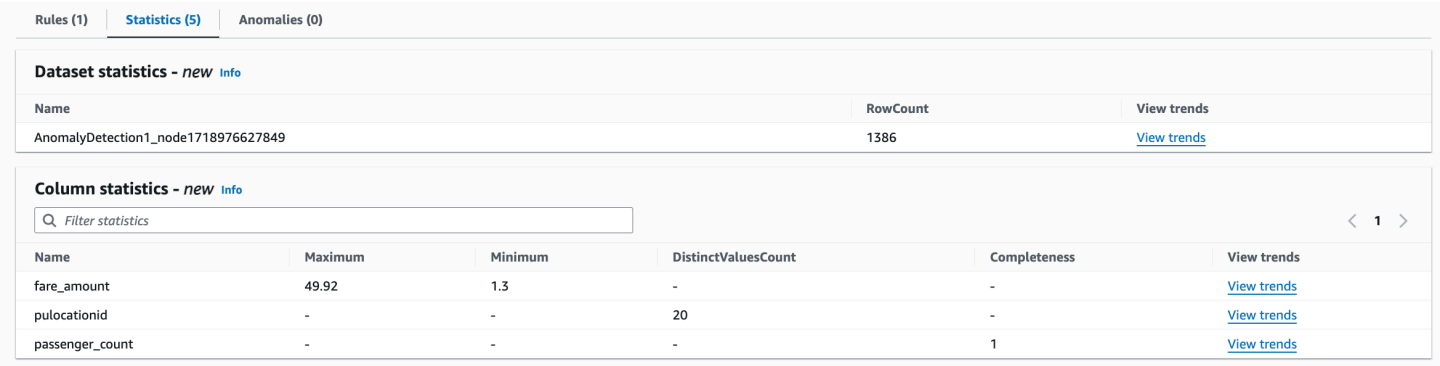
Appels pour l'image ci-dessus :

1. Lorsque des anomalies sont détectées, cliquez sur l'anomalie ou sélectionnez l'onglet Anomalies.
2. AWS Glue Data Quality fournit une explication détaillée de l'anomalie, de la valeur réelle et de la plage prévue
3. AWS Glue Data Quality indique une courbe de tendance. Elle présente la valeur réelle, une tendance dérivée basée sur les valeurs réelles (ligne rouge), la limite supérieure et la limite inférieure
4. AWS Glue Data Quality recommande des règles de qualité des données qui peuvent être utilisées pour capturer les modèles futurs. Vous pouvez copier toutes les règles qui vous sont recommandées et les appliquer à votre nœud de qualité des données pour capturer efficacement ces modèles.

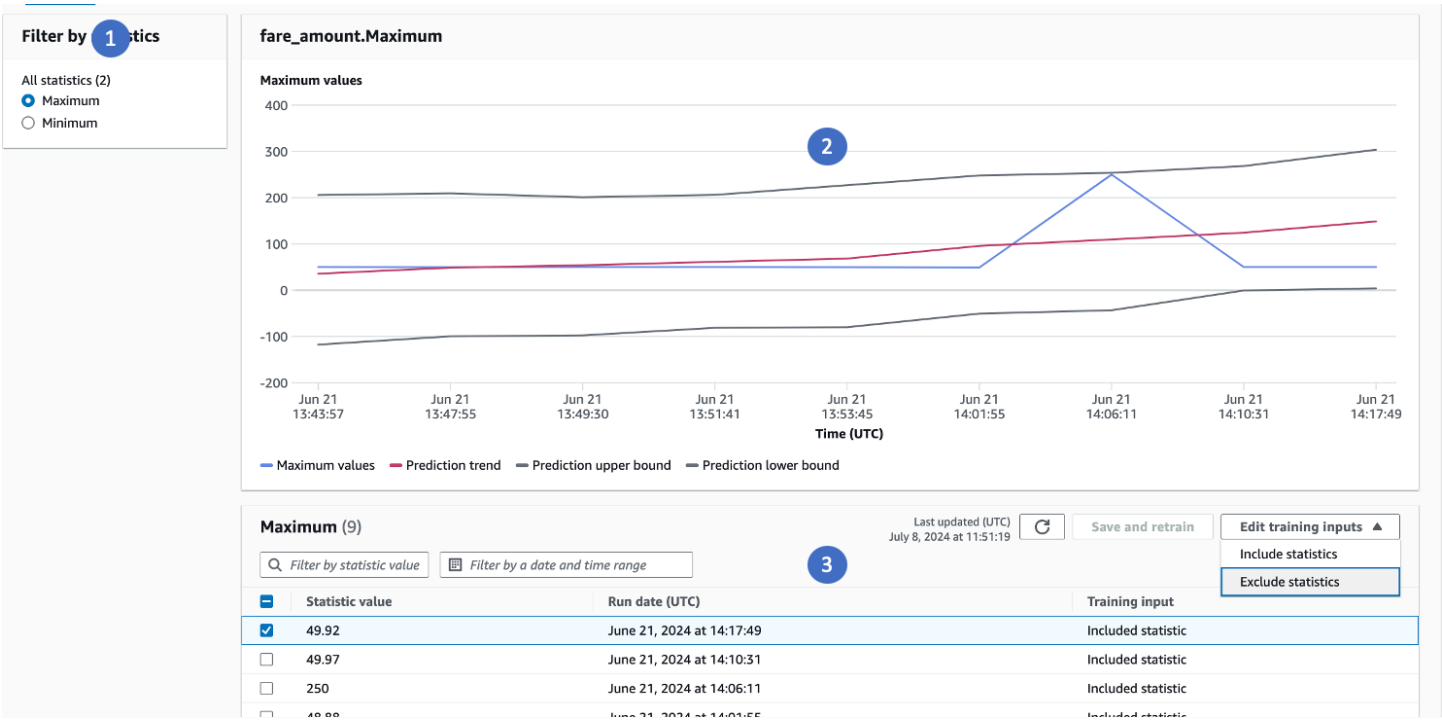
5. Vous pouvez fournir des entrées au modèle de machine learning (ML) pour exclure les valeurs anormales, afin de garantir que les futures exécutions détectent les anomalies avec précision. Si vous n'excluez pas explicitement les anomalies, AWS Glue Data Quality les prendra automatiquement en compte dans le modèle pour les prévisions futures. Il est important de noter que seule la dernière exécution reflétera les entrées de modèle que vous fournissez. Par exemple, si vous revenez en arrière et que vous avez exclu des points anormaux lors de quelques exécutions précédentes, le modèle ne reflétera pas ces modifications à moins que vous ne consultiez et ne mettiez à jour les entrées du modèle lors de la dernière exécution. Le modèle continuera à utiliser les entrées fournies précédemment jusqu'à ce que vous apportiez les ajustements nécessaires lors de la dernière exécution. En gérant activement l'exclusion des valeurs anormales, vous pouvez affiner la compréhension par le modèle de ML de ce qui constitue une anomalie par rapport à vos modèles de données et à vos exigences spécifiques, ce qui permet une détection plus précise des anomalies au fil du temps.

Affichage des statistiques des données au fil du temps et fourniture des informations d'entraînement

Parfois, vous souhaiterez peut-être consulter les statistiques ou les profils de données et voir comment ils progressent au fil du temps. Pour ce faire, choisissez Statistiques ou ouvrez l'onglet Statistiques. Vous pouvez ensuite consulter les dernières statistiques recueillies par AWS Glue Data Quality.



Cliquez sur Afficher les tendances pour voir l'évolution de chacune des statistiques au fil du temps.



1. Vous pouvez sélectionner la statistique pour une colonne spécifiée.
2. Vous pouvez suivre l'évolution des tendances.
3. Vous pouvez sélectionner des valeurs anormales et choisir de les exclure ou de les inclure. En fournissant ces informations, l'algorithme exclura ou inclura les points de données anormaux identifiés et réentraînera le modèle. Ce processus de réentraînement garantit une détection précise des anomalies à l'avenir, car le modèle tire les leçons des commentaires que vous avez fournis pour déterminer les valeurs qui doivent être considérées comme anormales ou non.

   Grâce à cette boucle de rétroaction, vous pouvez affiner la compréhension par l'algorithme de ce qui constitue une anomalie par rapport à vos modèles de données spécifiques et à vos exigences métier. En excluant les valeurs qui ne doivent pas être signalées comme des anomalies, ou en incluant des valeurs qui ont été oubliées, le modèle réentraîné pourra mieux différencier les points de données attendus des points de données réellement anormaux.

Qualité des données pour les tâches ETL dans les blocs-notes AWS Glue Studio

Dans ce didacticiel, vous apprendrez à utiliser la qualité des données deAWS Glue pour les tâches Extraction, transformation et chargement (ETL) dans les blocs-notes AWS Glue Studio.

Vous pouvez utiliser des blocs-notes dans AWS Glue Studio pour modifier des scripts de tâche et afficher le résultat sans avoir à exécuter une tâche complète. Vous pouvez également ajouter du markdown et enregistrer les blocs-notes sous forme de fichiers `.ipynb` et de scripts de tâche. Notez que vous pouvez démarrer un bloc-notes sans installation locale de logiciels ni gestion de serveurs. Lorsque vous êtes satisfait de votre code, vous pouvez utiliser AWS Glue Studio pour convertir facilement votre bloc-notes en une tâche AWS Glue.

L'ensemble de données que vous utilisez dans cet exemple comprend les données de paiement des prestataires de soins médicaux téléchargées à partir de deux ensembles de données Data.cms.gov : « Résumé des fournisseurs de systèmes de paiement potentiels pour les 100 principaux groupes liés au diagnostic - FY2011 » et « Données sur les frais d'hospitalisation pour l'exercice 2011 ».

Après le téléchargement des données, nous avons modifié le jeu de données de manière à introduire quelques enregistrements erronés à la fin du fichier. Ce fichier modifié est situé dans un compartiment Amazon S3 public à l'adresse `s3://awsglue-datasets/examples/medicare/Medicare_Hospital_Provider.csv`.

Conditions préalables

- Rôle AWS Glue avec l'autorisation Amazon S3 d'écrire dans votre compartiment Amazon S3 de destination
- Un nouveau bloc-notes (consultez Getting started with notebooks in AWS Glue Studio)

Création d'une tâche ETL dans AWS Glue Studio

Pour créer une tâche ETL

1. Changez la version de la séance pour AWS Glue 3.0.

   Pour ce faire, supprimez toutes les cellules de code standard avec le magic suivant et exécutez la cellule. Notez que ce code standard est automatiquement fourni dans la première cellule lorsqu'un nouveau bloc-notes est créé.

   ```
   %glue_version 3.0
   ```

2. Copiez et collez le code suivant et exécutez-le dans la cellule.

   ```
   import sys
   from awsglue.transforms import *
   ```

```
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job

sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
job = Job(glueContext)
```

3. Dans la cellule suivante, importez la classe `EvaluateDataQuality` qui évalue la qualité des données de AWS Glue.

```
from awsgluedq.transforms import EvaluateDataQuality
```

4. Dans la cellule suivante, lisez les données source à l'aide du fichier .csv stocké dans le compartiment public Amazon S3.

```
medicare = spark.read.format(
"csv").option(
"header", "true").option(
"inferSchema", "true").load(
's3://awsglue-datasets/examples/medicare/Medicare_Hospital_Provider.csv')
medicare.printSchema()
```

5. Convertissez les données au format AWS Glue DynamicFrame.

```
from awsglue.dynamicframe import DynamicFrame
medicare_dyf = DynamicFrame.fromDF(medicare,glueContext,"medicare_dyf")
```

6. Créez l'ensemble de règles au format DQDL (Data Quality Definition Language).

```
EvaluateDataQuality_ruleset = """
    Rules = [
        ColumnExists "Provider Id",
        IsComplete "Provider Id",
        ColumnValues  " Total Discharges " > 15
]
    ]
```

```
"""
```

7. Validez le jeu de données par rapport à l'ensemble de règles.

```
EvaluateDataQualityMultiframe = EvaluateDataQuality().process_rows(
    frame=medicare_dyf,
    ruleset=EvaluateDataQuality_ruleset,
    publishing_options={
        "dataQualityEvaluationContext": "EvaluateDataQualityMultiframe",
        "enableDataQualityCloudWatchMetrics": False,
        "enableDataQualityResultsPublishing": False,
    },
    additional_options={"performanceTuning.caching": "CACHE_NOTHING"},
)
```

8. Passez en revue les résultats.

```
ruleOutcomes = SelectFromCollection.apply(
    dfc=EvaluateDataQualityMultiframe,
    key="ruleOutcomes",
    transformation_ctx="ruleOutcomes",
)

ruleOutcomes.toDF().show(truncate=False)
```

Sortie :

```
-------------------------------------+-------
+-----------------------------------------------------
+-----------------------------------------+
|Rule                                 |Outcome|FailureReason
                  |EvaluatedMetrics                           |
+-------------------------------------+-------
+-----------------------------------------------------
+-----------------------------------------+
|ColumnExists "Provider Id"           |Passed |null
                  |{}                                         |
|IsComplete "Provider Id"             |Passed |null
                  |{Column.Provider Id.Completeness -> 1.0}   |
```

```
|ColumnValues " Total Discharges " > 15|Failed |Value: 11.0 does not meet the
 constraint requirement!|{Column. Total Discharges .Minimum -> 11.0}|
+-------------------------------------+-------
+-------------------------------------------------
+-----------------------------------------+
```

9. Filtrez les lignes ayant réussi et examinez les lignes ayant échoué dans les résultats au niveau des lignes de la qualité des données.

```
owLevelOutcomes = SelectFromCollection.apply(
dfc=EvaluateDataQualityMultiframe,
key="rowLevelOutcomes",
transformation_ctx="rowLevelOutcomes",
)

rowLevelOutcomes_df = rowLevelOutcomes.toDF() # Convert Glue DynamicFrame to
 SparkSQL DataFrame
rowLevelOutcomes_df_passed =
 rowLevelOutcomes_df.filter(rowLevelOutcomes_df.DataQualityEvaluationResult ==
 "Passed") # Filter only the Passed records.
rowLevelOutcomes_df.filter(rowLevelOutcomes_df.DataQualityEvaluationResult ==
 "Failed").show(5, truncate=False) # Review the Failed records
```

Sortie :

```
+---------------------------------------+-----------
+-----------------------------------+-------------------------+-------------
+--------------+-----------------+-------------------------------------
+------------------+-------------------------+----------------------
+-------------------------+-------------------------
+-----------------------------------------+---------------------------
+---------------------------+
|DRG Definition                           |Provider Id|Provider Name
           |Provider Street Address   |Provider City|Provider State|Provider Zip
 Code|Hospital Referral Region Description| Total Discharges | Average Covered
 Charges | Average Total Payments |Average Medicare Payments|DataQualityRulesPass
      |DataQualityRulesFail                    |DataQualityRulesSkip        |
DataQualityEvaluationResult|
+---------------------------------------+-----------
+-----------------------------------+-------------------------+-------------
+--------------+-----------------+-------------------------------------
```

```
+-----------------+-------------------------+-----------------------
+-------------------------+-------------------------
+----------------------------------------+---------------------------
+---------------------------+
|039 - EXTRACRANIAL PROCEDURES W/O CC/MCC|10005      |MARSHALL MEDICAL CENTER SOUTH
        |2505 U S HIGHWAY 431 NORTH|BOAZ         |AL            |35957
 |AL - Birmingham                      |14               |$15131.85
  |$5787.57               |$4976.71                 |[IsComplete "Provider Id"]|
[ColumnValues " Total Discharges " > 15]|[ColumnExists "Provider Id"]|Failed
              |
|039 - EXTRACRANIAL PROCEDURES W/O CC/MCC|10046      |RIVERVIEW REGIONAL MEDICAL
 CENTER    |600 SOUTH THIRD STREET    |GADSDEN      |AL            |35901
   |AL - Birmingham                      |14               |$67327.92
   |$5461.57               |$4493.57                 |[IsComplete "Provider Id"]|
[ColumnValues " Total Discharges " > 15]|[ColumnExists "Provider Id"]|Failed
              |
|039 - EXTRACRANIAL PROCEDURES W/O CC/MCC|10083      |SOUTH BALDWIN REGIONAL
 MEDICAL CENTER|1613 NORTH MCKENZIE STREET|FOLEY        |AL            |36535
        |AL - Mobile                          |15               |$25411.33
         |$5282.93               |$4383.73                 |[IsComplete "Provider
 Id"]|[ColumnValues " Total Discharges " > 15]|[ColumnExists "Provider Id"]|Failed
                   |
|039 - EXTRACRANIAL PROCEDURES W/O CC/MCC|30002      |BANNER GOOD SAMARITAN MEDICAL
 CENTER |1111 EAST MCDOWELL ROAD   |PHOENIX      |AZ            |85006
 |AZ - Phoenix                         |11               |$34803.81
  |$7768.90               |$6951.45                 |[IsComplete "Provider Id"]|
[ColumnValues " Total Discharges " > 15]|[ColumnExists "Provider Id"]|Failed
              |
|039 - EXTRACRANIAL PROCEDURES W/O CC/MCC|30010      |CARONDELET ST  MARYS HOSPITAL
        |1601 WEST ST MARY'S ROAD  |TUCSON       |AZ            |85745
 |AZ - Tucson                          |12               |$35968.50
  |$6506.50               |$5379.83                 |[IsComplete "Provider Id"]|
[ColumnValues " Total Discharges " > 15]|[ColumnExists "Provider Id"]|Failed
              |
+----------------------------------------+----------
+-----------------------------------+-------------------------+------------
+-------------+-----------------+-----------------------------------
+-----------------+-------------------------+-----------------------
+-------------------------+-------------------------
+----------------------------------------+---------------------------
+---------------------------+
only showing top 5 rows
```

Notez que AWS Glue Data Quality a ajouté quatre nouvelles colonnes (DataQualityRulesPass DataQualityRulesFail, DataQualityRulesSkip, et DataQualityEvaluationResult). Ces colonnes indiquent les enregistrements qui ont réussi, les enregistrements qui ont échoué, les règles ignorées pour l'évaluation au niveau des lignes et les résultats globaux au niveau des lignes.

10. Écrivez la sortie dans un compartiment Amazon S3 pour analyser les données et visualiser les résultats.

```
#Write the Passed records to the destination.

glueContext.write_dynamic_frame.from_options(
       frame = rowLevelOutcomes_df_passed,
       connection_type = "s3",
       connection_options = {"path": "s3://glue-sample-target/output-dir/
medicare_parquet"},
       format = "parquet")
```

Référence DQDL (Data Quality Definition Language)

Le langage DQDL (Data Quality Definition Language) est un langage spécifique au domaine que vous utilisez pour définir des règles pour AWS Glue Data Quality.

Ce guide présente les concepts clés de DQDL pour vous aider à vous familiariser avec ce langage. Il fournit également une référence pour les types de règles DQDL, avec syntaxe et exemples. Avant d'utiliser ce guide, nous vous recommandons de vous familiariser avec AWS Glue Data Quality. Pour de plus amples informations, veuillez consulter AWS Glue Qualité des données.

> **Note**
>
> DynamicRules ne sont pris en charge que dans l'AWS GlueETL.

Table des matières

- Syntaxe DQDL
  - Structure des règles
  - Règles composites

        - Fonctionnement des règles composites
    - Expressions
        - Mots-clés pour NULL, EMPTY et WHITESPACES_ONLY
        - Filtrage avec la clause Where
    - Constantes
    - Étiquettes
        - Syntaxe pour les étiquettes DQDL
            - Contraintes liées aux étiquettes
        - Récupération d'étiquettes DQDL
            - Résultats des règles
            - Résultats au niveau des lignes
            - Réponse de l'API
    - Règles dynamiques
    - Analyseurs
    - Commentaires
- Référence du type de règle DQDL
    - AggregateMatch
    - ColumnCorrelation
    - ColumnCount
    - ColumnDataType
    - ColumnExists
    - ColumnLength
    - ColumnNamesMatchPattern
    - ColumnValues
    - Intégralité
    - CustomSQL
    - DataFreshness
    - DatasetMatch
    - DistinctValuesCount
    - Entropie

- IsComplete
- IsPrimaryKey
- IsUnique
- Mean
- ReferentialIntegrity
- RowCount
- RowCountMatch
- StandardDeviation
- Somme
- SchemaMatch
- Unicité
- UniqueValueRatio
- DetectAnomalies
- FileFreshness
- FileMatch
- FileUniqueness
- FileSize

Syntaxe DQDL

Un document DQDL est sensible à la casse et contient un jeu de règles, qui regroupe les règles individuelles de qualité des données. Pour construire un jeu de règles, vous devez créer une liste nommée `Rules` (en majuscules), délimitée par une paire de crochets. La liste doit contenir une ou plusieurs règles DQDL séparées par des virgules, comme dans l'exemple suivant.

```
Rules = [
   IsComplete "order-id",
   IsUnique "order-id"
]
```

Structure des règles

La structure d'une règle DQDL dépend du type de la règle. Toutefois, les règles DQDL respectent généralement le format suivant.

```
<RuleType> <Parameter> <Parameter> <Expression>
```

`RuleType` est le nom sensible à la casse du type de règle que vous souhaitez configurer. Par exemple, `IsComplete`, `IsUnique` ou `CustomSql`. Les paramètres varient en fonction du type de règle. Pour obtenir une référence complète des types de règles DQDL et de leurs paramètres, consultez Référence du type de règle DQDL.

Règles composites

DQDL prend en charge les opérateurs logiques suivants que vous pouvez utiliser pour combiner des règles. Ces règles sont appelées règles composites.

and

L'opérateur logique `and` génère `true` si et seulement si les règles qu'il connecte sont `true`. Sinon, la règle combinée génère `false`. Chaque règle que vous connectez à l'opérateur `and` doit être entourée de parenthèses.

L'exemple suivant utilise l'opérateur `and` pour combiner deux règles DQDL.

```
(IsComplete "id") and (IsUnique "id")
```

or

L'opérateur logique `or` génère `true` si et seulement si une ou plusieurs des règles qu'il connecte sont `true`. Chaque règle que vous connectez à l'opérateur `or` doit être entourée de parenthèses.

L'exemple suivant utilise l'opérateur `or` pour combiner deux règles DQDL.

```
(RowCount "id" > 100) or (IsPrimaryKey "id")
```

Le même opérateur peut être utilisé pour connecter plusieurs règles. La combinaison de règles suivante est donc autorisée.

```
(Mean "Star_Rating" > 3) and (Mean "Order_Total" > 500) and (IsComplete "Order_Id")
```

Vous pouvez combiner les opérateurs logiques en une seule expression. Par exemple :

```
(Mean "Star_Rating" > 3) and ((Mean "Order_Total" > 500) or (IsComplete "Order_Id"))
```

Vous pouvez également créer des règles imbriquées plus complexes.

```
(RowCount > 0) or ((IsComplete "colA") and (IsUnique "colA"))
```

Fonctionnement des règles composites

Par défaut, les règles composites sont évaluées en tant que règles individuelles pour l'ensemble du jeu de données ou de la table, puis les résultats sont combinés. En d'autres termes, elles évaluent d'abord l'ensemble de la colonne, puis appliquent l'opérateur. Ce comportement par défaut est expliqué ci-dessous à l'aide d'un exemple :

```
# Dataset

+------+------+
|myCol1|myCol2|
+------+------+
|     2|     1|
|     0|     3|
+------+------+

# Overall outcome

+-------------------------------------------------------------+-------+
|Rule                                                         |Outcome|
+-------------------------------------------------------------+-------+
|(ColumnValues "myCol1" > 1) OR (ColumnValues "myCol2" > 2)|Failed |
+-------------------------------------------------------------+-------+
```

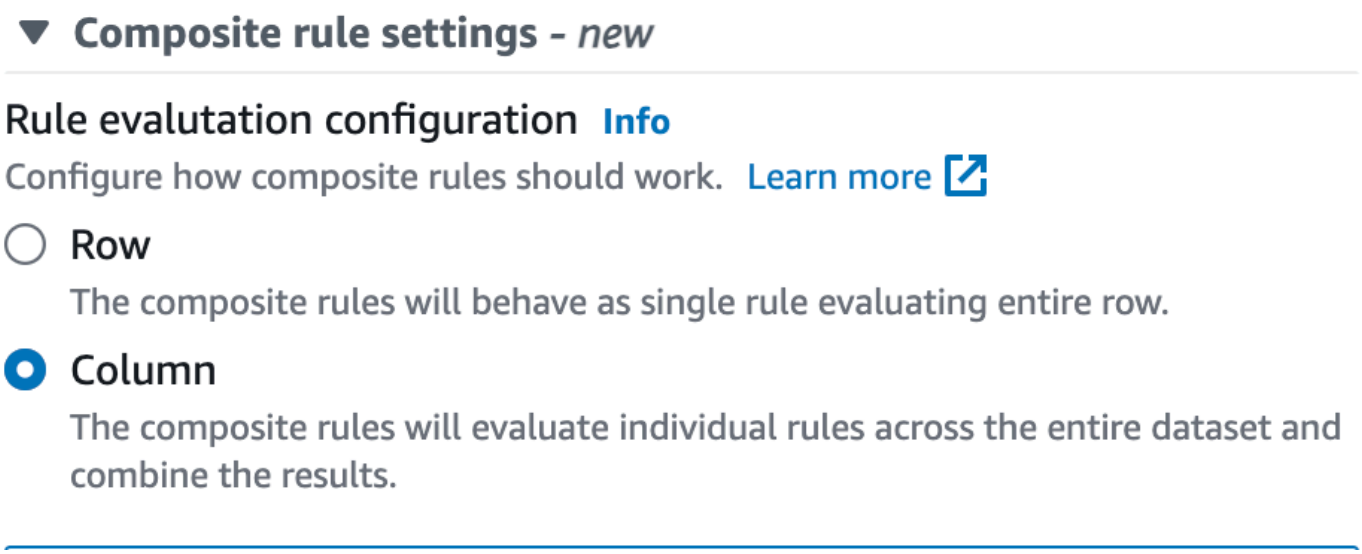
Dans l'exemple ci-dessus, AWS Glue Data Quality évalue d'abord `(ColumnValues "myCol1" > 1)`, ce qui entraînera un échec. Ensuite, il évaluera `(ColumnValues "myCol2" > 2)` ce qui entraînera également un échec. La combinaison des deux résultats sera considérée comme un ÉCHEC.

Toutefois, si vous préférez un comportement de type SQL, dans lequel vous devez évaluer la ligne entière, vous devez définir explicitement le paramètre `ruleEvaluation.scope` comme indiqué dans `additionalOptions`, dans l'extrait de code ci-dessous.

```
object GlueApp {
  val datasource = glueContext.getCatalogSource(
    database="<db>",
```

```
    tableName="<table>",
    transformationContext="datasource"
  ).getDynamicFrame()

  val ruleset = """
    Rules = [
        (ColumnValues "age" >= 26) OR (ColumnLength "name" >= 4)
    ]
  """

  val dq_results = EvaluateDataQuality.processRows(
    frame=datasource,
    ruleset=ruleset,
    additionalOptions=JsonOptions("""
        {
          "compositeRuleEvaluation.method":"ROW"
        }
      """
    )
  )
}
```

Dans AWS Glue Data Catalog, vous pouvez facilement configurer cette option dans l'interface utilisateur, comme indiqué ci-dessous.



Une fois définies, les règles composites se comporteront comme une règle unique évaluant la ligne entière. L'exemple suivant illustre ce comportement.

```
# Row Level outcome

+------+------+------------------------------------------------------------
+---------------------------+
|myCol1|myCol2|DataQualityRulesPass                                        |
DataQualityEvaluationResult|
+------+------+------------------------------------------------------------
+---------------------------+
|2     |1     |[(ColumnValues "myCol1" > 1) OR (ColumnValues "myCol2" > 2)]|Passed
               |
|0     |3     |[(ColumnValues "myCol1" > 1) OR (ColumnValues "myCol2" > 2)]|Passed
               |
+------+------+------------------------------------------------------------
+---------------------------+
```

Certaines règles ne peuvent pas être prises en charge dans cette fonctionnalité, car leur résultat global repose sur des seuils ou des ratios. Elles sont répertoriées ci-dessous.

Règles basées sur des ratios :

- Exhaustivité
- DatasetMatch
- ReferentialIntegrity
- Unicité

Règles dépendantes de seuils :

Lorsque les règles suivantes sont incluses dans un seuil, elles ne sont pas prises en charge. Cependant, les règles qui n'impliquent pas `with threshold` restent prises en charge.

- ColumnDataType
- ColumnValues
- CustomSQL

Expressions

Si un type de règle ne produit pas de réponse booléenne, vous devez fournir une expression en tant que paramètre afin de créer une réponse booléenne. Par exemple, la règle suivante vérifie la moyenne de toutes les valeurs d'une colonne par rapport à une expression afin de renvoyer un résultat vrai ou faux.

```
Mean "colA" between 80 and 100
```

Certains types de règles tels que `IsUnique` et `IsComplete` renvoient déjà une réponse booléenne.

Le tableau suivant répertorie les expressions pouvant être utilisées dans les règles DQDL.

Expressions DQDL prises en charge

Expression	Description	Exemple
=*x*	Correspond à `true` si la réponse du type de règle est égale à*x*.	`Completeness "colA" = "1.0", ColumnValues "colA" = "2022-06-30"`
!=*x*	x devient vrai si la réponse du type de règle n'est pas égale à*x*.	`ColumnValues "colA" != "a", ColumnValues "colA" != "2022-06-30"`
> *x*	Correspond à `true` si la réponse du type de règle est supérieure à*x*.	`ColumnValues "colA" > 10`
< *x*	Correspond à `true` si la réponse du type de règle est inférieure à*x*.	`ColumnValues "colA" < 1000, ColumnValues "colA" < "2022-06-30"`
>= *x*	Permet de déterminer `true` si la réponse du type de règle est supérieure ou égale à*x*.	`ColumnValues "colA" >= 10`

Expression	Description	Exemple
<= *x*	Permet de déterminer `true` si la réponse du type de règle est inférieure ou égale à*x*.	`ColumnValues "colA" <= 1000`
entre *x* et *y*	Se traduit par `true` si la réponse du type de règle se situe dans une plage spécifiée (exclusif). Utilisez ce type d'expression uniquement pour les types numériques et date.	`Mean "colA" between 8 and 100, ColumnValues "colA" between "2022-05-31" and "2022-06-30"`
pas entre *x* et *y*	Se traduit par true si la réponse du type de règle ne se situe pas dans une plage spécifiée (inclusif). Vous ne devez utiliser ce type d'expression que pour les types numériques et les dates.	`ColumnValues "colA" not between "2022-05-31" and "2022-06-30"`
dans [*a, b, c, ...*]	Se traduit par `true` si la réponse du type de règle se trouve dans le jeu spécifié.	`ColumnValues "colA" in [ 1, 2, 3 ], ColumnValues "colA" in [ "a", "b", "c" ]`
pas dans [*a, b, c, ...*]	Se traduit par `true` si la réponse du type de règle ne se trouve pas dans le jeu spécifié.	`ColumnValues "colA" not in [ 1, 2, 3 ], ColumnValues "colA" not in [ "a", "b", "c" ]`
allumettes */ab+c/i*	Se traduit par `true` si la réponse du type de règle correspond à une expression régulière.	`ColumnValues "colA" matches "[a-zA-Z]*"`

Expression	Description	Exemple
ne correspond pas */ab+c/i*	Se traduit par `true` si la réponse du type de règle ne correspond pas à une expression régulière.	`ColumnValues "colA" not matches "[a-zA-Z]*"`
`now()`	Fonctionne uniquement avec le type de règle `ColumnValues` pour créer une expression de date.	`ColumnValues "load_date" > (now() - 3 days)`

matches/in […]/not matches/notdans [...] `with threshold`	Spécifie le pourcentage de valeurs qui correspondent aux conditions de la règle. Fonctionne uniquement avec les types de règles `ColumnValues` , `ColumnDataType` et `CustomSQL` .	`ColumnValues "colA" in ["A", "B"] with threshold > 0.8,`
		`ColumnValues "colA" matches "[a-zA-Z]*" with threshold between 0.2 and 0.9`
		`ColumnDataType "colA" = "Timestamp" with threshold > 0.9`

Mots-clés pour NULL, EMPTY et WHITESPACES_ONLY

Si vous souhaitez vérifier si une colonne de chaîne contient une valeur nulle, vide ou une chaîne contenant uniquement des espaces blancs, vous pouvez utiliser les mots clés suivants :

- NULL/null : ce mot clé se traduit par true pour une valeur `null` d'une colonne de chaîne.

  `ColumnValues "colA" != NULL with threshold > 0.5` renverrait true si plus de 50 % de vos données ne contiennent pas de valeurs nulles.

  `(ColumnValues "colA" = NULL) or (ColumnLength "colA" > 5)` renverrait true pour toutes les lignes qui ont une valeur nulle ou dont la longueur est supérieure à 5. Notez que cela nécessitera l'utilisation de l'option « compositeRuleEvaluation .method » = « ROW ».
- EMPTY/empty : ce mot clé se traduit par true pour une valeur de chaîne vide (“”) dans une colonne de chaîne. Certains formats de données transforment les valeurs nulles d'une colonne de chaînes en chaînes vides. Ce mot clé permet de filtrer les chaînes vides dans vos données.

`(ColumnValues "colA" = EMPTY) or (ColumnValues "colA" in ["a", "b"])` renverrait true si une ligne est vide, « a » ou « b ». Notez que cela nécessite l'utilisation de l'option « compositeRuleEvaluation .method » = « ROW ».

- WHITESPACES_ONLY/whitespaces_only : ce mot clé prend la valeur true pour une chaîne contenant uniquement des espaces blancs (« ») dans une colonne de chaîne.

  `ColumnValues "colA" not in ["a", "b", WHITESPACES_ONLY]` renverrait true si une ligne n'est ni « a » ni « b », ni de simples espaces blancs.

  Règles prises en charge :

  - ColumnValues

Pour une expression numérique ou basée sur une date, si vous souhaitez vérifier si une colonne contient une valeur nulle, vous pouvez utiliser les mots clés suivants.

- NULL/null : ce mot clé se traduit par true pour une valeur nulle d'une colonne de chaîne.

  `ColumnValues "colA" in [NULL, "2023-01-01"]` renverrait true si une date de votre colonne est `2023-01-01` ou nulle.

  `(ColumnValues "colA" = NULL) or (ColumnValues "colA" between 1 and 9)` renverrait true pour toutes les lignes qui ont une valeur nulle ou dont les valeurs sont comprises entre 1 et 9. Notez que cela nécessitera l'utilisation de l'option « compositeRuleEvaluation .method » = « ROW ».

  Règles prises en charge :

  - ColumnValues

Filtrage avec la clause Where

> **Note**
>
> La clause Where n'est prise en charge que dans AWS Glue 4.0.

Vous pouvez filtrer vos données lorsque vous créez des règles. Cela s'avère utile lorsque vous souhaitez appliquer des règles conditionnelles.

```
<DQDL Rule> where "<valid SparkSQL where clause> "
```

Le filtre doit être spécifié avec le mot clé `where`, suivi d'une instruction SparkSQL valide placée entre guillemets (`""`).

Si vous souhaitez ajouter la clause Where à une règle avec un seuil, elle doit être spécifiée avant la condition de seuil.

```
<DQDL Rule> where "valid SparkSQL statement>" with threshold <threshold condition>
```

Avec cette syntaxe, vous pouvez écrire des règles comme les suivantes.

```
Completeness "colA" > 0.5 where "colB = 10"
ColumnValues "colB" in ["A", "B"] where "colC is not null" with threshold > 0.9
ColumnLength "colC" > 10 where "colD != Concat(colE, colF)"
```

Nous allons vérifier que l'instruction SparkSQL fournie est valide. Si elle n'est pas valide, l'évaluation des règles échouera et nous lancerons un `IllegalArgumentException` au format suivant :

```
Rule <DQDL Rule> where "<invalid SparkSQL>" has provided an invalid where clause :
<SparkSQL Error>
```

Comportement de la clause Where lorsque l'identification des enregistrements d'erreur au niveau des lignes est activée

Avec AWS Glue Data Quality, vous pouvez identifier les enregistrements spécifiques qui ont échoué. Lorsque vous appliquez une clause Where à des règles qui prennent en charge les résultats au niveau des lignes, nous étiquetons les lignes filtrées par la clause Where en tant que `Passed`.

Si vous préférez étiqueter séparément les lignes filtrées en tant que `SKIPPED`, vous pouvez définir les paramètres `additionalOptions` suivants pour la tâche ETL.

```
object GlueApp {
  val datasource = glueContext.getCatalogSource(
    database="<db>",
    tableName="<table>",
    transformationContext="datasource"
  ).getDynamicFrame()
```

```
  val ruleset = """
    Rules = [
       IsComplete "att2" where "att1 = 'a'"
    ]
  """

  val dq_results = EvaluateDataQuality.processRows(
    frame=datasource,
    ruleset=ruleset,
    additionalOptions=JsonOptions("""
        {
          "rowLevelConfiguration.filteredRowLabel":"SKIPPED"
        }
      """
    )
  )
}
```

À titre d'exemple, reportez-vous à la règle et à la trame de données suivantes :

```
IsComplete att2 where "att1 = 'a'"
```

id	att1	att2	Résultats au niveau des lignes (par défaut)	Résultats au niveau des lignes (option ignorée)	Commentaires
1	a	f	PASSED	PASSED	
2	b	d	PASSED	SKIPPED	La ligne est filtrée, car `att1` n'est pas `"a"`.
3	a	null	ÉCHEC	ÉCHEC	
4	a	f	PASSED	PASSED	
5	b	null	PASSED	SKIPPED	La ligne est filtrée, car

id	att1	att2	Résultats au niveau des lignes (par défaut)	Résultats au niveau des lignes (option ignorée)	Commentaires
					`att1` n'est pas `"a"`.
6	a	f	PASSED	PASSED	

Constantes

Dans DQDL, vous pouvez définir des valeurs constantes et les référencer tout au long de votre script. Cela permet d'éviter les problèmes liés aux limites de taille des requêtes, par exemple lorsque vous travaillez avec des instructions SQL volumineuses susceptibles de dépasser les limites autorisées. En affectant ces valeurs aux constantes, vous pouvez simplifier votre DQDL et éviter d'atteindre ces limites.

L'exemple suivant montre comment définir et utiliser une constante :

```
mySql = "select count(*) from primary"

Rules = [
    CustomSql $mySql between 0 and 100
]
```

Dans cet exemple, la requête SQL est affectée à la constante`mySql`, qui est ensuite référencée dans la règle à l'aide du $ préfixe.

Étiquettes

Les étiquettes constituent un moyen efficace d'organiser et d'analyser les résultats relatifs à la qualité des données. Vous pouvez interroger les résultats à l'aide d'étiquettes spécifiques pour identifier les règles défaillantes au sein de catégories particulières, compter les résultats des règles par équipe ou par domaine et créer des rapports ciblés pour les différentes parties prenantes.

Par exemple, vous pouvez appliquer toutes les règles relatives à l'équipe financière à l'aide d'une étiquette `"team=finance"` et générer un rapport personnalisé pour présenter les indicateurs de

qualité spécifiques à l'équipe financière. Vous pouvez étiqueter les règles de haute priorité avec `"criticality=high"` pour hiérarchiser les efforts de correction. Les étiquettes peuvent être créées dans le cadre du DQDL. Vous pouvez interroger les étiquettes dans le cadre des résultats des règles, des résultats au niveau des lignes et des réponses des API, ce qui facilite l'intégration à vos flux de travail de surveillance et de reporting existants.

> **Note**
>
> Les étiquettes ne sont disponibles que dans AWS Glue ETL et ne sont pas disponibles dans le Data Quality basé sur le catalogue de données AWS Glue.

Syntaxe pour les étiquettes DQDL

Le DQDL prend en charge les étiquettes par défaut et spécifiques aux règles. Les libellés par défaut sont définis au niveau de l'ensemble de règles et s'appliquent automatiquement à toutes les règles de cet ensemble de règles. Les règles individuelles peuvent également avoir leurs propres étiquettes, et comme les étiquettes sont implémentées sous forme de paires clé-valeur, les étiquettes spécifiques aux règles peuvent remplacer les étiquettes par défaut lorsque vous utilisez la même clé.

L'exemple suivant montre comment utiliser les libellés par défaut et spécifiques aux règles :

```
DefaultLabels=["frequency"="monthly"]

Rules = [
    // Auto includes the default label ["frequency"="monthly"]
    ColumnValues "col" > 21,
    // Add ["foo"="bar"] to default label. Labels for this rule would be
 ["frequency"="monthly", "foo"="bar"]
    Rule 1 with threshold > 0.8 labels=["foo"="bar"],
    // Override default label. Labels for this rule would be ["frequency"="daily",
 "foo"="bar"]
    Rule 2 with threshold > 0.8 labels=["foo"="bar", "frequency"="daily"]
    // Labels must be applied to the entire composite rule (parentheses required)
    (Rule 1 AND Rule 2) labels=["foo"="bar]
]
```

Contraintes liées aux étiquettes

Les étiquettes sont soumises aux contraintes suivantes :

- Un maximum de 10 étiquettes par règle DQDL.
- Les étiquettes sont spécifiées sous forme de liste de paires clé-valeur.
- La clé et la valeur de l'étiquette distinguent les majuscules et minuscules.
- La longueur maximale de la clé d'étiquette est de 128 caractères. La clé d'étiquette ne doit pas être vide ou nulle.
- La longueur maximale de la valeur d'étiquette est de 256 caractères. La valeur de l'étiquette peut être vide ou nulle.

Récupération d'étiquettes DQDL

Vous pouvez récupérer des étiquettes DQDL à partir des résultats des règles, des résultats au niveau des lignes et des réponses d'API.

Résultats des règles

Les étiquettes DQDL sont toujours visibles dans les résultats des règles. Aucune configuration supplémentaire n'est nécessaire pour les activer.

Résultats au niveau des lignes

Les étiquettes DQDL sont désactivées par défaut dans les résultats au niveau des lignes, mais elles peuvent être activées dans. `AdditionalOptions` `EvaluateDataQuality`

L'exemple suivant montre comment activer les étiquettes dans les résultats au niveau des lignes :

```
val evaluateResult = EvaluateDataQuality.processRows(
    frame=AmazonS3_node1754591511068,
    ruleset=example_ruleset,
    publishingOptions=JsonOptions("""{
        "dataQualityEvaluationContext": "evaluateResult",
        "enableDataQualityCloudWatchMetrics": "true",
        "enableDataQualityResultsPublishing": "true"
    }"""),
    additionalOptions=JsonOptions("""{
        "performanceTuning.caching":"CACHE_NOTHING",
        "observations.scope":"ALL",
        "rowLevelConfiguration.ruleWithLabels":"ENABLED"
    }""")
)
```

Lorsqu'elle est activée, la trame de données de résultats au niveau des lignes inclut des étiquettes pour chaque règle dans les colonnes`DataQualityRulesPass`, et`DataQualityRulesFail`. `DataQualityRulesSkip`

Réponse de l'API

Les étiquettes DQDL sont toujours visibles dans les réponses de l'API sous un nouveau champ `Labels` de l'`RuleResults`objet.

L'exemple suivant montre des libellés dans une réponse d'API :


```
{
    "ResultId": "dqresult-example",
    "ProfileId": "dqprofile-example",
    "Score": 0.6666666666666666,
    "RulesetName": "EvaluateDataQuality_node1754591514205",
    "EvaluationContext": "EvaluateDataQuality_node1754591514205",
    "StartedOn": "2025-08-22T19:36:10.448000+00:00",
    "CompletedOn": "2025-08-22T19:36:16.368000+00:00",
    "JobName": "anniezc-test-labels",
    "JobRunId": "jr_068f6d7a45074d9105d14e4dee09db12c3b95664b45f6ee44fa29ed7e5619ba8",
    "RuleResults": [
        {
            "Name": "Rule_0",
            "Description": "IsComplete colA",
            "EvaluationMessage": "Input data does not include column colA!",
            "Result": "FAIL",
            "EvaluatedMetrics": {},
            "EvaluatedRule": "IsComplete colA",
            "Labels": {
                "frequency": "monthly"
            }
        },
        {
            "Name": "Rule_1",
            "Description": "Rule 1 with threshold > 0.8",
            "Result": "PASS",
            "EvaluatedMetrics": {},
            "EvaluatedRule": "Rule 1 with threshold > 0.8",
            "Labels": {
                "frequency": "monthly",
                "foo": "bar"
            }
```

```
        },
        {
            "Name": "Rule_3",
            "Description": "Rule 2 with threshold > 0.8",
            "Result": "PASS",
            "EvaluatedMetrics": {},
            "EvaluatedRule": "Rule 2 with threshold > 0.8",
            "Labels": {
                "frequency": "daily",
                "foo": "bar"
            }
        }
    ]
}
```

Règles dynamiques

> **Note**
>
> Les règles dynamiques ne sont prises en charge que dans AWS Glue ETL et ne le sont pas dans AWS Glue Data Catalog.

Vous pouvez désormais créer des règles dynamiques pour comparer les métriques actuelles produites par vos règles avec leurs valeurs historiques. Ces comparaisons historiques sont rendues possibles en utilisant l'opérateur `last()` dans les expressions. Par exemple, la règle `RowCount > last()` fonctionne lorsque le nombre de lignes de l'exécution en cours est supérieur au nombre de lignes précédent le plus récent pour le même jeu de données. `last()` prend un argument de nombre naturel facultatif décrivant le nombre de métriques précédentes à prendre en compte ; `last(k)` où `k >= 1` renvoyera aux dernières métriques `k`.

- Si aucun point de données n'est disponible, `last(k)` renverra la valeur par défaut 0,0.
- Si moins de `k` métriques sont disponibles, `last(k)` renverra toutes les métriques précédentes.

Pour former des expressions valides, utilisez `last(k)`, où `k > 1` nécessite une fonction d'agrégation pour réduire plusieurs résultats historiques en un seul nombre. Par exemple, `RowCount > avg(last(5))` vérifiera si le nombre de lignes du jeu de données actuel est strictement supérieur à la moyenne des cinq dernières lignes du même jeu de données. `RowCount > last(5)` générera

une erreur, car le nombre de lignes du jeu de données actuel ne peut pas être comparé de manière significative à une liste.

Fonctions d'agrégation prises en charge :

- `avg`
- `median`
- `max`
- `min`
- `sum`
- `std` (écart-type)
- `abs` (valeur absolue)
- `index(last(k), i)` permet de sélectionner la `i`e valeur la plus récente parmi les dernières `k`. `i` est indexé à partir de zéro, donc `index(last(3), 0)` renverra le point de données le plus récent et `index(last(3), 3)` générera une erreur, car il n'y a que trois points de données et nous essayons d'indexer le 4e le plus récent.

Exemples d'expressions

ColumnCorrelation

- `ColumnCorrelation "colA" "colB" < avg(last(10))`

DistinctValuesCount

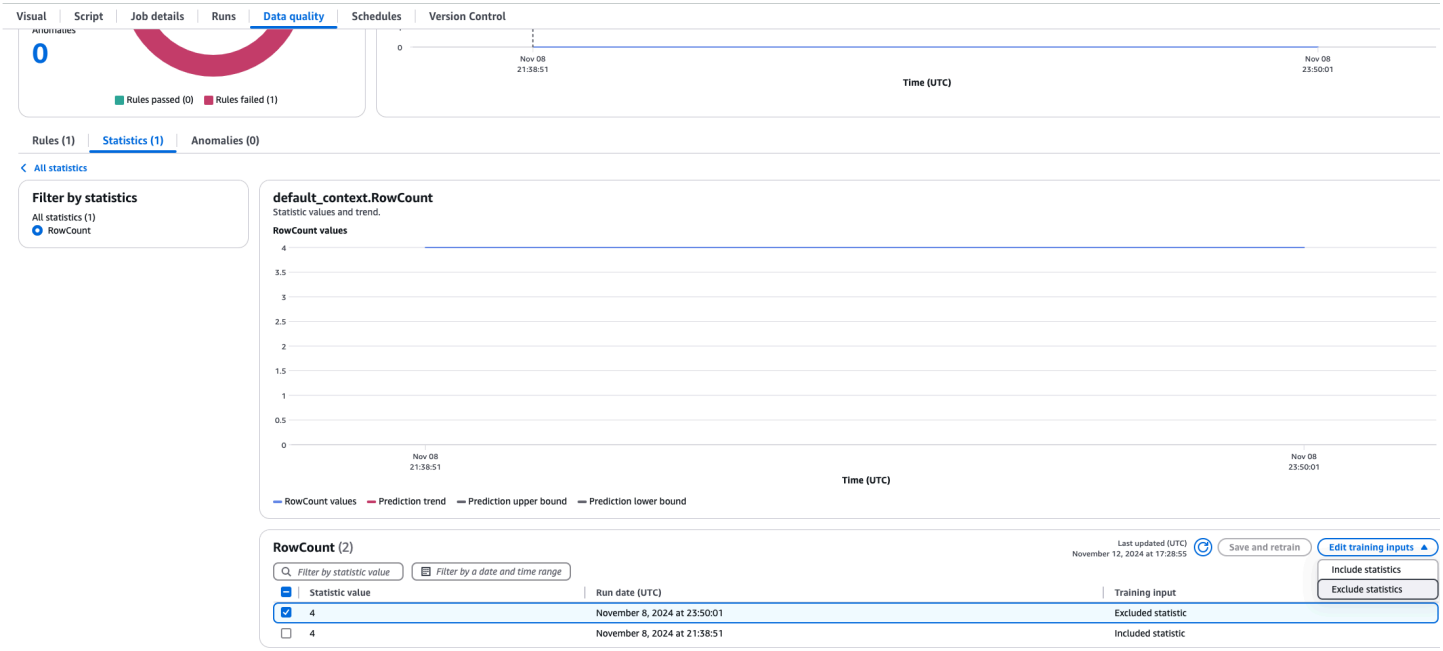
- `DistinctValuesCount "colA" between min(last(10))-1 and max(last(10))+1`

La plupart des types de règles comportant des conditions ou des seuils numériques prennent en charge les règles dynamiques. Consultez le tableau fourni, Analyseurs et règles pour vérifier si les règles dynamiques sont disponibles pour le type de règle que vous utilisez.

Exclure les statistiques des règles dynamiques

Vous devrez parfois exclure les statistiques de données de vos calculs de règles dynamiques. Supposons que vous ayez effectué un chargement de données historiques et que vous ne vouliez pas que cela ait un impact sur vos moyennes. Pour ce faire, ouvrez le job dans AWS Glue ETL et

choisissez l'onglet Data Quality, puis sélectionnez Statistics et sélectionnez les statistiques que vous souhaitez exclure. Vous pourrez voir un graphique de tendance ainsi qu'un tableau de statistiques. Sélectionnez les valeurs que vous souhaitez exclure, puis choisissez Exclure les statistiques. Les statistiques exclues ne seront désormais pas incluses dans les calculs de règles dynamiques.



Analyseurs

> **Note**
>
> Les analyseurs ne sont pas pris en charge dans AWS Glue Data Catalog.

Les règles DQDL utilisent des fonctions appelées analyseurs pour rassembler des informations sur vos données. Ces informations sont utilisées par l'expression booléenne d'une règle afin de décider si cette dernière doit être considérée comme réussie ou échouée. Par exemple, la RowCount règle `RowCount > 5` utilisera un analyseur de nombre de lignes pour découvrir le nombre de lignes de votre ensemble de données et comparer ce nombre à l'expression `> 5` pour vérifier s'il existe plus de cinq lignes dans le jeu de données actuel.

Il est parfois préférable, plutôt que de créer des règles, de créer des analyseurs et de les configurer pour qu'ils génèrent des statistiques utiles à la détection d'anomalies. Dans de tels cas, vous pouvez créer des analyseurs. Les analyseurs se distinguent des règles de la manière suivante.

Caractéristiques	Analyseurs	Rules
Partie du jeu de règles	Oui	Oui
Génère des statistiques	Oui	Oui
Génère des observations	Oui	Oui
Peut évaluer et affirmer une condition	Non	Oui
Vous pouvez configurer des actions telles que l'arrêt des tâches en cas d'échec ou la poursuite du traitement des tâches	Non	Oui

Les analyseurs peuvent exister indépendamment sans règles, vous offrant ainsi la possibilité de les configurer rapidement et de créer progressivement des règles de qualité des données.

Certains types de règles peuvent être saisis dans le bloc `Analyzers` de votre jeu de règles afin d'exécuter les règles nécessaires aux analyseurs et rassembler des informations sans avoir à valider une quelconque condition. Certains analyseurs ne sont pas associés à des règles et ne peuvent être saisis que dans le bloc `Analyzers`. Le tableau ci-dessous précise pour chaque élément s'il est pris en charge en tant que règle ou en tant qu'analyseur autonome, avec des informations supplémentaires pour chaque type de règle.

Exemple d'ensemble de règles avec analyseur

Le jeu de règles suivant utilise :

- une règle dynamique permettant de vérifier si la croissance d'un jeu de données est supérieure à sa moyenne mobile au cours des trois dernières exécutions de tâches
- un analyseur `DistinctValuesCount` permettant d'enregistrer le nombre de valeurs distinctes dans la colonne `Name` du jeu de données
- un analyseur `ColumnLength` permettant de suivre la taille `Name` minimale et maximale au fil du temps

Les résultats des métriques de l'analyseur peuvent être consultés dans l'onglet Qualité des données de votre exécution de tâche.

```
Rules = [
   RowCount > avg(last(3))
]
Analyzers = [
   DistinctValuesCount "Name",
   ColumnLength "Name"
]
```

AWS Glue Data Quality prend en charge les analyseurs suivants.

Nom de l'analyseur	Fonctionnalité
`RowCount`	Calcule le nombre de lignes pour un jeu de données
`Completeness`	Calcule le pourcentage de complétude d'une colonne
`Uniqueness`	Calcule le pourcentage d'unicité d'une colonne
`Mean`	Calcule la moyenne d'une colonne numérique
`Sum`	Calcule la somme d'une colonne numérique
`StandardDeviation`	Calcule l'écart type d'une colonne numérique
`Entropy`	Calcule l'entropie d'une colonne numérique
`DistinctValuesCount`	Calcule le nombre de valeurs distinctes dans une colonne
`UniqueValueRatio`	Calcule le ratio de valeurs uniques dans une colonne
`ColumnCount`	Calcule le nombre de colonnes d'un jeu de données

Nom de l'analyseur	Fonctionnalité
`ColumnLength`	Calcule la longueur d'une colonne
`ColumnValues`	Calcule le minimum et le maximum pour les colonnes numériques Calcule le minimum ColumnLength et le maximum ColumnLength pour les colonnes non numériques
`ColumnCorrelation`	Calcule les corrélations de colonnes pour des colonnes données
`CustomSql`	Calcule les statistiques renvoyées par le CustomSQL
`AllStatistics`	Calcule les statistiques suivantes : • RowCount, ColumnCount • Chaque colonne : complétude, unicité • Numérique : min, max, entropie, moyenne, développement standard, somme • Chaîne : MinLength, MaxLength

Commentaires

Vous pouvez utiliser le caractère « # » pour ajouter un commentaire à votre document DQDL. Tout ce qui se trouve après le caractère « # » et jusqu'à la fin de la ligne est ignoré par DQDL.

```
Rules = [
    # More items should generally mean a higher price, so correlation should be
 positive
    ColumnCorrelation "price" "num_items" > 0
]
```

Référence du type de règle DQDL

Cette section fournit une référence pour chaque type de règle pris en charge par AWS Glue Data Quality.

> **Note**
>
> - Le langage DQDL ne prend actuellement pas en charge les données de colonnes imbriquées ou de type liste.
> - Les valeurs entre crochets dans le tableau ci-dessous seront remplacées par les informations fournies dans les arguments des règles.
> - Les règles nécessitent généralement un argument supplémentaire pour l'expression.

Ruletype	Descripti on	Argumer	Métrique rapporté s	Prise en charge en tant que règle ?	Prise en charge en tant qu'analys eur ?	Renvoie des résultats au niveau des lignes ?	Prise en charge des règles dynamiq s ?	Génère des observat ons	Prise en charge de la syntaxe de clause Where ?
Aggregat Match	Vérifie si deux jeux de données correspo dent en compara des métrique récapitul atives telles que le montant total	Une ou plusieurs agrégatic ns	Lorsque les noms de la première et de la deuxièm colonne d'agréga ion correspo dent : `Column [C olumn]`	Oui	Non	Non	Non	Non	Non

Ruletype	Descripti on	Argumer	Métrique rapporté s	Prise en charge en tant que règle ?	Prise en charge en tant qu'analys eur ?	Renvoie des résultats au niveau des lignes ?	Prise en charge des règles dynamiq s ?	Génère des observat ons	Prise en charge de la syntaxe de clause Where ?

	des ventes. Utile pour permettr aux instituti ons financièr es de vérifier si toutes les données sont ingérées à partir de systèmes sources.		`gregate tch`
			
			Lorsque les noms de la première et de la deuxièm colonne d'agréga ion sont différent s :
			
			`Column. [C olumn1, lumn2]. gregate tch`						

Ruletype	Descripti on	Argumer	Métrique rapporté s	Prise en charge en tant que règle ?	Prise en charge en tant qu'analys eur ?	Renvoie des résultats au niveau des lignes ?	Prise en charge des règles dynamiq s ?	Génère des observat ons	Prise en charge de la syntaxe de clause Where ?

AllStatis tics	Analyseu autonom permetta t de rassemb r plusieurs métrique pour la colonne fournie d'un jeu de données	Un seul nom de colonne	Pour les colonnes de tous types :
			`Dataset .RowCou`
			`Column. [C olumn]. mpleter s`
			`Column. [C olumn]. iquenes`
			Métrique supplém taires pour les colonnes à
				Non	Oui	Non	Non	Non	Non

Ruletype	Descripti on	Argumer	Métrique rapporté s	Prise en charge en tant que règle ?	Prise en charge en tant qu'analys eur ?	Renvoie des résultats au niveau des lignes ?	Prise en charge des règles dynamiq s ?	Génère des observat ons	Prise en charge de la syntaxe de clause Where ?

			valeur de chaîne :
			
			`ColumnL gth metrics`
			
			Métrique supplém taires pour les colonnes à valeur numériqu
			
			`Column\ ues metrics`						

Ruletype	Descripti on	Argumer	Métrique rapporté s	Prise en charge en tant que règle ?	Prise en charge en tant qu'analys eur ?	Renvoie des résultats au niveau des lignes ?	Prise en charge des règles dynamiq s ?	Génère des observat ons	Prise en charge de la syntaxe de clause Where ?
ColumnC relation	Vérifie dans quelle mesure deux colonnes sont corrélées .	Exactem t deux noms de colonne	`Multic mn. [Colum n1,Colu 2].Colu Correla on`	Oui	Oui	Non	Oui	Non	Oui
ColumnC nt	Vérifie si des colonnes sont supprimé s.	Aucune	`Dataset .Columr unt`	Oui	Oui	Non	Oui	Oui	Non
ColumnD aType	Vérifie si une colonne est conforme à un type de données	Exactem t un nom de colonne	`Column. [C olumn]. lumnDat ype.Cor iance`	Oui	Non	Non	Oui, dans l'express ion de seuil au niveau de la ligne	Non	Oui

Ruletype	Descripti on	Argumer	Métrique rapporté s	Prise en charge en tant que règle ?	Prise en charge en tant qu'analys eur ?	Renvoie des résultats au niveau des lignes ?	Prise en charge des règles dynamiq s ?	Génère des observat ons	Prise en charge de la syntaxe de clause Where ?
ColumnE sts	Vérifie si des colonnes existent dans un jeu de données Cela permet aux clients de créer des plateform es de données en libre-service pour s'assurer que certaines colonnes sont	Exactem t un nom de colonne	N/A	Oui	Non	Non	Non	Non	Non

Ruletype	Descripti on	Argumer	Métrique rapporté s	Prise en charge en tant que règle ?	Prise en charge en tant qu'analys eur ?	Renvoie des résultats au niveau des lignes ?	Prise en charge des règles dynamiq s ?	Génère des observat ons	Prise en charge de la syntaxe de clause Where ?
	disponibl es.								

Ruletype	Descripti on	Argumer	Métrique rapporté s	Prise en charge en tant que règle ?	Prise en charge en tant qu'analys eur ?	Renvoie des résultats au niveau des lignes ?	Prise en charge des règles dynamiq s ?	Génère des observat ons	Prise en charge de la syntaxe de clause Where ?

ColumnL gth	Vérifie si la longueur des données est cohérent .	Exactem t un nom de colonne	`Column [C olumn]. ximumLe th`
			
			`Column [C olumn]. nimumLe th`
			
			Métrique supplém taire lorsque le seuil au niveau de la ligne est fourni :
			
			`Column [C olumn]. lumnVa`
				Oui	Oui	Oui, lorsque le seuil au niveau de la ligne est fourni	Non	Oui. Génère uniquem t des observat ons en analysan les longueur minimale et maximal	Oui

Ruletype	Descripti on	Argumer	Métrique rapporté s	Prise en charge en tant que règle ?	Prise en charge en tant qu'analys eur ?	Renvoie des résultats au niveau des lignes ?	Prise en charge des règles dynamiq s ?	Génère des observat ons	Prise en charge de la syntaxe de clause Where ?
			`s.Compl nce`						
ColumnN esMatchl ttern	Vérifie si les noms de colonnes correspo dent aux modèles définis. Utile pour les équipes de gouverna ce afin d'assure la cohéren des noms de colonnes	Une expressi n régulière pour les noms de colonne	`Dataset .Columr mesPatt nMatchR io`	Oui	Non	Non	Non	Non	Non

Ruletype	Descripti on	Argumer	Métrique rapporté s	Prise en charge en tant que règle ?	Prise en charge en tant qu'analys eur ?	Renvoie des résultats au niveau des lignes ?	Prise en charge des règles dynamiq s ?	Génère des observat ons	Prise en charge de la syntaxe de clause Where ?

ColumnV ues	Vérifie si les données sont cohérent s par rapport aux valeurs définies. Cette règle prend en charge les expressi ns régulière s.	Exactem t un nom de colonne	`Column [C olumn]. ximum`
			
			`Column [C olumn]. nimum`
			
			Métrique supplém taire lorsque le seuil au niveau de la ligne est fourni :
			
			`Column [C olumn]. lumnVa s.Comp nce`
				Oui	Oui	Oui, lorsque le seuil au niveau de la ligne est fourni	Non	Oui. Génère uniquem t des observat ons en analysan les valeurs minimale et maximal	Oui

Ruletype	Descripti on	Argumer	Métrique rapporté s	Prise en charge en tant que règle ?	Prise en charge en tant qu'analys eur ?	Renvoie des résultats au niveau des lignes ?	Prise en charge des règles dynamiq s ?	Génère des observat ons	Prise en charge de la syntaxe de clause Where ?
Exhausti ité	Vérifie la présence de données vides ou NULLs incomplè es.	Exactem t un nom de colonne	`Column [C olumn]. mpleter s`	Oui	Oui	Oui	Oui	Oui	Oui

Ruletype	Descripti on	Argumer	Métrique rapporté s	Prise en charge en tant que règle ?	Prise en charge en tant qu'analys eur ?	Renvoie des résultats au niveau des lignes ?	Prise en charge des règles dynamiq s ?	Génère des observat ons	Prise en charge de la syntaxe de clause Where ?
CustomS	Les clients peuvent implémer er presque tous les types de contrôles de qualité des données dans SQL.	Une instructi on SQL (Facultat if) Un seuil au niveau de la ligne	Dataset .Custor L Métrique supplém taire lorsque le seuil au niveau de la ligne est fourni : Dataset .Custor L.Compl nce	Oui	Non	Oui, lorsque le seuil au niveau de la ligne est fourni	Oui	Non	Non
DataFres ness	Vérifie si les données sont récentes	Exactem t un nom de colonne	Column [C olumn]. taFresh ss.Comp ance	Oui	Non	Oui	Non	Non	Oui

Ruletype	Descripti on	Argumer	Métrique rapporté s	Prise en charge en tant que règle ?	Prise en charge en tant qu'analys eur ?	Renvoie des résultats au niveau des lignes ?	Prise en charge des règles dynamiq s ?	Génère des observat ons	Prise en charge de la syntaxe de clause Where ?
DatasetM tch	Compare deux jeux de données et détermin s'ils sont synchror sés.	Nom d'un jeu de données de référence Un mappage de colonnes (Facultat if) Colonne pour vérifier les correspo dances	`Dataset [ Referer Dataset ias].Da setMatc`	Oui	Non	Oui	Oui	Non	Non
DistinctV aluesCou t	Vérifie la présence de valeurs dupliqué s.	Exactem t un nom de colonne	`Column [C olumn]. stinctV uesCour`	Oui	Oui	Oui	Oui	Oui	Oui

Ruletype	Descripti on	Argumer	Métrique rapporté s	Prise en charge en tant que règle ?	Prise en charge en tant qu'analys eur ?	Renvoie des résultats au niveau des lignes ?	Prise en charge des règles dynamiq s ?	Génère des observat ons	Prise en charge de la syntaxe de clause Where ?
DetectAr malies	Vérifie la présence d'anoma es dans les métrique rapporté s par un autre type de règle.	Un type de règle	Métrique s) rapporté (s) par l'argume t du type de règle	Oui	Non	Non	Non	Non	Non
Entropie	Vérifie l'entropi e des données	Exactem t un nom de colonne	`Column [C olumn] tropy`	Oui	Oui	Non	Oui	Non	Oui

Ruletype	Descripti on	Argumer	Métrique rapporté s	Prise en charge en tant que règle ?	Prise en charge en tant qu'analys eur ?	Renvoie des résultats au niveau des lignes ?	Prise en charge des règles dynamiq s ?	Génère des observat ons	Prise en charge de la syntaxe de clause Where ?
IsComple e	Vérifie si 100 % des données sont complète .	Exactem t un nom de colonne	`Column [C olumn]. mpleter s`	Oui	Non	Oui	Non	Non	Oui

Ruletype	Descripti on	Argumer	Métrique rapporté s	Prise en charge en tant que règle ?	Prise en charge en tant qu'analys eur ?	Renvoie des résultats au niveau des lignes ?	Prise en charge des règles dynamiq s ?	Génère des observat ons	Prise en charge de la syntaxe de clause Where ?

IsPrimar Key	Vérifie si une colonne est une clé primaire (non NULL et unique).	Exactem t un nom de colonne	Pour une seule colonne
			
			`Column. [C olumn]. iquenes`
			
			Pour plusieurs colonnes
			
			`Multicc mn. [Comma Delimit Columns Uniquer s`
				Oui	Non	Oui	Non	Non	Oui

Ruletype	Descripti on	Argumer	Métrique rapporté s	Prise en charge en tant que règle ?	Prise en charge en tant qu'analys eur ?	Renvoie des résultats au niveau des lignes ?	Prise en charge des règles dynamiq s ?	Génère des observat ons	Prise en charge de la syntaxe de clause Where ?
IsUnique	Vérifie si 100 % des données sont uniques.	Exactem t un nom de colonne	`Column [C olumn]. iquenes`	Oui	Non	Oui	Non	Non	Oui
Mean	Vérifie si la moyenne correspo d au seuil défini.	Exactem t un nom de colonne	`Column [C olumn]. an`	Oui	Oui	Oui	Oui	Non	Oui

Ruletype	Description	Argumer	Métrique rapporté s	Prise en charge en tant que règle ?	Prise en charge en tant qu'analys eur ?	Renvoie des résultats au niveau des lignes ?	Prise en charge des règles dynamiq s ?	Génère des observat ons	Prise en charge de la syntaxe de clause Where ?
Referent alIntegri ty	Vérifie si deux jeux de données ont une intégrité référenti elle.	Un ou plusieurs noms de colonne provenar du jeu de données Un ou plusieurs noms de colonne provenar du jeu de données de référence	`Column [R eferenc ataset/ as].Re ential egrity`	Oui	Non	Oui	Oui	Non	Non

Ruletype	Description	Argumer	Métrique rapporté s	Prise en charge en tant que règle ?	Prise en charge en tant qu'analys eur ?	Renvoie des résultats au niveau des lignes ?	Prise en charge des règles dynamiq s ?	Génère des observat ons	Prise en charge de la syntaxe de clause Where ?
RowCou	Vérifie si le nombre d'enregis trements correspo d à un seuil.	Aucune	`Dataset .RowCou`	Oui	Oui	Non	Oui	Oui	Oui
RowCou atch	Vérifie si le nombre d'enregis trements entre deux jeux de données correspo d.	Alias de jeu de données de référence	`Dataset [ Referer Dataset ias].Ro ountMat`	Oui	Non	Non	Oui	Non	Non

Ruletype	Descripti on	Argumer	Métrique rapporté s	Prise en charge en tant que règle ?	Prise en charge en tant qu'analys eur ?	Renvoie des résultats au niveau des lignes ?	Prise en charge des règles dynamiq s ?	Génère des observat ons	Prise en charge de la syntaxe de clause Where ?
Standarc eviation	Vérifie si l'écart type correspo d au seuil.	Exactem t un nom de colonne	`Column [C olumn]. andardD iation`	Oui	Oui	Oui	Oui	Non	Oui
Schemal ch	Vérifie si le schéma entre deux jeux de données correspo d.	Alias de jeu de données de référence	`Dataset [ Referer Dataset ias].Sc maMatch`	Oui	Non	Non	Oui	Non	Non
Somme	Vérifie si la somme correspo d à un seuil défini.	Exactem t un nom de colonne	`Column [C olumn]. m`	Oui	Oui	Non	Oui	Non	Oui

Ruletype	Descripti on	Argumer	Métrique rapporté s	Prise en charge en tant que règle ?	Prise en charge en tant qu'analys eur ?	Renvoie des résultats au niveau des lignes ?	Prise en charge des règles dynamiq s ?	Génère des observat ons	Prise en charge de la syntaxe de clause Where ?
Unicité	Vérifie si l'unicité du jeu de données correspo d au seuil.	Exactem t un nom de colonne	`Column [C olumn]. iquenes`	Oui	Oui	Oui	Oui	Non	Oui
UniqueV ueRatio	Vérifie si le ratio de valeur unique correspo d au seuil.	Exactem t un nom de colonne	`Column [C olumn]. iqueVa Ratio`	Oui	Oui	Oui	Oui	Non	Oui

FileFresh ness	Vérifie si les fichiers dans Amazon S3 sont récents.	Chemin d'accès au fichier ou au dossier et seuil.	`Dataset .FileF hness.( pliance`
			
			`Dataset .FileCo t`
				Oui	Non	Non	Non	Non	Non

Ruletype	Description	Argumer	Métrique rapporté s	Prise en charge en tant que règle ?	Prise en charge en tant qu'analys eur ?	Renvoie des résultats au niveau des lignes ?	Prise en charge des règles dynamiq s ?	Génère des observat ons	Prise en charge de la syntaxe de clause Where ?
FileMatcl	Vérifie si le contenu du fichier correspo d à une somme de contrôle ou à un autre fichier. Cette règle utilise des sommes de contrôle pour s'assurer que deux fichiers	Chemin d'accès au fichier ou au dossier source et chemin d'accès au fichier ou au dossier cible.	Aucune statistiq ue n'est générée.	Oui	Non	Non	Non	Non	Non

Ruletype	Description	Argumer	Métrique rapporté s	Prise en charge en tant que règle ?	Prise en charge en tant qu'analys eur ?	Renvoie des résultats au niveau des lignes ?	Prise en charge des règles dynamiq s ?	Génère des observat ons	Prise en charge de la syntaxe de clause Where ?
	sont identique s.								

FileSize	Vérifie si la taille d'un fichier correspo d à une condition spécifiée .	Chemin d'accès au fichier ou au dossier et seuil.	`Dataset .FileSi .Compli ce`
			`Dataset .FileCo t`
			`Dataset .Maximu ileSize`
			`Dataset .Minimu ileSize`
				Oui	Non	Non	Non	Non	Non

Ruletype	Descripti on	Argumer	Métrique rapporté s	Prise en charge en tant que règle ?	Prise en charge en tant qu'analys eur ?	Renvoie des résultats au niveau des lignes ?	Prise en charge des règles dynamiq s ?	Génère des observat ons	Prise en charge de la syntaxe de clause Where ?

FileUniqu eness	Vérifie si les fichiers sont uniques à l'aide de sommes de contrôle.	Chemin d'accès au fichier ou au dossier et seuil.	`Dataset .FileUr uenessR io`
			
			`Dataset .FileCo t`
				Oui	Non	Non	Non	Non	Non

Rubriques

- AggregateMatch
- ColumnCorrelation
- ColumnCount
- ColumnDataType
- ColumnExists
- ColumnLength
- ColumnNamesMatchPattern
- ColumnValues
- Intégralité
- CustomSQL
- DataFreshness
- DatasetMatch

- DistinctValuesCount
- Entropie
- IsComplete
- IsPrimaryKey
- IsUnique
- Mean
- ReferentialIntegrity
- RowCount
- RowCountMatch
- StandardDeviation
- Somme
- SchemaMatch
- Unicité
- UniqueValueRatio
- DetectAnomalies
- FileFreshness
- FileMatch
- FileUniqueness
- FileSize

AggregateMatch

Vérifie le ratio de deux agrégations de colonnes par rapport à une expression donnée. Ce type de règle fonctionne sur plusieurs jeux de données. Les deux agrégations de colonnes sont évaluées et un ratio est produit en divisant le résultat de la première agrégation de colonnes par le résultat de la seconde agrégation de colonnes. Le ratio est vérifié par rapport à l'expression fournie pour produire une réponse booléenne.

Syntaxe

Agrégation de colonnes

```
AggregateMatch <AGG_OPERATION> (<OPTIONAL_REFERENCE_ALIAS>.<COL_NAME>)
```

- AGG_OPERATION : l'opération à utiliser pour l'agrégation. À l'heure actuelle, `sum` et `avg` sont pris en charge.

  Types de colonnes pris en charge : octet, décimal, double, virgule flottante, entier, long, court

- OPTIONAL_REFERENCE_ALIAS : ce paramètre doit être fourni si la colonne provient d'un jeu de données de référence et non du jeu de données principal. Si vous utilisez cette règle dans le catalogue de données AWS Glue, votre alias de référence doit suivre le format « »<database_name>. <table_name>. <column_name>

  Types de colonnes pris en charge : octet, décimal, double, virgule flottante, entier, long, court

- COL_NAME : le nom de la colonne à agréger.

  Types de colonnes pris en charge : octet, décimal, double, virgule flottante, entier, long, court

Exemple : moyenne

```
"avg(rating)"
```

Exemple : somme

```
"sum(amount)"
```

Exemple : moyenne de la colonne dans le jeu de données de référence

```
"avg(reference.rating)"
```

Règle

```
AggregateMatch <AGG_EXP_1> <AGG_EXP_2> <EXPRESSION>
```

- AGG_EXP_1 : la première agrégation de colonnes.

  Types de colonnes pris en charge : octet, décimal, double, virgule flottante, entier, long, court

  Types de colonnes pris en charge : octet, décimal, double, virgule flottante, entier, long, court

- AGG_EXP_2 : la deuxième agrégation de colonnes.

  Types de colonnes pris en charge : octet, décimal, double, virgule flottante, entier, long, court

Types de colonnes pris en charge : octet, décimal, double, virgule flottante, entier, long, court

- EXPRESSION – Expression à exécuter en fonction de la réponse du type de règle afin de produire une valeur booléenne. Pour de plus amples informations, veuillez consulter Expressions.

Exemple : correspondance agrégée à l'aide de la somme

L'exemple de règle suivant vérifie si la somme des valeurs de la colonne `amount` est exactement égale à la somme des valeurs de la colonne `total_amount`.

```
AggregateMatch "sum(amount)" "sum(total_amount)" = 1.0
```

Exemple : correspondance agrégée à l'aide de la moyenne

L'exemple de règle suivant vérifie si la moyenne des valeurs de la colonne `ratings` est égale à au moins 90 % de la moyenne des valeurs de la colonne `ratings` dans le jeu de données `reference`. Le jeu de données de référence est fourni en tant que source de données supplémentaire dans l'expérience ETL ou le catalogue de données.

Dans AWS Glue ETL, vous pouvez utiliser :

```
AggregateMatch "avg(ratings)" "avg(reference.ratings)" >= 0.9
```

Dans le catalogue de données AWS Glue, vous pouvez utiliser :

```
AggregateMatch "avg(ratings)" "avg(database_name.tablename.ratings)" >= 0.9
```

Comportement null

La règle `AggregateMatch` ignorera les lignes contenant des valeurs NULL dans le calcul des méthodes d'agrégation (somme/moyenne). Par exemple :

```
+---+-----------+
|id |units      |
+---+-----------+
|100|0          |
|101|null       |
|102|20         |
|103|null       |
|104|40         |
```

```
+---+-----------+
```

La moyenne de la colonne `units` sera (0 + 20 + 40) / 3 = 20. Les lignes 101 et 103 ne sont pas prises en compte dans ce calcul.

ColumnCorrelation

Vérifie la corrélation entre deux colonnes par rapport à une expression donnée. AWS Glue Data Quality utilise le coefficient de corrélation de Pearson pour mesurer la corrélation linéaire entre deux colonnes. Le résultat est un nombre compris entre -1 et 1 qui mesure la force et la direction de la relation.

Syntaxe

```
ColumnCorrelation <COL_1_NAME> <COL_2_NAME> <EXPRESSION>
```

- COL_1_NAME – Nom de la première colonne par rapport à laquelle vous souhaitez évaluer la règle de qualité des données.

  Types de colonnes pris en charge : octet, décimal, double, virgule flottante, entier, long, court
- COL_2_NAME – Nom de la deuxième colonne par rapport à laquelle vous souhaitez évaluer la règle de qualité des données.

  Types de colonnes pris en charge : octet, décimal, double, virgule flottante, entier, long, court
- EXPRESSION – Expression à exécuter en fonction de la réponse du type de règle afin de produire une valeur booléenne. Pour plus d'informations, consultez Expressions.

Exemple : corrélation de colonnes

L'exemple de règle suivant vérifie si le coefficient de corrélation entre les colonnes `height` et `weight` présente une forte corrélation positive (valeur de coefficient supérieure à 0,8).

```
ColumnCorrelation "height" "weight" > 0.8
```

```
ColumnCorrelation "weightinkgs" "Salary" > 0.8 where "weightinkgs > 40"
```

Exemples de règles dynamiques

- `ColumnCorrelation "colA" "colB" between min(last(10)) and max(last(10))`

- `ColumnCorrelation "colA" "colB" < avg(last(5)) + std(last(5))`

Comportement null

La règle `ColumnCorrelation` ignorera les lignes contenant des valeurs `NULL` dans le calcul de la corrélation. Par exemple :

```
+---+-----------+
|id |units      |
+---+-----------+
|100|0          |
|101|null       |
|102|20         |
|103|null       |
|104|40         |
+---+-----------+
```

Les lignes 101 et 103 seront ignorées, et la valeur `ColumnCorrelation` sera 1.0.

ColumnCount

Vérifie le nombre de colonnes du jeu de données principal par rapport à une expression donnée. Dans l'expression, vous pouvez spécifier le nombre de colonnes ou une plage de colonnes à l'aide d'opérateurs tels que > et <.

Syntaxe

```
ColumnCount <EXPRESSION>
```

- EXPRESSION – Expression à exécuter en fonction de la réponse du type de règle afin de produire une valeur booléenne. Pour de plus amples informations, veuillez consulter Expressions.

Exemple : vérification numérique du nombre de colonnes

L'exemple de règle suivant vérifie si le nombre de colonnes est compris dans une plage donnée.

```
ColumnCount between 10 and 20
```

Exemples de règles dynamiques

- `ColumnCount >= avg(last(10))`
- `ColumnCount between min(last(10))-1 and max(last(10))+1`

ColumnDataType

Vérifie si les valeurs d'une colonne donnée peuvent être converties vers le type fourni dans Apache Spark. Accepte une expression `with threshold` pour vérifier la présence d'un sous-ensemble des valeurs de la colonne.

Syntaxe

```
ColumnDataType <COL_NAME> = <EXPECTED_TYPE>
```

- COL_NAME – Nom de la colonne par rapport à laquelle la règle de qualité des données doit être évaluée.

  Types de colonnes pris en charge : type de chaîne

  Types de colonnes pris en charge : octet, décimal, double, virgule flottante, entier, long, court
- EXPECTED_TYPE : le type attendu des valeurs de la colonne.

  Valeurs prises en charge : booléen, date, horodatage, entier, double, flottant, long

  Types de colonnes pris en charge : octet, décimal, double, virgule flottante, entier, long, court
- EXPRESSION : une expression facultative pour spécifier le pourcentage de valeurs qui doivent être du type attendu.

  Types de colonnes pris en charge : octet, décimal, double, virgule flottante, entier, long, court

Exemple : les entiers de type colonne en tant que chaînes

L'exemple de règle suivant vérifie si les valeurs de la colonne donnée, qui est de type chaîne, peuvent être converties en entiers.

```
ColumnDataType "colA" = "INTEGER"
```

Exemple : les entiers de type colonne en tant que chaînes de caractères vérifient un sous-ensemble de valeurs.

L'exemple de règle suivant vérifie si plus de 90 % des valeurs de la colonne donnée, qui est de type chaîne, peuvent être converties en entiers.

```
ColumnDataType "colA" = "INTEGER" with threshold > 0.9
```

ColumnExists

Vérifie si une colonne existe.

Syntaxe

```
ColumnExists <COL_NAME>
```

- COL_NAME – Nom de la colonne par rapport à laquelle la règle de qualité des données doit être évaluée.

  Types de colonnes pris en charge : n'importe quel type de colonne

Exemple : une colonne existe

L'exemple de règle suivant vérifie si la colonne nommée `Middle_Name` existe.

```
ColumnExists "Middle_Name"
```

ColumnLength

Vérifie si la longueur de chaque ligne d'une colonne est conforme à une expression donnée.

Syntaxe

```
ColumnLength <COL_NAME><EXPRESSION>
```

- COL_NAME – Nom de la colonne par rapport à laquelle la règle de qualité des données doit être évaluée.

  Types de colonnes pris en charge : chaîne
- EXPRESSION – Expression à exécuter en fonction de la réponse du type de règle afin de produire une valeur booléenne. Pour plus d'informations, consultez Expressions.

Exemple : longueur de ligne de colonne

L'exemple de règle suivant vérifie si la valeur de chaque ligne de la colonne nommée `Postal_Code` est constituée de 5 caractères.

```
ColumnLength "Postal_Code" = 5
ColumnLength "weightinkgs" = 2 where "weightinkgs > 10"
```

Comportement null

La règle `ColumnLength` traite les `NULL` comme des chaînes de longueur 0. Pour une ligne `NULL` :

```
ColumnLength "Postal_Code" > 4 # this will fail
```

```
ColumnLength "Postal_Code" < 6 # this will succeed
```

L'exemple de règle composée suivant fournit un moyen d'annuler explicitement des valeurs `NULL` :

```
(ColumnLength "Postal_Code" > 4) AND (ColumnValues "Postal_Code" != NULL)
```

ColumnNamesMatchPattern

Vérifie si les noms de toutes les colonnes du jeu de données principal correspondent à l'expression régulière donnée.

Syntaxe

```
ColumnNamesMatchPattern <PATTERN>
```

- PATTERN : le modèle par rapport auquel vous souhaitez évaluer la règle de qualité des données.

  Types de colonnes pris en charge : octet, décimal, double, virgule flottante, entier, long, court

Exemple : les noms de colonnes correspondent au modèle

L'exemple de règle suivant vérifie si toutes les colonnes commencent par le préfixe « aws_ »

```
ColumnNamesMatchPattern "aws_.*"
```

```
ColumnNamesMatchPattern "aws_.*" where "weightinkgs > 10"
```

ColumnValues

Exécute une expression en fonction des valeurs d'une colonne.

Syntaxe

```
ColumnValues <COL_NAME> <EXPRESSION>
```

- COL_NAME – Nom de la colonne par rapport à laquelle la règle de qualité des données doit être évaluée.

  Types de colonnes pris en charge : n'importe quel type de colonne
- EXPRESSION – Expression à exécuter en fonction de la réponse du type de règle afin de produire une valeur booléenne. Pour plus d'informations, consultez Expressions.

Exemple : valeurs autorisées

L'exemple de règle suivant vérifie si chaque valeur de la colonne spécifiée fait partie d'un ensemble de valeurs autorisées (dont null, empty, et strings uniquement avec des espaces blancs).

```
ColumnValues "Country" in [ "US", "CA", "UK", NULL, EMPTY, WHITESPACES_ONLY ]
ColumnValues "gender" in ["F", "M"] where "weightinkgs < 10"
```

Exemple : expression régulière

L'exemple de règle suivant compare les valeurs d'une colonne à une expression régulière.

```
ColumnValues "First_Name" matches "[a-zA-Z]*"
```

Exemple : valeurs de date

L'exemple de règle suivant compare les valeurs d'une colonne de date à une expression de date.

```
ColumnValues "Load_Date" > (now() - 3 days)
```

Exemple : valeurs numériques

L'exemple de règle suivant vérifie si les valeurs des colonnes correspondent à une certaine contrainte numérique.

```
ColumnValues "Customer_ID" between 1 and 2000
```

Comportement null

Pour toutes les règles `ColumnValues` (à l'exception de `!=` et de `NOT IN`), les lignes `NULL` dérogent à la règle. Si la règle échoue en raison d'une valeur null, la raison de l'échec s'affichera comme suit :

```
Value: NULL does not meet the constraint requirement!
```

L'exemple de règle composée suivant fournit un moyen d'autoriser explicitement des valeurs `NULL` :

```
(ColumnValues "Age" > 21) OR (ColumnValues "Age" = NULL)
```

ColumnValues Les règles annulées utilisant la `not in` syntaxe `!=` et seront transmises aux `NULL` lignes. Par exemple :

```
ColumnValues "Age" != 21
```

```
ColumnValues "Age" not in [21, 22, 23]
```

Les exemples suivants fournissent un moyen de déroger explicitement aux valeurs `NULL`.

```
(ColumnValues "Age" != 21) AND (ColumnValues "Age" != NULL)
```

```
ColumnValues "Age" not in [21, 22, 23, NULL]
```

Intégralité

Compare le pourcentage de valeurs complètes (non nulles) d'une colonne à une expression donnée.

Syntaxe

```
Completeness <COL_NAME> <EXPRESSION>
```

- COL_NAME – Nom de la colonne par rapport à laquelle la règle de qualité des données doit être évaluée.

  Types de colonnes pris en charge : n'importe quel type de colonne
- EXPRESSION – Expression à exécuter en fonction de la réponse du type de règle afin de produire une valeur booléenne. Pour plus d'informations, consultez Expressions.

Exemple : pourcentage de valeur nulle

Les exemples de règles suivants vérifient si plus de 95 % des valeurs d'une colonne sont complètes.

```
Completeness "First_Name" > 0.95
Completeness "First_Name" > 0.95 where "weightinkgs > 10"
```

Exemples de règles dynamiques

- `Completeness "colA" between min(last(5)) - 1 and max(last(5)) + 1`
- `Completeness "colA" <= avg(last(10))`

Comportement null

Remarque sur les formats de données CSV : les lignes vides des colonnes CSV peuvent afficher plusieurs comportements.

- Si une colonne est de type `String`, la ligne vide sera reconnue comme une chaîne vide et ne dérogera pas à la règle `Completeness`.
- Si une colonne est d'un autre type de données, comme `Int`, la ligne vide sera reconnue comme `NULL` et dérogera à la règle `Completeness`.

CustomSQL

Ce type de règle a été étendu pour prendre en charge deux cas d'utilisation :

- Exécutez une instruction SQL personnalisée sur un jeu de données et compare la valeur renvoyée à une expression donnée.
- Exécutez une instruction SQL personnalisée dans laquelle vous spécifiez un nom de colonne dans votre instruction SELECT, que vous comparez à une certaine condition pour obtenir des résultats au niveau des lignes.

Syntaxe

```
CustomSql <SQL_STATEMENT> <EXPRESSION>
```

- SQL_STATEMENT – Instruction SQL qui renvoie une valeur numérique unique, entourée de guillemets doubles.
- EXPRESSION – Expression à exécuter en fonction de la réponse du type de règle afin de produire une valeur booléenne. Pour de plus amples informations, veuillez consulter Expressions.

Exemple : SQL personnalisé pour récupérer le résultat global d'une règle

Cet exemple de règle utilise une instruction SQL pour récupérer le nombre d'enregistrements d'un jeu de données. La règle vérifie ensuite que le nombre d'enregistrements est compris entre 10 et 20.

```
CustomSql "select count(*) from primary" between 10 and 20
```

Exemple : SQL personnalisé pour récupérer les résultats au niveau des lignes

Cet exemple de règle utilise une instruction SQL dans laquelle vous spécifiez un nom de colonne dans votre instruction SELECT, que vous comparez à une certaine condition pour obtenir des résultats au niveau des lignes. Une expression de condition de seuil définit le nombre d'enregistrements qui doivent échouer pour que l'ensemble de la règle échoue. Notez qu'une règle ne peut pas contenir à la fois une condition et un mot clé.

```
CustomSql "select Name from primary where Age  > 18"
```

or

```
CustomSql "select Name from primary where Age > 18" with threshold  > 3
```

> **Important**
>
> L'alias `primary` est le nom du jeu de données que vous souhaitez évaluer. Lorsque vous utilisez des tâches ETL visuelles sur la console, `primary` représente toujours le `DynamicFrame` qui est transmis à la transformation `EvaluateDataQuality.apply()`. Lorsque vous utilisez le catalogue de données AWS Glue pour exécuter des tâches de qualité des données sur une table, `primary` représente la table.

Si vous êtes dans le catalogue AWS Glue, vous pouvez également utiliser les noms de table réels :

```
CustomSql "select count(*) from database.table" between 10 and 20
```

Vous pouvez également joindre plusieurs tables pour comparer différents éléments de données :

```
CustomSql "select count(*) from database.table inner join database.table2 on id1 = id2"
 between 10 and 20
```

Dans AWS Glue ETL, l'utilisation de CustomSQL permet d'identifier les enregistrements qui ont échoué aux contrôles de qualité des données. Pour que cela fonctionne, vous devez renvoyer les enregistrements qui font partie de la table principale pour laquelle vous évaluez la qualité des données. Les enregistrements renvoyés dans le cadre de la requête sont considérés comme réussis et les enregistrements non renvoyés sont considérés comme ayant échoué. Cela fonctionne en joignant le résultat de votre requête CustomSQL au jeu de données d'origine. La complexité de votre requête SQL peut avoir des répercussions sur les performances.

Pour cela :

- Vous devez sélectionner au moins une colonne dans votre table principale.
  - `select count(*) from primary` est une requête valide pour la règle OVERALL CustomSQL DQ, mais pas pour le SQL personnalisé au niveau des lignes.
  - Cette règle va générer une erreur lors de l'évaluation : `The output from CustomSQL must contain at least one column that matches the input dataset for AWS Glue Data Quality to provide row level results. The SQL query is a valid query but the columns from the SQL result are not present in the Input Dataset. Ensure that matching columns are returned from the SQL.`
- Dans votre requête SQL, sélectionnez une « clé primaire » dans votre table ou sélectionnez un ensemble de colonnes qui forment une clé composite. À défaut, vous risquez d'obtenir des résultats incohérents en raison de la correspondance de lignes dupliquées et d'une dégradation des performances.
- Sélectionnez les clés UNIQUEMENT dans votre table principale et non dans vos tables de référence.

La règle suivante permet de s'assurer que les enregistrements dont l'âge est inférieur à 100 sont identifiés comme réussis et que les enregistrements dont l'âge est supérieur sont marqués comme échoués.

```
CustomSql "select id from primary where age < 100"
```

Cette règle CustomSQL sera validée si 50 % des enregistrements ont un âge supérieur à 10 et identifiera également les enregistrements qui ont échoué. Les enregistrements renvoyés par cette règle CustomSQL seront considérés comme réussis, tandis que ceux non renvoyés seront considérés comme ayant échoué.

```
CustomSQL "select ID, CustomerID from primary where age > 10" with threshold > 0.5
```

Remarque : la règle CustomSQL échoue si vous renvoyez des enregistrements qui ne sont pas disponibles dans le jeu de données.

DataFreshness

Vérifie l'actualisation des données d'une colonne en évaluant la différence entre l'heure actuelle et les valeurs d'une colonne de date. Pour ce type de règle, vous pouvez spécifier une expression temporelle afin de vérifier que les valeurs des colonnes sont à jour.

Syntaxe

```
DataFreshness <COL_NAME> <EXPRESSION>
```

- COL_NAME – Nom de la colonne par rapport à laquelle la règle de qualité des données doit être évaluée.

  Types de colonne pris en charge : Date
- EXPRESSION – Expression numérique exprimée en heures ou en jours. Vous devez spécifier l'unité de temps dans votre expression.

Exemple : actualisation des données

Les exemples de règles suivants vérifient l'actualisation des données.

```
DataFreshness "Order_Date" <= 24 hours
DataFreshness "Order_Date" between 2 days and 5 days
```

Comportement null

Les règles `DataFreshness` échoueront pour les lignes contenant des valeurs `NULL`. Si la règle échoue en raison d'une valeur null, la raison de l'échec s'affichera comme suit :

```
80.00 % of rows passed the threshold
```

où 20 % des lignes qui ont échoué incluent les lignes avec `NULL`.

L'exemple de règle composée suivant fournit un moyen d'autoriser explicitement des valeurs `NULL` :

```
(DataFreshness "Order_Date" <= 24 hours) OR (ColumnValues "Order_Date" = NULL)
```

Actualité des données pour les objets Amazon S3

Vous devrez parfois valider l'actualité des données en fonction de l'heure de création du fichier Amazon S3. Pour ce faire, vous pouvez utiliser le code suivant pour obtenir l'horodatage et l'ajouter à votre trame de données, puis appliquer des contrôles d'actualité des données.

```
df = glueContext.create_data_frame.from_catalog(database = "default", table_name =
 "mytable")
df = df.withColumn("file_ts", df["_metadata.file_modification_time"])

Rules = [
 DataFreshness "file_ts" < 24 hours
]
```

DatasetMatch

Vérifie si les données du jeu de données principal correspondent aux données d'un jeu de données de référence. Les deux jeux de données sont joints à l'aide des mappages de colonnes clés fournis. Des mappages de colonnes supplémentaires peuvent être fournis si vous souhaitez vérifier l'égalité des données uniquement dans ces colonnes. Notez que DataSetMatchpour fonctionner, vos clés de jointure doivent être uniques et ne pas être NULL (elles doivent être une clé primaire). Si vous ne remplissez pas ces conditions, le message d'erreur suivant affiche : « La clé de distribution fournie ne convient pas à des cadres de données donnés ». Dans les cas où vous ne pouvez pas avoir de clés jointes uniques, envisagez d'utiliser d'autres types de règles, par exemple AggregateMatchpour établir une correspondance sur des données récapitulatives.

Syntaxe

```
DatasetMatch <REFERENCE_DATASET_ALIAS> <JOIN CONDITION WITH
 MAPPING> <OPTIONAL_MATCH_COLUMN_MAPPINGS> <EXPRESSION>
```

- REFERENCE_DATASET_ALIAS : l'alias du jeu de données de référence avec lequel vous comparez les données du jeu de données principal.
- KEY_COLUMN_MAPPINGS : une liste de noms de colonnes séparés par des virgules qui forment une clé dans les jeux de données. Si les noms de colonnes ne sont pas identiques dans les deux jeux de données, vous devez les séparer par `->`
- OPTIONAL_MATCH_COLUMN_MAPPINGS : vous pouvez fournir ce paramètre si vous souhaitez vérifier les données correspondantes uniquement dans certaines colonnes. Il utilise la même syntaxe que les mappages de colonnes clés. Si ce paramètre n'est pas fourni, la correspondance portera sur les données de toutes les autres colonnes. Les autres colonnes non-clés doivent porter le même nom dans les deux jeux de données.
- EXPRESSION – Expression à exécuter en fonction de la réponse du type de règle afin de produire une valeur booléenne. Pour de plus amples informations, veuillez consulter Expressions.

Exemple : faire correspondre des jeux de données à l'aide de la colonne ID

L'exemple de règle suivant vérifie que plus de 90 % du jeu de données principal correspond au jeu de données de référence, en utilisant la colonne « ID » pour joindre les deux jeux de données. Dans ce cas, toutes les colonnes sont comparées.

```
DatasetMatch "reference" "ID" >= 0.9
```

Exemple : faire correspondre des jeux de données en utilisant plusieurs colonnes clés

Dans l'exemple suivant, le jeu de données principal et le jeu de données de référence portent des noms différents pour les colonnes clés. `ID_1` et `ID_2` forment ensemble une clé composite dans le jeu de données principal. `ID_ref1` et `ID_ref2` forment ensemble une clé composite dans le jeu de données de référence. Dans ce scénario, vous pouvez utiliser la syntaxe spéciale pour fournir les noms des colonnes.

```
DatasetMatch "reference" "ID_1->ID_ref1,ID_2->ID_ref2" >= 0.9
```

Exemple : faire correspondre des jeux de données à l'aide de plusieurs colonnes clés et vérifie qu'une colonne spécifique correspond

Cet exemple s'appuie sur l'exemple précédent. Nous voulons vérifier que seule la colonne contenant les montants correspond. Cette colonne est nommée `Amount1` dans le jeu de données principal et `Amount2` dans le jeu de données de référence. Une correspondance exacte est désirée.

```
DatasetMatch "reference" "ID_1->ID_ref1,ID_2->ID_ref2" "Amount1->Amount2" >= 0.9
```

DistinctValuesCount

Vérifie le nombre de valeurs distinctes dans une colonne par rapport à une expression donnée.

Syntaxe

```
DistinctValuesCount <COL_NAME> <EXPRESSION>
```

- COL_NAME – Nom de la colonne par rapport à laquelle la règle de qualité des données doit être évaluée.

  Types de colonnes pris en charge : n'importe quel type de colonne
- EXPRESSION – Expression à exécuter en fonction de la réponse du type de règle afin de produire une valeur booléenne. Pour plus d'informations, consultez Expressions.

Exemple : nombre de valeurs de colonne distinctes

L'exemple de règle suivant vérifie que la colonne nommée `State` contient plus de 3 valeurs distinctes.

```
DistinctValuesCount "State" > 3
DistinctValuesCount "Customer_ID" < 6  where "Customer_ID < 10"
```

Exemples de règles dynamiques

- `DistinctValuesCount "colA" between avg(last(10))-1 and avg(last(10))+1`
- `DistinctValuesCount "colA" <= index(last(10),2) + std(last(5))`

Entropie

Vérifie si la valeur d'entropie d'une colonne correspond à une expression donnée. L'entropie mesure le niveau d'information contenu dans un message. Compte tenu de la distribution de probabilité des valeurs d'une colonne, l'entropie décrit le nombre de bits nécessaires pour identifier une valeur.

Syntaxe

```
Entropy <COL_NAME> <EXPRESSION>
```

- COL_NAME – Nom de la colonne par rapport à laquelle la règle de qualité des données doit être évaluée.

  Types de colonnes pris en charge : n'importe quel type de colonne
- EXPRESSION – Expression à exécuter en fonction de la réponse du type de règle afin de produire une valeur booléenne. Pour plus d'informations, consultez Expressions.

Exemple : entropie de colonne

L'exemple de règle suivant vérifie que la colonne nommée `Feedback` possède une valeur d'entropie supérieure à un.

```
Entropy "Star_Rating" > 1
Entropy "First_Name" > 1  where "Customer_ID < 10"
```

Exemples de règles dynamiques

- `Entropy "colA" < max(last(10))`
- `Entropy "colA" between min(last(10)) and max(last(10))`

IsComplete

Vérifie si toutes les valeurs d'une colonne sont complètes (non nulles).

Syntaxe

```
IsComplete <COL_NAME>
```

- COL_NAME – Nom de la colonne par rapport à laquelle la règle de qualité des données doit être évaluée.

  Types de colonnes pris en charge : n'importe quel type de colonne

Exemple : valeurs nulles

L'exemple suivant vérifie si toutes les valeurs d'une colonne nommée `email` ne sont pas nulles.

```
IsComplete "email"
IsComplete "Email" where "Customer_ID between 1 and 50"
IsComplete "Customer_ID"  where "Customer_ID < 16 and Customer_ID != 12"
IsComplete "passenger_count" where "payment_type<>0"
```

Comportement null

Remarque sur les formats de données CSV : les lignes vides des colonnes CSV peuvent afficher plusieurs comportements.

- Si une colonne est de type `String`, la ligne vide sera reconnue comme une chaîne vide et ne dérogera pas à la règle `Completeness`.
- Si une colonne est d'un autre type de données, comme `Int`, la ligne vide sera reconnue comme `NULL` et dérogera à la règle `Completeness`.

IsPrimaryKey

Vérifie si une colonne contient une clé primaire. Une colonne contient une clé primaire si toutes les valeurs de la colonne sont uniques et complètes (non nulles). Vous pouvez également vérifier la présence de clés primaires comportant plusieurs colonnes.

Syntaxe

```
IsPrimaryKey <COL_NAME>
```

- COL_NAME – Nom de la colonne par rapport à laquelle la règle de qualité des données doit être évaluée.

  Types de colonnes pris en charge : n'importe quel type de colonne

Exemple : clé primaire

L'exemple de règle suivant vérifie si la colonne nommée `Customer_ID` contient une clé primaire.

```
IsPrimaryKey "Customer_ID"
IsPrimaryKey "Customer_ID" where "Customer_ID < 10"
```

Exemple : clé primaire avec plusieurs colonnes. Tous les exemples suivants sont valides.

```
IsPrimaryKey "colA" "colB"
IsPrimaryKey "colA" "colB" "colC"
IsPrimaryKey colA "colB" "colC"
```

IsUnique

Vérifie si toutes les valeurs d'une colonne sont uniques et renvoie une valeur booléenne.

Syntaxe

```
IsUnique <COL_NAME>
```

- COL_NAME – Nom de la colonne par rapport à laquelle la règle de qualité des données doit être évaluée.

  Types de colonnes pris en charge : n'importe quel type de colonne

Exemples

L'exemple de règle suivant vérifie si toutes les valeurs d'une colonne nommée `email` sont uniques.

```
IsUnique "email"
IsUnique "Customer_ID" where "Customer_ID < 10"]
```

L'exemple de règle suivant vérifie plusieurs colonnes.

```
IsUnique "vendorid" "tpep_pickup_datetime"
```

Mean

Vérifie si la moyenne de toutes les valeurs d'une colonne correspond à une expression donnée.

Syntaxe

```
Mean <COL_NAME> <EXPRESSION>
```

- COL_NAME – Nom de la colonne par rapport à laquelle la règle de qualité des données doit être évaluée.

  Types de colonnes pris en charge : octet, décimal, double, virgule flottante, entier, long, court
- EXPRESSION – Expression à exécuter en fonction de la réponse du type de règle afin de produire une valeur booléenne. Pour plus d'informations, consultez Expressions.

Exemple : valeur moyenne

L'exemple de règle suivant vérifie si la moyenne de toutes les valeurs d'une colonne dépasse un seuil.

```
Mean "Star_Rating" > 3
Mean "Salary" < 6200 where "Customer_ID < 10"
```

Exemples de règles dynamiques

- `Mean "colA" > avg(last(10)) + std(last(2))`
- `Mean "colA" between min(last(5)) - 1 and max(last(5)) + 1`

Comportement null

La règle `Mean` ignorera les lignes contenant des valeurs `NULL` dans le calcul de la moyenne. Par exemple :

```
+---+-----------+
|id |units      |
+---+-----------+
|100|0          |
|101|null       |
|102|20         |
|103|null       |
|104|40         |
+---+-----------+
```

La moyenne de la colonne `units` sera (0 + 20 + 40) / 3 = 20. Les lignes 101 et 103 ne sont pas prises en compte dans ce calcul.

ReferentialIntegrity

Vérifie dans quelle mesure les valeurs d'un ensemble de colonnes du jeu de données principal sont un sous-ensemble des valeurs d'un ensemble de colonnes d'un jeu de données de référence.

Syntaxe

```
ReferentialIntegrity <PRIMARY_COLS> <REFERENCE_DATASET_COLS> <EXPRESSION>
```

- PRIMARY_COLS : une liste de noms de colonnes séparés par des virgules dans le jeu de données principal.

  Types de colonnes pris en charge : octet, décimal, double, virgule flottante, entier, long, court
- REFERENCE_DATASET_COLS : ce paramètre contient deux parties séparées par un point. La première partie est l'alias du jeu de données de référence. La deuxième partie est la liste des noms de colonnes du jeu de données de référence, séparés par des virgules et placés entre accolades.

  Types de colonnes pris en charge : octet, décimal, double, virgule flottante, entier, long, court
- EXPRESSION – Expression à exécuter en fonction de la réponse du type de règle afin de produire une valeur booléenne. Pour de plus amples informations, veuillez consulter Expressions.

Exemple : vérifier l'intégrité référentielle d'une colonne de code postal

L'exemple de règle suivant vérifie que plus de 90 % des valeurs de la colonne `zipcode` du jeu de données principal sont présentes dans la colonne `zipcode` du jeu de données `reference`.

```
ReferentialIntegrity "zipcode" "reference.zipcode" >= 0.9
```

Exemple : vérifier l'intégrité référentielle des colonnes de ville et d'État

Dans l'exemple suivant, des colonnes contenant des informations sur la ville et l'État existent dans le jeu de données principal et dans le jeu de données de référence. Les noms des colonnes sont différents dans les deux jeux de données. La règle vérifie si l'ensemble de valeurs des colonnes du jeu de données principal est exactement égal à l'ensemble de valeurs des colonnes du jeu de données de référence.

```
ReferentialIntegrity "city,state" "reference.{ref_city,ref_state}" = 1.0
```

Exemples de règles dynamiques

- `ReferentialIntegrity "city,state" "reference.{ref_city,ref_state}" > avg(last(10))`
- `ReferentialIntegrity "city,state" "reference.{ref_city,ref_state}" between min(last(10)) - 1 and max(last(10)) + 1`

RowCount

Vérifie le nombre de lignes d'un jeu de données par rapport à une expression donnée. Dans l'expression, vous pouvez spécifier le nombre de lignes ou une plage de lignes à l'aide d'opérateurs tels que > et <.

Syntaxe

```
RowCount <EXPRESSION>
```

- EXPRESSION – Expression à exécuter en fonction de la réponse du type de règle afin de produire une valeur booléenne. Pour plus d'informations, consultez Expressions.

Exemple : vérification numérique du nombre de lignes

L'exemple de règle suivant vérifie si le nombre de lignes est compris dans une plage donnée.

```
RowCount between 10 and 100
RowCount between 1 and 50 where "Customer_ID < 10"
```

Exemples de règles dynamiques

```
RowCount > avg(last(10)) *0.8
```

RowCountMatch

Vérifie le rapport entre le nombre de lignes du jeu de données principal et le nombre de lignes d'un jeu de données de référence par rapport à l'expression donnée.

Syntaxe

```
RowCountMatch <REFERENCE_DATASET_ALIAS> <EXPRESSION>
```

- REFERENCE_DATASET_ALIAS : alias du jeu de données de référence par rapport auquel comparer le nombre de lignes.

  Types de colonnes pris en charge : octet, décimal, double, virgule flottante, entier, long, court
- EXPRESSION – Expression à exécuter en fonction de la réponse du type de règle afin de produire une valeur booléenne. Pour de plus amples informations, veuillez consulter Expressions.

Exemple : vérification du nombre de lignes par rapport à un jeu de données de référence

L'exemple de règle suivant vérifie si le nombre de lignes du jeu de données principal est au moins égal à 90 % du nombre de lignes du jeu de données de référence.

```
RowCountMatch "reference" >= 0.9
```

StandardDeviation

Vérifie l'écart type de toutes les valeurs d'une colonne par rapport à une expression donnée.

Syntaxe

```
StandardDeviation <COL_NAME> <EXPRESSION>
```

- COL_NAME – Nom de la colonne par rapport à laquelle la règle de qualité des données doit être évaluée.

  Types de colonnes pris en charge : octet, décimal, double, virgule flottante, entier, long, court
- EXPRESSION – Expression à exécuter en fonction de la réponse du type de règle afin de produire une valeur booléenne. Pour plus d'informations, consultez Expressions.

Exemple : écart type

L'exemple de règle suivant vérifie si l'écart type des valeurs d'une colonne nommée `colA` est inférieur à une valeur spécifiée.

```
StandardDeviation "Star_Rating" < 1.5
StandardDeviation "Salary" < 3500 where "Customer_ID < 10"
```

Exemples de règles dynamiques

- `StandardDeviation "colA" > avg(last(10) + 0.1`
- `StandardDeviation "colA" between min(last(10)) - 1 and max(last(10)) + 1`

Comportement null

La règle `StandardDeviation` ignorera les lignes contenant des valeurs `NULL` dans le calcul de l'écart type. Par exemple :

```
+---+-----------+-----------+
|id |units1     |units2     |
+---+-----------+-----------+
|100|0          |0          |
|101|null       |0          |
|102|20         |20         |
|103|null       |0          |
|104|40         |40         |
+---+-----------+-----------+
```

L'écart type de la colonne `units1` ne tiendra pas compte des lignes 101 et 103 et donnera 16,33. L'écart type pour la colonne `units2` sera de 16.

Somme

Vérifie la somme de toutes les valeurs d'une colonne par rapport à une expression donnée.

Syntaxe

```
Sum <COL_NAME> <EXPRESSION>
```

- COL_NAME – Nom de la colonne par rapport à laquelle la règle de qualité des données doit être évaluée.

  Types de colonnes pris en charge : octet, décimal, double, virgule flottante, entier, long, court
- EXPRESSION – Expression à exécuter en fonction de la réponse du type de règle afin de produire une valeur booléenne. Pour plus d'informations, consultez Expressions.

Exemple : somme

L'exemple de règle suivant vérifie si la somme de toutes les valeurs d'une colonne dépasse un seuil donné.

```
Sum "transaction_total" > 500000
Sum "Salary" < 55600 where "Customer_ID < 10"
```

Exemples de règles dynamiques

- `Sum "ColA" > avg(last(10))`
- `Sum "colA" between min(last(10)) - 1 and max(last(10)) + 1`

Comportement null

La règle `Sum` ignorera les lignes contenant des valeurs `NULL` dans le calcul de la somme. Par exemple :

```
+---+-----------+
|id |units      |
+---+-----------+
|100|0          |
|101|null       |
|102|20         |
|103|null       |
|104|40         |
+---+-----------+
```

La somme des colonnes `units` ne tiendra pas compte des lignes 101 et 103 et donnera (0 + 20 + 40) = 60.

SchemaMatch

Vérifie si le schéma du jeu de données principal correspond au schéma d'un jeu de données de référence. La vérification du schéma s'effectue colonne par colonne. Le schéma de deux colonnes correspond si les noms et les types sont identiques. L'ordre des colonnes n'importe pas.

Syntaxe

```
SchemaMatch <REFERENCE_DATASET_ALIAS> <EXPRESSION>
```

- REFERENCE_DATASET_ALIAS : alias du jeu de données de référence par rapport auquel comparer les schémas.

  Types de colonnes pris en charge : octet, décimal, double, virgule flottante, entier, long, court
- EXPRESSION – Expression à exécuter en fonction de la réponse du type de règle afin de produire une valeur booléenne. Pour de plus amples informations, veuillez consulter Expressions.

Exemple : SchemaMatch

L'exemple de règle suivant vérifie si le schéma du jeu de données principal correspond exactement au schéma d'un jeu de données de référence.

```
SchemaMatch "reference" = 1.0
```

Unicité

Vérifie le pourcentage de valeurs uniques dans une colonne par rapport à une expression donnée. Les valeurs uniques n'apparaissent qu'une seule fois.

Syntaxe

```
Uniqueness <COL_NAME> <EXPRESSION>
```

- COL_NAME – Nom de la colonne par rapport à laquelle la règle de qualité des données doit être évaluée.

  Types de colonnes pris en charge : n'importe quel type de colonne
- EXPRESSION – Expression à exécuter en fonction de la réponse du type de règle afin de produire une valeur booléenne. Pour de plus amples informations, veuillez consulter Expressions.

Exemple

L'exemple de règle suivant vérifie si le pourcentage de valeurs uniques d'une colonne correspond à certains critères numériques.

```
Uniqueness "email" = 1.0
Uniqueness "Customer_ID" != 1.0 where "Customer_ID < 10"
```

L'exemple de règle suivant vérifie plusieurs colonnes.

```
Uniqueness "vendorid" "tpep_pickup_datetime" = 1
```

Exemples de règles dynamiques

- `Uniqueness "colA" between min(last(10)) and max(last(10))`
- `Uniqueness "colA" >= avg(last(10))`

UniqueValueRatio

Vérifie le ratio de valeurs uniques d'une colonne par rapport à une expression donnée. Un ratio de valeurs uniques correspond à la fraction de valeurs uniques divisée par le nombre de toutes les valeurs distinctes d'une colonne. Les valeurs uniques n'apparaissent qu'une seule fois, alors que les valeurs distinctes apparaissent au moins une fois.

Par exemple, le jeu `[a, a, b]` contient une valeur unique (b) et deux valeurs distinctes (a et b). Le ratio de valeurs uniques du jeu est donc ½ = 0,5.

Syntaxe

```
UniqueValueRatio <COL_NAME> <EXPRESSION>
```

- COL_NAME – Nom de la colonne par rapport à laquelle la règle de qualité des données doit être évaluée.

  Types de colonnes pris en charge : n'importe quel type de colonne
- EXPRESSION – Expression à exécuter en fonction de la réponse du type de règle afin de produire une valeur booléenne. Pour de plus amples informations, veuillez consulter Expressions.

Exemple : ratio de valeurs uniques

Cet exemple vérifie le ratio de valeurs uniques d'une colonne par rapport à une plage de valeurs.

```
UniqueValueRatio "test_score" between 0 and 0.5
UniqueValueRatio "Customer_ID" between 0 and 0.9 where "Customer_ID < 10"
```

Exemples de règles dynamiques

- `UniqueValueRatio "colA" > avg(last(10))`
- `UniqueValueRatio "colA" <= index(last(10),2) + std(last(5))`

DetectAnomalies

Détecte les anomalies pour une règle de qualité des données donnée. Chaque exécution d'une DetectAnomalies règle entraîne l'enregistrement de la valeur évaluée pour la règle donnée. Lorsque suffisamment de données sont collectées, l'algorithme de détection des anomalies prend toutes les données historiques pour cette règle donnée et exécute la détection des anomalies. DetectAnomalies la règle échoue lorsqu'une anomalie est détectée. Plus d'informations sur l'anomalie détectée peuvent être obtenues à partir des observations.

Syntaxe

```
DetectAnomalies <RULE_NAME> <RULE_PARAMETERS>
```

`RULE_NAME` : le nom de la règle pour laquelle vous souhaitez évaluer et détecter des anomalies. Règles prises en charge :

- "RowCount"
- "Completeness"
- "Uniqueness"
- "Mean"
- "Sum"
- "StandardDeviation"
- "Entropy"
- "DistinctValuesCount"
- "UniqueValueRatio"
- "ColumnLength"
- "ColumnValues"
- "ColumnCorrelation"
- "CustomSQL"
- "ColumnCount"

RULE_PARAMETERS : certaines règles nécessitent des paramètres supplémentaires pour s'exécuter. Reportez-vous à la documentation de la règle concernée pour connaître les paramètres requis.

Exemple : Anomalies pour RowCount

Par exemple, si nous voulons détecter des RowCount anomalies, nous indiquons RowCount le nom de la règle.

```
DetectAnomalies "RowCount"
```

Exemple : Anomalies pour ColumnLength

Par exemple, si nous voulons détecter des ColumnLength anomalies, nous indiquons le nom de la ColumnLength règle et le nom de la colonne.

```
DetectAnomalies "ColumnLength" "id"
```

FileFreshness

FileFreshness garantit que vos fichiers de données sont à jour en fonction de l'état que vous fournissez. Il utilise l'heure de dernière modification de vos fichiers pour s'assurer que les fichiers de données ou l'ensemble du dossier le sont up-to-date.

Cette règle regroupe deux métriques :

- FileFreshness conformité en fonction de la règle que vous avez définie
- Nombre de fichiers analysés par la règle.

```
{"Dataset.*.FileFreshness.Compliance":1,"Dataset.*.FileCount":1}
```

La détection des anomalies ne prend pas en compte ces métriques.

Vérification de l'actualité des fichiers

La règle suivante garantit que tickets.parquet a été créé au cours des dernières 24 heures.

```
FileFreshness "s3://amzn-s3-demo-bucket/artifacts/file/tickets/tickets.parquet" >
 (now() - 24 hours)
```

Vérification de l'actualité du dossier

La règle suivante s'applique si tous les fichiers du dossier ont été créés ou modifiés au cours des dernières 24 heures.

```
FileFreshness "s3://bucket/" >= (now() -1 days)
FileFreshness "s3://amzn-s3-demo-bucket/artifacts/file/tickets/" >= (now() - 24 hours)
```

Vérification de l'actualité d'un dossier ou d'un fichier avec un seuil

La règle suivante s'applique si 10 % des fichiers du dossier « tickets » ont été créés ou modifiés au cours des 10 derniers jours.

```
FileFreshness "s3://amzn-s3-demo-bucket/artifacts/file/tickets/" < (now() - 10 days)
 with threshold > 0.1
```

Vérification de fichiers ou de dossiers avec des dates spécifiques

Vous pouvez vérifier l'actualité des fichiers pour des jours spécifiques.

```
FileFreshness "s3://amzn-s3-demo-bucket/artifacts/file/tickets/" > "2020-01-01"
FileFreshness "s3://amzn-s3-demo-bucket/artifacts/file/tickets/" between "2023-01-01"
 and "2024-01-01"
```

Vérification de fichiers ou de dossiers avec l'heure

Vous pouvez l'utiliser FileFreshness pour vous assurer que les fichiers sont arrivés à certaines heures.

```
FileFreshness "s3://amzn-s3-demo-bucket/artifacts/file/tickets/" between now() and
 (now() - 45 minutes)
FileFreshness "s3://amzn-s3-demo-bucket/artifacts/file/tickets/" between "9:30 AM" and
 "9:30 PM"
FileFreshness "s3://amzn-s3-demo-bucket/artifacts/file/tickets/" > (now() - 10 minutes)
FileFreshness "s3://amzn-s3-demo-bucket/artifacts/file/tickets/" > now()
FileFreshness "s3://amzn-s3-demo-bucket/artifacts/file/tickets/" between (now() - 2
 hours) and (now() + 15 minutes)
FileFreshness "s3://amzn-s3-demo-bucket/artifacts/file/tickets/" between (now() - 3
 days) and (now() + 15 minutes)
FileFreshness "s3://amzn-s3-demo-bucket/artifacts/file/tickets/" between "2001-02-07"
 and (now() + 15 minutes)
FileFreshness "s3://amzn-s3-demo-bucket/artifacts/file/tickets/" > "21:45""
FileFreshness "s3://amzn-s3-demo-bucket/artifacts/file/tickets/" > "2024-01-01"
```

```
FileFreshness "s3://amzn-s3-demo-bucket/artifacts/file/tickets/" between "02:30" and
 "04:30"
FileFreshness "s3://amzn-s3-demo-bucket/artifacts/file/tickets/" between "9:30 AM" and
 "22:15"
```

Considérations clés :

- FileFreshness peut évaluer les fichiers en utilisant des unités de jours, d'heures et de minutes
- Parfois, elle prend en charge le format matins/après-midi et 24 heures
- Les heures sont calculées en UTC, sauf si une dérogation est spécifiée.
- Les dates sont calculées en UTC à 00:00.

FileFreshness qui sont basés sur le temps fonctionnent comme suit :

```
FileFreshness "s3://amzn-s3-demo-bucket/artifacts/file/tickets/" > "21:45"
```

- Tout d'abord, l'heure « 21:45 » est combinée à la date du jour au format UTC pour créer un champ date-heure.
- Ensuite, le champ date-heure est converti en un fuseau horaire que vous avez spécifié.
- Enfin, la règle est évaluée.

Balises de règles facultatives basées sur des fichiers :

Les balises vous permettent de contrôler le comportement des règles.

recentFiles

Cette balise limite le nombre de fichiers traités en conservant le fichier le plus récent en premier.

```
FileFreshness "s3://amzn-s3-demo-bucket/" between (now() - 100 minutes) and (now() + 10
 minutes) with recentFiles = 1
```

URI Regex

> **Note**
>
> Le `uriRegex` tag est disponible dans AWS Glue 5.0 et versions ultérieures.

Cette balise filtre les fichiers en appliquant un modèle regex au chemin du fichier. Seuls les fichiers dont le chemin correspond au modèle sont traités. Vous pouvez également utiliser une prévision négative pour exclure les fichiers qui correspondent à un modèle.

```
# Match only files with a .csv extension
FileFreshness "s3://amzn-s3-demo-bucket/" > (now() - 24 hours) with uriRegex = "\.csv$"
# Match Parquet files that contain "orders_" in the path
FileFreshness "s3://amzn-s3-demo-bucket/" > (now() - 24 hours) with uriRegex =
 ".*orders_.*\.parquet"
# Exclude files ending in .tmp using a negative lookahead
FileFreshness "s3://amzn-s3-demo-bucket/" > (now() - 24 hours) with uriRegex = "(?!.*
\.tmp$).*"
```

Ordre du filtre

> **Note**
>
> Le `filterOrder` tag est disponible dans AWS Glue 5.0 et versions ultérieures.

Lorsque vous utilisez plusieurs balises de filtre, telles que `recentFiles` et `uriRegex` ensemble, la `filterOrder` balise contrôle l'ordre dans lequel elles sont appliquées. L'ordre par défaut est `recentFiles` le premier, puis`uriRegex`.

```
FileFreshness "s3://amzn-s3-demo-bucket/" > (now() - 24 hours) with recentFiles = 1
 with uriRegex = "inventory_" with filterOrder = ["uriRegex","recentFiles"]
```

Dans l'exemple ci-dessus, le `uriRegex` filtre est d'abord appliqué pour sélectionner uniquement les fichiers correspondant à « inventory_ », puis `recentFiles = 1` prend le fichier le plus récent de cet ensemble filtré. Dans le cas contraire`filterOrder`, le comportement par défaut prendrait d'abord le fichier le plus récent, puis appliquerait l'expression régulière, ce qui pourrait entraîner l'absence de correspondance de fichiers si le fichier le plus récent ne correspond pas au modèle.

> **Note**
>
> Toutes les valeurs de la `filterOrder` liste doivent faire référence à d'autres balises de filtre (`recentFilesouuriRegex`) également présentes sur la même règle. Les balises non filtrantes telles que `timeZone` ou ne `failFast` sont pas valides dans`filterOrder`.

Fail-vite

Lorsqu'elle est définie sur''`true`'', la règle renvoie un échec immédiatement au premier fichier qui ne répond pas à la condition de fraîcheur, au lieu d'évaluer tous les fichiers et de calculer un ratio de conformité.

```
FileFreshness "s3://amzn-s3-demo-bucket/" > (now() - 24 hours) with failFast = "true"
```

timeZone

Dérogations de fuseau horaire acceptées, consultez Allowed Time Zones pour connaître les fuseaux horaires pris en charge.

```
FileFreshness "s3://path/" > "21:45" with timeZone = "America/New_York"
```

```
FileFreshness "s3://path/" > "21:45" with timeZone = "America/Chicago"
```

```
FileFreshness "s3://path/" > "21:45" with timeZone = "Europe/Paris"
```

```
FileFreshness "s3://path/" > "21:45" with timeZone = "Asia/Shanghai"
```

```
FileFreshness "s3://path/" > "21:45" with timeZone = "Australia/Darwin"
```

Déduction des noms de fichiers directement à partir de trames de données

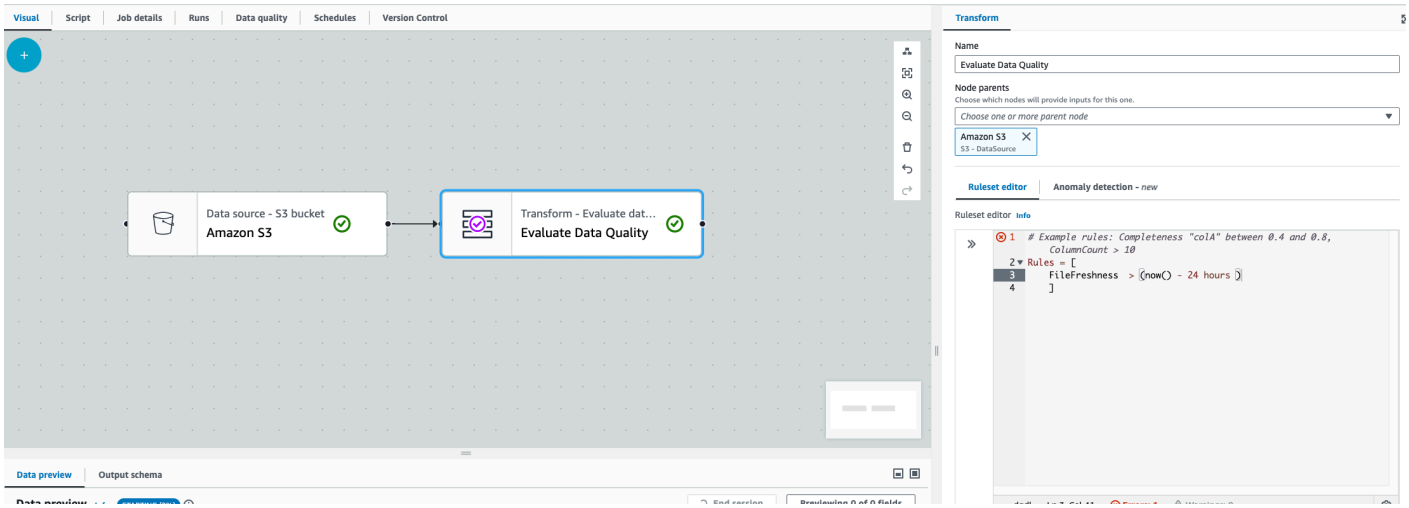
Il n'est pas toujours nécessaire de fournir un chemin d'accès au fichier. Par exemple, lorsque vous créez la règle dans le catalogue de données AWS Glue, il peut être difficile de trouver les dossiers utilisés par les tables du catalogue. AWS Glue Data Quality peut trouver les dossiers ou fichiers spécifiques utilisés pour remplir votre dataframe et détecter s'ils sont récents.

> **Note**
>
> Cette fonctionnalité ne fonctionne que lorsque les fichiers sont correctement lus dans le DynamicFrame ou DataFrame.

```
FileFreshness > (now() - 24 hours)
```

Cette règle permet de trouver le chemin d'accès au dossier ou aux fichiers utilisés pour renseigner l'image dynamique ou la trame de données. Cela fonctionne pour les chemins Amazon S3 ou les tables AWS Glue Data Catalog basées sur Amazon S3. Voici quelques considérations à prendre en compte :

1. Dans AWS Glue ETL, vous devez disposer de la EvaluateDataQualitytransformation immédiatement après une transformation d'Amazon S3 ou de AWS Glue Data Catalog.



2. Cette règle ne fonctionnera pas dans les sessions AWS Glue Interactive.

Si vous essayez de le faire dans les deux cas, ou si AWS Glue ne trouve pas les fichiers, AWS Glue générera le message d'erreur suivant : `"Unable to parse file path from DataFrame"`

FileMatch

La FileMatch règle vous permet de comparer des fichiers à d'autres fichiers ou à des checksums. Elle peut être utile dans certains scénarios :

1. Validation des fichiers reçus de sources externes : vous pouvez vous FileMatch assurer que vous avez reçu les bons fichiers provenant de sources externes en les comparant aux sommes de contrôle. Cela permet de valider l'intégrité des données que vous ingérez.
2. Comparaison de données dans deux dossiers différents : FileMatch peut être utilisé pour comparer des fichiers entre deux dossiers.

Cette règle regroupe une métrique : le nombre de fichiers analysés par la règle.

```
{"Dataset.*.FileCount":1}
```

Valider le fichier avec une somme de contrôle :

FileMatch accepte un fichier et une somme de contrôle définie pour s'assurer qu'au moins une somme de contrôle correspond au fichier.

```
FileMatch "s3://amzn-s3-demo-bucket/file.json" in ["3ee0d8617ac041793154713e5ef8f319"]
 with hashAlgorithm = "MD5"
FileMatch "s3://amzn-s3-demo-bucket/file.json" in ["3ee0d8617ac041793154713e5ef8f319"]
 with hashAlgorithm = "SHA-1"
FileMatch "s3://amzn-s3-demo-bucket/file.json" in ["3ee0d8617ac041793154713e5ef8f319"]
 with hashAlgorithm = "SHA-256"
FileMatch "s3://amzn-s3-demo-bucket/file.json" in ["3ee0d8617ac041793154713e5ef8f319"]
```

Les algorithmes standard suivants sont pris en charge :

- MD5
- SHA-1
- SHA-256

Si vous ne fournissez aucun algorithme, la valeur par défaut est SHA-256.

Valider tous les fichiers d'un dossier avec un ensemble de sommes de contrôle :

```
FileMatch "s3://amzn-s3-demo-bucket /" in ["3ee0d8617ac041793154713e5ef8f319",
 "7e8617ac041793154713e5ef8f319"] with hashAlgorithm = "MD5"
FileMatch "s3://amzn-s3-demo-bucket /internal-folder/" in
 ["3ee0d8617ac041793154713e5ef8f319", "7e8617ac041793154713e5ef8f319"]
```

Comparer des fichiers dans différents dossiers

```
# Compare all files across two buckets
FileMatch "s3://original_bucket/" "s3://archive_bucket/"
# Compare files within specific subfolders
FileMatch "s3://original_bucket/internal-folder/" "s3://original_bucket/other-folder/"
# Compare only .json files across two folders
FileMatch "s3://original_bucket/" "s3://archive_bucket/" with uriRegex = "\.json$"
# Compare only the 5 most recent .csv files
FileMatch "s3://original_bucket/" "s3://archive_bucket/" with recentFiles = 5 with
 uriRegex = "\.csv$" with filterOrder = ["uriRegex","recentFiles"]
```

FileMatch vérifiera le contenu des fichiers `original_bucket` et s'assurera qu'ils correspondent à ce qu'ils contiennent`archive_bucket`. La règle échouera s'ils ne correspondent pas exactement. Elle peut également vérifier le contenu de dossiers internes ou de fichiers individuels.

FileMatch peut également vérifier les fichiers individuels les uns par rapport aux autres.

```
FileMatch "s3://amzn-s3-demo-bucket /file_old.json" "s3://amzn-s3-demo-bucket /
file_new.json"
```

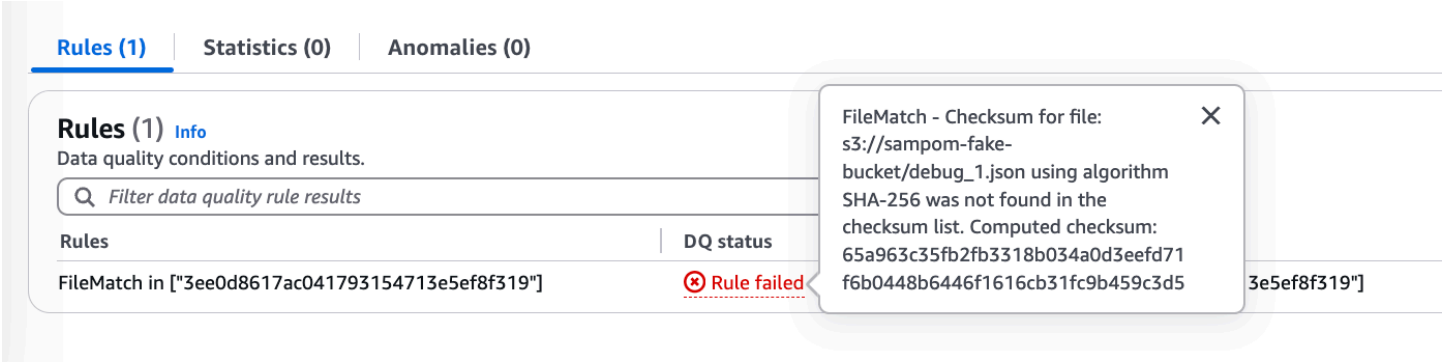
Déduction des noms de fichiers directement à partir de trames de données

Il n'est pas toujours nécessaire de fournir un chemin d'accès au fichier. Par exemple, lorsque vous créez la règle dans le catalogue de données AWS Glue (soutenu par Amazon S3), il peut être difficile de trouver les dossiers utilisés par les tables du catalogue. AWS Glue Data Quality peut trouver les dossiers ou fichiers spécifiques utilisés pour remplir votre bloc de données.

> **Note**
>
> Cette fonctionnalité ne fonctionne que lorsque les fichiers sont correctement lus dans le DynamicFrame ou DataFrame.

```
FileMatch in ["3ee0d8617ac041793154713e5ef8f319"] with hashAlgorithm = "MD5"
FileMatch in ["3ee0d8617ac041793154713e5ef8f319"] with hashAlgorithm = "SHA-1"
FileMatch in ["3ee0d8617ac041793154713e5ef8f319"] with hashAlgorithm = "SHA-256"
FileMatch in ["3ee0d8617ac041793154713e5ef8f319"]
```

Si la somme de contrôle fournie est différente de celle calculée, elle vous FileMatch avertira de la différence.



Balises de règles facultatives basées sur des fichiers :

Les balises vous permettent de contrôler le comportement des règles.

recentFiles

Cette balise limite le nombre de fichiers traités en conservant le fichier le plus récent en premier.

```
FileMatch "s3://bucket/" in ["3ee0d8617ac04179sam4713e5ef8f319"] with recentFiles = 1
```

URI Regex

> **Note**
>
> Le `uriRegex` tag est disponible dans AWS Glue 5.0 et versions ultérieures.

Cette balise filtre les fichiers en appliquant un modèle regex au chemin du fichier. Seuls les fichiers dont le chemin correspond au modèle sont traités. Vous pouvez également utiliser une prévision négative pour exclure les fichiers qui correspondent à un modèle.

```
# Match only files with a .json extension
FileMatch "s3://bucket/" in ["3ee0d8617ac04179sam4713e5ef8f319"] with uriRegex =
 "\.json$"
# Exclude files ending in .tmp using a negative lookahead
FileMatch "s3://bucket/" in ["3ee0d8617ac04179sam4713e5ef8f319"] with uriRegex = "(?!.*
\.tmp$).*"
```

Ordre du filtre

> **Note**
>
> Le `filterOrder` tag est disponible dans AWS Glue 5.0 et versions ultérieures.

Lorsque vous utilisez plusieurs balises de filtre, telles que `recentFiles` et `uriRegex` ensemble, la `filterOrder` balise contrôle l'ordre dans lequel elles sont appliquées. L'ordre par défaut est `recentFiles` le premier, puis`uriRegex`.

```
FileMatch "s3://bucket/" in ["3ee0d8617ac04179sam4713e5ef8f319"] with recentFiles = 1
 with uriRegex = "\.json$" with filterOrder = ["uriRegex","recentFiles"]
```

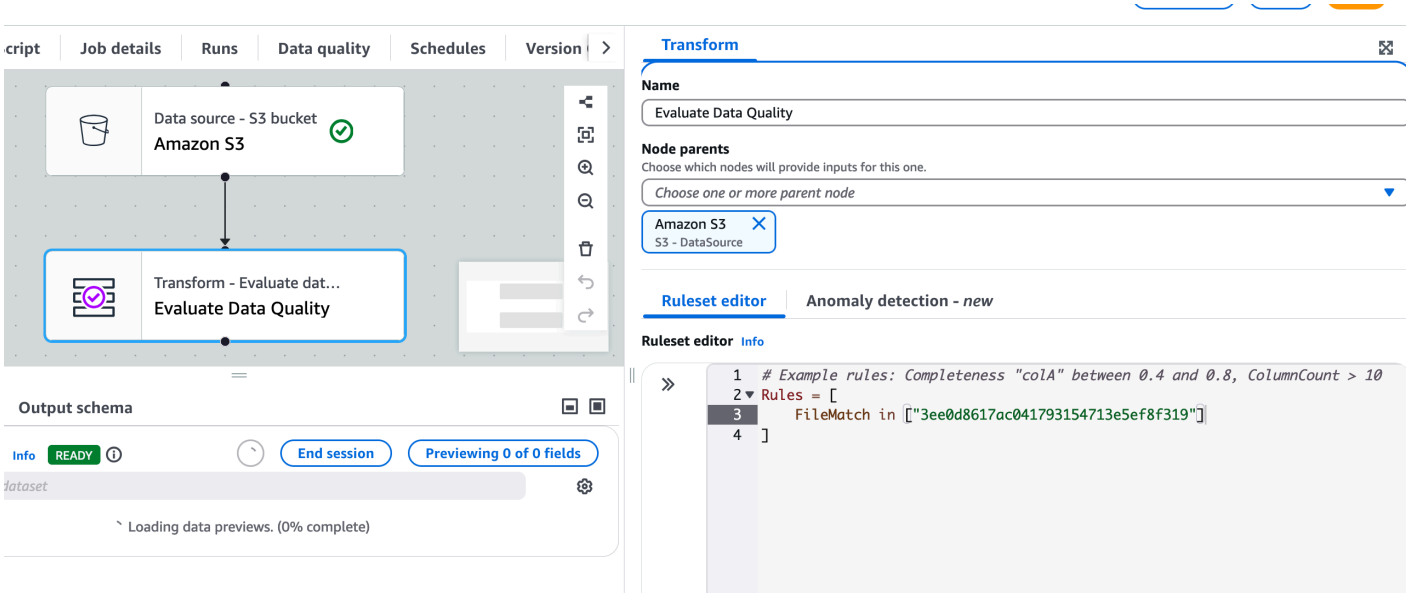
matchFileName

Cette balise garantit que les fichiers ne comportent pas de noms en double. Le comportement par défaut est false.

```
FileMatch "s3://amzn-s3-demo-bucket/file.json" in ["3ee0d8617ac04179sam4713e5ef8f319"]
 with matchFileName = "true"
```

Voici quelques considérations à prendre en compte :

1. Dans AWS Glue ETL, vous devez disposer de la EvaluateDataQualitytransformation immédiatement après une transformation d'Amazon S3 ou de AWS Glue Data Catalog.



2. Cette règle ne fonctionnera pas dans les sessions AWS Glue Interactive.

FileUniqueness

L'unicité des fichiers vous permet de vous assurer qu'il n'y a pas de doublons dans les données que vous avez reçues de vos producteurs de données.

Elle rassemble les statistiques de données suivantes :

1. Nombre de fichiers analysés par la règle.
2. Le taux d'unicité des fichiers

```
Dataset.*.FileUniquenessRatio: 1.00, Dataset.*.FileCount: 8.00
```

Rechercher des fichiers en double dans un dossier :

```
FileUniqueness "s3://bucket/" > 0.5
FileUniqueness "s3://bucket/folder/" = 1
```

Déduire les noms de dossiers directement à partir des trames de données pour détecter les doublons :

Il n'est pas toujours nécessaire de fournir un chemin d'accès au fichier. Par exemple, lorsque vous créez la règle dans le catalogue de données AWS Glue, il peut être difficile de trouver les dossiers utilisés par les tables du catalogue. AWS Glue Data Quality peut trouver les dossiers ou fichiers spécifiques utilisés pour remplir votre bloc de données.

> **Note**
>
> Lors de l'utilisation de l'inférence, les règles basées sur les fichiers peuvent uniquement détecter les fichiers lus avec succès dans le DynamicFrame ou. DataFrame

```
FileUniqueness > 0.5
```

Balises de règles facultatives basées sur des fichiers :

Les balises vous permettent de contrôler le comportement des règles.

recentFiles

Cette balise limite le nombre de fichiers traités en conservant le fichier le plus récent en premier.

```
FileUniqueness "s3://amzn-s3-demo-bucket/" > 0.5 with recentFiles = 1
```

URI Regex

> **Note**
>
> Le `uriRegex` tag est disponible dans AWS Glue 5.0 et versions ultérieures.

Cette balise filtre les fichiers en appliquant un modèle regex au chemin du fichier. Seuls les fichiers dont le chemin correspond au modèle sont traités. Vous pouvez également utiliser une prévision négative pour exclure les fichiers qui correspondent à un modèle.

```
# Match only files with a .csv extension
FileUniqueness "s3://bucket/" > 0.5 with uriRegex = "\.csv$"
# Exclude files ending in .tmp using a negative lookahead
FileUniqueness "s3://bucket/" > 0.5 with uriRegex = "(?!.*\.tmp$).*"
```

Ordre du filtre

> **Note**
>
> Le `filterOrder` tag est disponible dans AWS Glue 5.0 et versions ultérieures.

Lorsque vous utilisez plusieurs balises de filtre, telles que `recentFiles` et `uriRegex` ensemble, la `filterOrder` balise contrôle l'ordre dans lequel elles sont appliquées. L'ordre par défaut est `recentFiles` le premier, puis`uriRegex`.

```
FileUniqueness "s3://bucket/" > 0.5 with recentFiles = 5 with uriRegex = "\.csv$" with
 filterOrder = ["uriRegex","recentFiles"]
```

matchFileName

Cette balise garantit que les fichiers ne comportent pas de noms en double. Le comportement par défaut est false.

```
FileUniqueness "s3://amzn-s3-demo-bucket/" > 0.5 with matchFileName = "true"
```

Voici quelques considérations à prendre en compte :

1. Dans AWS Glue ETL, vous devez disposer de la EvaluateDataQualitytransformation immédiatement après une transformation d'Amazon S3 ou de AWS Glue Data Catalog.
2. Cette règle ne fonctionnera pas dans les sessions AWS Glue Interactive.

FileSize

Le FileSize type de règle vous permet de vous assurer que les fichiers répondent à certains critères de taille de fichier. Cela s'avère utile dans les cas d'utilisation suivants :

1. Assurez-vous que les producteurs n'envoient pas de fichiers vides ou nettement plus petits en traitement.
2. Assurez-vous que vos compartiments cible ne contiennent pas de fichiers plus petits, ce qui pourrait entraîner des problèmes de performances.

FileSize rassemble les métriques suivantes :

1. Compliance : renvoie le pourcentage de fichiers qui atteignent le seuil de règle que vous avez établi
2. File Count : nombre de fichiers analysés par la règle
3. Taille minimale du fichier en octets
4. Taille maximale du fichier en octets

```
Dataset.*.FileSize.Compliance: 1.00,
Dataset.*.FileCount: 8.00,
Dataset.*.MaximumFileSize: 327413121.00,
Dataset.*.MinimumFileSize: 204558920.00
```

La détection des anomalies n'est pas prise en charge pour ces métriques.

Valider la taille des fichiers

Cette règle est acceptée lorsque l'élément file.dat est supérieur à 2 Mo.

```
FileSize "s3://amzn-s3-demo-bucket/file.dat" > 2 MB
```

Les unités prises en charge sont les suivantes : o (octets), Mo (mégaoctets), Go (gigaoctets) et To (téraoctets).

Valider la taille des fichiers dans les dossiers

```
FileSize "s3://bucket/" > 5 B
FileSize "s3://bucket/" < 2 GB
```

Cette règle sera acceptée si 70 % des fichiers dans s3://amzn-s3-demo-bucket ont une taille comprise entre 2 Go et 1 To.

```
FileSize "s3://amzn-s3-demo-bucket/" between 2 GB and 1 TB  with threshold > 0.7
```

Déduction des noms de fichiers directement à partir de trames de données

Il n'est pas toujours nécessaire de fournir un chemin d'accès au fichier. Par exemple, lorsque vous créez la règle dans le catalogue de données, il peut être difficile de trouver les dossiers utilisés par les tables du catalogue. AWS Glue Data Quality peut trouver les dossiers ou fichiers spécifiques utilisés pour remplir votre bloc de données.

> **Note**
>
> Cette fonctionnalité ne fonctionne que lorsque les fichiers sont correctement lus dans le DynamicFrame ou DataFrame.

```
FileSize < 10 MB with threshold > 0.7
```

Balises de règles facultatives basées sur des fichiers :

Les balises vous permettent de contrôler le comportement des règles.

recentFiles

Cette balise limite le nombre de fichiers traités en conservant le fichier le plus récent en premier.

```
FileSize "s3://amzn-s3-demo-bucket/" > 5 B with recentFiles = 1
```

URI Regex

> **Note**
>
> Le `uriRegex` tag est disponible dans AWS Glue 5.0 et versions ultérieures.

Cette balise filtre les fichiers en appliquant un modèle regex au chemin du fichier. Seuls les fichiers dont le chemin correspond au modèle sont traités. Vous pouvez également utiliser une prévision négative pour exclure les fichiers qui correspondent à un modèle.

```
# Match only files with a .dat extension
FileSize "s3://bucket/" > 5 B with uriRegex = "\.dat$"
# Exclude files ending in .tmp using a negative lookahead
FileSize "s3://bucket/" > 5 B with uriRegex = "(?!.*\.tmp$).*"
```

Ordre du filtre

> **Note**
>
> Le `filterOrder` tag est disponible dans AWS Glue 5.0 et versions ultérieures.

Lorsque vous utilisez plusieurs balises de filtre telles que `recentFiles` et `uriRegex` ensemble, la `filterOrder` balise contrôle l'ordre dans lequel elles sont appliquées. L'ordre par défaut est `recentFiles` le premier, puis`uriRegex`.

```
FileSize "s3://bucket/" > 5 B with recentFiles = 5 with uriRegex = "\.dat$" with
 filterOrder = ["uriRegex","recentFiles"]
```

FailFast

Lorsqu'elle est définie sur`"true"`, la règle renvoie l'échec immédiatement au premier fichier qui ne répond pas à la condition de taille, au lieu d'évaluer tous les fichiers et de calculer un ratio de conformité.

```
FileSize "s3://bucket/" > 2 MB with failFast = "true"
```

Voici quelques considérations à prendre en compte :

1. Dans AWS Glue ETL, vous devez disposer d' DataQuality Evaluate Transform immédiatement après la transformation d'Amazon S3 ou de Data Catalog.
2. Cette règle ne fonctionnera pas dans les sessions AWS Glue Interactive.

Utilisation APIs pour mesurer et gérer la qualité des données

Cette rubrique décrit comment l'utiliser APIs pour mesurer et gérer la qualité des données.

Table des matières

- Conditions préalables
- Utilisation des recommandations relatives à AWS la qualité des données de Glue
- Travailler avec les ensembles de règles de qualité des données de AWS Glue
- Travailler avec AWS Glue Data Quality s'exécute
- Utilisation des résultats AWS de Glue Data Quality

Conditions préalables

- Assurez-vous que votre version de boto3 est à jour afin d'inclure la dernière API AWS Glue Data Quality.
- Assurez-vous que la version de votre AWS CLI est à jour, afin d'inclure la dernière version de la CLI.

Si vous utilisez une tâche AWS Glue pour les exécuter APIs, vous pouvez utiliser l'option suivante pour mettre à jour la dernière version de la bibliothèque boto3 :

```
–additional-python-modules boto3==<version>
```

Utilisation des recommandations relatives à AWS la qualité des données de Glue

Pour démarrer une recommandation de AWS Glue Data Quality, exécutez :

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 AWS Glue client.
        """
        self.glue_client = glue_client
```

```
   def start_data_quality_rule_recommendation_run(self, database_name, table_name,
role_arn):
       """
       Starts a recommendation run that is used to generate rules when you don't know
what rules to write. AWS Glue Data Quality
       analyzes the data and comes up with recommendations for a potential ruleset.
You can then triage the ruleset
       and modify the generated ruleset to your liking.

       :param database_name: The name of the AWS Glue database which contains the
dataset.
       :param table_name: The name of the AWS Glue table against which we want a
recommendation
       :param role_arn: The Amazon Resource Name (ARN) of an AWS Identity and Access
Management (IAM) role that grants permission to let AWS Glue access the resources it
needs.

       """
       try:
           response = self.client.start_data_quality_rule_recommendation_run(
               DataSource={
                   'GlueTable': {
                       'DatabaseName': database_name,
                       'TableName': table_name
                   }
               },
               Role=role_arn
           )
       except ClientError as err:
           logger.error(
               "Couldn't start data quality recommendation run %s. Here's why: %s:
%s", name,
               err.response['Error']['Code'], err.response['Error']['Message'])
           raise
       else:
           return response['RunId']
```

Pour une exécution de recommandation, vous pouvez utiliser vos `pushDownPredicates` ou vos `catalogPartitionPredicates` pour améliorer les performances et exécuter des recommandations uniquement sur des partitions spécifiques de vos sources de catalogue.

```
client.start_data_quality_rule_recommendation_run(
           DataSource={
```

```
                'GlueTable': {
                    'DatabaseName': database_name,
                    'TableName': table_name,
                    'AdditionalOptions': {
                        'pushDownPredicate': "year=2022"
                    }
                }
            },
            Role=role_arn,
            NumberOfWorkers=2,
            CreatedRulesetName='<rule_set_name>'
  )
```

Pour obtenir les résultats d'une recommandation de AWS Glue Data Quality, exécutez :

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 AWS Glue client.
        """
        self.glue_client = glue_client

    def get_data_quality_rule_recommendation_run(self, run_id):
        """
        Gets the specified recommendation run that was used to generate rules.

        :param run_id: The id of the data quality recommendation run

        """
        try:
            response =
 self.client.get_data_quality_rule_recommendation_run(RunId=run_id)
        except ClientError as err:
            logger.error(
                "Couldn't get data quality recommendation run %. Here's why: %s: %s",
 run_id,
                err.response['Error']['Code'], err.response['Error']['Message'])
            raise
        else:
            return response
```

À partir de l'objet de réponse ci-dessus, vous pouvez extraire celui RuleSet qui a été recommandé par l'exécution, pour l'utiliser dans les étapes suivantes :

```
print(response['RecommendedRuleset'])

Rules = [
    RowCount between 2000 and 8000,
    IsComplete "col1",
    IsComplete "col2",
    StandardDeviation "col3" between 58138330.8 and 64258155.09,
    ColumnValues "col4" between 1000042965 and 1214474826,
    IsComplete "col5"
]
```

Pour obtenir une liste de toutes vos exécutions de recommandation qui peuvent être filtrées et répertoriées :

```
response = client.list_data_quality_rule_recommendation_runs(
    Filter={
        'DataSource': {
            'GlueTable': {
                'DatabaseName': '<database_name>',
                'TableName': '<table_name>'
            }
        }
)
```

Pour annuler les tâches de recommandation de AWS Glue Data Quality existantes, procédez comme suit :

```
response = client.cancel_data_quality_rule_recommendation_run(
    RunId='dqrun-d4b6b01957fdd79e59866365bf9cb0e40fxxxxxxx'
)
```

Travailler avec les ensembles de règles de qualité des données de AWS Glue

Pour créer un ensemble de règles AWS Glue Data Quality, procédez comme suit :

```
response = client.create_data_quality_ruleset(
```

```
    Name='<ruleset_name>',
    Ruleset='Rules = [IsComplete "col1", IsPrimaryKey "col2", RowCount between 2000 and
 8000]',
    TargetTable={
        'TableName': '<table_name>',
        'DatabaseName': '<database_name>'
    }
)
```

Pour obtenir un ensemble de règles de qualité des données :

```
response = client.get_data_quality_ruleset(
    Name='<ruleset_name>'
)
print(response)
```

Vous pouvez utiliser cette API pour ensuite extraire l'ensemble de règles :

```
print(response['Ruleset'])
```

Pour répertorier tous les ensembles de règles de qualité des données pour une table :

```
response = client.list_data_quality_rulesets()
```

Vous pouvez utiliser la condition de filtrage dans l'API pour filtrer tous les ensembles de règles attachés à une base de données ou à une table spécifique :

```
response = client.list_data_quality_rulesets(
    Filter={
        'TargetTable': {
            'TableName': '<table_name>',
            'DatabaseName': '<database_name>'
        }
    },
)
```

Pour mettre à jour un ensemble de règles de qualité des données :

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""
```

```
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 AWS Glue client.
        """
        self.glue_client = glue_client

    def update_data_quality_ruleset(self, ruleset_name, ruleset_string):
        """
        Update an AWS Glue Data Quality Ruleset

        :param ruleset_name: The name of the AWS Glue Data Quality ruleset to update
        :param ruleset_string: The DQDL ruleset string to update the ruleset with

        """
        try:
            response = self.client.update_data_quality_ruleset(
                Name=ruleset_name,
                Ruleset=ruleset_string
            )
        except ClientError as err:
            logger.error(
                "Couldn't update the AWS Glue Data Quality ruleset. Here's why: %s:
 %s",
                err.response['Error']['Code'], err.response['Error']['Message'])
            raise
        else:
            return response
```

Pour supprimer un ensemble de règles de qualité des données :

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 AWS Glue client.
        """
        self.glue_client = glue_client

    def delete_data_quality_ruleset(self, ruleset_name):
        """
        Delete a AWS Glue Data Quality Ruleset

        :param ruleset_name: The name of the AWS Glue Data Quality ruleset to delete
```

```
        """
        try:
            response = self.client.delete_data_quality_ruleset(
                Name=ruleset_name
            )
        except ClientError as err:
            logger.error(
                "Couldn't delete the AWS Glue Data Quality ruleset. Here's why: %s:
%s",
                err.response['Error']['Code'], err.response['Error']['Message'])
            raise
        else:
            return response
```

Travailler avec AWS Glue Data Quality s'exécute

Pour démarrer une exécution de AWS Glue Data Quality, procédez comme suit :

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 AWS Glue client.
        """
        self.glue_client = glue_client

    def start_data_quality_ruleset_evaluation_run(self, database_name, table_name,
 role_name, ruleset_list):
        """
        Start an AWS Glue Data Quality evaluation run

        :param database_name: The name of the AWS Glue database which contains the
 dataset.
        :param table_name: The name of the AWS Glue table against which we want to
 evaluate.
        :param role_arn: The Amazon Resource Name (ARN) of an AWS Identity and Access
 Management (IAM) role that grants permission to let AWS Glue access the resources it
 needs.
        :param ruleset_list: The list of AWS Glue Data Quality ruleset names to
 evaluate.

        """
```

```
        try:
            response = client.start_data_quality_ruleset_evaluation_run(
                DataSource={
                    'GlueTable': {
                        'DatabaseName': database_name,
                        'TableName': table_name
                    }
                },
                Role=role_name,
                RulesetNames=ruleset_list
            )
        except ClientError as err:
            logger.error(
                "Couldn't start the AWS Glue Data Quality Run. Here's why: %s: %s",
                err.response['Error']['Code'], err.response['Error']['Message'])
            raise
        else:
            return response['RunId']
```

N'oubliez pas que vous pouvez transmettre un paramètre `pushDownPredicate` ou `catalogPartitionPredicate` pour garantir que votre exécution de qualité des données ne cible qu'un ensemble spécifique de partitions dans votre table de catalogue. Par exemple :

```
response = client.start_data_quality_ruleset_evaluation_run(
    DataSource={
        'GlueTable': {
            'DatabaseName': '<database_name>',
            'TableName': '<table_name>',
            'AdditionalOptions': {
                'pushDownPredicate': 'year=2023'
            }
        }
    },
    Role='<role_name>',
    NumberOfWorkers=5,
    Timeout=123,
    AdditionalRunOptions={
        'CloudWatchMetricsEnabled': False
    },
    RulesetNames=[
        '<ruleset_name>',
    ]
```

```
)
```

Vous pouvez également configurer la manière dont les règles composites de votre ensemble de règles sont évaluées, au niveau de la LIGNE ou de la COLONNE. Pour plus d'informations sur le fonctionnement des règles composites, reportez-vous à How composite rules work dans la documentation.

Exemple expliquant comment définir la méthode d'évaluation des règles composites dans votre demande :

```
response = client.start_data_quality_ruleset_evaluation_run(
    DataSource={
        'GlueTable': {
            'DatabaseName': '<database_name>',
            'TableName': '<table_name>',
            'AdditionalOptions': {
                'pushDownPredicate': 'year=2023'
            }
        }
    },
    Role='<role_name>',
    NumberOfWorkers=5,
    Timeout=123,
    AdditionalRunOptions={
        'CompositeRuleEvaluationMethod':ROW
    },
    RulesetNames=[
        '<ruleset_name>',
    ]
)
```

Pour obtenir des informations sur un AWS Glue Data Quality, exécutez :

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 AWS Glue client.
        """
        self.glue_client = glue_client

    def get_data_quality_ruleset_evaluation_run(self, run_id):
```

```
        """
        Get details about an AWS Glue Data Quality Run

        :param run_id: The AWS Glue Data Quality run ID to look up

        """
        try:
            response = self.client.get_data_quality_ruleset_evaluation_run(
                RunId=run_id
            )
        except ClientError as err:
            logger.error(
                "Couldn't look up the AWS Glue Data Quality run ID. Here's why: %s:
 %s",
                err.response['Error']['Code'], err.response['Error']['Message'])
            raise
        else:
            return response
```

Pour obtenir les résultats d'une exécution de AWS Glue Data Quality :

Pour une exécution de AWS Glue Data Quality donnée, vous pouvez extraire les résultats de l'évaluation de l'exécution à l'aide de la méthode suivante :

```
response = client.get_data_quality_ruleset_evaluation_run(
    RunId='d4b6b01957fdd79e59866365bf9cb0e40fxxxxxxx'
)

resultID = response['ResultIds'][0]

response = client.get_data_quality_result(
    ResultId=resultID
)

print(response['RuleResults'])
```

Pour répertorier toutes vos exécutions de AWS Glue Data Quality, procédez comme suit :

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 AWS Glue client.
```

```
        """
        self.glue_client = glue_client

    def list_data_quality_ruleset_evaluation_runs(self, database_name, table_name):
        """
        Lists all the AWS Glue Data Quality runs against a given table

        :param database_name: The name of the database where the data quality runs
        :param table_name: The name of the table against which the data quality runs
 were created

        """
        try:
            response = self.client.list_data_quality_ruleset_evaluation_runs(
                Filter={
                    'DataSource': {
                        'GlueTable': {
                            'DatabaseName': database_name,
                            'TableName': table_name
                        }
                    }
                }
            )
        except ClientError as err:
            logger.error(
                "Couldn't list the AWS Glue Quality runs. Here's why: %s: %s",
                err.response['Error']['Code'], err.response['Error']['Message'])
            raise
        else:
            return response
```

Vous pouvez modifier la clause de filtrage pour n'afficher que les résultats obtenus à des moments précis ou concernant des tables spécifiques.

Pour arrêter une exécution de AWS Glue Data Quality en cours, procédez comme suit :

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 AWS Glue client.
        """
        self.glue_client = glue_client
```

```
    def cancel_data_quality_ruleset_evaluation_run(self, result_id):
        """
        Cancels a given AWS Glue Data Quality run

        :param result_id: The result id of a AWS Glue Data Quality run to cancel

        """
        try:
            response = self.client.cancel_data_quality_ruleset_evaluation_run(
                ResultId=result_id
            )
        except ClientError as err:
            logger.error(
                "Couldn't cancel the AWS Glue Data Quality run. Here's why: %s: %s",
                err.response['Error']['Code'], err.response['Error']['Message'])
            raise
        else:
            return response
```

Utilisation des résultats AWS de Glue Data Quality

Pour obtenir les résultats de votre exécution avec AWS Glue Data Quality, procédez comme suit :

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 AWS Glue client.
        """
        self.glue_client = glue_client

    def get_data_quality_result(self, result_id):
        """
        Outputs the result of an AWS Glue Data Quality Result

        :param result_id: The result id of an AWS Glue Data Quality run

        """
        try:
            response = self.client.get_data_quality_result(
                ResultId=result_id
            )
```

```
        except ClientError as err:
            logger.error(
                "Couldn't get the AWS Glue Data Quality result. Here's why: %s: %s",
                err.response['Error']['Code'], err.response['Error']['Message'])
            raise
        else:
            return response
```

Pour consulter les statistiques recueillies pour un résultat de qualité des données donné :

```
import boto3
from botocore.exceptions import ClientError
import logging

logger = logging.getLogger(__name__)
class GlueWrapper:
    """Encapsulates AWS Glue actions."""
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 AWS Glue client.
        """
        self.glue_client = glue_client

    def get_profile_for_data_quality_result(self, result_id):
        """
        Outputs the statistic profile for a AWS Glue Data Quality Result

        :param result_id: The result id of a AWS Glue Data Quality run

        """
        try:
            response = self.glue_client.get_data_quality_result(
                ResultId=result_id
            )

            # the profile contains all statistics gathered for the result
            profile_id = response['ProfileId']
            profile = self.glue_client.list_data_quality_statistics(
                ProfileId = profile_id
            )
            return profile
        except ClientError as err:
            logger.error(
```

```
                "Couldn't retrieve Data Quality profile. Here's why: %s: %s",
                err.response['Error']['Code'], err.response['Error']['Message'])
            raise
```

Pour consulter les séries temporelles d'une statistique collectée au cours de plusieurs cycles de qualité des données :

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 AWS Glue client.
        """
        self.glue_client = glue_client

    def get_statistics_for_data_quality_result(self, profile_id):
        """
        Outputs an array of datapoints for each statistic in the input result.

        :param result_id: The profile id of a AWS Glue Data Quality run

        """
        try:
            profile = self.glue_client.list_data_quality_statistics(
                ProfileId = profile_id
            )
            statistics = [self.glue_client.list_data_quality_statistics(
                StatisticId = s['StatisticId']
            ) for s in profile['Statistics']]
            return statistics
        except ClientError as err:
            logger.error(
                "Couldn't retrieve Data Quality statistics. Here's why: %s: %s",
                err.response['Error']['Code'], err.response['Error']['Message'])
            raise
```

Pour afficher le modèle de détection d'anomalies pour une statistique spécifique :

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 AWS Glue client.
```

```
        """
        self.glue_client = glue_client

    def get_model_training_result_for_statistic(self, statistic_id, profile_id):
        """
        Outputs the details (bounds) of anomaly detection training for the given
 statistic at the given profile.

        :param statistic_id the model's statistic (the timeseries it is tracking)
        :param profile_id the profile associated with the model (a point in the
 timeseries)

        """
        try:
            model = self.glue_client.get_data_quality_model_result(
                ProfileId = profile_id, StatisticId = statistic_id
            )
            return model
        except ClientError as err:
            logger.error(
                "Couldn't retrieve Data Quality model results. Here's why: %s: %s",
                err.response['Error']['Code'], err.response['Error']['Message'])
            raise
```

Pour exclure un point de données de la ligne de base de détection des anomalies de son modèle statistique :

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 AWS Glue client.
        """
        self.glue_client = glue_client

    def apply_exclusions_to_statistic(self, statistic_id, profile_ids):
        """
        Annotate some points along a given statistic timeseries.

        This example excludes the provided values; INCLUDE can also be used to undo
 this action.

        :param statistic_id the statistic timeseries to annotate
```

```
        :param profile_id the profiles we want to exclude (points in the timeseries)

        """

        try:
            response = self.glue_client.batch_put_data_quality_statistic_annotation(
                    InclusionAnnotations = [
                        {'ProfileId': prof_id,
                        'StatisticId': statistic_id,
                        'InclusionAnnotation': 'EXCLUDE'} for prof_id in profile_ids
                    ]
            )
            return response['FailedInclusionAnnotations']
        except ClientError as err:
            logger.error(
                "Couldn't store Data Quality annotations. Here's why: %s: %s",
                err.response['Error']['Code'], err.response['Error']['Message'])
            raise
```

Pour consulter l'état de l'entraînement des modèles de détection des anomalies pour une statistique spécifique :

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 AWS Glue client.
        """
        self.glue_client = glue_client

    def get_model_training_status_for_statistic(self, statistic_id, profile_id):
        """
        Outputs the status of anomaly detection training for the given statistic at the
 given profile.

        :param statistic_id the model's statistic (the timeseries it is tracking)
        :param profile_id the profile associated with the model (a point in the
 timeseries)

        """
        try:
            model = self.glue_client.get_data_quality_model(
                ProfileId = profile_id, StatisticId = statistic_id
```

```
            )
            return model
        except ClientError as err:
            logger.error(
                "Couldn't retrieve Data Quality statistics. Here's why: %s: %s",
                err.response['Error']['Code'], err.response['Error']['Message'])
            raise
```

Pour exclure tous les résultats d'une analyse de qualité des données spécifique des références de détection des anomalies :

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 AWS Glue client.
        """
        self.glue_client = glue_client

    def apply_exclusions_to_profile(self, profile_id):
        """
        Exclude datapoints produced by a run across statistic timeseries.

        This example excludes the provided values; INCLUDE can also be used to undo
 this action.

        :param profile_id the profiles we want to exclude (points in the timeseries)

        """
        try:
            response = self.glue_client.put_data_quality_profile_annotation(
                    ProfileId = profile_id,
                    InclusionAnnotation = "EXCLUDE"
            )
            return response
        except ClientError as err:
            logger.error(
                "Couldn't store Data Quality annotations. Here's why: %s: %s",
                err.response['Error']['Code'], err.response['Error']['Message'])
            raise
```

Pour obtenir les résultats d'une exécution de la qualité des données et afficher les résultats :

Avec une AWS Glue Data Quality`runID`, vous pouvez les extraire `resultID` pour obtenir les résultats réels, comme indiqué ci-dessous :

```
response = client.get_data_quality_ruleset_evaluation_run(
    RunId='dqrun-abca77ee126abe1378c1da1ae0750d7dxxxx'
)

resultID = response['ResultIds'][0]

response = client.get_data_quality_result(
    ResultId=resultID
)

print(resp['RuleResults'])
```

Configuration des alertes, des déploiements et de la planification

Cette rubrique explique comment configurer les alertes, les déploiements et la planification pour AWS Glue Data Quality.

Table des matières

- Configuration des alertes et des notifications dans le cadre de EventBridge l'intégration Amazon
  - Options de configuration supplémentaires pour le modèle d'événement
  - Formatage des notifications sous forme d'e-mails
- Configurer des alertes et des notifications lors de l' CloudWatch intégration
- Interrogation des résultats de la qualité des données pour créer des tableaux de bord
- Déploiement de règles de qualité des données en utilisant AWS CloudFormation
- Planification de règles de qualité des données

Configuration des alertes et des notifications dans le cadre de EventBridge l'intégration Amazon

AWS Glue Data Quality prend en charge la publication d' EventBridge événements, qui sont émis à la fin d'une évaluation d'un ensemble de règles de qualité des données. Vous pouvez ainsi facilement configurer des alertes en cas d'échec des règles de qualité des données.

Voici un exemple d'événement lorsque vous évaluez des ensembles de règles de qualité des données dans le catalogue de données. Grâce à ces informations, vous pouvez consulter les données mises à disposition par Amazon EventBridge. Vous pouvez lancer des appels d'API supplémentaires pour obtenir plus de détails. Par exemple, appelez l'API `get_data_quality_result` avec l'ID du résultat pour obtenir les détails d'une exécution particulière.

```
{
    "version":"0",
    "id":"abcdef00-1234-5678-9abc-def012345678",
    "detail-type":"Data Quality Evaluation Results Available",
    "source":"aws.glue-dataquality",
    "account":"123456789012",
    "time":"2017-09-07T18:57:21Z",
    "region":"us-west-2",
    "resources":[],
    "detail":{
        "context": {
                    "contextType": "GLUE_DATA_CATALOG",
                    "runId":"dqrun-12334567890",
                    "databaseName": "db-123",
                    "tableName": "table-123",
                    "catalogId": "123456789012"
                    },
        "resultID": "dqresult-12334567890",
        "rulesetNames": ["rulset1"],
        "state":"SUCCEEDED",
        "score": 1.00,
        "rulesSucceeded": 100,
        "rulesFailed": 0,
        "rulesSkipped": 0
    }
}
```

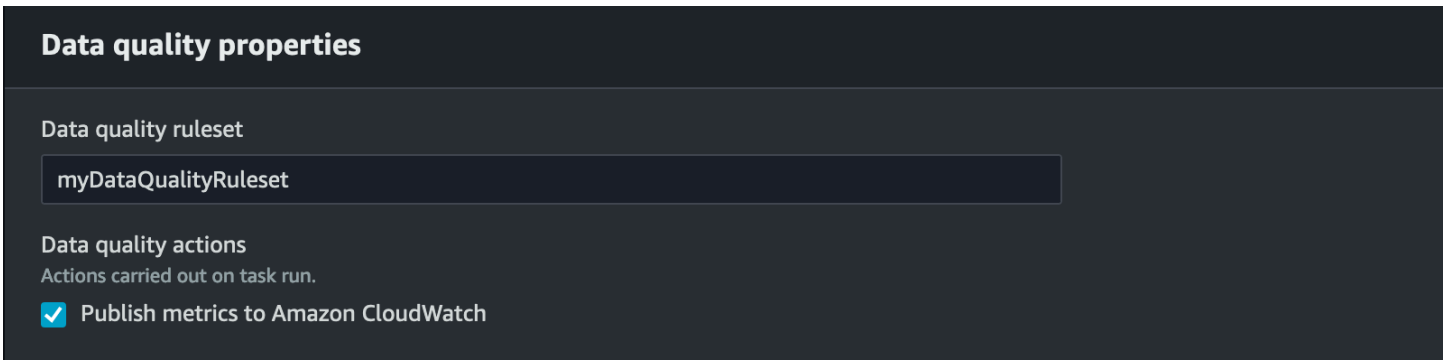
Voici un exemple d'événement publié lorsque vous évaluez des ensembles de règles de qualité des données dans les blocs-notes AWS Glue ETL ou AWS Glue Studio.

```
{
    "version":"0",
    "id":"abcdef00-1234-5678-9abc-def012345678",
    "detail-type":"Data Quality Evaluation Results Available",
    "source":"aws.glue-dataquality",
```

```
    "account":"123456789012",
    "time":"2017-09-07T18:57:21Z",
    "region":"us-west-2",
    "resources":[],
    "detail":{
        "context": {
                    "contextType": "GLUE_JOB",
                    "jobId": "jr-12334567890",
                    "jobName": "dq-eval-job-1234",
                    "evaluationContext": "",
                    }
        "resultID": "dqresult-12334567890",
        "rulesetNames": ["rulset1"],
        "state":"SUCCEEDED",
        "score": 1.00
        "rulesSucceeded": 100,
        "rulesFailed": 0,
        "rulesSkipped": 0
    }
}
```

Pour que l'évaluation de la qualité des données soit exécutée à la fois dans le catalogue de données et dans les tâches ETL, l' Amazon CloudWatch option Publier les métriques sur, sélectionnée par défaut, doit rester sélectionnée pour que la EventBridge publication fonctionne.

Configuration des EventBridge notifications



Pour recevoir les événements émis et définir des cibles, vous devez configurer les EventBridge règles Amazon. Pour créer des règles :

1. Ouvrez la EventBridge console Amazon.
2. Choisissez Règles dans la section Bus de la barre de navigation.
3. Choisissez Create Rule (Créer une règle).

4. Sous Définir les détails de la règle :
   a. Pour le nom, saisissez `myDQRule`.
   b. Saisissez une description (facultatif).
   c. Pour le bus d'événements, sélectionnez votre bus d'événements. Si vous n'en avez pas, laissez le bus par défaut.
   d. Pour Type de règle, sélectionnez Règle avec un modèle d'événement, puis choisissez Suivant.
5. Sous Créer un modèle d'événement :
   a. Pour la source de l'événement, sélectionnez AWS les événements ou les événements EventBridge partenaires.
   b. Ignorez la section de l'exemple d'événement.
   c. Pour la méthode de création, sélectionnez Utiliser le formulaire de modèle.
   d. Pour le modèle d'événement :
      i. Sélectionnez Services AWS pour Source de l'événement.
      ii. Sélectionnez Glue Data Quality pour le AWS service.
      iii. Sélectionnez Résultats de l'évaluation de la qualité des données disponibles pour Type d'événement.
      iv. Sélectionnez FAILED pour Statuts spécifiques. Vous voyez alors un modèle d'événement similaire à celui présenté ci-dessous :

```
{
  "source": ["aws.glue-dataquality"],
  "detail-type": ["Data Quality Evaluation Results Available"],
  "detail": {
    "state": ["FAILED"]
  }
}
```

      v. Pour plus d'options de configuration, consultez Options de configuration supplémentaires pour le modèle d'événement.
6. Sous Cibles sélectionnées :
   a. Pour Types de cibles, sélectionnez Service AWS .
   b. Utilisez le menu déroulant Sélectionnez une cible pour choisir le AWS service auquel vous souhaitez vous connecter (SNS, Lambda, SQS, etc.), puis choisissez Next.
7. Sous Configurer les balises, cliquez sur Ajouter de nouvelles balises pour ajouter des balises facultatives, puis choisissez Suivant.

8. Une page récapitulative de toutes les sélections s'affiche. Choisissez Créer une règle en bas de la page.

Options de configuration supplémentaires pour le modèle d'événement

En plus de filtrer de votre événement en fonction de sa réussite ou de son échec, vous pouvez également filtrer les événements en fonction de différents paramètres.

Pour ce faire, accédez à la section Modèle d'événement et sélectionnez Modifier le modèle pour spécifier des paramètres supplémentaires. Notez que les champs du modèle d'événements sont sensibles à la casse. Voici des exemples de configuration de modèle d'événement.

Pour capturer les événements d'une table particulière évaluant des ensembles de règles spécifiques, utilisez ce type de modèle :

```
{
  "source": ["aws.glue-dataquality"],
  "detail-type": ["Data Quality Evaluation Results Available"],
  "detail": {
    "context": {
      "contextType": ["GLUE_DATA_CATALOG"],
      "databaseName": "db-123",
       "tableName": "table-123",
    },
    "rulesetNames": ["ruleset1", "ruleset2"]
    "state": ["FAILED"]
  }
}
```

Pour capturer des événements à partir de tâches spécifiques dans l'expérience ETL, utilisez ce type de modèle :

```
{
  "source": ["aws.glue-dataquality"],
  "detail-type": ["Data Quality Evaluation Results Available"],
  "detail": {
    "context": {
      "contextType": ["GLUE_JOB"],
      "jobName": ["dq_evaluation_job1", "dq_evaluation_job2"]
    },
    "state": ["FAILED"]
```

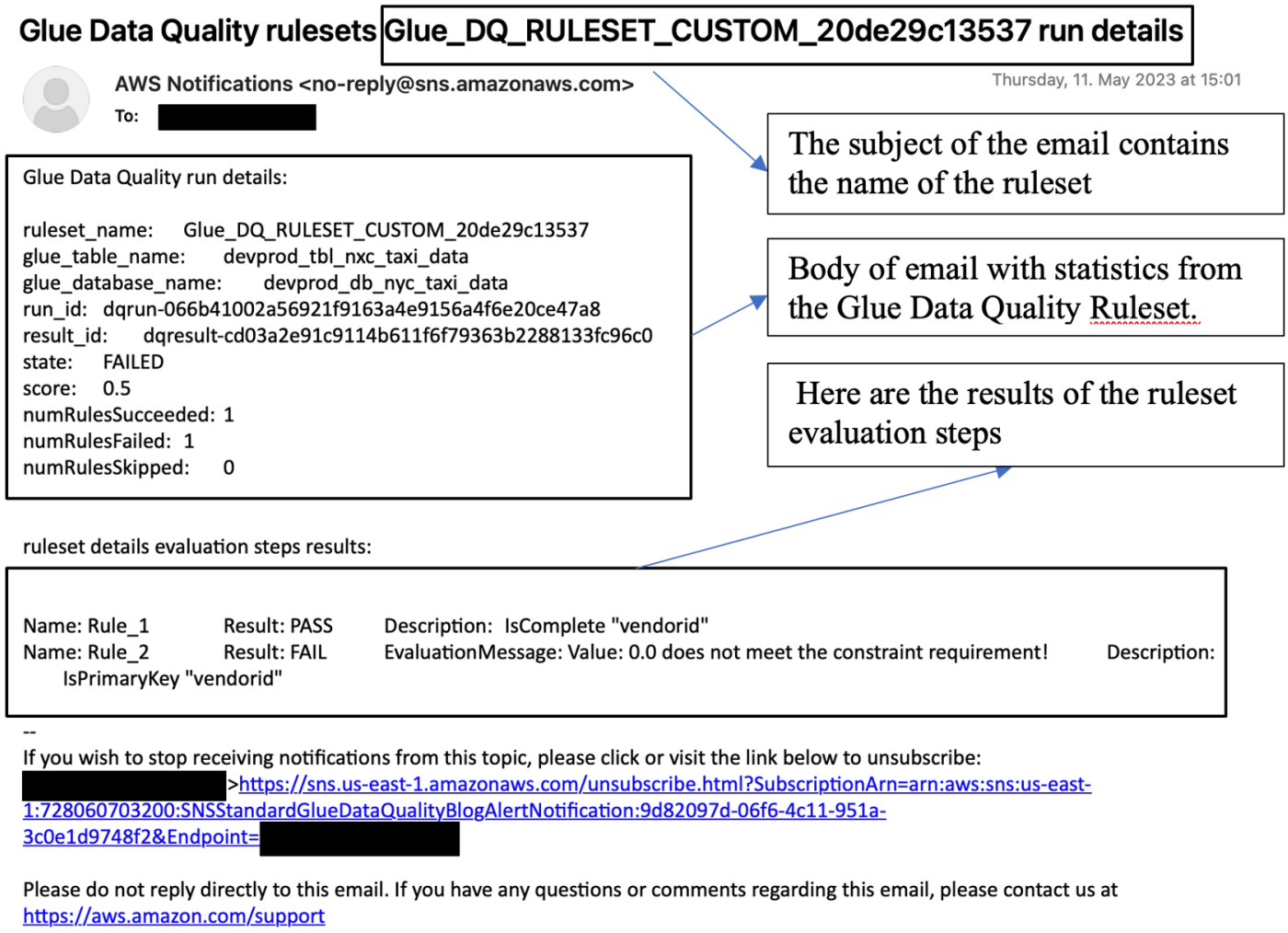
```
  }
}
```

Pour capturer les événements dont le score est inférieur à un seuil spécifique (par exemple 70 %) :

```
{
  "source": ["aws.glue-dataquality"],
  "detail-type": ["Data Quality Evaluation Results Available"],
  "detail": {
    "score": [{
      "numeric": ["<=", 0.7]
    }]
  }
}
```

Formatage des notifications sous forme d'e-mails

Vous devez parfois envoyer une notification par e-mail correctement formatée à vos équipes métier. Vous pouvez utiliser Amazon EventBridge et AWS Lambda pour y parvenir.



L'exemple de code suivant peut être utilisé pour formater vos notifications de qualité des données afin de générer des e-mails.

```
import boto3
import json
from datetime import datetime

sns_client = boto3.client('sns')
glue_client = boto3.client('glue')

sns_topic_arn = 'arn:aws:sns:<region-code>:<account-id>:<sns-topic-name>'



def lambda_handler(event, context):
    log_metadata = {}
```

```
    message_text = ""
    subject_text = ""

    if event['detail']['context']['contextType'] == 'GLUE_DATA_CATALOG':
        log_metadata['ruleset_name'] = str(event['detail']['rulesetNames'][0])
        log_metadata['tableName'] = str(event['detail']['context']['tableName'])
        log_metadata['databaseName'] = str(event['detail']['context']['databaseName'])
        log_metadata['runId'] = str(event['detail']['context']['runId'])
        log_metadata['resultId'] = str(event['detail']['resultId'])
        log_metadata['state'] = str(event['detail']['state'])
        log_metadata['score'] = str(event['detail']['score'])
        log_metadata['numRulesSucceeded'] = str(event['detail']['numRulesSucceeded'])
        log_metadata['numRulesFailed'] = str(event['detail']['numRulesFailed'])
        log_metadata['numRulesSkipped'] = str(event['detail']['numRulesSkipped'])

        message_text += "Glue Data Quality run details:\n"
        message_text += "ruleset_name: {}\n".format(log_metadata['ruleset_name'])
        message_text += "glue_table_name: {}\n".format(log_metadata['tableName'])
        message_text += "glue_database_name: {}\n".format(log_metadata['databaseName'])
        message_text += "run_id: {}\n".format(log_metadata['runId'])
        message_text += "result_id: {}\n".format(log_metadata['resultId'])
        message_text += "state: {}\n".format(log_metadata['state'])
        message_text += "score: {}\n".format(log_metadata['score'])
        message_text += "numRulesSucceeded:
{}\n".format(log_metadata['numRulesSucceeded'])
        message_text += "numRulesFailed: {}\n".format(log_metadata['numRulesFailed'])
        message_text += "numRulesSkipped: {}\n".format(log_metadata['numRulesSkipped'])

        subject_text = "Glue Data Quality ruleset {} run
details".format(log_metadata['ruleset_name'])

    else:
        log_metadata['ruleset_name'] = str(event['detail']['rulesetNames'][0])
        log_metadata['jobName'] = str(event['detail']['context']['jobName'])
        log_metadata['jobId'] = str(event['detail']['context']['jobId'])
        log_metadata['resultId'] = str(event['detail']['resultId'])
        log_metadata['state'] = str(event['detail']['state'])
        log_metadata['score'] = str(event['detail']['score'])

        log_metadata['numRulesSucceeded'] = str(event['detail']['numRulesSucceeded'])
        log_metadata['numRulesFailed'] = str(event['detail']['numRulesFailed'])
        log_metadata['numRulesSkipped'] = str(event['detail']['numRulesSkipped'])

        message_text += "Glue Data Quality run details:\n"
```

```
        message_text += "ruleset_name: {}\n".format(log_metadata['ruleset_name'])
        message_text += "glue_job_name: {}\n".format(log_metadata['jobName'])
        message_text += "job_id: {}\n".format(log_metadata['jobId'])
        message_text += "result_id: {}\n".format(log_metadata['resultId'])
        message_text += "state: {}\n".format(log_metadata['state'])
        message_text += "score: {}\n".format(log_metadata['score'])
        message_text += "numRulesSucceeded:
{}\n".format(log_metadata['numRulesSucceeded'])
        message_text += "numRulesFailed: {}\n".format(log_metadata['numRulesFailed'])
        message_text += "numRulesSkipped: {}\n".format(log_metadata['numRulesSkipped'])

        subject_text = "Glue Data Quality ruleset {} run
details".format(log_metadata['ruleset_name'])

    resultID = str(event['detail']['resultId'])
    response = glue_client.get_data_quality_result(ResultId=resultID)
    RuleResults = response['RuleResults']
    message_text += "\n\nruleset details evaluation steps results:\n\n"
    subresult_info = []

    for dic in RuleResults:
        subresult = "Name: {}\t\tResult: {}\t\tDescription: \t{}".format(dic['Name'],
dic['Result'], dic['Description'])
        if 'EvaluationMessage' in dic:
            subresult += "\t\tEvaluationMessage: {}".format(dic['EvaluationMessage'])
        subresult_info.append({
            'Name': dic['Name'],
            'Result': dic['Result'],
            'Description': dic['Description'],
            'EvaluationMessage': dic.get('EvaluationMessage', '')
        })
        message_text += "\n" + subresult

    log_metadata['resultrun'] = subresult_info



    sns_client.publish(
        TopicArn=sns_topic_arn,
        Message=message_text,
        Subject=subject_text
    )

    return {
```

```
        'statusCode': 200,
        'body': json.dumps('Message published to SNS topic')
    }
```

Configurer des alertes et des notifications lors de l' CloudWatch intégration

Notre approche recommandée consiste à configurer des alertes relatives à la qualité des données à l'aide d'Amazon EventBridge, car Amazon EventBridge nécessite une configuration unique pour alerter les clients. Cependant, certains clients préfèrent Amazon pour des CloudWatch raisons de familiarité. Pour ces clients, nous proposons une intégration avec Amazon CloudWatch.

Chaque évaluation de la qualité des données de AWS Glue émet une paire de métriques nommées `glue.data.quality.rules.passed` (indiquant le nombre de règles passées) et `glue.data.quality.rules.failed` (indiquant le nombre de règles ayant échoué) par cycle de qualité des données. Vous pouvez utiliser cette métrique émise pour créer des alarmes afin d'avertir les utilisateurs lorsqu'une exécution d'évaluation de la qualité des données se situe en dessous du seuil. Pour commencer à configurer une alarme qui enverrait un e-mail via une notification Amazon SNS, suivez les étapes ci-dessous :

Pour commencer à configurer une alarme qui enverrait un e-mail via une notification Amazon SNS, suivez les étapes ci-dessous :

1. Ouvrez la CloudWatch console Amazon.
2. Choisissez Toutes les métriques sous Métriques. Vous verrez un espace de noms supplémentaire sous Espaces de noms personnalisés nommé qualité des données Glue.

> **Note**
>
> Lorsque vous lancez une AWS analyse de Glue Data Quality, assurez-vous que la CloudWatch case Publish metrics to Amazon est cochée. Dans le cas contraire, les statistiques relatives à cette course en particulier ne seront pas publiées sur Amazon CloudWatch.

Sous l'espace de noms `Glue Data Quality`, vous pouvez voir les métriques émises par table, par ensemble de règles. Dans le cadre de cette rubrique, nous utiliserons la règle `glue.data.quality.rules.failed` et l'alarme si cette valeur est supérieure à 1 (ce qui

signifie que si le nombre d'échecs d'évaluation de la règle est supérieur à 1, nous voulons en être informés).

3. Pour créer l'alarme, sélectionnez Toutes les alarmes sous Alarmes.
4. Choisissez Create alarm (Créer une alerte).
5. Choisissez Sélectionner une métrique.
6. Sélectionnez la métrique `glue.data.quality.rules.failed` correspondant à la table que vous avez créée, puis choisissez Sélectionner une métrique.
7. Sous l'onglet Spécifier les métriques et les conditions, dans la section Métriques :
   a. Pour Statistics (Statistique), choisissez Sum (Somme).
   b. Pour Période, choisissez 1 minute.
8. Sous la section Conditions :
   a. Pour Threshold type (Type de seuil), choisissez Static (Statique).
   b. Pour Chaque fois que l'échec des règles de qualité des données est…, sélectionnez Supérieur ou égal.
   c. Pour à…, saisissez 1 comme valeur de seuil.

   Ces options impliquent que si la métrique `glue.data.quality.rules.failed` émet une valeur supérieure ou égale à 1, nous déclencherons une alarme. Toutefois, s'il n'y a pas de données, nous les traiterons comme acceptables.
9. Choisissez Suivant.
10. Sous Configurer les actions :
    a. Pour la section Déclencheur d'état d'alarme, choisissez En alarme.
    b. Pour la section Envoyer une notification à la rubrique SNS suivante, choisissez Créer une rubrique pour envoyer une notification via une nouvelle rubrique SNS.
    c. Pour Points de terminaison de l'e-mail qui recevra la notification, saisissez votre adresse e-mail. Cliquez ensuite sur Créer une rubrique.
    d. Choisissez Suivant.
11. Pour Nom de l'alarme, saisissez `myFirstDQAlarm` et choisissez Suivant.
12. Une page récapitulative de toutes les sélections s'affiche. Choisissez Créer une alarme en bas de la page.

Vous pouvez désormais voir l'alarme en cours de création depuis le tableau de bord des CloudWatch alarmes Amazon.

Interrogation des résultats de la qualité des données pour créer des tableaux de bord

Vous souhaiterez peut-être créer un tableau de bord pour afficher vos résultats de qualité de données. Il existe deux façons de procéder :

Configurez Amazon EventBridge avec le code suivant pour écrire les données dans Amazon S3 :

```
import boto3
import json
from datetime import datetime


s3_client = boto3.client('s3')
glue_client = boto3.client('glue')


s3_bucket = 's3-bucket-name'

def write_logs(log_metadata):
    try:
        filename = datetime.now().strftime("%m%d%Y%H%M%S") + ".json"
        key_opts = {
            'year': datetime.now().year,
            'month': "{:02d}".format(datetime.now().month),
            'day': "{:02d}".format(datetime.now().day),
            'filename': filename
        }
        s3key = "gluedataqualitylogs/year={year}/month={month}/day={day}/
{filename}".format(**key_opts)
        s3_client.put_object(Bucket=s3_bucket, Key=s3key,
 Body=json.dumps(log_metadata))
    except Exception as e:
        print(f'Error writing logs to S3: {e}')


def lambda_handler(event, context):
    log_metadata = {}
    message_text = ""
    subject_text = ""

    if event['detail']['context']['contextType'] == 'GLUE_DATA_CATALOG':
        log_metadata['ruleset_name'] = str(event['detail']['rulesetNames'][0])
```

```
        log_metadata['tableName'] = str(event['detail']['context']['tableName'])
        log_metadata['databaseName'] = str(event['detail']['context']['databaseName'])
        log_metadata['runId'] = str(event['detail']['context']['runId'])
        log_metadata['resultId'] = str(event['detail']['resultId'])
        log_metadata['state'] = str(event['detail']['state'])
        log_metadata['score'] = str(event['detail']['score'])
        log_metadata['numRulesSucceeded'] = str(event['detail']['numRulesSucceeded'])
        log_metadata['numRulesFailed'] = str(event['detail']['numRulesFailed'])
        log_metadata['numRulesSkipped'] = str(event['detail']['numRulesSkipped'])

        message_text += "Glue Data Quality run details:\n"
        message_text += "ruleset_name: {}\n".format(log_metadata['ruleset_name'])
        message_text += "glue_table_name: {}\n".format(log_metadata['tableName'])
        message_text += "glue_database_name: {}\n".format(log_metadata['databaseName'])
        message_text += "run_id: {}\n".format(log_metadata['runId'])
        message_text += "result_id: {}\n".format(log_metadata['resultId'])
        message_text += "state: {}\n".format(log_metadata['state'])
        message_text += "score: {}\n".format(log_metadata['score'])
        message_text += "numRulesSucceeded:
{}\n".format(log_metadata['numRulesSucceeded'])
        message_text += "numRulesFailed: {}\n".format(log_metadata['numRulesFailed'])
        message_text += "numRulesSkipped: {}\n".format(log_metadata['numRulesSkipped'])

        subject_text = "Glue Data Quality ruleset {} run
details".format(log_metadata['ruleset_name'])

    else:
        log_metadata['ruleset_name'] = str(event['detail']['rulesetNames'][0])
        log_metadata['jobName'] = str(event['detail']['context']['jobName'])
        log_metadata['jobId'] = str(event['detail']['context']['jobId'])
        log_metadata['resultId'] = str(event['detail']['resultId'])
        log_metadata['state'] = str(event['detail']['state'])
        log_metadata['score'] = str(event['detail']['score'])

        log_metadata['numRulesSucceeded'] = str(event['detail']['numRulesSucceeded'])
        log_metadata['numRulesFailed'] = str(event['detail']['numRulesFailed'])
        log_metadata['numRulesSkipped'] = str(event['detail']['numRulesSkipped'])

        message_text += "Glue Data Quality run details:\n"
        message_text += "ruleset_name: {}\n".format(log_metadata['ruleset_name'])
        message_text += "glue_job_name: {}\n".format(log_metadata['jobName'])
        message_text += "job_id: {}\n".format(log_metadata['jobId'])
        message_text += "result_id: {}\n".format(log_metadata['resultId'])
        message_text += "state: {}\n".format(log_metadata['state'])
```

```
        message_text += "score: {}\n".format(log_metadata['score'])
        message_text += "numRulesSucceeded:
{}\n".format(log_metadata['numRulesSucceeded'])
        message_text += "numRulesFailed: {}\n".format(log_metadata['numRulesFailed'])
        message_text += "numRulesSkipped: {}\n".format(log_metadata['numRulesSkipped'])

        subject_text = "Glue Data Quality ruleset {} run
details".format(log_metadata['ruleset_name'])

    resultID = str(event['detail']['resultId'])
    response = glue_client.get_data_quality_result(ResultId=resultID)
    RuleResults = response['RuleResults']
    message_text += "\n\nruleset details evaluation steps results:\n\n"
    subresult_info = []

    for dic in RuleResults:
        subresult = "Name: {}\t\tResult: {}\t\tDescription: \t{}".format(dic['Name'],
dic['Result'], dic['Description'])
        if 'EvaluationMessage' in dic:
            subresult += "\t\tEvaluationMessage: {}".format(dic['EvaluationMessage'])
        subresult_info.append({
            'Name': dic['Name'],
            'Result': dic['Result'],
            'Description': dic['Description'],
            'EvaluationMessage': dic.get('EvaluationMessage', '')
        })
        message_text += "\n" + subresult

    log_metadata['resultrun'] = subresult_info

    write_logs(log_metadata)


    return {
        'statusCode': 200,
        'body': json.dumps('Message published to SNS topic')
    }
```

Après avoir écrit à Amazon S3, vous pouvez utiliser les robots d'exploration AWS Glue pour vous enregistrer auprès d'Athena et interroger les tables.

Configurez un emplacement Amazon S3 lors d'une évaluation de la qualité des données :

Lorsque vous exécutez des tâches de qualité des données dans le AWS Glue Data Catalog ou AWS Glue ETL, vous pouvez fournir un emplacement Amazon S3 pour écrire les résultats de qualité des données sur Amazon S3. Vous pouvez utiliser la syntaxe ci-dessous pour créer une table en référençant la cible afin de lire les résultats de qualité des données.

Notez que vous devez exécuter les requêtes `CREATE EXTERNAL TABLE` et `MSCK REPAIR TABLE` séparément.

```
CREATE EXTERNAL TABLE <my_table_name>(
    catalogid string,
    databasename string,
    tablename string,
    dqrunid string,
    evaluationstartedon timestamp,
    evaluationcompletedon timestamp,
    rule string,
    outcome string,
    failurereason string,
    evaluatedmetrics string)
PARTITIONED BY (
    `year` string,
    `month` string,
    `day` string)
ROW FORMAT SERDE 'org.openx.data.jsonserde.JsonSerDe'
WITH SERDEPROPERTIES (

 'paths'='catalogId,databaseName,dqRunId,evaluatedMetrics,evaluationCompletedOn,evaluationStart
STORED AS INPUTFORMAT 'org.apache.hadoop.mapred.TextInputFormat'
OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION 's3://glue-s3-dq-bucket-us-east-2-results/'
TBLPROPERTIES (
    'classification'='json',
    'compressionType'='none',
    'typeOfData'='file');
```

```
MSCK REPAIR TABLE <my_table_name>;
```

Une fois la table ci-dessus créée, vous pouvez exécuter des requêtes analytiques à l'aide d'Amazon Athena.

Déploiement de règles de qualité des données en utilisant AWS CloudFormation

Vous pouvez l'utiliser AWS CloudFormation pour créer des règles de qualité des données. Pour plus d'informations, consultez AWS CloudFormation AWS Glue.

Planification de règles de qualité des données

Vous pouvez planifier des règles de qualité des données à l'aide des méthodes suivantes :

- Planifiez les règles de qualité des données à partir du catalogue de données : aucun code Les utilisateurs ne peuvent utiliser cette option pour planifier facilement leurs analyses de qualité des données. AWS Glue Data Quality créera le calendrier sur Amazon EventBridge. Pour planifier des règles de qualité des données :
  - Accédez à l'ensemble de règles et cliquez sur Exécuter.
  - Dans la section Fréquence d'exécution, sélectionnez le calendrier souhaité et indiquez un nom de tâche. Ce nom de tâche est le nom de votre calendrier dans EventBridge.
- Utilisez Amazon EventBridge et AWS Step Functions pour orchestrer les évaluations et les recommandations relatives aux règles de qualité des données.

Le chiffrement des données est arrêté pour AWS Glue Data Quality

AWS Glue Data Qualityfournit un chiffrement par défaut pour protéger les données sensibles des clients au repos à l'aide de clés de chiffrement AWS détenues par nos soins.

AWS clés possédées

AWS Glue Data Quality utilise ces clés pour chiffrer automatiquement les actifs de qualité des données des clients. Vous ne pouvez pas consulter, gérer ou utiliser les clés AWS détenues, ni auditer leur utilisation. Toutefois, vous n'avez pas besoin de prendre de mesure ou de modifier les programmes pour protéger les clés qui chiffrent vos données. Pour plus d'informations, consultez la section sur les clés AWS détenues dans le Guide du AWS KMS développeur.

Le chiffrement des données au repos par défaut permet de réduire les frais opérationnels et la complexité liés à la protection des données sensibles. Dans le même temps, il vous permet de créer des applications sécurisées qui répondent aux exigences réglementaires et de conformité strictes en matière de chiffrement.

Bien que vous ne puissiez pas désactiver cette couche de chiffrement ou sélectionner un autre type de chiffrement, vous pouvez ajouter une deuxième couche de chiffrement aux clés de chiffrement AWS détenues existantes en choisissant une clé gérée par le client lorsque vous créez vos ressources de qualité des données.

Clés gérées par le client

Clés gérées par le client : AWS Glue Data Quality prend en charge l'utilisation d'une clé symétrique gérée par le client que vous créez, détenez et gérez. Cela ajoute une deuxième couche de chiffrement par rapport au chiffrement AWS propre existant. Étant donné que vous avez le contrôle total de cette couche de chiffrement, vous pouvez effectuer les tâches suivantes :

- Établissement et gestion des stratégies de clé
- Établissement et gestion des politiques IAM
- Activation et désactivation des stratégies de clé
- Rotation des matériaux de chiffrement de clé
- Ajout de balises
- Création d'alias de clé
- Planification des clés pour la suppression

Pour plus d'informations, consultez la section Clés gérées par le client dans le Guide du AWS KMS développeur.

Le tableau suivant résume la manière dont AWS Glue Data Quality chiffre les différents actifs de Data Quality.

Type de données	AWS chiffrement par clé détenue	Chiffrement par clé gérée par le client
Ensemble de règles de qualité des données Chaîne d'ensemble de règles DQDL référencée par l'ensemble de règles DQ persistant. Pour le moment, ces ensembles de règles	Activé	Activé

Type de données	AWS chiffrement par clé détenue	Chiffrement par clé gérée par le client
persistants ne sont utilisés que dans l’expérience catalogue de données AWS Glue.		
Rule/Analyzer Résultats relatifs à la qualité des données Artefacts de résultat qui contiennent le pass/fail statut de chaque règle d'un ensemble de règles ainsi que les métriques collectées par les règles et les analyseurs.	Activé	Activé
Observations Des observations sont générées lorsqu’une anomalie est détectée dans les données. Elles contienne nt des informations sur les limites supérieure et inférieur e attendues et une règle suggérée basée sur ces limites. Si elles sont générées, elles sont affichées avec les résultats de qualité des données.	Activé	Activé

Type de données	AWS chiffrement par clé détenue	Chiffrement par clé gérée par le client
Statistiques Contient des informations sur les métriques collectées après avoir évalué les données selon un ensemble de règles, telles que la valeur de la métrique (par exemple RowCount, Exhaustivité), les noms des colonnes et d'autres métadonnées.	Activé	Activé
Modèles statistiques de détection des anomalies Les modèles statistiques contiennent les séries temporelles des limites supérieure et inférieure pour une métrique donnée, générées sur la base d'évaluations précédentes des données client.	Activé	Activé

> **Note**
>
> AWS Data Quality active automatiquement le chiffrement au repos à l'aide de clés AWS détenues pour protéger gratuitement les données personnelles identifiables. Toutefois, AWS KMS des frais s'appliquent pour l'utilisation d'une clé gérée par le client. Pour plus d'informations sur la tarification, consultez Tarification d'AWS KMS.
> Pour plus d'informations sur AWS KMS, voir AWS KMS.

Création d'une clé gérée par le client

Vous pouvez créer une clé symétrique gérée par le client en utilisant le AWS Management Console, ou le AWS KMS APIs.

Pour créer une clé symétrique gérée par le client :

- Suivez les étapes de création de AWS KMS clés de chiffrement symétriques décrites dans le manuel du AWS Key Management Service développeur.

Stratégie de clé

Les stratégies de clé contrôlent l'accès à votre clé gérée par le client. Chaque clé gérée par le client doit avoir exactement une stratégie de clé, qui contient des instructions qui déterminent les personnes pouvant utiliser la clé et comment elles peuvent l'utiliser. Lorsque vous créez votre clé gérée par le client, vous pouvez spécifier une stratégie de clé. Pour plus d'informations, consultez la section Politiques relatives aux AWS KMS clés dans le Guide du AWS Key Management Service développeur.

Pour utiliser votre clé gérée par le client avec les ressources de qualité des données, les opérations d'API suivantes doivent être autorisées dans la stratégie de clé :

- `kms:Decrypt`— Déchiffre le texte chiffré par une clé en utilisant AWS KMS `GenerateDataKeyWithoutPlaintext`
- `kms:DescribeKey` : fournit les détails des clés gérées par le client pour permettre à Amazon Location de valider la clé.
- `kms:GenerateDataKeyWithoutPlaintext`— Renvoie une clé de données symétrique unique à utiliser en dehors de AWS KMS. Cette opération renvoie une clé de données chiffrée sous une clé KMS de chiffrement symétrique que vous spécifiez. Les octets de la clé sont aléatoires. Ils ne sont pas liés à l'appelant ni à la clé KMS. Sert à réduire le nombre d'appels KMS que le client doit passer.
- `kms:ReEncrypt*` : déchiffre le texte chiffré, puis le rechiffre entièrement au sein d' AWS KMS. Vous pouvez utiliser cette opération pour modifier la clé KMS sous laquelle les données sont chiffrées, par exemple lorsque vous faites pivoter manuellement une clé KMS ou que vous modifiez la clé KMS qui protège un texte chiffré. Vous pouvez également l'utiliser pour rechiffrer du texte chiffré sous la même clé KMS, par exemple pour modifier le contexte de chiffrement d'un texte chiffré.

Voici des exemples de déclarations de stratégie que vous pouvez ajouter pour Amazon Location :

```
"Statement" : [
    {
        "Sid" : "Allow access to principals authorized to use AWS Glue Data Quality",
        "Effect" : "Allow",
        "Principal" : {
            "AWS": "arn:aws:iam::<account_id>:role/ExampleRole"
        },
        "Action" : [
            "kms:Decrypt",
            "kms:DescribeKey",
            "kms:GenerateDataKeyWithoutPlaintext",
            "kms:ReEncrypt*"
        ],
        "Resource" : "*",
        "Condition" : {
            "StringEquals" : {
                "kms:ViaService" : "glue.amazonaws.com",
                "kms:CallerAccount" : "111122223333"
            }
        },
    {
        "Sid": "Allow access for key administrators",
        "Effect": "Allow",
        "Principal": {
            "AWS": "arn:aws:iam::111122223333:root"
          },
        "Action" : [
            "kms:*"
         ],
        "Resource": "arn:aws:kms:region:111122223333:key/key_ID"
    },
    {
        "Sid" : "Allow read-only access to key metadata to the account",
        "Effect" : "Allow",
        "Principal" : {
            "AWS" : "arn:aws:iam::111122223333:root"
        },
        "Action" : [
            "kms:Describe*",
            "kms:Get*",
            "kms:List*",
        ],
```

```
        "Resource" : "*"
    }
]
```

Remarques concernant l'utilisation des clés KMS dans AWS Glue Data Quality

AWS Glue Data Quality ne prend pas en charge les transitions clés. Cela signifie que si vous chiffrez vos actifs de qualité des données avec la clé A et que vous décidez de passer à la clé B, nous ne chiffrerons pas à nouveau les données chiffrées avec la clé A pour utiliser la clé B. Vous pouvez toujours passer à la clé B, mais vous devrez conserver l'accès à la clé A pour accéder aux données précédemment chiffrées avec la clé A.

Pour plus d'informations sur la spécification des autorisations dans une politique, consultez la section Autorisations pour les AWS services dans les politiques clés du Guide du AWS Key Management Service développeur.

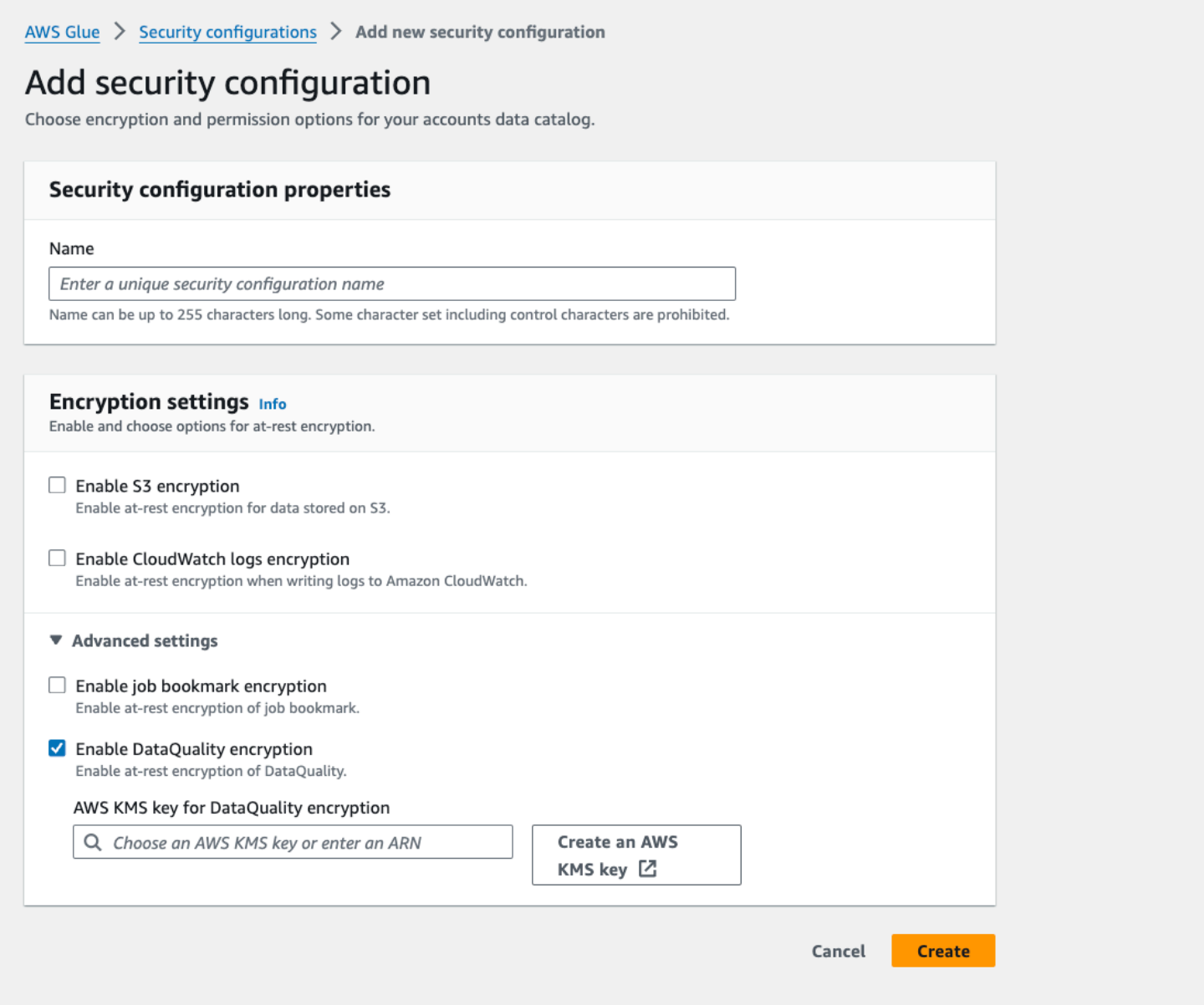
Pour plus d'informations sur la résolution des problèmes d'accès par clé, consultez la section Résolution des problèmes d'accès par clé dans le Guide du AWS Key Management Service développeur.

Création d'une configuration de sécurité

Dans AWS Glue, la ressource Security Configurations contient les propriétés nécessaires pour écrire des données chiffrées.

Pour chiffrer les actifs liés à la qualité de vos données :

1. Dans Paramètres de chiffrement, sous Paramètres avancés, choisissez Activer le chiffrement de la qualité des données.
2. Sélectionnez votre clé KMS ou choisissez Créer une AWS KMS clé



AWS Contexte de chiffrement de Glue Data Quality

Un contexte de chiffrement est un ensemble facultatif de paires clé-valeur qui contient des informations contextuelles supplémentaires sur les données.

AWS KMS utilise le contexte de chiffrement comme données authentifiées supplémentaires pour prendre en charge le chiffrement authentifié. Lorsque vous incluez un contexte de chiffrement dans une demande de chiffrement de données, AWS KMS lie le contexte de chiffrement aux données chiffrées. Pour déchiffrer les données, vous devez inclure le même contexte de chiffrement dans la demande.

AWS Exemple de contexte de chiffrement de Glue Data Quality

```
"encryptionContext": {
    "kms-arn": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
    "branch-key-id": "111122223333+arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
    "hierarchy-version": "1",
    "aws-crypto-ec:aws:glue:securityConfiguration": "111122223333:customer-security-
configuration-name",
    "create-time": "2024-06-07T13:47:23:000861Z",
    "tablename": "AwsGlueMlEncryptionKeyStore",
    "type": "beacon:ACTIVE"
}
```

Utilisation du contexte de chiffrement à des fins de surveillance

Lorsque vous utilisez une clé symétrique gérée par le client pour chiffrer les données de votre suivi ou de votre collection de périmètres virtuels, vous pouvez également utiliser le contexte de chiffrement dans les enregistrements et les journaux d'audit pour identifier la manière dont la clé gérée par le client est utilisée. Le contexte de chiffrement apparaît également dans les journaux générés par AWS CloudTrail ou Amazon CloudWatch Logs.

Surveillance de vos clés de chiffrement pour AWS Glue Data Quality

Lorsque vous utilisez une clé gérée par le AWS KMS client avec vos ressources AWS Glue Data Quality, vous pouvez utiliser AWS CloudTrail ou Amazon CloudWatch Logs suivre les demandes auxquelles AWS Glue Data Quality envoie AWS KMS.

Les exemples suivants sont des AWS CloudTrail événements pour `GenerateDataKeyWithoutPlainText` et `Decrypt` pour surveiller les opérations KMS appelées par AWS Glue Data Quality afin d'accéder aux données chiffrées par votre clé gérée par le client.

Decrypt

```
{
    "eventVersion": "1.09",
    "userIdentity": {
        "type": "AssumedRole",
        "arn": "arn:aws:sts::111122223333:role/CustomerRole",
        "accountId": "111122223333",
```

```
        "invokedBy": "glue.amazonaws.com"
    },
    "eventTime": "2024-07-02T20:03:10Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Decrypt",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "glue.amazonaws.com",
    "userAgent": "glue.amazonaws.com",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
        "encryptionAlgorithm": "SYMMETRIC_DEFAULT",
        "encryptionContext": {
            "kms-arn": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
            "branch-key-id": "111122223333+arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
            "hierarchy-version": "1",
            "aws-crypto-ec:aws:glue:securityConfiguration": "111122223333:customer-
security-configuration-name",
            "create-time": "2024-06-07T13:47:23:000861Z",
            "tablename": "AwsGlueMlEncryptionKeyStore",
            "type": "branch:ACTIVE",
            "version": "branch:version:ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE"
        }
    },
    "responseElements": null,
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

GenerateDataKeyWithoutPlaintext


```
{
    "eventVersion": "1.09",
    "userIdentity": {
        "type": "AssumedRole",
        "arn": "arn:aws:sts::111122223333:role/CustomerRole",
        "accountId": "111122223333",
        "invokedBy": "glue.amazonaws.com"
    },
    "eventTime": "2024-07-02T20:03:10Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKeyWithoutPlaintext",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "glue.amazonaws.com",
    "userAgent": "glue.amazonaws.com",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
        "encryptionAlgorithm": "SYMMETRIC_DEFAULT",
        "encryptionContext": {
            "kms-arn": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
            "branch-key-id": "111122223333+arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
            "hierarchy-version": "1",
            "aws-crypto-ec:aws:glue:securityConfiguration": "111122223333:customer-
security-configuration-name",
            "create-time": "2024-06-07T13:47:23:000861Z",
            "tablename": "AwsGlueMlEncryptionKeyStore",
            "type": "branch:version:ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE"
        }
    },
    "responseElements": null,
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
```

```
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

ReEncyrpt

```
{
    "eventVersion": "1.09",
    "userIdentity": {
        "type": "AssumedRole",
        "arn": "arn:aws:sts::111122223333:role/CustomerRole",
        "accountId": "111122223333",
        "invokedBy": "glue.amazonaws.com"
    },
    "eventTime": "2024-07-17T21:34:41Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "ReEncrypt",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "glue.amazonaws.com",
    "userAgent": "glue.amazonaws.com",
    "requestParameters": {
        "destinationEncryptionContext": {
             "kms-arn": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
            "branch-key-id": "111122223333+arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
            "hierarchy-version": "1",
            "aws-crypto-ec:aws:glue:securityConfiguration": "111122223333:customer-
security-configuration-name",
            "create-time": "2024-06-07T13:47:23:000861Z",
            "tablename": "AwsGlueMlEncryptionKeyStore",
            "type": "branch:ACTIVE"
            "version": "branch:version:12345678-SAMPLE"
        },
        "destinationKeyId": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
        "sourceAAD": "1234567890-SAMPLE+Z+lqoYOHj7VtWxJLrvh+biUFbliYDAQkobM=",
        "sourceKeyId": "arn:aws:kms:ap-southeast-2:585824196334:key/17ca05ca-a8c1-40d7-
b7fd-30abb569a53a",
        "destinationEncryptionAlgorithm": "SYMMETRIC_DEFAULT",
```

```
        "sourceEncryptionContext": {
            "kms-arn": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
            "branch-key-id": "111122223333+arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
            "hierarchy-version": "1",
            "aws-crypto-ec:aws:glue:securityConfiguration": "111122223333:customer-
security-configuration-name",
            "create-time": "2024-06-07T13:47:23:000861Z",
            "tablename": "AwsGlueMlEncryptionKeyStore",
            "type": "branch:version:ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE"
        },
        "destinationAAD": "1234567890-SAMPLE",
        "sourceEncryptionAlgorithm": "SYMMETRIC_DEFAULT"
    },
    "responseElements": null,
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        },
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```


En savoir plus

Les ressources suivantes fournissent plus d'informations sur le chiffrement des données au repos.

- Pour plus d'informations sur les concepts AWS Key Management Service de base, consultez le guide du AWS Key Management Service développeur.
- Pour plus d'informations sur les meilleures pratiques en matière de sécurité, AWS Key Management Service consultez le guide du AWS Key Management Service développeur.

Résolution des erreurs liées à la qualité des données de AWS Glue

Si vous rencontrez des erreurs dans AWS Glue Data Quality, utilisez les solutions suivantes pour trouver la source des problèmes et les résoudre.

Table des matières

- Erreur : module AWS Glue Data Quality manquant
- Erreur : autorisations insuffisantes pour AWS Lake Formation
- Erreur : les ensembles de règles ne sont pas nommés de manière unique
- Erreur : tables contenant des caractères spéciaux
- Erreur : erreur de débordement avec un ensemble de règles volumineux
- Erreur : échec du statut général de la règle
- AnalysisException: Impossible de vérifier l'existence de la base de données par défaut
- Message d'erreur : la clé de distribution fournie ne convient pas aux cadres de données fournies.
- Exception dans la classe utilisateur : java.lang. RuntimeException : Impossible de récupérer les données. Vérifiez les connexions CloudWatch pour obtenir plus de détails
- ERREUR DE LANCEMENT : erreur lors du téléchargement depuis S3 pour le compartiment
- InvalidInputException (statut : 400) : DataQuality les règles ne peuvent pas être analysées
- Erreur : Eventbridge ne déclenche pas les tâches Glue DQ selon le calendrier que j'ai configuré.
- Erreurs CustomSQL
- Règles dynamiques
- Exception dans la classe utilisateur : org.apache.spark.sql. AnalysisException: org.apache.hadoop.hive.ql.metadata. HiveException
- UNCLASSIFIED_ERROR ; IllegalArgumentException : Erreur d'analyse : aucune règle ni analyseur fournis., aucune alternative viable en entrée

Erreur : module AWS Glue Data Quality manquant

Message d'erreur : aucun module nommé « awsgluedq ».

Résolution : Cette erreur se produit lorsque vous exécutez AWS Glue Data Quality dans une version non prise en charge. AWS Glue Data Quality n'est prise en charge que dans les versions 3.0 et ultérieures de Glue.

Erreur : autorisations insuffisantes pour AWS Lake Formation

Message d'erreur : exception dans la classe utilisateur `com.amazonaws.services.glue.model.AccessDeniedException` : autorisations de Lake Formation insuffisantes sur impact_sdg_involvement (Service : ; Code d'état : 400 ; Code d'erreur : AWS Glue ; ID de demande : AccessDeniedException 465ae693-b7ba-4df0-a4e4-6b17xxxxxxxx ; Proxy : nul).

Résolution : Vous devez fournir des autorisations suffisantes dans AWS Lake Formation.

Erreur : les ensembles de règles ne sont pas nommés de manière unique

Message d'erreur : Exception dans la classe utilisateur :... services.glue.model. AlreadyExistsException: Un autre ensemble de règles portant le même nom existe déjà.

Résolution : les ensembles de règles sont globaux et doivent être uniques.

Erreur : tables contenant des caractères spéciaux

Message d'erreur : Exception dans la classe utilisateur : org.apache.spark.sql. AnalysisException: impossible de résoudre les colonnes d'entrée « C » données : [primary.data_end_time, primary.data_start_time, primary.end_time, primary.last_updated, primary.message, primary.process_date, primary.rowhash, primary.run_by, primary.run_id, primary.start_time, primary.status] ; ligne 1 pos 44 ;.

Résolution : Il existe actuellement une limite selon laquelle AWS Glue Data Quality ne peut pas être exécuté sur des tables contenant des caractères spéciaux tels que «. ».

Erreur : erreur de débordement avec un ensemble de règles volumineux

Message d'erreur : Exception dans la classe utilisateur : java.lang. StackOverflowError.

Résolution : si vous disposez d'un ensemble de règles de plus de 2 000 règles, vous pouvez rencontrer ce problème. Divisez vos règles en plusieurs ensembles de règles.

Erreur : échec du statut général de la règle

Condition d'erreur : mon ensemble de règles est réussi, mais le statut général de mes règles est un échec.

Résolution : Cette erreur s'est probablement produite parce que vous avez choisi l'option de publier les statistiques sur Amazon CloudWatch lors de la publication. Si votre ensemble de données se trouve dans un VPC, celui-ci n'autorise peut-être pas AWS Glue à publier des métriques sur Amazon. CloudWatch Dans ce cas, vous devez configurer un point de terminaison pour que votre VPC puisse accéder à Amazon. CloudWatch

AnalysisException: Impossible de vérifier l'existence de la base de données par défaut

Condition d'erreur AnalysisException : Impossible de vérifier l'existence de la base de données par défaut : com.amazonaws.services.glue.model. AccessDeniedException: Autorisations de Lake Formation insuffisantes par défaut (Service : AWS Glue ; Code d'état : 400 ; Code d'erreur : ; ID de demande : AccessDeniedException XXXXXXXX-XXXX-XXXX-XXXX -XXXXXXXXXXXX ; Proxy : nul)

Résolution : dans l'intégration du catalogue de la tâcheAWS Glue, AWS Glue essaie toujours de vérifier si la base de données par défaut existe ou non en utilisant AWS Glue `GetDatabase API`. Lorsque l'autorisation `DESCRIBE` Lake Formation n'est pas accordée ou que l'autorisation `GetDatabase IAM` est accordée, la tâche échoue lors de la vérification de l'existence de la base de données par défaut.

Pour résoudre le problème :

1. Ajoutez l'autorisation `DESCRIBE` dans Lake Formation pour la base de données par défaut.
2. Configurez le rôle IAM associé à la tâche AWS Glue en tant que créateur de base de données dans Lake Formation. Cela crée automatiquement une base de données par défaut et accorde les autorisations Lake Formation requises pour le rôle.
3. Désactivez l'option `--enable-data-catalog`. (Elle est affichée en tant que Utiliser Data Catalog en tant que métastore Hive dans AWS Glue Studio).

   Si vous n'avez pas besoin de l'intégration Data Catalog de Spark SQL dans la tâche, vous pouvez la désactiver.

Message d'erreur : la clé de distribution fournie ne convient pas aux cadres de données fournies.

Condition d'erreur : la clé de distribution fournie ne convient pas aux cadres de données fournies.

Résolution : vous utilisez le DataSetMatchtype de règle et les clés de jointure sont dupliquées. Vos clés de jointure doivent être uniques et ne doivent pas être nulles. Dans les cas où vous ne pouvez pas avoir de clés de jointure uniques, envisagez d'utiliser d'autres types de règles, par exemple AggregateMatchpour établir une correspondance sur des données récapitulatives.

Exception dans la classe utilisateur : java.lang. RuntimeException : Impossible de récupérer les données. Vérifiez les connexions CloudWatch pour obtenir plus de détails

Condition d'erreur : exception dans la classe utilisateur : java.lang. RuntimeException : Impossible de récupérer les données. Vérifiez les connexions CloudWatch pour obtenir plus de détails.

Résolution : Cela se produit lorsque vous créez des règles DQ sur un tableau basé sur Amazon S3 qui est comparé à Amazon RDS ou. Amazon Redshift Dans ces cas, AWS Glue ne peut pas charger la connexion. Essayez plutôt de configurer une règle DQ sur le jeu de données Amazon Redshift ou Amazon RDS. Il s'agit d'un bogue connu.

ERREUR DE LANCEMENT : erreur lors du téléchargement depuis S3 pour le compartiment

Condition d'erreur : ERREUR DE LANCEMENT : erreur lors du téléchargement depuis S3 pour le compartiment : `aws-glue-ml-data-quality-assets-us-east-1, key: jars/aws-glue-ml-data-quality-etl.jar.Access Denied (Service: Amazon S3; Status Code: 403; Please refer logs for details)` .

Résolution : Les autorisations associées au rôle transmises à AWS Glue Data Quality doivent permettre la lecture depuis l'emplacement Amazon S3 précédent. Cette politique IAM doit être associée au rôle :

```
{
  "Sid": "allowS3",
  "Effect": "Allow",
  "Action": "s3:GetObject",
```

```
  "Resource": "arn:aws:s3:::aws-glue-ml-data-quality-assets-<region>/*"
}
```

Reportez-vous à Data Quality authorization pour obtenir la liste des autorisations détaillées. Ces bibliothèques sont nécessaires pour évaluer la qualité des données de vos jeux de données.

InvalidInputException (statut : 400) : DataQuality les règles ne peuvent pas être analysées

Condition d'erreur : InvalidInputException (statut : 400) : DataQuality les règles ne peuvent pas être analysées.

Résolution : il existe de nombreuses possibilités pour cette erreur. Il est possible que vos règles comportent des guillemets simples. Assurez-vous qu'ils sont entre guillemets. Par exemple :

```
Rules = [
ColumnValues "tipo_vinculo" in ["CODO", "DOCO", "COCO", "DODO"] AND "categoria" = 'ES'
    AND "cod_bandera" = 'CEP'
```

Remplacer par :

```
Rules = [
(ColumnValues "tipovinculo" in [ "CODO", "DOCO", "COCO", "DODO"]) AND (ColumnValues
 "categoria" = "ES")
    AND (ColumnValues "codbandera" = "CEP")
]
```

Erreur : Eventbridge ne déclenche pas les tâches Glue DQ selon le calendrier que j'ai configuré.

Condition d'erreur : erreur : Eventbridge ne déclenche pas les tâches AWS Glue Data Quality selon le calendrier que j'ai configuré.

Résolution : le rôle déclenchant la tâche ne dispose peut-être pas des autorisations appropriées. Assurez-vous que le rôle que vous utilisez pour lancer les tâches dispose des autorisations mentionnées dans la rubrique Configuration IAM nécessaire à la planification des évaluations.

Erreurs CustomSQL

Condition d'erreur : `The output from CustomSQL must contain at least one column that matches the input dataset for AWS Glue Data Quality to provide row level results. The SQL query is a valid query but no columns from the SQL result are present in the Input Dataset. Ensure that matching columns are returned from the SQL`.

Résolution : la requête SQL est valide, mais assurez-vous que vous ne sélectionnez que les colonnes de la table principale. La sélection de fonctions d'agrégation telles que somme, nombre sur les colonnes à partir de la table principale peut entraîner cette erreur.

Condition d'erreur : `There was a problem when executing your SQL statement: cannot resolve "Col"`.

Résolution : cette colonne n'est pas présente dans la table principale.

Condition d'erreur : `The columns that are returned from the SQL statement should only belong to the primary table. "In this case, some columns ( Col ) belong to reference table"`.

Résolution : dans les requêtes SQL, lorsque vous attachez la table principale à d'autres tables de référence, assurez-vous que votre instruction SELECT ne contient que des noms de colonnes de votre table principale afin de générer des résultats au niveau de la ligne pour la table principale.

Règles dynamiques

Condition d'erreur: `Dynamic rules require job context, and cannot be evaluated in interactive session or data preview.`.

Cause : ce message d'erreur peut apparaître dans les résultats de l'aperçu des données, ou dans d'autres sessions interactives, lorsque votre jeu de règles contient des règles DQ dynamiques. Les règles dynamiques font référence à des métriques historiques associées à un nom de tâche et à un contexte d'évaluation particuliers, de sorte qu'elles ne peuvent pas être évaluées lors de sessions interactives.

Résolution : l'exécution de votre tâche AWS Glue produira des métriques historiques, qui pourront être référencées lors d'exécutions ultérieures de la même tâche.

Condition d'erreur :

- `[RuleType] rule only supports simple atomic operands in thresholds.`.
- `Function last not yet implemented for [RuleType] rule.`

Résolution : les règles dynamiques sont généralement prises en charge pour tous les types de règles DQDL dans les expressions numériques (voir Référence DQDL). Cependant, certaines règles qui produisent plusieurs métriques ColumnValues et ColumnLength ne sont pas encore prises en charge.

Condition d'erreur : `Binary expression operands must resolve to a single number.`.

Cause : les règles dynamiques prennent en charge les expressions binaires, telles que `RowCount > avg(last(5)) * 0.9`. Ici, l'expression binaire est `avg(last(5)) * 0.9`. Cette règle est valide parce que les opérandes `avg(last(5))` et `0.9` se résolvent en un seul nombre. Un exemple incorrect est `RowCount > last(5) * 0.9` parce que `last(5)` produira une liste qui ne pourra pas être comparée de manière significative au nombre actuel de lignes.

Résolution : utilisez des fonctions d'agrégation pour réduire un opérande à valeurs multiples en un seul nombre.

Condition d'erreur :

- `Rule threshold results in list, and a single value is expected. Use aggregation functions to produce a single value. Valid example: sum(last(10)), avg(last(10)).`
- `Rule threshold results in empty list, and a single value is expected.`

Cause : les règles dynamiques peuvent être utilisées pour comparer certaines fonctionnalités de votre jeu de données avec ses valeurs historiques. La fonction « last » permet de récupérer plusieurs valeurs historiques, à condition de fournir un argument entier positif. Par exemple, `last(5)` récupérera les cinq dernières valeurs observées lors des exécutions de tâches pour votre règle.

Résolution : une fonction d'agrégation doit être utilisée pour réduire ces valeurs en un seul nombre afin de réaliser une comparaison significative avec la valeur observée dans l'exécution de tâche actuelle.

Exemples valides :

- `RowCount >= avg(last(5))`
- `RowCount > last(1)`
- `RowCount < last()`

Exemple non valide : `RowCount > last(5)`.

Condition d'erreur :

- `Function index used in threshold requires positive integer argument.`
- `Index argument must be an integer. Valid syntax example: RowCount > index(last(10, 2)), which means RowCount must be greater than third most recent execution from last 10 job runs.`

Résolution : lors de la création de règles dynamiques, vous pouvez utiliser la fonction d'agrégation `index` pour sélectionner une valeur historique à partir d'une liste. Par exemple, `RowCount > index(last(5)`, 1) vérifiera si le nombre de lignes observé dans l'exécution de tâche actuelle est strictement supérieur au nombre de lignes le plus récent, après le dernier, observé pour votre tâche. `index` est indexé à partir de zéro.

Condition d'erreur : `IllegalArgumentException: Parsing Error: Rule Type: DetectAnomalies is not valid`.

Résolution : la détection d'anomalies est uniquement disponible dans AWS Glue 4.0.

Condition d'erreur : `IllegalArgumentException: Parsing Error: Unexpected condition for rule of type ... no viable alternative at input ...`.

Remarque : `...` est dynamique. Exemple: `IllegalArgumentException: Parsing Error: Unexpected condition for rule of type RowCount with number return type, line 4:19 no viable alternative at input '>last'`.

Résolution : la détection d'anomalies est uniquement disponible dans AWS Glue 4.0.

Exception dans la classe utilisateur : org.apache.spark.sql. AnalysisException: org.apache.hadoop.hive.ql.metadata. HiveException

Condition d'erreur: `Exception in User Class: org.apache.spark.sql.AnalysisException:`

`org.apache.hadoop.hive.ql.metadata.HiveException: Unable to fetch table mailpiece_submitted. StorageDescriptor#InputFormat cannot be null for table: mailpiece_submitted (Service: null; Status Code: 0; Error Code: null; Request ID: null; Proxy: null)`

Cause : vous utilisez Apache Iceberg dans AWS Glue Data Catalog et l'attribut Input Format est vide dans AWS Glue Data Catalog.

Résolution : ce problème se produit lorsque vous utilisez le type de règle CustomSQL dans votre règle DQ. Une façon de résoudre ce problème consiste à utiliser « principal » ou à ajouter un nom de catalogue `glue_catalog.` à `<database>.<table> in Custom ruletype`.

UNCLASSIFIED_ERROR ; IllegalArgumentException : Erreur d'analyse : aucune règle ni analyseur fournis., aucune alternative viable en entrée

Condition d’erreur: `UNCLASSIFIED_ERROR; IllegalArgumentException: Parsing Error: No rules or analyzers provided., no viable alternative at input`

Résolution : le DQDL n’est pas analysable. Cela peut se produire dans quelques instances. Si vous utilisez des règles composites, assurez-vous qu’elles comportent des parenthèses droites.

```
(RowCount >= avg(last(10)) * 0.6) and (RowCount <= avg(last(10)) * 1.4) instead of
      RowCount >= avg(last(10)) * 0.6 and RowCount <= avg(last(10)) * 1.4
```

Intégration des données Amazon Q dans AWS Glue

L'intégration de données dans Amazon Q AWS Glue est une nouvelle fonctionnalité d'IA générative AWS Glue qui permet aux ingénieurs de données et aux développeurs ETL de créer des tâches d'intégration de données en langage naturel. Les ingénieurs et les développeurs peuvent demander à Amazon Q de créer des jobs, de résoudre des problèmes et de répondre à des questions sur l'intégration des données AWS Glue et sur l'intégration des données.

Qu'est-ce qu'Amazon Q ?

> **Note**
>
> Propulsé par Amazon Bedrock : AWS implémente la détection automatique des abus. L'intégration de données Amazon Q étant basée sur Amazon Bedrock, les utilisateurs peuvent pleinement tirer parti des contrôles mis en œuvre dans Amazon Bedrock pour renforcer la sûreté, la sécurité et l'utilisation responsable de l'intelligence artificielle (IA).

Amazon Q est un assistant conversationnel basé sur l'intelligence artificielle générative (IA) qui peut vous aider à comprendre, créer, étendre et exploiter AWS des applications. Le modèle sur lequel repose Amazon Q a été enrichi d'un AWS contenu de haute qualité afin de vous fournir des réponses plus complètes, exploitables et référencées afin d'accélérer votre développement. AWS Pour plus d'informations, consultez la rubrique Qu'est-ce qu'Amazon Q ?

Qu'est-ce que l'intégration des données Amazon Q dans AWS Glue ?

L'intégration des données Amazon Q AWS Glue inclut les fonctionnalités suivantes :

- Chat : Amazon Q Data Integration in AWS Glue peut répondre à des questions en langage naturel en anglais sur AWS Glue des domaines tels que les connecteurs AWS Glue source et de destination, les tâches AWS Glue ETL, le catalogue de données, les robots d'exploration et AWS Lake Formation d'autres documents relatifs aux fonctionnalités, ainsi que les meilleures pratiques. L'intégration des données Amazon Q AWS Glue répond par des step-by-step instructions et inclut des références à ses sources d'informations.

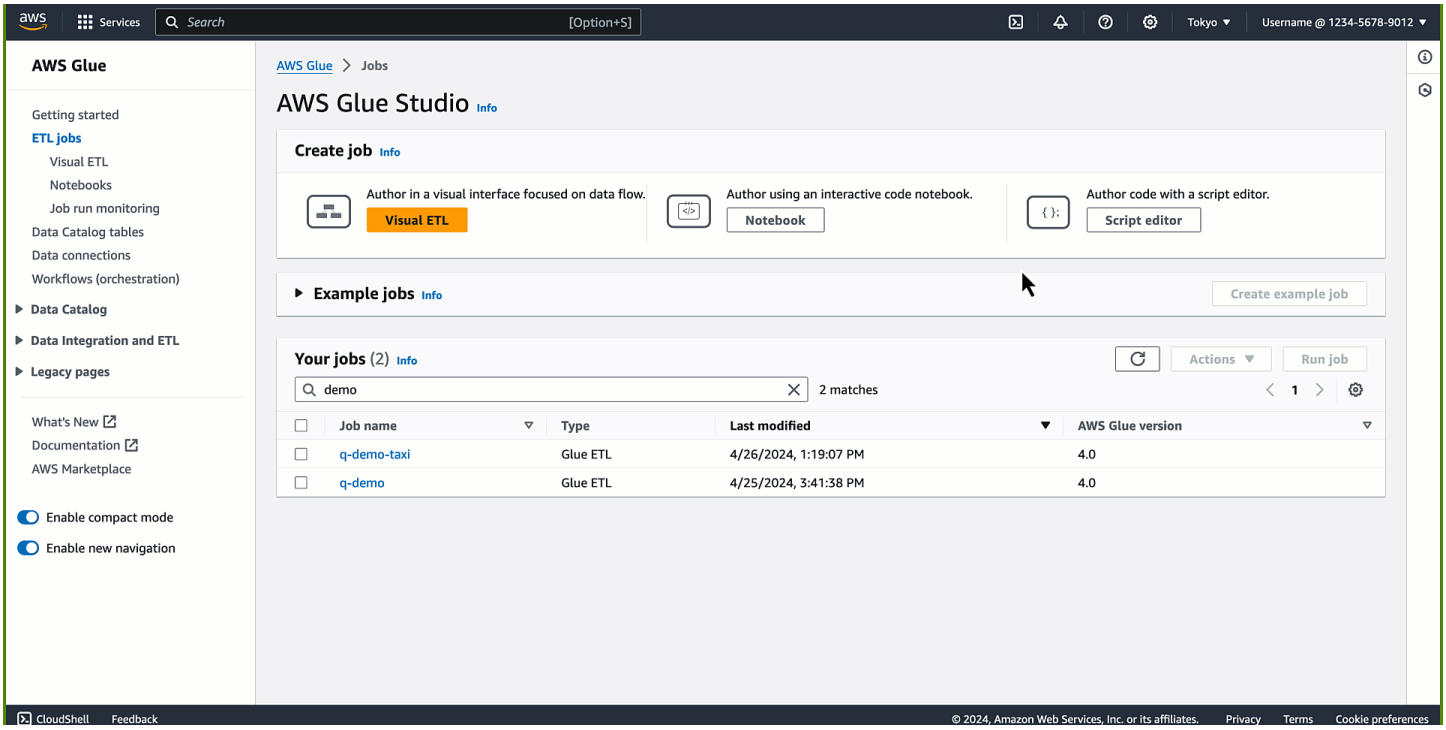
- Génération de code d'intégration de données — L'intégration de données Amazon Q AWS Glue permet de répondre à des questions sur les scripts AWS Glue ETL et de générer du nouveau code à partir d'une question en langage naturel en anglais.
- Résolution des problèmes : l'intégration des données dans Amazon Q AWS Glue est spécialement conçue pour vous aider à comprendre les erreurs dans les AWS Glue tâches et fournit des step-by-step instructions pour déterminer la cause de vos problèmes et les résoudre.

> **Note**
>
> L'intégration des données Amazon Q dans Amazon Q AWS Glue n'utilise pas le contexte de votre conversation pour éclairer les réponses futures pendant toute la durée de votre conversation. Chaque conversation avec l'intégration de données Amazon Q AWS Glue est indépendante de vos conversations précédentes ou futures.

Vous utilisez l'intégration des données Amazon Q dans AWS Glue ?

Dans le panneau Amazon Q, vous pouvez demander à Amazon Q de générer du code pour un script AWS Glue ETL, ou de répondre à une question sur les AWS Glue fonctionnalités ou de résoudre une erreur. La réponse est un script ETL contenant PySpark des step-by-step instructions pour personnaliser le script, le réviser et l'exécuter. Pour les questions, la réponse est générée à partir de la base de connaissances sur l'intégration des données, avec un résumé et l'URL de la source pour les références.

Par exemple, vous pouvez demander à Amazon Q de « Veuillez fournir un script Glue qui lit depuis Snowflake, renomme les champs et écrit sur Redshift » et en réponse, l'intégration des données Amazon Q AWS Glue renverra un script de AWS Glue tâche capable d'effectuer l'action demandée. Vous pouvez vérifier le code généré pour vous assurer qu'il répond à l'intention demandée. Si vous êtes satisfait, vous pouvez le déployer en tant que AWS Glue tâche en production. Vous pouvez résoudre les problèmes liés aux tâches en demandant à l'intégration d'expliquer les erreurs et les échecs et de proposer des solutions. Amazon Q peut répondre aux questions concernant les AWS Glue meilleures pratiques en matière d'intégration des données.



Voici des exemples de questions qui montrent comment l'intégration des données Amazon Q dans Amazon Q AWS Glue peut vous aider à tirer parti de ce qui suit AWS Glue :

AWS Glue Génération de code ETL :

- Écrivez un AWS Glue script qui lit le JSON depuis S3, transforme les champs à l'aide du mappage d'application et écrit sur Amazon Redshift
- Comment écrire un AWS Glue script pour lire depuis DynamoDB, appliquer la transformation et écrire DropNullFields dans S3 en tant que Parquet ?
- Donnez-moi un AWS Glue script qui lit depuis MySQL, supprime certains champs en fonction de ma logique métier et écrit dans Snowflake
- Rédigez une AWS Glue tâche à lire depuis DynamoDB et à écrire dans S3 au format JSON
- Aidez-moi à développer un AWS Glue script pour AWS Glue Data Catalog to S3
- Rédigez une AWS Glue tâche pour lire le JSON depuis S3, supprimer les valeurs nulles et écrire dans Redshift

AWS Glue explications sur les fonctionnalités :

- Comment utiliser la qualité AWS Glue des données ?
- Comment utiliser les signets AWS Glue d'offres d'emploi ?

- Comment activer la mise à l' AWS Glue échelle automatique ?
- Quelle est la différence entre les cadres AWS Glue dynamiques et les blocs de données Spark ?
- Quels sont les différents types de connexions pris en charge AWS Glue ?

AWS Glue résolution des problèmes :

- Comment résoudre les erreurs de mémoire insuffisante (OOM) sur les tâches ? AWS Glue
- Quels sont les messages d'erreur susceptibles de s'afficher lors de la configuration de la qualité AWS Glue des données et comment pouvez-vous les corriger ?
- Comment corriger une AWS Glue tâche avec le message d'erreur « Accès refusé à Amazon S3 » ?
- Comment résoudre les problèmes liés au transfert des données sur AWS Glue les tâches ?

Bonnes pratiques en matière d’interaction avec l’intégration des données Amazon Q

Les bonnes pratiques pour interagir avec l’intégration des données Amazon Q sont les suivantes :

- Lorsque vous interagissez avec l’intégration des données Amazon Q, posez des questions spécifiques, itérez lorsque vous avez des demandes complexes et vérifiez le degré d’exactitude des réponses.
- Lorsque vous fournissez des invites d’intégration des données en langage naturel, soyez aussi spécifique que possible pour aider l’assistant de codage à comprendre exactement ce dont vous avez besoin. Au lieu de demander « extraire les données de S3 », fournissez plus de détails, tels que « écrivez un AWS Glue script qui extrait les fichiers JSON de S3 ».
- Passez en revue le script généré avant de l’exécuter, afin de garantir son exactitude. Si le script généré comporte des erreurs ou ne correspond pas à votre intention, fournissez des instructions à l’assistant sur la façon de le corriger.
- La technologie d’IA générative est nouvelle et les réponses peuvent comporter des erreurs, parfois appelées hallucinations. Testez et passez en revue tout le code pour détecter des erreurs et des vulnérabilités avant de l’utiliser dans votre environnement ou votre charge de travail.

Intégration des données Amazon Q dans l'amélioration des AWS Glue services

Pour faciliter l'intégration des données Amazon Q et AWS Glue fournir les informations les plus pertinentes sur les AWS services, nous pouvons utiliser certains contenus d'Amazon Q, tels que les questions que vous posez à Amazon Q et ses réponses, afin d'améliorer le service.

Pour plus d'informations sur le contenu que nous utilisons et sur la manière de vous désinscrire, consultez Amazon Q Developer service improvement dans le Guide de l'utilisateur Amazon Q Developer.

Considérations

Prenez en compte les éléments suivants avant d'utiliser l'intégration de données Amazon Q dans AWS Glue :

- Actuellement, la génération de code ne fonctionne qu'avec PySpark le noyau. Le code généré est destiné aux AWS Glue tâches basées sur Python Spark.
- Pour plus d'informations sur les combinaisons de fonctionnalités de génération de code prises en charge pour l'intégration de données Amazon Q dans AWS Glue, consultezCapacités de génération de code prises en charge.

Configuration de l'intégration des données Amazon Q dans AWS Glue

Les sections suivantes présentent des informations sur la configuration de l'intégration des données Amazon Q dans AWS Glue.

Rubriques

- Configuration des autorisations IAM

Configuration des autorisations IAM

Cette rubrique décrit les autorisations IAM que vous configurez pour l'expérience de chat Amazon Q et l'expérience du bloc-notes AWS Glue Studio.

Rubriques

- Configuration des autorisations IAM pour le chat Amazon Q
- Configuration des autorisations IAM pour les blocs-notes AWS Glue Studio

Configuration des autorisations IAM pour le chat Amazon Q

L'octroi d'autorisations à l'intégration des données APIs Used by Amazon Q AWS Glue nécessite des autorisations AWS Identity and Access Management (IAM) appropriées. Vous pouvez obtenir des autorisations en associant la AWS politique personnalisée suivante à votre identité IAM (comme un utilisateur, un rôle ou un groupe) :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:StartCompletion",
        "glue:GetCompletion"
      ],
      "Resource": [
        "arn:aws:glue:*:*:completion/*"
      ]
    }
  ]
}
```

Configuration des autorisations IAM pour les blocs-notes AWS Glue Studio

Pour activer l'intégration des données Amazon Q dans les blocs-notes AWS Glue Studio, assurez-vous que l'autorisation suivante est attachée au rôle IAM du bloc-notes :

> **Note**
>
> Le préfixe `codewhisperer` est un nom hérité d'un service qui a fusionné avec Amazon Q Developer. Pour plus d'informations, consultez Amazon Q Developer rename - Summary of changes.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:StartCompletion",
        "glue:GetCompletion"
      ],
      "Resource": [
        "arn:aws:glue:*:*:completion/*"
      ]
    },
    {
      "Sid": "AmazonQDeveloperPermissions",
      "Effect": "Allow",
      "Action": [
        "codewhisperer:GenerateRecommendations"
      ],
      "Resource": "*"
    }
  ]
}
```

> **Note**
>
> L'intégration des données dans Amazon Q AWS Glue n' APIs est pas disponible via le AWS SDK que vous pouvez utiliser par programmation. APIs Les deux éléments suivants sont

utilisés dans la politique IAM pour permettre cette expérience via le panneau de discussion Amazon Q ou les blocs-notes AWS Glue Studio : `StartCompletion` et. `GetCompletion`

Attribution d'autorisations

Pour activer l'accès, ajoutez des autorisations à vos utilisateurs, groupes ou rôles :

- Utilisateurs et groupes dans AWS IAM Identity Center : créez un ensemble d'autorisations. Suivez les instructions de la rubrique Create a permission set du Guide de l'utilisateur AWS IAM Identity Center.
- Utilisateurs gérés dans IAM via un fournisseur d'identité : créez un rôle pour la fédération d'identité. Pour plus d'informations, voir la rubrique Création d'un rôle pour un fournisseur d'identité tiers (fédération) du Guide de l'utilisateur IAM.
- Utilisateurs IAM :
  - Créez un rôle que votre utilisateur peut assumer. Suivez les instructions de la rubrique Création d'un rôle pour un utilisateur IAM du Guide de l'utilisateur IAM.
  - (Non recommandé) Attachez une politique directement à un utilisateur ou ajoutez un utilisateur à un groupe d'utilisateurs. Suivez les instructions de la rubrique Ajout d'autorisations à un utilisateur (console) du Guide de l'utilisateur IAM.

Capacités de génération de code prises en charge

Voici les combinaisons des fonctionnalités de génération de code associées à l'intégration de données Amazon Q.

Sources et cibles	Transformation
S3 avec les types de format suivants : json, csv, parquet, hudi, delta	Drop
AWS Glue Data Catalog	Regrouper
Redlake	DropDuplicates
Amazon DynamoDB	Joindre

Sources et cibles	Transformation
MySQL	Filtre
Oracle	RenameColumns
PostgreSQL	FillNull
Microsoft SQL Server	DropNull
Amazon DocumentDB/MongoDB	WithColumns
Snowflake	Requête SQL
Google BigQuery	Union
Teradata	Select
Amazon OpenSearch Service	
Vertica	
SAP HANA	
Amazon Redshift	

Exemples d'interactions

L'intégration des données Amazon Q vous AWS Glue permet de saisir votre question dans le panneau Amazon Q. Vous pouvez saisir une question concernant la fonctionnalité d'intégration des données proposée par AWS Glue. Une réponse détaillée, accompagnée de documents de référence, vous sera renvoyée.

Un autre cas d'utilisation est la génération de scripts de travail AWS Glue ETL. Vous pouvez poser une question sur la façon d'effectuer une tâche d'extraction, de transformation et de chargement de données. Un PySpark script généré sera renvoyé.

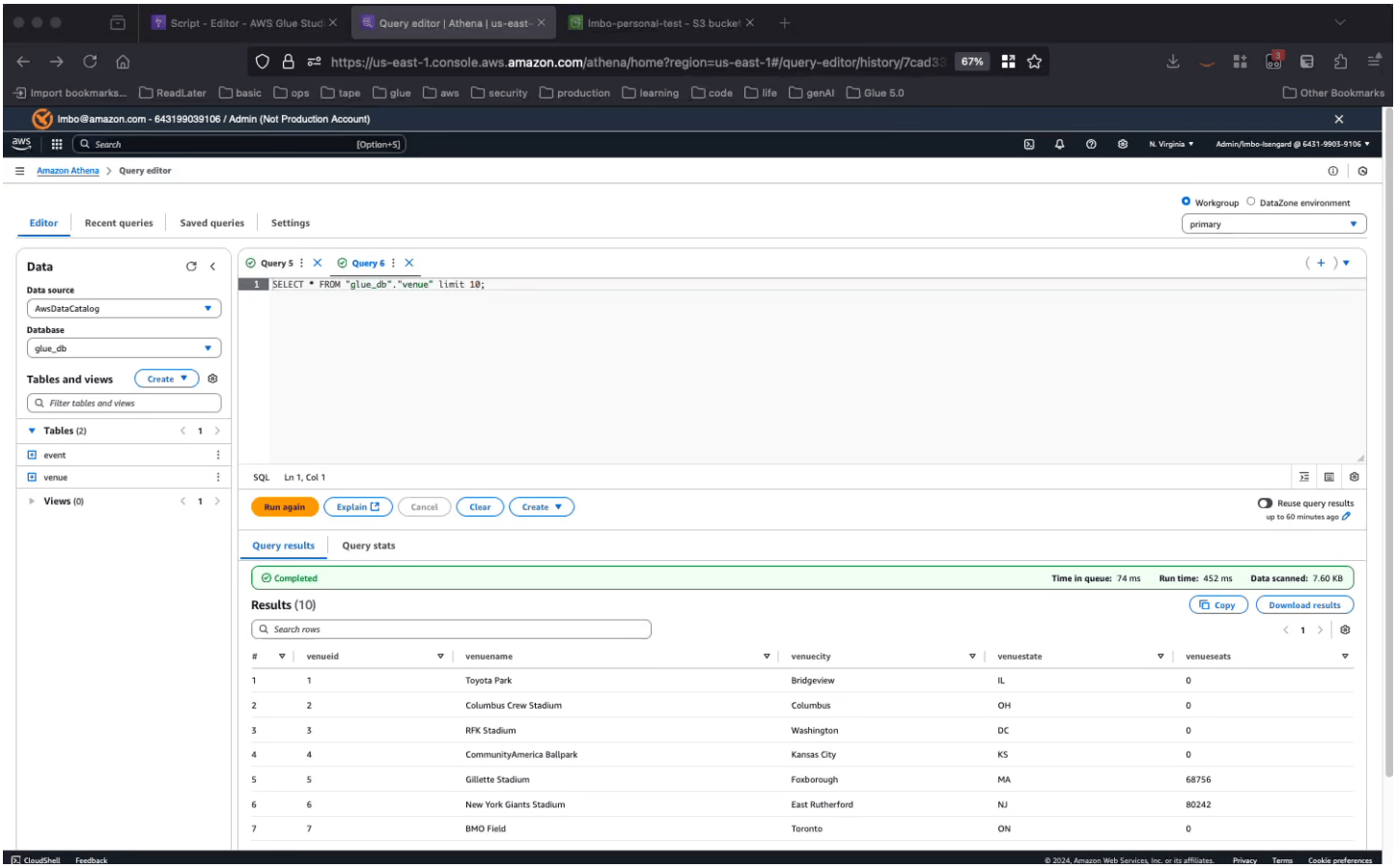
Rubriques

- Interactions dans le chat Amazon Q

- AWS Glue Interactions avec Studio Notebook

Interactions dans le chat Amazon Q

Sur la AWS Glue console, commencez à créer une nouvelle tâche et demandez à Amazon Q : « Créez un flux Glue ETL, connectez-vous à deux tables du catalogue Glue, lieu et événement, dans ma base de données glue_db, joignez les résultats sur le lieu et le e_venueid de l'événement, puis filtrez sur l'état du lieu avec la condition VenueState=='DC' et écrivez sur s3://amzn-s3-demo -/au format CSV. » bucket/codegen/BDB-9999/output



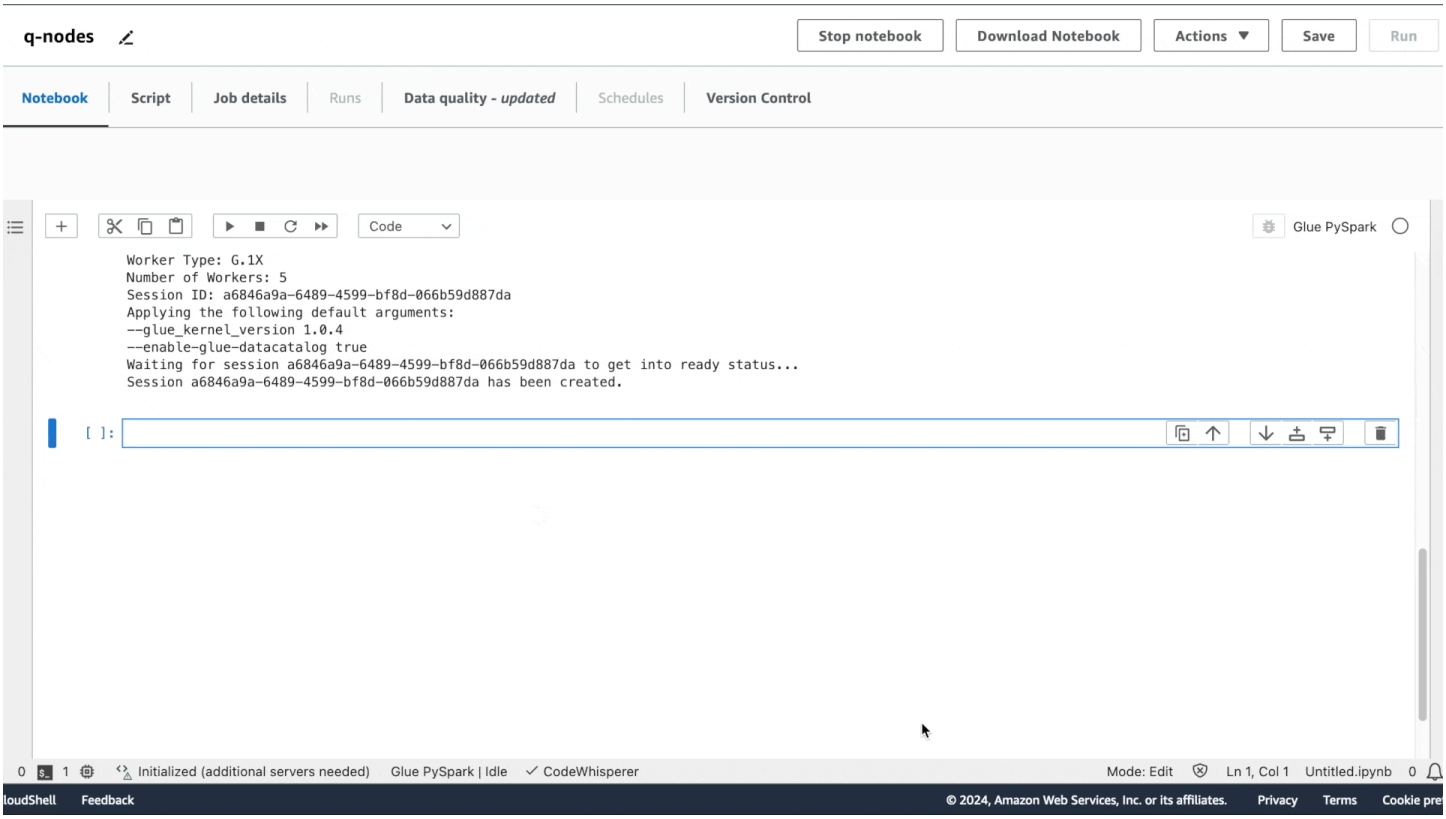
Vous remarquerez que le code est généré. Grâce à cette réponse, vous pouvez apprendre et comprendre comment créer AWS Glue du code adapté à vos besoins. Vous pouvez envoyer copy/paste le code généré dans l'éditeur de script et configurer des espaces réservés. Après avoir configuré un rôle IAM et AWS Glue les connexions sur la tâche, enregistrez et exécutez la tâche. Une fois la tâche terminée, vous pouvez vérifier que les données récapitulatives sont conservées dans Amazon S3 comme prévu et qu'elles peuvent être utilisées par vos charges de travail en aval.

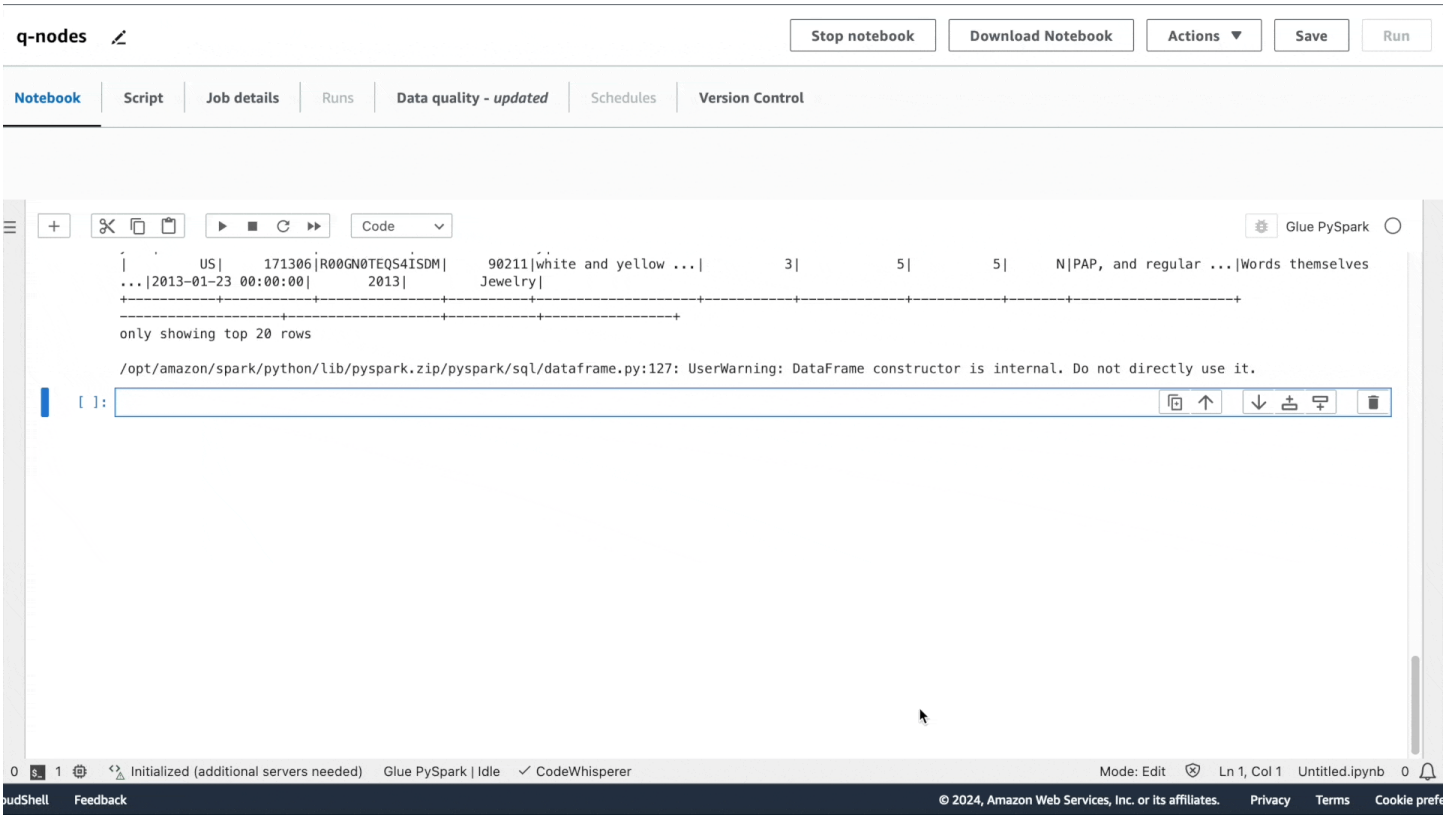
AWS Glue Interactions avec Studio Notebook

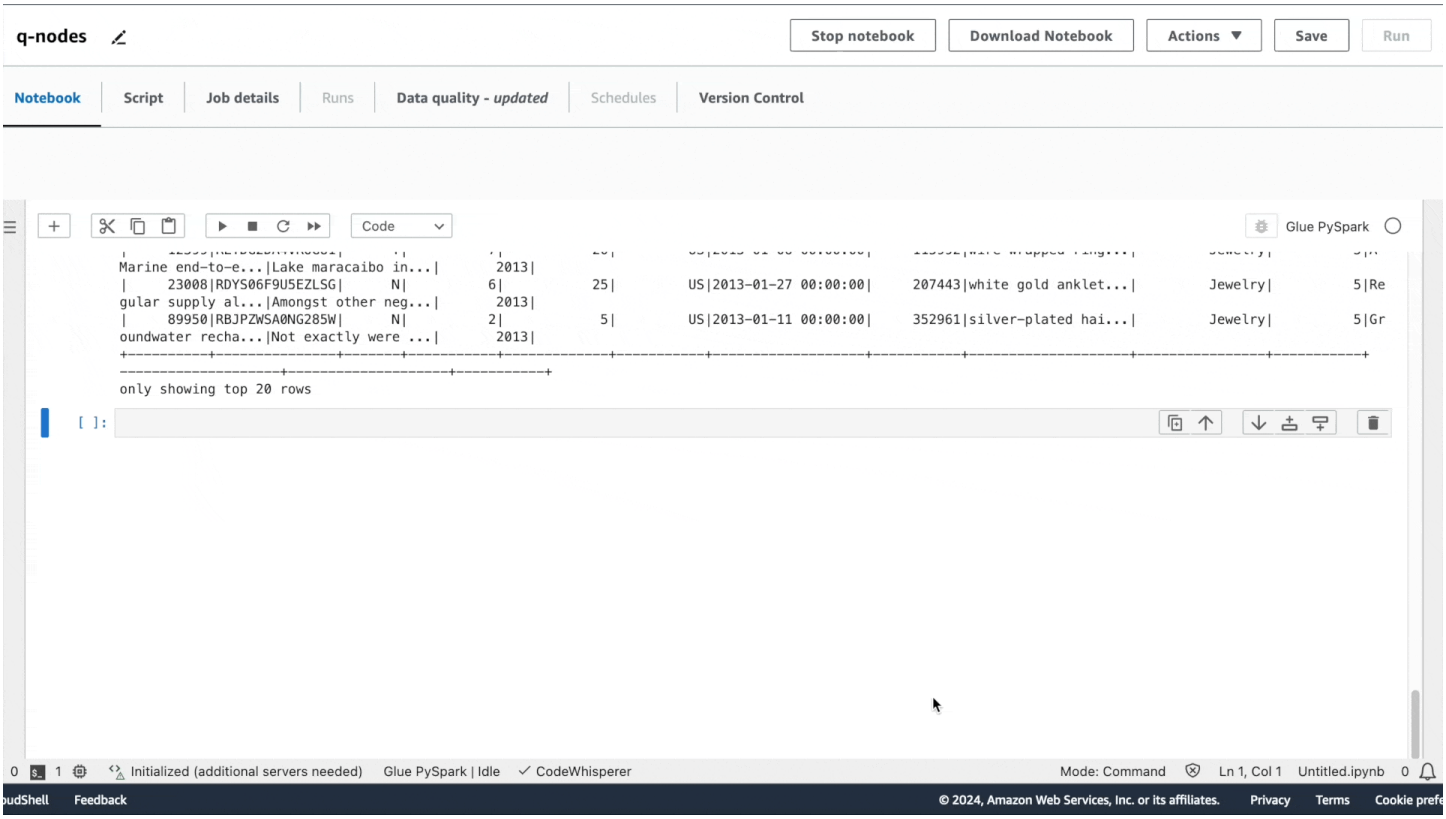
> **Note**
>
> L'expérience d'intégration des données Amazon Q dans le AWS Glue Studio bloc-notes se concentre toujours sur le flux d'intégration de données DynamicFrame basé sur le flux.

Ajoutez une nouvelle cellule et saisissez votre commentaire pour décrire ce que vous souhaitez réaliser. Après avoir appuyé sur Tabulation et sur Entrée, le code recommandé s'affiche.

Le premier objectif est d'extraire les données : « Donne moi du code qui lit une table du catalogue de données Glue », suivi de « Donne-moi du code pour appliquer une transformation de filtre avec star_rating>3 » et « Donne-moi du code qui écrit le cadre dans S3 au format Parquet ».




q-nodes
Stop notebook
Download Notebook
Actions
Save
Run
Notebook
Script
Job details
Runs
Data quality - updated
Schedules
Version Control
Code
Glue PySpark
...|2013-01-23 00:00:00|
2013|
Jewelry|
only showing top 20 rows
/opt/amazon/spark/python/lib/pyspark.zip/pyspark/sql/dataframe.py:127: UserWarning: DataFrame constructor is internal. Do not directly use it.
[ ]:
Initialized (additional servers needed)
Glue PySpark | Idle
CodeWhisperer
Mode: Edit
Ln 1, Col 1
Untitled.ipynb
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms


q-nodes
Stop notebook
Download Notebook
Actions
Save
Run
Notebook
Script
Job details
Runs
Data quality - updated
Schedules
Version Control
Code
Glue PySpark
Marine end-to-e...|Lake maracaibo in...| 2013|
| 23008|RDYS06F9U5EZLSG| N| 6| 25| US|2013-01-27 00:00:00| 207443|white gold anklet...| Jewelry| 5|Re
gular supply al...|Amongst other neg...| 2013|
| 89950|RBJPZWSA0NG285W| N| 2| 5| US|2013-01-11 00:00:00| 352961|silver-plated hai...| Jewelry| 5|Gr
oundwater recha...|Not exactly were ...| 2013|
only showing top 20 rows
[ ]:
Initialized (additional servers needed)
Glue PySpark | Idle
CodeWhisperer
Mode: Command
Ln 1, Col 1
Untitled.ipynb
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms

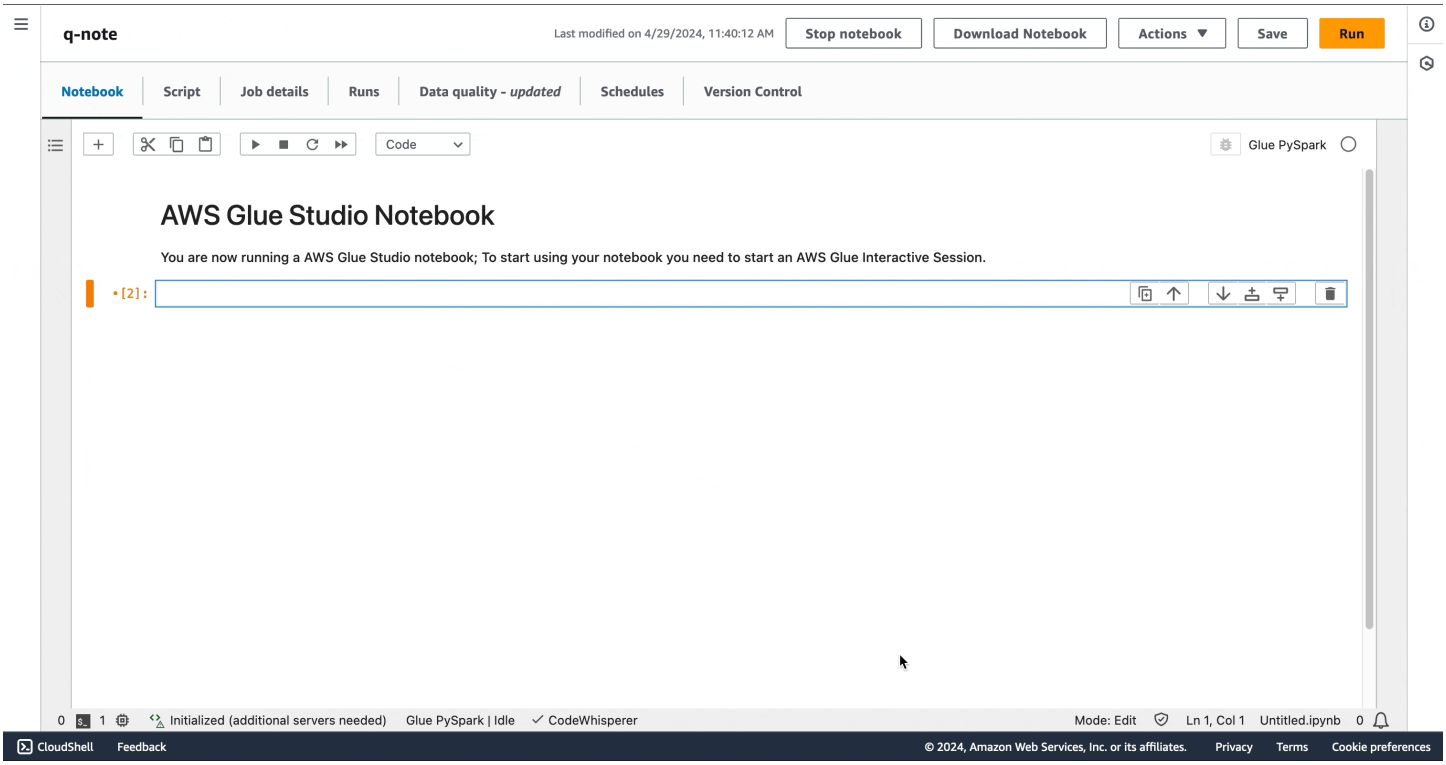
Tout comme pour l'expérience de chat Amazon Q, le code est recommandé. Si vous appuyez sur Tabulation, le code recommandé est sélectionné.

Vous pouvez exécuter chaque cellule en renseignant les options appropriées pour vos sources dans le code généré. À tout moment au cours des exécutions, vous pouvez également prévisualiser un exemple de votre jeu de données à l'aide de la méthode `show()`.

Vous pouvez exécuter le bloc-notes en tant que tâche, soit par programmation, soit en choisissant Exécuter.

Invites complexes

Vous pouvez générer un script complet à l'aide d'une seule invite complexe. « J'ai des données JSON dans S3 et des données dans Oracle qui doivent être combinées. Donne-moi un script Glue qui lit les deux sources, les combine, puis écrit les résultats dans Redshift. »



Vous remarquerez peut-être que, dans le bloc-notes, l'intégration des données Amazon Q a AWS Glue généré le même extrait de code que celui généré dans le chat Amazon Q.

Vous pouvez exécuter le bloc-notes en tant que tâche, soit par programmation, soit en choisissant Exécuter.

Utiliser la connaissance du contexte avec Amazon Q Data Integration in AWS Glue

Vous pouvez désormais créer des tâches de traitement de données de manière plus efficace grâce à la prise en compte du contexte basée sur les requêtes et à la génération de PySpark DataFrame code dans Amazon Q Data Integration. Par exemple, vous pouvez utiliser cette invite pour générer PySpark du code : « créez une tâche pour charger les données de vente depuis la table Redshift 'analytics.salesorder' en utilisant la connexion 'erp_conn', filtrez order_amount en dessous de 50 dollars et enregistrez sur Amazon S3 au format parquet ».

Amazon Q générera le script en fonction de votre demande et configurera le flux de travail d'intégration des données avec les détails fournis dans votre question, tels que les configurations de connexion, les détails du schéma, les database/table noms et les spécifications des colonnes pour les transformations. Les informations sensibles, telles que les mots de passe des options de connexion, continuent d'être expurgées.

Si les informations requises ne sont pas fournies à partir de votre invite, Amazon Q placera des espaces réservés, dans lesquels vous devrez mettre à jour le code généré avec les valeurs appropriées avant d'exécuter le code.

Vous trouverez ci-dessous des exemples d'utilisation de la prise en compte du contexte.

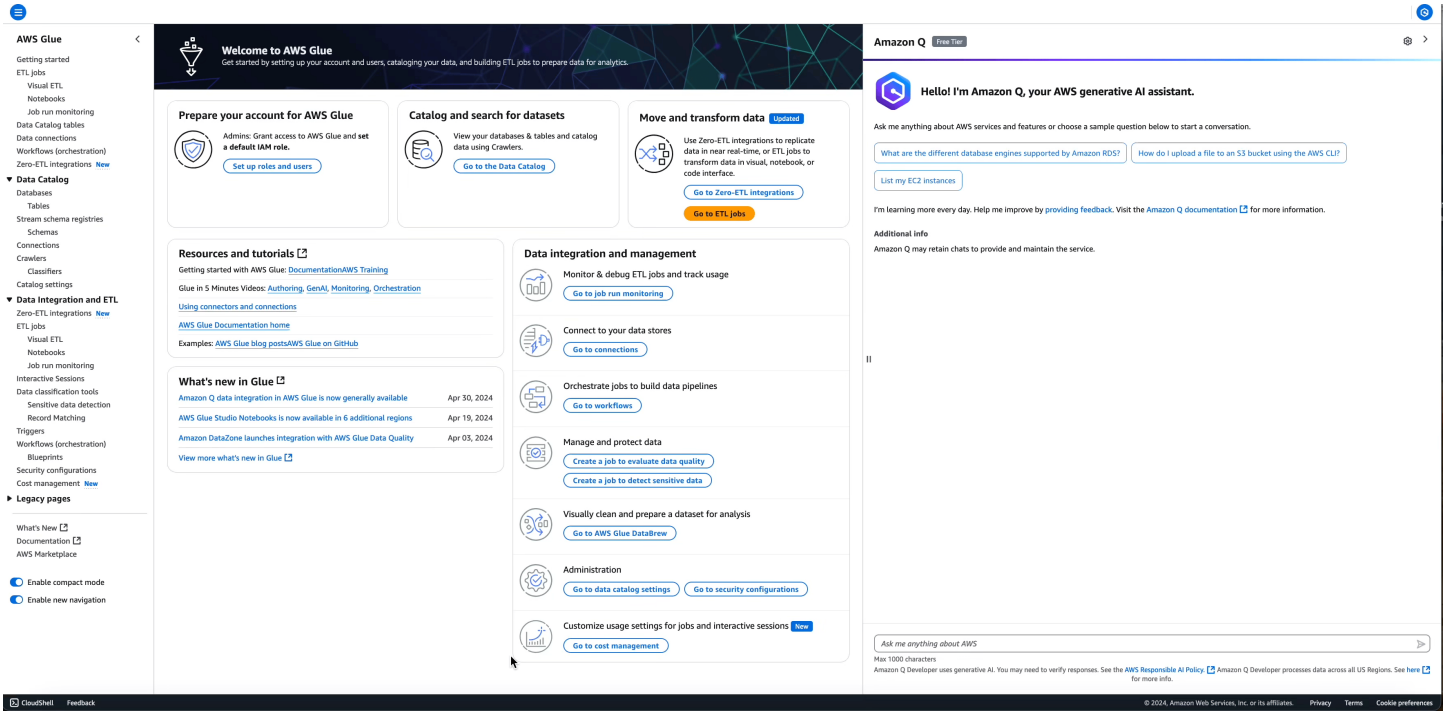
Exemple : interactions

Lecture d'un fichier Amazon S3 et écriture dans un compartiment Amazon S3

Invite : `Create an AWS Glue spark job that reads a file s3://amzn-s3-demo-bucket-input-bucket-name/input-filename.csv and saves it into the parquet file under directory s3://amzn-s3-demo-bucket-output-bucket-name/output-s3-prefix/`

```
import sys
from awsglue.transforms import *
from pyspark.context import SparkContext
from pyspark.sql import SparkSession

sc = SparkContext.getOrCreate()
spark = SparkSession.builder.getOrCreate()

# Script generated for node S3DataSource
```

```
S3DataSource_dsource1 = spark.read.format("csv").load(
    "s3://amzn-s3-demo-bucket-input-bucket-name/input-filename.csv"
)

# Script generated for node S3DataSink
S3DataSource_dsource1.write.format("parquet").mode("append").save(
    "s3://amzn-s3-demo-bucket-output-bucket-name/output-s3-prefix/"
)
```



Récupération des données de Lakehouse et écriture dans la base de données

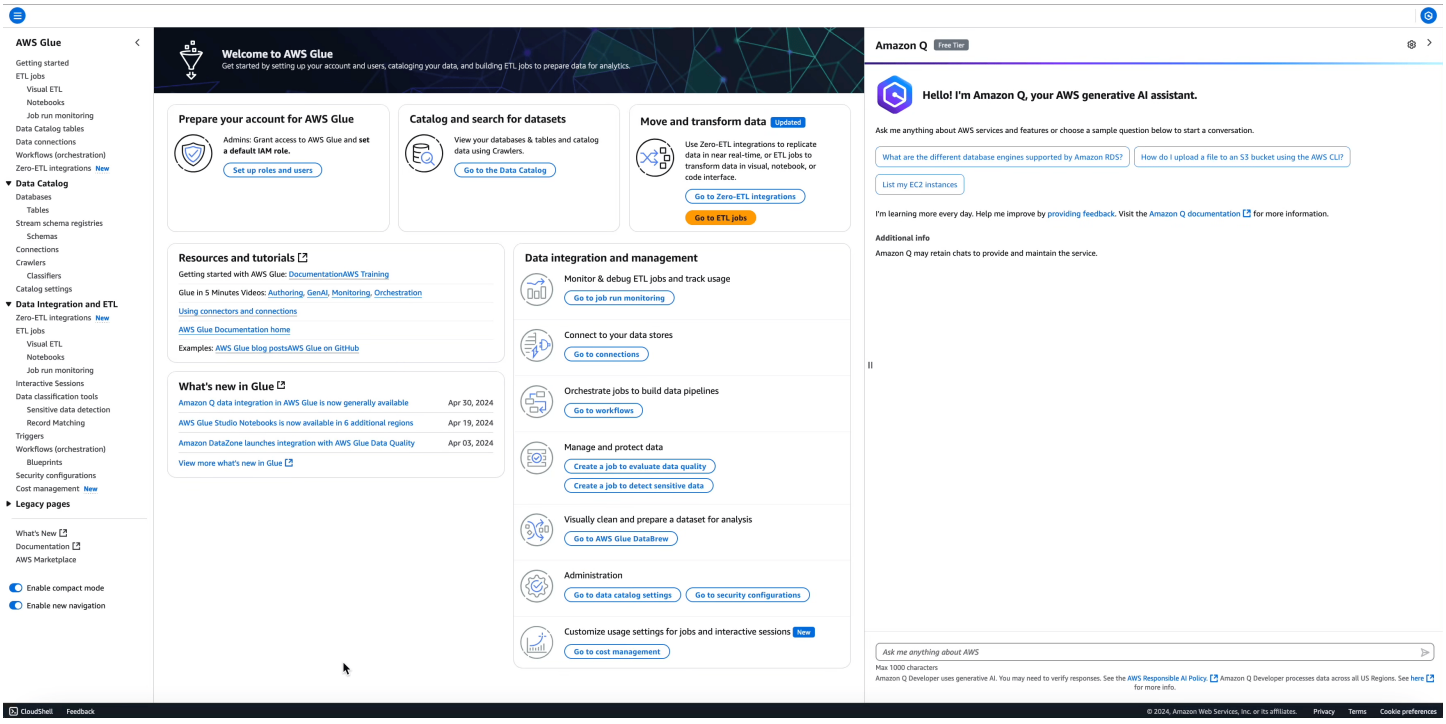
Invite : `write an ETL script to read from a Lakehouse table my-table in database my-database and write it to a RDS MySQL table my-target-table`

Pour les champs où vous n'avez pas fourni d'informations (par exemple, le nom de connexion requis est pour le récepteur de données MySQL et contient par défaut un espace réservé <connection-name> dans le code généré), un espace réservé est conservé pour que vous puissiez renseigner les informations requises avant d'exécuter le script.

Script généré :

```
import sys
from awsglue.transforms import *
from pyspark.context import SparkContext
```

```
from pyspark.sql import SparkSession
from connectivity.adapter import CatalogConnectionHelper

sc = SparkContext.getOrCreate()
spark = SparkSession.builder.getOrCreate()

# Script generated for node S3DataSource
S3DataSource_dsource1 = spark.read.format("parquet").load(
    "s3://amzn-lakehouse-demo-bucket/my-database/my-table"
)

# Script generated for node ConnectionV2DataSink
ConnectionV2DataSink_dsink1_additional_options = {"dbtable": "my-target-table"}
CatalogConnectionHelper(spark).write(
    S3DataSource_dsource1,
    "mysql",
    "<connection-name>",
    ConnectionV2DataSink_dsink1_additional_options,
)
```



Exemple : flux de travail ETL complet

L'exemple suivant montre comment vous pouvez demander à AWS Glue de créer un script AWS Glue pour terminer un flux de travail ETL complet avec l'invite suivante :`Create a AWS Glue ETL`

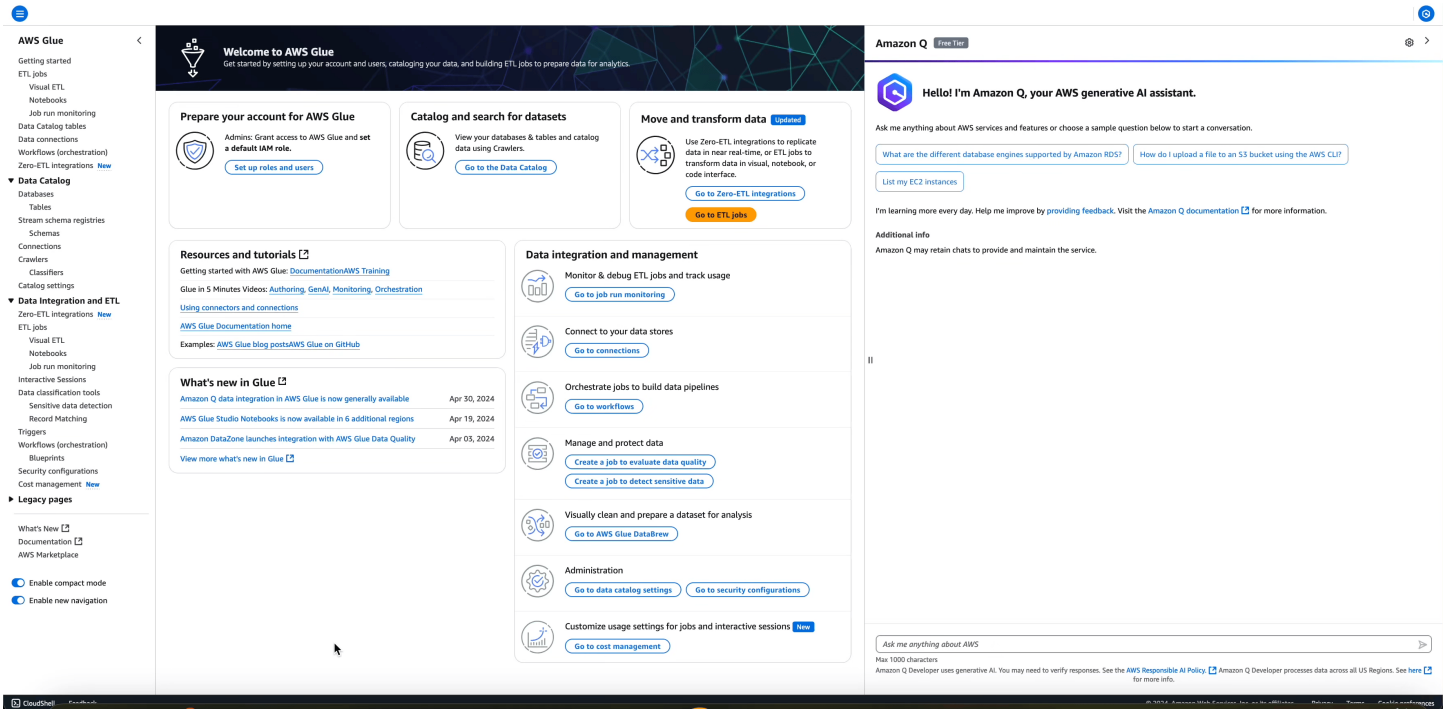
`Script read from two AWS Glue Data Catalog tables venue and event in my database glue_db_4fthqih3vvk1if, join the results on the field venueid, filter on venue state with condition as venuestate=='DC' after joining the results and write output to an Amazon S3 S3 location s3://amz-s3-demo-bucket/output/ in CSV format.`

Le flux de travail comprend la lecture à partir de différentes sources de données (deux tables AWS Glue Data Catalog) et quelques transformations après la lecture en joignant le résultat de deux lectures, en filtrant en fonction de certaines conditions et en écrivant la sortie transformée dans une destination Amazon S3 au format CSV.

La tâche générée renseignera les informations détaillées sur la source de données, les opérations de transformation et de réception avec les informations correspondantes extraites de la question de l'utilisateur, comme ci-dessous.

```
import sys
from awsglue.transforms import *
from pyspark.context import SparkContext
from pyspark.sql import SparkSession

sc = SparkContext.getOrCreate()
spark = SparkSession.builder.getOrCreate()

# Script generated for node CatalogDataSource
CatalogDataSource_dsource1 = spark.sql("select * from
 `glue_db_4fthqih3vvk1if`.`venue`")

# Script generated for node CatalogDataSource
CatalogDataSource_dsource2 = spark.sql("select * from
 `glue_db_4fthqih3vvk1if`.`event`")

# Script generated for node JoinTransform
JoinTransform_transform1 = CatalogDataSource_dsource1.join(
    CatalogDataSource_dsource2,
    (CatalogDataSource_dsource1["venueid"] == CatalogDataSource_dsource2["venueid"]),
    "inner",
)

# Script generated for node FilterTransform
FilterTransform_transform2 = JoinTransform_transform1.filter("venuestate=='DC'")
```

```
# Script generated for node S3DataSink
FilterTransform_transform2.write.format("csv").mode("append").save(
    "s3://amz-s3-demo-bucket/output//output/"
)
```



Limitations

- Report du contexte :
  - La fonctionnalité de prise en compte du contexte ne reporte que le contexte de la requête précédente de l'utilisateur au sein de la même conversation. Elle ne conserve pas le contexte au-delà de la requête immédiatement précédente.
- Prise en charge des configurations de nœuds :
  - À l'heure actuelle, la prise en compte du contexte ne prend en charge qu'un sous-ensemble des configurations requises pour les différents nœuds.
  - La prise en charge des champs facultatifs est prévue dans les prochaines versions.
- Disponibilité :
  - La connaissance du contexte et le DataFrame support sont disponibles dans les blocs-notes Q Chat et SageMaker Unified Studio. Toutefois, ces fonctionnalités ne sont pas encore disponibles dans les blocs-notes AWS Glue Studio.

Orchestration dans AWS Glue

Les sections suivantes vous renseignent sur l'orchestration d'applications dans AWS Glue.

Rubriques

- Démarrage des tâches et des crawlers à l'aide de déclencheurs
- Exécution d'activités ETL complexes à l'aide de plans et de flux de travail dans AWS Glue
- Développement des plans dans AWS Glue

Démarrage des tâches et des crawlers à l'aide de déclencheurs

Dans AWS Glue, vous pouvez créer des objets de catalogue de données appelés déclencheurs, que vous pouvez utiliser pour démarrer manuellement ou automatiquement un ou plusieurs crawlers, ou des tâches ETL (extraction, transformation et chargement). À l'aide de déclencheurs, vous pouvez concevoir une chaîne de tâches et d'crawlers dépendants.

> **Note**
>
> Vous pouvez accomplir la même chose en définissant des flux de travail. Les flux de travail sont préférables pour créer des opérations ETL multitâches complexes. Pour de plus amples informations, veuillez consulter the section called "Exécution d'activités ETL complexes à l'aide de plans et de flux de travail".

Rubriques

- Déclencheurs AWS Glue
- Ajout de déclencheurs
- Activation et désactivation des déclencheurs

Déclencheurs AWS Glue

Lorsqu'il est lancé, un déclencheur peut démarrer des tâches et des crawlers spécifiés. Il se déclenche à la demande en fonction d'une planification ou d'une combinaison d'événements.

> **Note**
>
> Seuls deux crawlers peuvent être activés par un même déclencheur. Si vous souhaitez analyser plusieurs magasins de données, utilisez plusieurs sources pour chaque crawler au lieu d'exécuter plusieurs crawlers simultanément.

Un déclencheur peut exister dans différents états : `CREATED`, `ACTIVATED` ou `DEACTIVATED`. Il peut aussi passer par des états transitoires, comme `ACTIVATING`. Pour arrêter temporairement le lancement d'un déclencheur, vous pouvez le désactiver. Vous pourrez le réactiver ultérieurement.

Il existe trois types de tâches :

Planifié

Déclencheur temporel basé sur `cron`.

Vous pouvez créer un déclencheur pour un ensemble de tâches ou de crawlers en fonction d'une planification. Vous pouvez spécifier des contraintes, telles que la fréquence d'exécution des tâches ou des crawlers, les jours de la semaine qu'ils s'exécutent et à quelle heure. Ces contraintes sont basées sur `cron`. Lorsque vous configurez la planification d'un déclencheur, prenez en compte les fonctions et limitations de cron. Par exemple, si vous choisissez d'exécuter votre crawler le 31 de chaque mois, n'oubliez pas que certains mois ne comportent pas 31 jours. Pour plus d'informations sur cron, consultez Planifications temporelles pour les tâches et les crawlers.

Conditionnel

Déclencheur qui se lance lorsqu'une ou plusieurs tâches précédentes, ou encore un ou plusieurs crawlers précédents satisfont à une liste de conditions.

Lorsque vous créez un déclencheur conditionnel, vous spécifiez une liste de tâches et une liste de crawlers à surveiller. Pour chaque tâche surveillée ou crawler, vous spécifiez un état à surveiller, tel que réussite, échec, expiré, etc. Le déclencheur se lance si les tâches ou les crawlers surveillés se finalisent par les statuts spécifiés. Vous pouvez configurer le déclencheur pour qu'il se lance lorsque tout ou partie des événements surveillés se produisent.

Par exemple, vous pouvez configurer un déclencheur T1 pour démarrer la tâche J3 lorsque la tâche J1 et la tâche J2 sont terminées avec succès, et un autre déclencheur T2 pour démarrer la tâche J4 si la tâche J1 ou la tâche J2 échoue.

Le tableau suivant répertorie les états de fin de tâche et de crawler qui déclenchent la surveillance.

États d'achèvement des tâches	États d'achèvement des crawlers
• `SUCCEEDED` • `STOPPED` • `FAILED` • `TIMEOUT`	• `SUCCEEDED` • `FAILED` • `CANCELLED`

À la demande

Déclencheur qui se lance lorsque vous l'activez. Les déclencheurs à la demande ne peuvent pas comporter l'état `DEACTIVATED` ou `ACTIVATED`. Ils restent toujours à l'état `CREATED`.

Afin qu'ils soient prêts à se déclencher dès qu'ils existent, vous pouvez définir un indicateur pour activer les déclencheurs planifiés et conditionnels lors de leur création.

> **Important**
>
> Les tâches ou les crawlers qui s'exécutent à la suite de l'exécution d'autres tâches ou crawlers sont considérés comme dépendants. Les tâches ou les crawlers dépendants ne sont lancés que si la tâche ou le crawler qui se finalise a été démarré par un déclencheur. Toutes les tâches d'une chaîne de dépendance doivent descendre d'un seul calendrier ou d'un déclencheur à la demande.

Transmission de paramètres de tâches avec des déclencheurs

Un déclencheur permet de transmettre des paramètres aux tâches qu'il démarre. Ceux-ci incluent les arguments des tâches, la valeur de délai d'attente, la configuration de sécurité, etc. Si le déclencheur démarre plusieurs tâches, les paramètres sont transmis à chacune d'elles.

Voici les règles pour les arguments de tâche transmis par un déclencheur :

- Si la clé de la paire clé-valeur correspond à un argument de tâche par défaut, l'argument transmis remplace l'argument par défaut. Si la clé ne correspond pas à un argument par défaut, l'argument est transmis en tant qu'argument supplémentaire pour la tâche.
- Si la clé de la paire clé-valeur correspond à un argument non remplaçable, l'argument transmis est ignoré.

Pour plus d'informations, consultez the section called "Triggers" dans l'API AWS Glue.

Ajout de déclencheurs

Vous pouvez ajouter un déclencheur à l'aide de la AWS Glue console, du AWS Command Line Interface (AWS CLI) ou de l'AWS GlueAPI.

Pour ajouter un déclencheur (console)

1. Connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/.
2. Dans le panneau de navigation, sous ETL, choisissez Triggers (Déclencheurs). Choisissez Add trigger (Ajouter un déclencheur).
3. Fournissez les propriétés suivantes :

   Name

   Attribuez un nom unique à votre déclencheur.

   Type de déclencheur

   Spécifiez l'un des éléments suivants :

   - Schedule (Planification) : le déclencheur se lance à une fréquence et à une heure spécifiques.
   - Job Events (Événements de tâche) : déclencheur conditionnel. Le déclencheur est exécuté lorsqu'une ou plusieurs tâches de la liste correspondent à l'événement de tâche sélectionné. Pour que le déclencheur s'exécute, les tâches surveillées doivent avoir été lancées par des déclencheurs. Pour chaque tâche que vous choisissez, vous ne pouvez surveiller qu'un événement de tâche (état d'achèvement).
   - On-demand (À la demande) : le déclencheur se lance lorsqu'il est activé.
4. Terminez l'assistant de déclenchement. Sur la page Review (Vérifier), vous pouvez activer immédiatement les déclencheurs Schedule (Planification) et Job events (Événements de tâche)

(conditionnels) en sélectionnant Enable trigger on creation (Activer le déclencheur lors de la création).

Pour ajouter un déclencheur (AWS CLI)

- Utilisez une commande similaire à la suivante.

```
aws glue create-trigger --name MyTrigger --type SCHEDULED --schedule  "cron(0 12 *
 * ? *)" --actions CrawlerName=MyCrawler --start-on-creation
```

Cette commande crée un déclencheur de planification nommé `MyTrigger`, qui s'exécute tous les jours à 12h00 UTC et démarre un crawler nommé `MyCrawler`. Le déclencheur est créé dans l'état activé.

Pour de plus amples informations, veuillez consulter the section called "Déclencheurs AWS Glue".

Planifications temporelles pour les tâches et les crawlers

Vous pouvez définir une planification temporelle pour vos crawlers et vos tâches dans AWS Glue. La définition de ces planifications utilise la syntaxe de type Unix cron. Vous spécifiez l'heure en heure UTC, et la précision minimale d'une planification est de 5 minutes.

Pour en savoir plus sur la configuration des tâches et des crawlers à exécuter à l'aide d'une planification, veuillez consulter Démarrage des tâches et des crawlers à l'aide de déclencheurs.

Expressions Cron

Ces expressions se composent de six champs obligatoires qui sont séparés par des espaces.

Syntaxe

```
cron(Minutes Hours Day-of-month Month Day-of-week Year)
```

Champs	Valeurs	Caractères génériques
Minutes	0–59	, - * /
Heures	0 – 23	, - * /

Champs	Valeurs	Caractères génériques
D ay-of-month	1–31	, - * ? / L W
Mois	1–12 ou JAN–DEC	, - * /
D ay-of-week	1–7 ou dim.–sam.	, - * ? / L
Année	1970-2199	, - * /

Caractères génériques

- Le caractère générique , (virgule) inclut des valeurs supplémentaires. Dans le champ `Month`, `JAN,FEB,MAR` contiendrait janvier, février et mars.
- Le caractère générique - (tiret) spécifie des plages. Dans le champ `Day`, 1–15 inclurait les jours 1 à 15 du mois spécifié.
- Le caractère générique * (astérisque) inclut toutes les valeurs du champ. Dans le champ `Hours`, * inclut chaque heure.
- Le caractère générique / (barre oblique) spécifie les incréments. Dans le champ `Minutes`, vous pouvez saisir **`1/10`** pour spécifier un dixième de minute, en démarrant de la première minute de l'heure (par exemple, les 11e, 21e et 31e minutes).
- Le caractère générique ? (point d'interrogation) indique l'un ou l'autre. Dans le `Day-of-month` champ, tu pouvais saisir 7, et si tu ne te souciais pas du jour de la semaine le septième, tu pourrais entrer ? sur le Day-of-week terrain.
- Le caractère générique L dans les champs ou spécifie le dernier jour du mois ou de la semaine.`Day-of-month``Day-of-week`
- Le caractère générique W dans le champ spécifie un jour de la semaine. `Day-of-month` Dans le champ `Day-of-month`, `3W` spécifie le jour le plus proche du troisième jour de semaine du mois.

Restrictions

- Vous ne pouvez pas spécifier les champs `Day-of-month` et `Day-of-week` de la même expression cron. Si vous spécifiez une valeur dans l'un de ces champs, vous devez utiliser un signe ? (point d'interrogation) dans l'autre.
- Les expressions cron qui entraînent des fréquences d'une rapidité supérieure à 5 minutes ne sont pas prises en charge.

Exemples

Lors de la création d'une planification, vous pouvez utiliser les exemples de chaînes cron suivants.

Minutes	Heures	Jour du mois	Mois	Jour de la semaine	Année	Signification
0 USD	10	*	*	?	*	Exécuter à 10 h 00 (UTC) chaque jour
15	12	*	*	?	*	Exécuter à 12 h 15 (UTC) chaque jour
0	18	?	*	MON-FRI	*	Exécuter à 18 h 00 (UTC) du lundi au vendredi
0	8	1	*	?	*	Exécuter à 8 h 00 (UTC) chaque 1er jour du mois
0/15	*	*	*	?	*	Exécuter toutes les 15 minutes
0/10	*	?	*	MON-FRI	*	Exécuter toutes les 10 minutes

Minutes	Heures	Jour du mois	Mois	Jour de la semaine	Année	Signification
						du lundi au vendredi
0/5	8–17	?	*	MON-FRI	*	Exécuter toutes les 5 minutes du lundi au vendredi entre 8 h 00 et 17 h 55 (UTC)

Par exemple, pour une exécution quotidienne à 12 h 15 UTC, spécifiez :

```
cron(15 12 * * ? *)
```

Activation et désactivation des déclencheurs

Vous pouvez activer ou désactiver un déclencheur à l'aide de la AWS Glue console, du AWS Command Line Interface (AWS CLI) ou de l'AWS GlueAPI.

Pour activer ou désactiver un déclencheur (console)

1. Connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/.
2. Dans le panneau de navigation, sous ETL, choisissez Triggers (Déclencheurs).
3. Activez la case à cocher en regard du déclencheur souhaité et, dans le menu Action, choisissez Enable trigger (Activer le déclencheur) pour activer le déclencheur ou Disable trigger (Désactiver le déclencheur) pour désactiver le déclencheur.

Pour activer ou désactiver un déclencheur (AWS CLI)

- Entrez l'une des commandes suivantes.

```
aws glue start-trigger --name MyTrigger

aws glue stop-trigger --name MyTrigger
```

  Le démarrage d'un déclencheur l'active, tandis que l'arrêt d'un déclencheur le désactive. Lorsque vous activez un déclencheur à la demande, il s'exécute immédiatement.

Pour de plus amples informations, veuillez consulter the section called "Déclencheurs AWS Glue".

Exécution d'activités ETL complexes à l'aide de plans et de flux de travail dans AWS Glue

Certains des processus d'extraction, transformation et chargement (ETL) complexes de votre organisation peuvent être mieux mis en œuvre en utilisant plusieurs tâches et crawlers qui dépendent de AWS Glue. Grâce aux flux de travail AWS Glue, vous pouvez concevoir un processus ETL complexe multi-tâches et à crawlers multiples que AWS Glue peut exécuter et suivre en tant qu'entité unique. Une fois que vous avez créé un flux de travail et spécifié les tâches, les crawlers et les déclencheurs dans le flux de travail, vous pouvez exécuter le flux de travail à la demande ou selon un planning.

Rubriques

- Présentation des flux de travail dans AWS Glue
- Création et développement d'un flux de travail manuellement dans AWS Glue
- Démarrer un AWS Glue flux de travail avec un EventBridge événement Amazon
- Afficher les EventBridge événements qui ont lancé un flux de travail
- Exécution et surveillance d'un flux de travail dans AWS Glue
- Arrêt d'une exécution de flux de travail
- Réparation et reprise de l'exécution d'un flux de travail
- Obtention et définition des propriétés d'exécution du flux de travail dans AWS Glue
- Interrogation des flux de travail à l'aide du AWS Glue API
- Restrictions du plan et du flux de travail dans AWS Glue

- Résolution des erreurs de plans dans AWS Glue
- Autorisations de personas et de rôles pour les plans AWS Glue

Présentation des flux de travail dans AWS Glue

Dans AWS Glue, vous pouvez utiliser les flux de travail pour créer et visualiser les activités complexes d'extraction, de transformation et de chargement (ETL) impliquant plusieurs crawlers, tâches et déclencheurs. Chaque flux de travail gère l'exécution et la surveillance de l'ensemble de ses tâches et crawlers. Tandis qu'un flux de travail exécute chaque composant, il enregistre le statut et la progression de l'exécution. Vous trouverez ainsi une présentation de la tâche la plus volumineuses et les détails de chaque étape. La console AWS Glue fournit une représentation visuelle d'un flux de travail sous forme de graphique.

Vous pouvez créer un flux de travail à partir d'un AWS Glue plan, ou vous pouvez créer manuellement un flux de travail composant par composant à l'aide du AWS Management Console ou du AWS Glue API. Pour plus d'informations sur les modèles de présentation, consultez the section called “Présentation des plans”.

Les déclencheurs dans les flux de travail peuvent démarrer les tâches ou les crawlers, et peuvent être déclenchés lorsque les tâches et les crawlers ont terminé leurs opérations. En utilisant des déclencheurs, vous pouvez créer de grandes chaînes de tâches et d'crawlers interdépendants. En plus des déclencheurs au sein d'un flux de travail qui définissent les dépendances des tâches et de l'crawler, chaque flux de travail a un déclencheur de démarrage. Il existe trois types de déclencheurs :

- Schedule (Planification) - le flux de travail est démarré selon un planificateur que vous définissez. La planification peut être quotidienne, hebdomadaire, mensuelle, etc., ou peut être une planification personnalisée basée sur une expression `cron`.
- À la demande : le flux de travail est démarré manuellement depuis la AWS Glue console, l'API ou AWS CLI.
- EventBridge événement — Le flux de travail est lancé lors de la survenance d'un seul EventBridge événement Amazon ou d'un lot d' EventBridge événements Amazon. Avec ce type de déclencheur, AWS Glue peut être un consommateur d'événements dans une architecture événementielle. Tout type EventBridge d'événement peut démarrer un flux de travail. L'arrivée d'un nouvel objet dans un compartiment Amazon S3 (l'opération `PutObject` S3) fait partie des cas d'utilisation les plus courants.

Démarrer un flux de travail avec un lot d'événements signifie attendre qu'un nombre spécifié d'événements ait été reçu ou qu'un laps de temps spécifié se soit écoulé. Lorsque vous créez le déclencheur EventBridge d'événement, vous pouvez éventuellement spécifier les conditions du lot. Si vous spécifiez des conditions de lot, vous devez spécifier la taille du lot (nombre d'événements) et pouvez éventuellement spécifier une fenêtre de lot (nombre de secondes). La fenêtre de lot par défaut et maximale est de 900 secondes (15 minutes). La condition de lot qui est remplie démarre en premier le flux de travail. La fenêtre de traitement par lots démarre lorsque le premier événement arrive. Si vous ne spécifiez pas de conditions de lot lors de la création d'un déclencheur, la taille du lot par défaut est 1.

Lorsque le flux de travail démarre, les conditions du lot sont réinitialisées et le déclencheur d'événement commence à surveiller la prochaine condition de lot à remplir pour redémarrer le flux de travail.

Le tableau suivant montre comment la taille du lot et la fenêtre de lot fonctionnent ensemble pour déclencher un flux de travail.

Taille de lot	Fenêtre de lot	Condition de déclenchement obtenue
10		Le flux de travail est déclenché à l'arrivée de 10 EventBridge événements, ou 15 minutes après l'arrivée du premier événement, selon la première éventualité. (Si la taille de la fenêtre temporelle n'est pas spécifiée, la valeur par défaut est de 15 minutes.)
10	2 min	Le flux de travail est déclenché à l'arrivée de 10 EventBridge événements, ou 2 minutes après l'arrivée du premier événement, selon la première éventualité.
1		Le flux de travail est déclenché dès l'arrivée du premier événement. La taille de la fenêtre n'est pas pertinent e. La taille du lot est définie par défaut sur 1 si vous ne spécifiez pas les conditions du lot lors de la création du déclencheur EventBridge d'événement.

L'opération d'API `GetWorkflowRun` renvoie la condition de lot qui a déclenché le flux de travail.

Quel que soit le mode de démarrage d'un flux de travail, vous pouvez spécifier le nombre maximal d'exécutions simultanées de flux de travail lorsque vous créez le flux de travail.

Si un événement ou un lot d'événements démarre une exécution de flux de travail qui échoue finalement, cet événement ou lot d'événements n'est plus pris en compte pour démarrer une exécution de flux de travail. Une nouvelle exécution de flux de travail est démarrée uniquement lorsque le prochain événement ou lot d'événements se présente.

> **Important**
>
> Limitez le nombre total de tâches, de crawlers et de déclencheurs au sein d'un flux de travail à 100 ou moins. Si vous en incluez plus de 100, vous risquez de rencontrer des erreurs lorsque vous tentez de reprendre ou d'arrêter les exécutions du flux de travail.

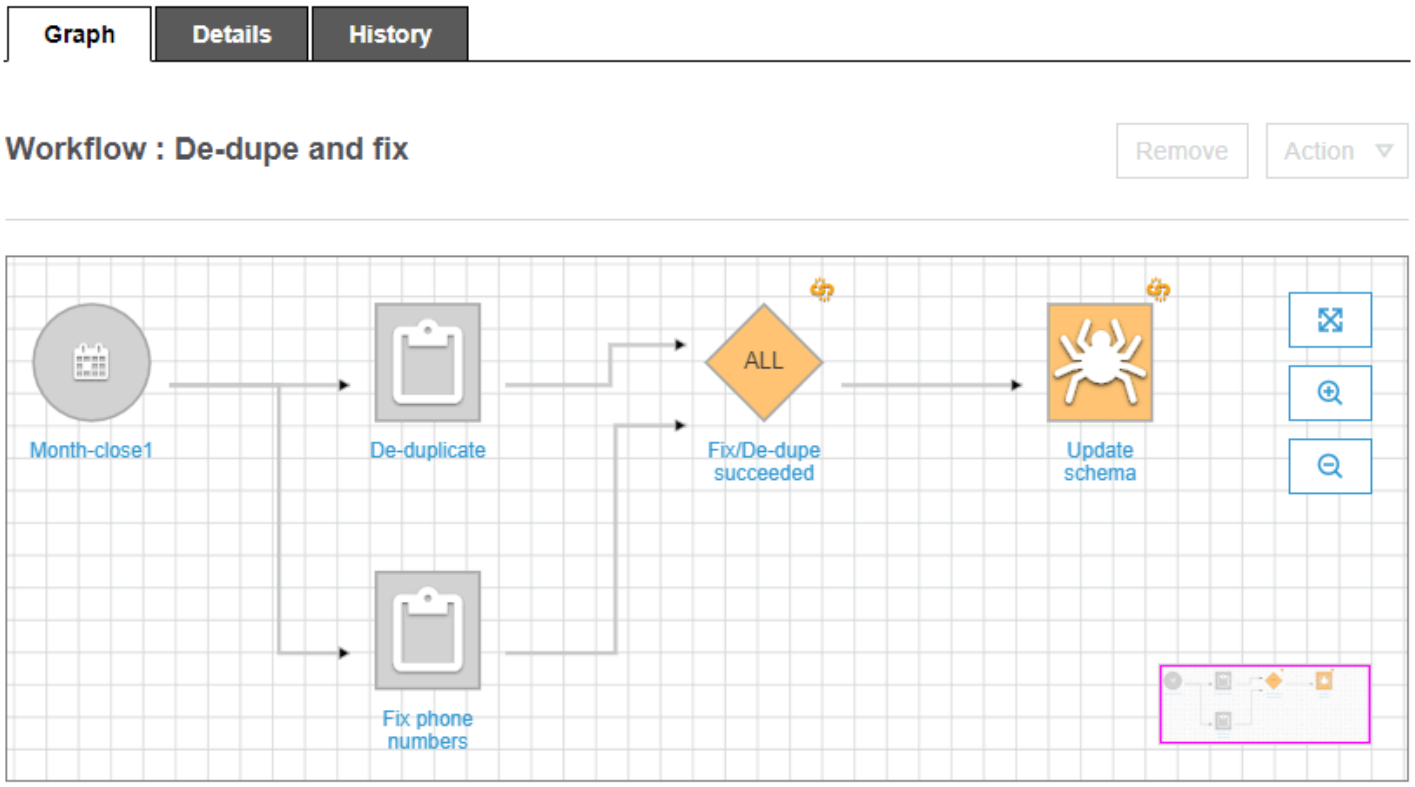
Une exécution de flux de travail ne démarrera pas si elle dépasse la limite de simultanéité définie pour le flux de travail, même si la condition d'événement est remplie. Il est conseillé d'ajuster les limites de simultanéité du flux de travail en fonction du volume d'événements attendu. AWS Glue ne retente pas les exécutions de flux de travail qui échouent en raison du dépassement des limites de simultanéité. De même, il est conseillé d'ajuster les limites de simultanéité pour les tâches et les crawlers dans les flux de travail en fonction du volume d'événements attendu.

Propriétés de l'exécution du flux de travail

Pour partager et gérer l'état tout au long de l'exécution d'un flux de travail, vous pouvez définir des propriétés d'exécution de flux de travail par défaut. Ces propriétés, qui sont des name/value paires, sont disponibles pour toutes les tâches du flux de travail. À l'aide de AWS Glue API, les tâches peuvent récupérer les propriétés d'exécution du flux de travail et les modifier pour les tâches qui seront ajoutées ultérieurement dans le flux de travail.

Graphique de flux de travail

L'image suivante illustre le graphique d'un flux de travail vraiment de base sur la console AWS Glue. Votre flux de travail peut avoir des douzaines de composants.



Ce flux de travail est démarré par un déclencheur de calendrier, `Month-close1`, qui lance deux tâches, `De-duplicate` et `Fix phone numbers`. Lorsque vous avez validé les deux tâches, un déclencheur d'événement, `Fix/De-dupe succeeded`, lance un crawler, `Update schema`.

Affichages de flux de travail statiques et dynamiques

Pour chaque flux de travail, il existe la notion de vue statique et de vue dynamique. La conception statique indique la conception du flux de travail. L'affichage dynamique est une vue de l'exécution qui inclut les dernières informations d'exécution pour chacune des tâches et chacun des crawlers. Les informations d'exécution inclut le statut de succès et les informations détaillées d'erreur.

Lorsqu'un flux de travail est en cours d'exécution, la console affiche la vue dynamique, indiquant graphiquement les tâches qui se sont terminées et celles qu'il reste à exécuter. Vous pouvez également extraire la vue dynamique d'un flux de travail en cours d'exécution avec AWS Glue API. Pour de plus amples informations, veuillez consulter Interrogation des flux de travail à l'aide du AWS Glue API.

Consultez aussi

- the section called “Création d'un flux de travail à partir d'un plan”

- the section called “Création et développement d'un flux de travail manuellement”
- Flux de travail (pour l'API de flux de travail)

Création et développement d'un flux de travail manuellement dans AWS Glue

Vous pouvez utiliser la console AWS Glue pour créer et développer manuellement un flux de travail un nœud à la fois.

Un flux de travail contient des tâches, des crawlers et des déclencheurs. Avant de créer un flux de travail manuellement, créez les tâches et les crawlers que le flux de travail doit inclure. Il est préférable de spécifier des robots d' run-on-demandexploration pour les flux de travail. Vous pouvez créer de nouveaux déclencheurs lors du développement de de votre flux de travail ou vous pouvez cloner des déclencheurs existants dans le flux de travail. Lorsque vous clonez un déclencheur, tous les objets du catalogue associés au déclencheur (les tâches ou les crawlers qui le déclenchent et les tâches ou les crawlers qu'il démarre) sont ajoutés au flux de travail.

> **Important**
>
> Limitez le nombre total de tâches, de crawlers et de déclencheurs au sein d'un flux de travail à 100 ou moins. Si vous en incluez plus de 100, vous risquez de rencontrer des erreurs lorsque vous tentez de reprendre ou d'arrêter les exécutions du flux de travail.

Vous développez votre flux de travail en ajoutant des déclencheurs au graphique du flux de travail et en définissant les événements et les actions observés pour chaque déclencheur. Vous commencez par un déclencheur de début, qui peut être un déclencheur à la demande ou un déclencheur de planification, et complétez le graphique en ajoutant des déclencheurs d'événements (conditionnels).

Étape 1 : Créer le flux de travail

1. Connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/.
2. Dans le panneau de navigation, sous ETL, sélectionnez Workflows (Flux de travail).
3. Sélectionnez Add workflow (Ajouter un flux de travail) et remplissez le formulaire Add a new ETL workflow (Ajouter un nouveau flux de travail ETL).

Toutes les propriétés d'exécution par défaut facultatives que vous ajoutez sont rendues disponibles en tant qu'arguments à toutes les tâches du flux de travail. Pour de plus amples informations, veuillez consulter Obtention et définition des propriétés d'exécution du flux de travail dans AWS Glue.

4. Sélectionnez Add workflow (Ajouter un flux de travail).

   Le nouveau flux de travail s'affiche dans la liste sur la page Workflows (Flux de travail).

Étape 2 : Ajouter un déclencheur de début

1. Sur la page Workflows (Flux de travail), sélectionnez votre nouveau flux de travail. Ensuite, en bas de la page, assurez-vous que l'onglet Graph (Graphique) est sélectionné.
2. Choisissez Add trigger (Ajouter un déclencheur), puis, dans la boîte dedialogue Add trigger (Ajouter un déclencheur), effectuez l'une des actions suivantes :

   - Choisissez Clone existing (Clone existant), puis choisissez un déclencheur à cloner. Choisissez ensuite Ajouter.

     Le déclencheur s'affiche sur le graphique, ainsi que les tâches et les crawlers qu'il surveille, et les tâches et les crawlers qu'il démarre.

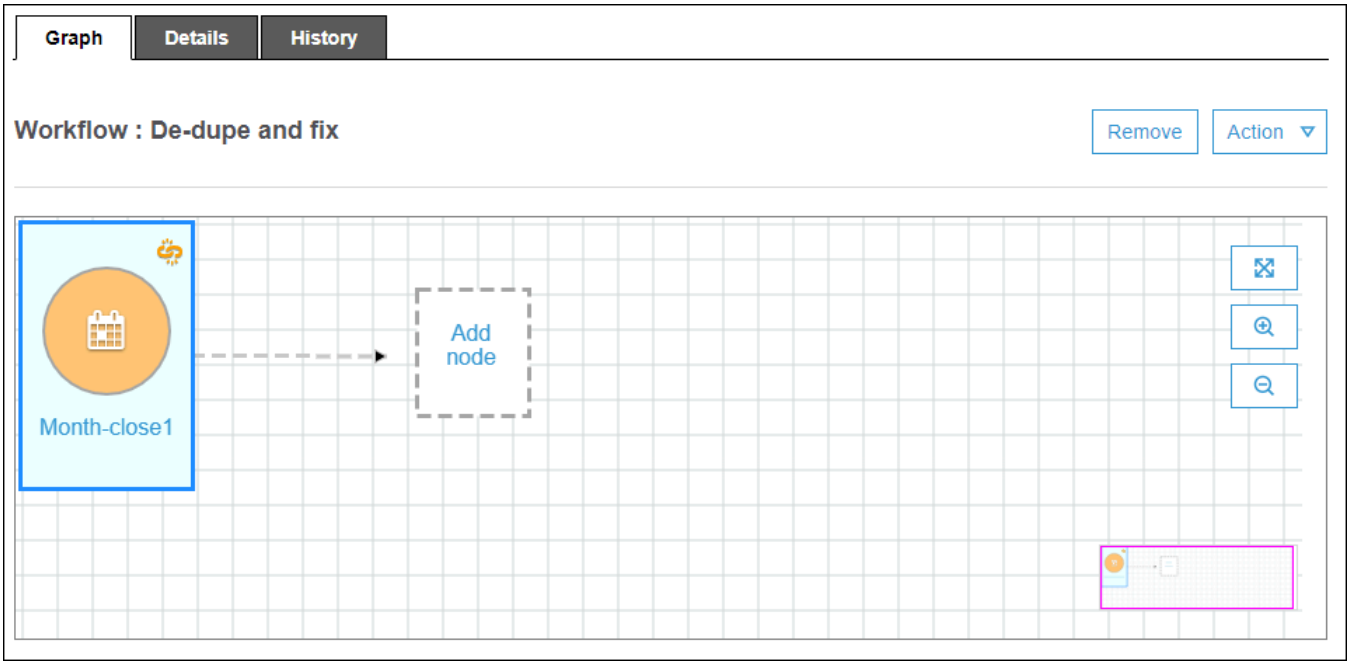
     Si vous avez sélectionné par erreur le mauvais déclencheur, sélectionnez le déclencheur sur le graphique, puis choisissez Remove (Supprimer).
   - Choisissez Add new (Ajouter un nouveau) et complétez le formulaire Add trigger (Ajouter un déclencheur).
     1. Pour le type de déclencheur, sélectionnez Planification, À la demande ou EventBridgeévénement.

        Pour le type de déclencheur Schedule (Planification), sélectionnez l'une des options Frequency (Fréquence). Sélectionnez Custom (Personnalisée) pour saisir une expression `cron`.

        Pour un EventBridge événement de type déclencheur, entrez Nombre d'événements (taille du lot) et entrez éventuellement Délai (fenêtre de lot). Si vous omettez Time delay (Délai), la fenêtre de lot est définie par défaut sur 15 minutes. Pour de plus amples informations, veuillez consulter Présentation des flux de travail dans AWS Glue.
     2. Choisissez Ajouter.

Le déclencheur s'affiche sur le graphique, ainsi qu'un nœud d'espace réservé (nommé Add node (Ajouter un nœud)). Dans l'exemple ci-dessous, le déclencheur de démarrage est un déclencheur de planification nommé `Month-close1`.

À ce stade, le déclencheur n'est pas encore enregistré.



3. Si vous avez ajouté un nouveau déclencheur, procédez comme suit :

   a. Effectuez l'une des actions suivantes :

      - Choisissez le nœud d'espace réservé (Add node (Ajouter un nœud)).
      - Assurez-vous que le déclencheur de démarrage est sélectionné, puis dans le menu Action au-dessus du graphique, choisissez Ajouter jobs/crawlers au déclencheur.

   b. Dans la boîte de dialogue Add jobs(s) and crawler(s) to trigger (Ajouter des tâches et des crawlers au déclencheur), sélectionnez une ou plusieurs tâches ou crawlers, puis sélectionnez Add (Ajouter).

      Le déclencheur est enregistré, et les tâches ou crawlers sélectionnés apparaissent sur le graphique avec les connecteurs du déclencheur.

      Si vous avez ajouté par erreur les mauvaises tâches ou les mauvais crawlers, vous pouvez sélectionner le déclencheur ou un connecteur et choisir Remove (Supprimer).

Étape 3 : Ajouter d'autres déclencheurs

Continuez de développer votre flux de travail en ajoutant d'autres déclencheurs de type Event (Événement). Pour faire un zoom avant ou arrière ou pour agrandir le canevas du graphique, utilisez les icônes à la droite du graphique. Pour chaque déclencheur à ajouter, effectuez les opérations suivantes :

> **Note**
>
> Il n'existe aucune action pour enregistrer le flux de travail. Après avoir ajouté votre dernier déclencheur et affecté des actions au déclencheur, le flux de travail est terminé et enregistré. Vous pouvez toujours revenir plus tard et ajouter plus de nœuds.

1. Effectuez l'une des actions suivantes :

   - Pour cloner un déclencheur existant, assurez-vous qu'aucun nœud sur le graphique ne soit sélectionné, et sur le menu Action, choisissez Add trigger (Ajouter un déclencheur).
   - Pour ajouter un nouveau déclencheur qui surveille une tâche particulière ou un crawler particulier sur le graphique, sélectionnez le nœud de la tâche ou de l'crawler, puis choisissez le nœud d'espace réservé Add trigger (Ajouter un déclencheur).

     Vous pouvez ajouter d'autres tâches ou crawlers à surveiller pour ce déclencheur dans une étape ultérieure.
2. Dans la boîte de dialogue Add trigger (Ajouter un déclencheur), exécutez l'une des actions suivantes :

   - Choisissez Add new (Ajouter un nouveau) et complétez le formulaire Add trigger (Ajouter un déclencheur). Choisissez ensuite Ajouter.

     Le déclencheur s'affiche sur le graphique. Vous complèterez le déclencheur lors d'une étape ultérieure.
   - Choisissez Clone existing (Clone existant), puis choisissez un déclencheur à cloner. Choisissez ensuite Ajouter.

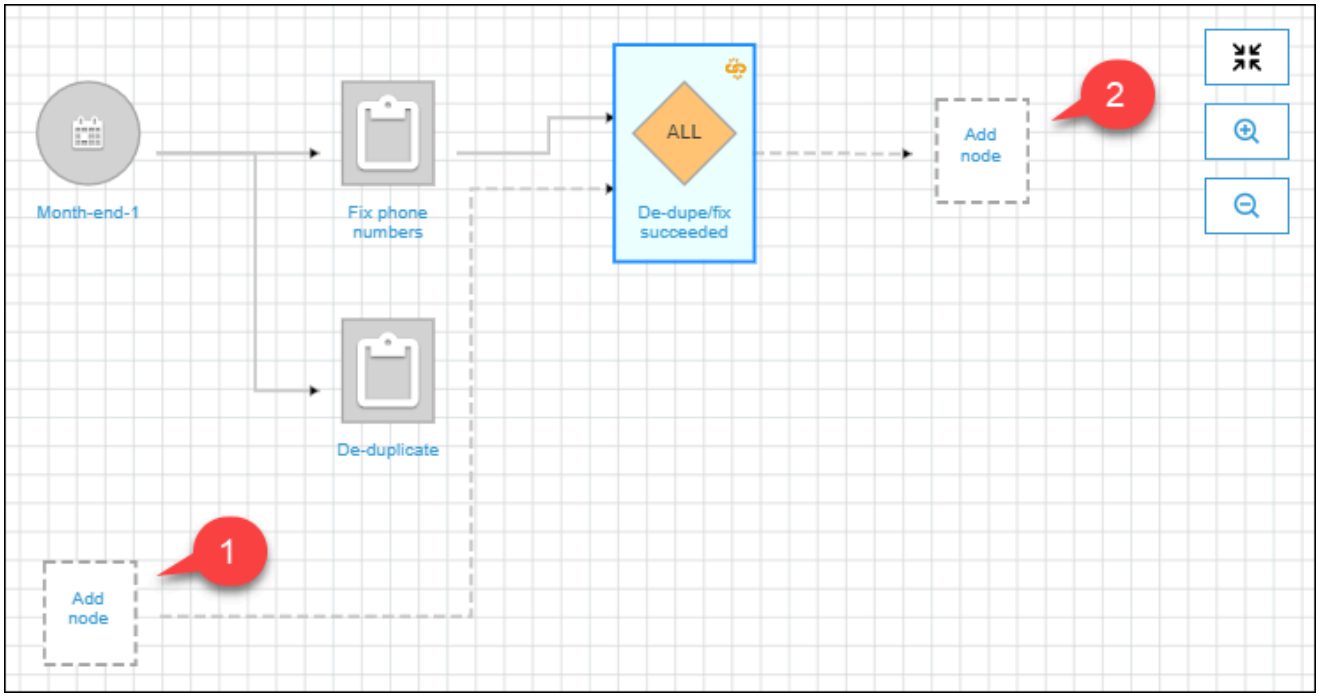
     Le déclencheur s'affiche sur le graphique, ainsi que les tâches et les crawlers qu'il surveille, et les tâches et les crawlers qu'il démarre.

Si vous avez choisi par erreur le mauvais déclencheur, sélectionnez le déclencheur sur le graphique, puis choisissez Remove (Supprimer).

3. Si vous avez ajouté un nouveau déclencheur, procédez comme suit :

   a. Sélectionnez le nouveau déclencheur.

      Comme l'illustre le graphique suivant, le déclencheur `De-dupe/fix succeeded` est sélectionné et les nœuds d'espace réservé apparaissent pour (1) les événements à surveiller et pour (2) les actions.



   b. (Facultatif si le déclencheur surveille déjà un événement et que vous souhaitez ajouter davantage de tâches ou d'crawlers à surveiller.) Choisissez le nœud events-to-watch réservé, puis dans la boîte de dialogue Ajouter des tâches et des robots à surveiller, sélectionnez un ou plusieurs travaux ou robots d'exploration. Choisissez un événement à surveiller (SUCCEEDED, FAILED, etc.), puis choisissez Add (Ajouter).

   c. Assurez-vous que le déclencheur est sélectionné, puis choisissez le nœud d'espace réservé des actions.

   d. Dans la boîte de dialogue Add job(s) and crawler(s) to watch (Ajouter des tâches et des crawlers à surveiller) sélectionnez une ou plusieurs tâches ou un ou plusieurs crawlers, et choisissez Add (Ajouter).

Les tâches et les crawlers sélectionnés apparaissent sur le graphique, avec les connecteurs du déclencheur.

Pour en savoir plus sur les flux de travail et les plans, consultez les rubriques suivantes.

- Présentation des flux de travail dans AWS Glue
- Exécution et surveillance d'un flux de travail dans AWS Glue
- Création d'un flux de travail à partir d'un plan dans AWS Glue

Démarrer un AWS Glue flux de travail avec un EventBridge événement Amazon

Amazon EventBridge, également connu sous le nom d' CloudWatch Events, vous permet d'automatiser vos AWS services et de répondre automatiquement aux événements du système tels que les problèmes de disponibilité des applications ou les modifications des ressources. Les événements AWS liés aux services sont diffusés EventBridge en temps quasi réel. Vous pouvez écrire des règles simples pour indiquer quels événements vous intéressent et les actions automatisées à effectuer quand un événement correspond à une règle.

Avec du EventBridge support, il AWS Glue peut servir de producteur et de consommateur d'événements dans une architecture axée sur les événements. Pour les flux de travail, AWS Glue prend en charge tout type d' EventBridge événement en tant que consommateur. Le cas d'utilisation le plus courant est l'arrivée d'un nouvel objet dans un compartiment Amazon S3. Si vous avez des données arrivant à des intervalles irréguliers ou indéfinis, vous pouvez traiter ces données aussi près que possible de leur arrivée.

> **Note**
>
> AWS Gluene garantit pas la livraison des EventBridge messages. AWS Gluen'effectue aucune déduplication s'il EventBridge délivre des messages dupliqués. Vous devez gérer l'idempotence en fonction de votre cas d'utilisation.
> Assurez-vous de configurer correctement EventBridge les règles pour éviter d'envoyer des événements indésirables.

Avant de commencer

Si vous souhaitez démarrer un flux de travail avec des événements de données Amazon S3, vous devez vous assurer que les événements du compartiment S3 qui vous intéresse sont enregistrés dans AWS CloudTrail et EventBridge. Pour ce faire, vous devez créer un CloudTrail parcours. Pour plus d'informations, consultez la section Création d'un historique pour votre AWS compte.

Pour démarrer un flux de travail avec un EventBridge événement

> **Note**
>
> Dans les commandes suivantes, remplacez :
>
> - *<workflow-name>*avec le nom à attribuer au flux de travail.
> - *<trigger-name>*avec le nom à attribuer au déclencheur.
> - *<bucket-name>*avec le nom du compartiment Amazon S3.
> - *<account-id>*avec un identifiant de AWS compte valide.
> - *<region>*avec le nom de la région (par exemple,`us-east-1`).
> - *<rule-name>*avec le nom à attribuer à la EventBridge règle.

1. Assurez-vous que vous disposez des autorisations Gestion des identités et des accès AWS (IAM) nécessaires pour créer et afficher les EventBridge règles et les cibles. Voici un modèle de politique de clé que vous pouvez attacher. Vous voudrez peut-être restreindre sa portée afin d'imposer des limites aux opérations et aux ressources.

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "events:PutRule",
        "events:DisableRule",
        "events:DeleteRule",
        "events:PutTargets",
        "events:RemoveTargets",
        "events:EnableRule",
        "events:List*",
```

```
        "events:Describe*"
      ],
      "Resource": "*"
    }
  ]
}
```

2. Créez un rôle IAM que le EventBridge service peut assumer lorsqu'il transmet un événement àAWS Glue.

   a. Sur la page Création d'un rôle de la console IAM, sélectionnez Service AWS . Choisissez ensuite le service CloudWatch Events.

   b. Suivez intégralement l'assistant de Création d'un rôle. L'Assistant attache automatiquement les politiques `CloudWatchEventsBuiltInTargetExecutionAccess` et `CloudWatchEventsInvocationAccess`.

   c. Attachez la politique en ligne suivante au rôle. Cette politique permet au EventBridge service de diriger les événements versAWS Glue.

      JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:notifyEvent"
      ],
      "Resource": [
        "arn:aws:glue:us-east-1:111122223333:workflow/workflow-name"
      ]
    }
  ]
}
```

3. Saisissez la commande suivante pour créer le flux de travail.

   Veuillez consulter la rubrique create-workflow dans la Référence des commandes AWS CLI pour plus d'informations sur les paramètres de ligne de commande facultatifs supplémentaires.

```
aws glue create-workflow --name <workflow-name>
```

4. Entrez la commande suivante pour créer un déclencheur d' EventBridge événement pour le flux de travail. Ce sera le déclencheur de démarrage du flux de travail. Remplacez *<actions>* par les actions à effectuer (les tâches et les robots d'exploration pour démarrer).

   Veuillez consulter la rubrique create-triggerdans la Référence des commandes AWS CLI pour obtenir des informations sur la manière de coder les arguments `actions`.

   ```
   aws glue create-trigger --workflow-name <workflow-name> --type EVENT --
   name <trigger-name> --actions <actions>
   ```

   Si vous souhaitez que le flux de travail soit déclenché par un lot d'événements plutôt que par un seul EventBridge événement, entrez plutôt la commande suivante.

   ```
   aws glue create-trigger --workflow-name <workflow-name> --type EVENT
    --name <trigger-name> --event-batching-condition BatchSize=<number-of-
   events>,BatchWindow=<seconds> --actions <actions>
   ```

   En ce qui concerne les arguments `event-batching-condition`, `BatchSize` est obligatoire et `BatchWindow` est facultatif. Si `BatchWindow` est omis, la fenêtre est définie par défaut à 900 secondes, ce qui correspond à la taille maximale de la fenêtre.

   Example

   L'exemple suivant crée un déclencheur qui démarre le `eventtest` flux de travail après l'arrivée de trois EventBridge événements, ou cinq minutes après l'arrivée du premier événement, selon la première éventualité.

   ```
   aws glue create-trigger --workflow-name eventtest --type EVENT --name objectArrival
    --event-batching-condition BatchSize=3,BatchWindow=300 --actions JobName=test1
   ```

5. Créez une règle sur Amazon EventBridge.

   a. Créer l'objet JSON définissant les détails de la règle dans votre éditeur de texte préféré.

      L'exemple suivant spécifie Amazon S3 comme source d'événement, `PutObject` comme nom de l'événement, et le nom du compartiment en tant que paramètre de requête. Cette règle démarre un flux de travail lorsqu'un nouvel objet arrive dans le compartiment.

```
{
  "source": [
    "aws.s3"
  ],
  "detail-type": [
    "AWS API Call via CloudTrail"
  ],
  "detail": {
    "eventSource": [
      "s3.amazonaws.com"
    ],
    "eventName": [
      "PutObject"
    ],
    "requestParameters": {
      "bucketName": [
        "<bucket-name>"
      ]
    }
  }
}
```

Pour démarrer le flux de travail lorsqu'un nouvel objet arrive dans un dossier du compartiment, vous pouvez introduire le code suivant dans `requestParameters`.

```
    "requestParameters": {
      "bucketName": [
        "<bucket-name>"
      ]
      "key" : [{ "prefix" : "<folder1>/<folder2>/*"}}]
  }
```

b. Utilisez votre outil préféré pour convertir l'objet JSON de définition de règle en une chaîne échappée.

```
{\n  \"source\": [\n    \"aws.s3\"\n  ],\n  \"detail-type\": [\n    \"AWS API
 Call via CloudTrail\"\n  ],\n  \"detail\": {\n    \"eventSource\": [\n
 \"s3.amazonaws.com\"\n    ],\n    \"eventName\": [\n      \"PutObject\"\n
  ],\n    \"requestParameters\": {\n      \"bucketName\": [\n        \"<bucket-
name>\"\n      ]\n    }\n  }\n}
```

c. Exécutez la commande suivante pour créer un modèle de paramètre JSON que vous pouvez modifier afin de spécifier les paramètres d'entrée d'une commande `put-rule` ultérieure. Enregistrez le résultat dans un fichier. Dans cet exemple, le fichier est nommé `ruleCommand`.

```
aws events put-rule --name <rule-name> --generate-cli-skeleton >ruleCommand
```

Pour de plus amples informations sur le paramètre `--generate-cli-skeleton`, veuillez consulter la rubrique Génération du squelette AWS CLI et des paramètres d'entrée à partir d'un fichier d'entrée JSON ou YAML dans le Guide de l'utilisateur de l'interface de ligne de commande AWS .

Le fichier de sortie devrait ressembler à ce qui suit.

```
{
    "Name": "",
    "ScheduleExpression": "",
    "EventPattern": "",
    "State": "ENABLED",
    "Description": "",
    "RoleArn": "",
    "Tags": [
        {
            "Key": "",
            "Value": ""
        }
    ],
    "EventBusName": ""
}
```

d. Modifiez le fichier pour supprimer éventuellement des paramètres et pour spécifier au minimum les paramètres `Name`, `EventPattern` et `State`. Pour le paramètre `EventPattern`, fournissez la chaîne d'échappement des détails de règle que vous avez créé à l'étape précédente.

```
{
    "Name": "<rule-name>",
    "EventPattern": "{\n  \"source\": [\n    \"aws.s3\"\n  ],\n  \"detail-
type\": [\n    \"AWS API Call via CloudTrail\"\n  ],\n  \"detail\": {\n
 \"eventSource\": [\n      \"s3.amazonaws.com\"\n    ],\n    \"eventName\": [\n
```

```
      \"PutObject\"\n    ],\n    \"requestParameters\": {\n      \"bucketName
\": [\n        \"<bucket-name>\"\n      ]\n    }\n  }\n}",
    "State": "DISABLED",
    "Description": "Start an AWS Glue workflow upon new file arrival in an
 Amazon S3 bucket"
}
```

> **Note**
>
> Il est préférable de laisser la règle désactivée jusqu'à ce que vous ayez terminé la création du flux de travail.

e. Saisissez la commande `put-rule` suivante, qui lit les paramètres d'entrée dans le fichier `ruleCommand`.

```
aws events put-rule --name <rule-name> --cli-input-json file://ruleCommand
```

La sortie suivante indique que l'opération a réussi.

```
{
    "RuleArn": "<rule-arn>"
}
```

6. Saisissez la commande suivante pour attacher la règle à une cible. La cible est le flux de travail dans AWS Glue. *<role-name>*Remplacez-le par le rôle que vous avez créé au début de cette procédure.

```
aws events put-targets --rule <rule-name> --targets
 "Id"="1","Arn"="arn:aws:glue:<region>:<account-id>:workflow/<workflow-
name>","RoleArn"="arn:aws:iam::<account-id>:role/<role-name>" --region <region>
```

La sortie suivante indique que l'opération a réussi.

```
{
    "FailedEntryCount": 0,
    "FailedEntries": []
}
```

7. Confirmez la connexion réussie de la règle et de la cible en saisissant la commande suivante.

```
aws events list-rule-names-by-target --target-arn arn:aws:glue:<region>:<account-id>:workflow/<workflow-name>
```

Le résultat suivant indique un succès, où *<rule-name>* est le nom de la règle que vous avez créée.

```
{
    "RuleNames": [
        "<rule-name>"
    ]
}
```

8. Connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/.
9. Sélectionnez le flux de travail et vérifiez que le déclencheur de démarrage et ses actions (les tâches ou les crawlers qu'il démarre) apparaissent sur le graphique du flux de travail. Procédez ensuite comme indiqué dans Étape 3 : Ajouter d'autres déclencheurs. Vous pouvez également ajouter d'autres composants au flux de travail en utilisant l'API AWS Glue ou AWS Command Line Interface.
10. Lorsque le flux de travail est entièrement défini, activez la règle.

```
aws events enable-rule --name <rule-name>
```

Le flux de travail est maintenant prêt à être lancé par un EventBridge événement ou un lot d'événements.

Consultez aussi

- Guide de EventBridge l'utilisateur Amazon
- Présentation des flux de travail dans AWS Glue
- Création et développement d'un flux de travail manuellement dans AWS Glue

Afficher les EventBridge événements qui ont lancé un flux de travail

Vous pouvez consulter l'ID de l' EventBridge événement Amazon qui a lancé votre flux de travail. Si votre flux de travail a été lancé par un lot d'événements, vous pouvez afficher l'événement IDs de tous les événements du lot.

Pour les flux de travail dont la taille de lot est supérieure à un, vous pouvez également voir quelle condition de lot a démarré le flux de travail : l'arrivée du nombre d'événements dans la taille du lot ou l'expiration de la fenêtre de lot.

Pour afficher les EventBridge événements qui ont lancé un flux de travail (console)

1. Connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/.
2. Dans le panneau de navigation, sous ETL, sélectionnez Workflows (Flux de travail).
3. Sélectionnez un flux de travail. Puis, en bas, sélectionnez l'onglet History (Historique).
4. Sélectionnez un flux de travail, puis sélectionnez View run details (Afficher les détails de l'exécution).
5. Sur la page de détails de l'exécution, repérez le champ Run properties (Propriétés d'exécution), et recherchez la clé aws:eventIds.

   La valeur de cette clé est une liste d' EventBridge événements IDs.

Pour afficher les EventBridge événements qui ont démarré un flux de travail (AWS API)

- Incluez le code suivant dans votre script Python.

```
workflow_params =
 glue_client.get_workflow_run_properties(Name=workflow_name,RunId=workflow_run_id)
batched_events = workflow_params['aws:eventIds']
```

  `batched_events` sera une liste de chaînes, chaque chaîne étant un ID d'événement.

Consultez aussi

- Guide de EventBridge l'utilisateur Amazon

- the section called "Présentation des flux de travail"

Exécution et surveillance d'un flux de travail dans AWS Glue

Si le déclencheur de début d'un flux de travail est un déclencheur à la demande, vous pouvez démarrer le flux de travail à partir de la console AWS Glue. Procédez comme suit pour exécuter et surveiller un flux de travail. Si le flux de travail échoue, vous pouvez afficher le graphique d'exécution pour déterminer le nœud qui a échoué. Pour faciliter le dépannage, si le flux de travail a été créé à partir d'un modèle, vous pouvez afficher l'exécution ce dernier pour voir les valeurs des paramètres de modèle qui ont été utilisées pour créer le flux de travail. Pour de plus amples informations, veuillez consulter the section called "Affichage des exécutions de plans".

Vous pouvez exécuter et surveiller un flux de travail à l'aide de la console AWS Glue, l'API ou AWS Command Line Interface (AWS CLI).

Pour exécuter et surveiller un flux de travail (console)

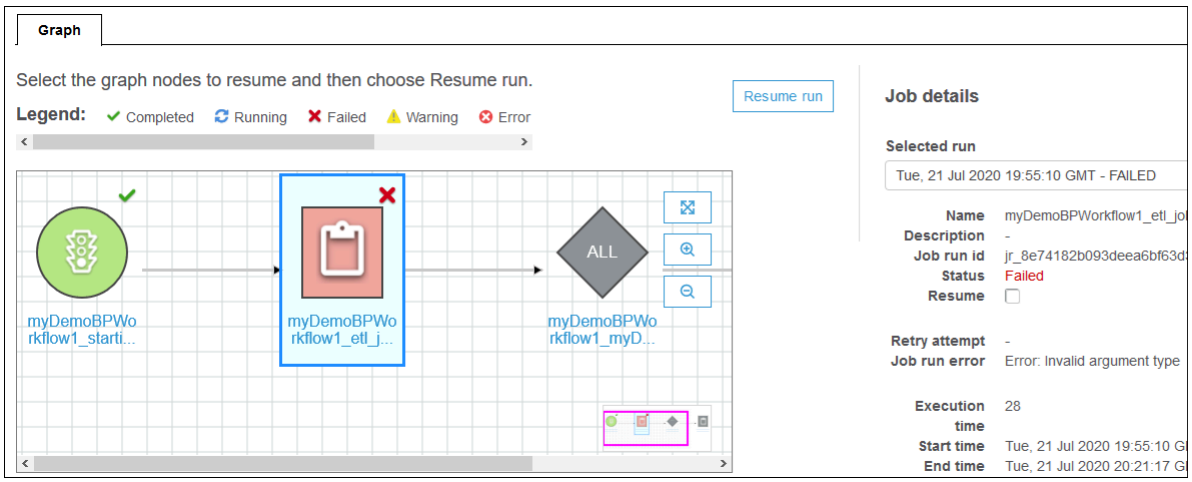
1. Ouvrez la AWS Glue console à l'adresse https://console.aws.amazon.com/glue/.
2. Dans le panneau de navigation, sous ETL, sélectionnez Workflows (Flux de travail).
3. Sélectionnez un flux de travail. Dans le menu Actions, choisissez Run (Exécuter).
4. Vérifiez la colonne Last run status (Statut de la dernière exécution) dans la liste des flux de travail. Cliquez sur le bouton Refresh (Actualiser) pour afficher le statut du flux de travail en cours.
5. Pendant l'exécution du flux de travail ou une fois qu'il est terminé (ou a échoué), affichez les détails de l'exécution en effectuant les étapes suivantes.

   a. Assurez-vous que le flux de travail est sélectionné, puis sélectionnez l'onglet History (Historique).

   b. Choisissez l'exécution de flux de travail actuelle ou la plus récente, puis sélectionnez View run details (Afficher les détails de l'exécution).

      Le graphique d'exécution du flux de travail affiche l'état d'exécution actuel.

   c. Choisissez n'importe quel nœud dans le graphique pour afficher ses détails et son état.



Pour exécuter et surveiller un flux de travail (AWS CLI)

1. Entrez la commande suivante. Remplacez *<workflow-name>* par le flux de travail à exécuter.

```
aws glue start-workflow-run --name <workflow-name>
```

   Si le flux de travail est démarré avec succès, la commande renvoie l'ID d'exécution.

2. Affichez le statut d'exécution du flux de travail à l'aide de la commande `get-workflow-run`. Indiquez le nom et l'ID d'exécution du flux de travail.

```
aws glue get-workflow-run --name myWorkflow --run-id
 wr_d2af14217e8eae775ba7b1fc6fc7a42c795aed3cbcd8763f9415452e2dbc8705
```

   Voici un exemple de sortie de commande.

```
{
    "Run": {
        "Name": "myWorkflow",
        "WorkflowRunId":
 "wr_d2af14217e8eae775ba7b1fc6fc7a42c795aed3cbcd8763f9415452e2dbc8705",
        "WorkflowRunProperties": {
            "run_state": "COMPLETED",
            "unique_id": "fee63f30-c512-4742-a9b1-7c8183bdaae2"
        },
        "StartedOn": 1578556843.049,
        "CompletedOn": 1578558649.928,
        "Status": "COMPLETED",
```

```
        "Statistics": {
            "TotalActions": 11,
            "TimeoutActions": 0,
            "FailedActions": 0,
            "StoppedActions": 0,
            "SucceededActions": 9,
            "RunningActions": 0,
            "ErroredActions": 0
        }
    }
}
```

Voir aussi :

- the section called “Présentation des flux de travail”
- the section called “Présentation des plans”

Arrêt d'une exécution de flux de travail

Vous pouvez utiliser la AWS Glue console AWS Command Line Interface (AWS CLI) ou l'AWS GlueAPI pour arrêter l'exécution d'un flux de travail. Lorsque vous arrêtez un cycle de flux de travail, toutes les tâches et tous les crawlers en cours d'exécution sont immédiatement arrêtés, tandis que ceux qui n'ont pas encore été lancés ne démarreront jamais. Cela peut prendre jusqu'à une minute pour que toutes les tâches et tous les crawlers en cours d'exécution s'arrêtent. L'état du cycle de flux de travail passe de Running (Exécution en cours) à Stopping (Arrêt en cours), puis lorsque le cycle de flux de travail est complètement arrêté, l'état passe à Stopped (Arrêté).

Une fois le cycle de flux de travail arrêté, vous pouvez afficher le graphique d'exécution pour voir les tâches et les crawlers abandonnés, ainsi que ceux qui n'ont jamais démarré. Vous pouvez ensuite déterminer si vous devez prendre des mesures spécifiques pour garantir l'intégrité des données. L'arrêt d'un cycle de flux de travail n'entraîne aucune opération de restauration automatique.

Pour arrêter l'exécution d'un flux de travail (console)

1. Ouvrez la AWS Glue console à l'adresse https://console.aws.amazon.com/glue/.
2. Dans le panneau de navigation, sous ETL, sélectionnez Workflows (Flux de travail).
3. Choisissez un flux de travail en cours d'exécution, puis choisissez l'onglet History (Historique).

4. Choisissez le cycle de flux de travail, puis choisissez Stop run (Arrêter ce cycle).

   L'état d'exécution passe à Stopping (Arrêt en cours).

5. (Facultatif) Choisissez le cycle de flux de travail, puis (Optional) Choose the workflow run, choose (Afficher les détails de l'exécution) et consultez le graphique d'exécution.

Pour arrêter un cycle de flux de travail (AWS CLI)

- Entrez la commande suivante. Remplacez *<workflow-name>* par le nom du flux de travail et *<run-id>* par l'ID d'exécution du flux de travail à arrêter.

```
aws glue stop-workflow-run --name <workflow-name> --run-id <run-id>
```

  Voici un exemple de la commande stop-workflow-run.

```
aws glue stop-workflow-run --name my-workflow --run-id
 wr_137b88917411d128081069901e4a80595d97f719282094b7f271d09576770354
```

Réparation et reprise de l'exécution d'un flux de travail

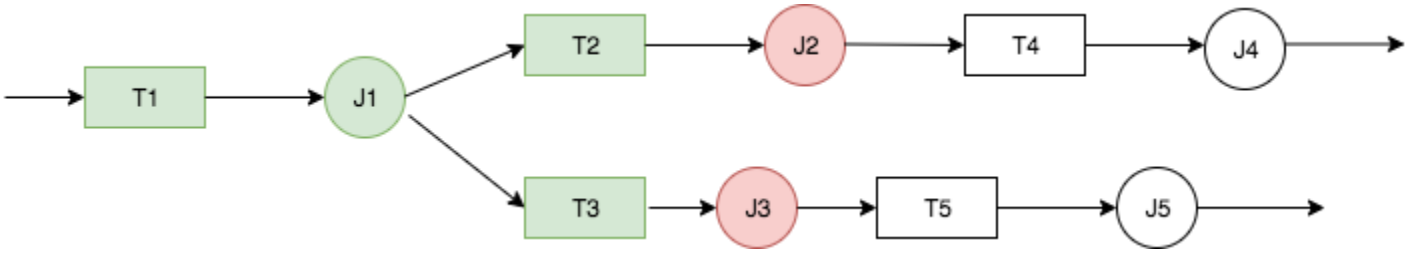
Si un ou plusieurs nœuds (tâches ou crawlers) d'un flux de travail ne se terminent pas correctement, cela signifie que le flux de travail n'a été exécuté que partiellement. Après avoir trouvé les causes premières et apporté des corrections, vous pouvez sélectionner un ou plusieurs nœuds à partir desquels reprendre l'exécution du flux de travail, puis reprendre celle-ci. Les nœuds sélectionnés et tous les nœuds en aval de ces nœuds sont ensuite exécutés.

Rubriques

- Reprise d'une exécution de flux de travail : comment ça marche
- Reprise de l'exécution d'un flux de travail
- Remarques et limitations pour la reprise des exécutions de flux de travail

Reprise d'une exécution de flux de travail : comment ça marche

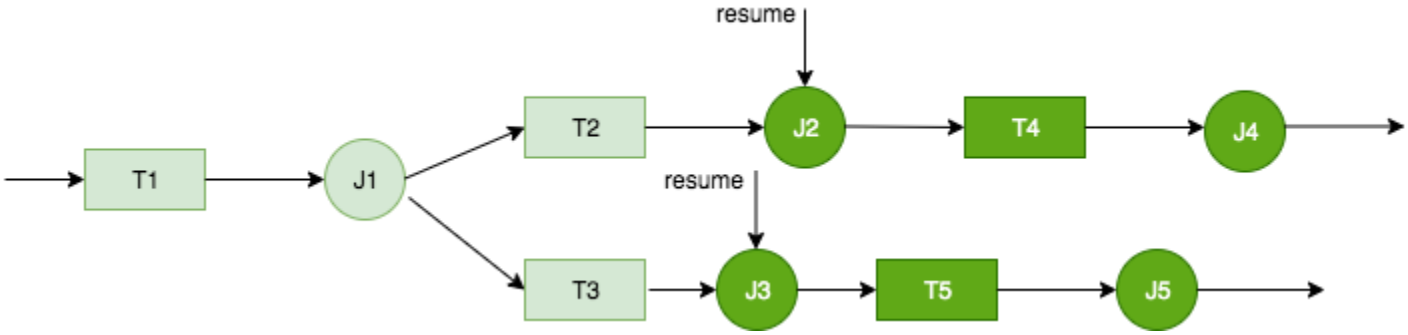
Examinez le flux de travail W1 dans le diagramme suivant.



L'exécution de flux de travail se déroule comme suit :

1. Le déclencheur T1 démarre la tâche J1.
2. La réussite de la tâche J1 déclenche T2 et T3, qui exécutent les tâches J2 et J3, respectivement.
3. Les tâches J2 et J3 échouent.
4. Les déclencheurs T4 et T5 dépendent de la réussite de J2 et J3, ils ne se déclenchent donc pas et les tâches J4 et J5 ne s'exécutent pas. Le flux de travail W1 n'est exécuté que partiellement.

À présent, supposons que les problèmes qui ont causé l'échec de J2 et J3 sont corrigés. J2 et J3 sont sélectionnés comme points de départ à partir desquels reprendre l'exécution du flux de travail.



L'exécution de flux de travail reprend comme suit :

1. Les tâches J2 et J3 sont exécutées avec succès.
2. Déclenche T4 et T5.
3. Les tâches J4 et J5 sont exécutées avec succès.

La reprise de l'exécution du flux de travail est suivie en tant qu'exécution de flux de travail distincte avec un nouvel ID d'exécution. Lorsque vous affichez l'historique du flux de travail, vous pouvez afficher l'ID d'exécution précédente pour toute exécution de flux de travail. Dans l'exemple de la capture d'écran suivante, le flux de travail exécuté avec l'ID d'exécution `wr_c7a22...` (la deuxième

ligne) avait un nœud qui ne s'est pas terminé. L'utilisateur a résolu le problème et a repris l'exécution du flux de travail, ce qui a entraîné l'ID d'exécution `wr_a07e55...` (la première ligne).

> **Note**
>
> Pour le reste de cette discussion, le terme « reprise de l'exécution du flux de travail » fait référence à l'exécution du flux de travail qui a été créée lors de la reprise de l'exécution du flux de travail précédente. L'« exécution du flux de travail d'origine » fait référence à l'exécution du flux de travail qui n'a été exécutée que partiellement et qui devait être reprise.

Graphique de la reprise d'une l'exécution du flux de travail

Dans une exécution de flux de travail reprise, bien que seul un sous-ensemble de nœuds soit exécuté, le graphique d'exécution est un graphique complet. Autrement dit, les nœuds qui ne se sont pas exécutés dans le flux de travail repris sont copiés à partir du graphique d'exécution de l'exécution du flux de travail d'origine. Les nœuds de tâche et d'crawler copiés qui ont été exécutés dans l'exécution du flux de travail d'origine incluent les détails de l'exécution.

Considérez à nouveau le flux de travail W1 dans le diagramme précédent. Lorsque l'exécution du flux de travail reprend en commençant par J2 et J3, le graphique d'exécution de l'exécution de flux de travail reprise affiche tous les travaux, J1 à J5, et tous les déclencheurs, T1 à T5. Les détails de l'exécution pour J1 sont copiés à partir de l'exécution du flux de travail d'origine.

Instantanés d'exécution de flux de travail

Lorsqu'une exécution de flux de travail est démarrée, AWS Glue prend un instantané du graphique de conception de flux de travail à ce moment-là. Cet instantané est utilisé pendant toute la durée de l'exécution du flux de travail. Si vous apportez des modifications à des déclencheurs après le démarrage de l'exécution, ces modifications n'affectent pas l'exécution du flux de travail en cours. Les instantanés garantissent que les exécutions du flux de travail se déroulent de manière cohérente.

Les instantanés rendent les déclencheurs inaltérables uniquement. Les modifications que vous apportez aux tâches en aval et aux crawlers pendant l'exécution du flux de travail prennent effet pour l'exécution en cours.

Reprise de l'exécution d'un flux de travail

Procédez comme suit pour reprendre l'exécution d'un flux de travail. Vous pouvez reprendre l'exécution d'un flux de travail à l'aide de la AWS Glue console, de l'API ou AWS Command Line Interface (AWS CLI).

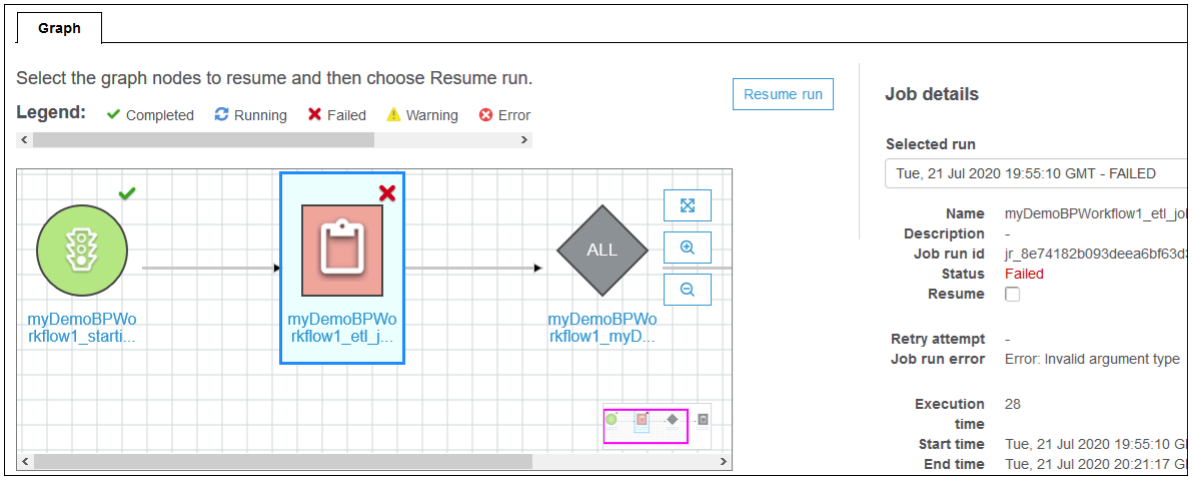
Pour reprendre l'exécution d'un flux de travail (console)

1. Ouvrez la AWS Glue console à l'adresse https://console.aws.amazon.com/glue/.

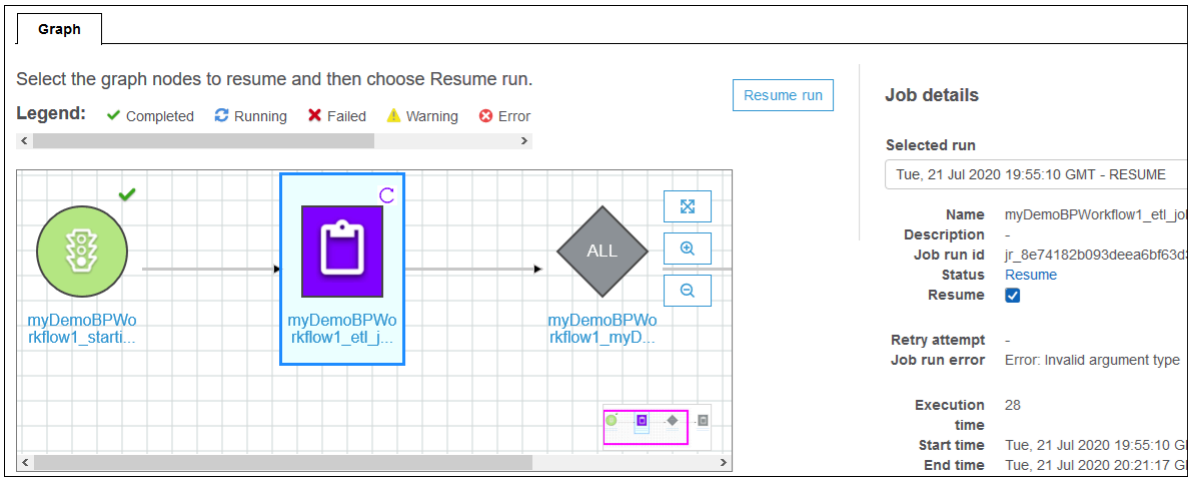
   Connectez-vous en tant qu'utilisateur autorisé à afficher les flux de travail et à reprendre les exécutions de flux de travail.

   > **Note**
   >
   > Pour reprendre les exécutions de flux de travail, vous devez disposer de l'autorisation `glue:ResumeWorkflowRun` Gestion des identités et des accès AWS (IAM).

2. Dans le panneau de navigation, sous ETL, sélectionnez Workflows (Flux de travail).
3. Sélectionnez un flux de travail, puis l'onglet History (Historique).
4. Sélectionnez l'exécution du flux de travail qui n'a été exécutée que partiellement, puis sélectionnez View run details (Afficher les détails de l'exécution).
5. Dans le graphique d'exécution, sélectionnez le premier (ou le seul) nœud que vous souhaitez redémarrer et à partir duquel vous souhaitez reprendre l'exécution du flux de travail.
6. Dans le volet de détails à droite du graphique, cochez la case Resume (Reprendre).



Le nœud change de couleur et affiche une petite icône de CV en haut à droite.



7. Effectuez les deux étapes précédentes pour redémarrer tous les nœuds supplémentaires.
8. Sélectionnez Resume (Reprendre l'exécution).

Pour reprendre l'exécution d'un flux de travail (AWS CLI)

1. Assurez-vous de disposer de l'autorisation IAM `glue:ResumeWorkflowRun`.
2. Récupérez le nœud IDs pour les nœuds que vous souhaitez redémarrer.

   a. Exécutez la commande `get-workflow-run` pour l'exécution du flux de travail d'origine. Fournissez le nom du flux de travail et l'ID d'exécution, puis ajoutez l'option `--include-graph`, comme illustré dans l'exemple suivant. Obtenez l'ID d'exécution dans l'onglet History (Historique) de la console ou en exécutant la commande `get-workflow`.

```
aws glue get-workflow-run --name cloudtrailtest1 --run-id
 wr_a07e55f2087afdd415a404403f644a4265278f68b13ba3da08c71924ebe3c3a8 --include-
graph
```

La commande renvoie les nœuds et les bords du graphique sous la forme d'un objet JSON volumineux.

b. Localisez les nœuds qui vous intéressent avec les propriétés `Type` et `Name` des objets de nœud.

Ce qui suit est un exemple d'objet nœud de la sortie.

```
{
    "Type": "JOB",
    "Name": "test1_post_failure_4592978",
    "UniqueId":
 "wnode_d1b2563c503078b153142ee76ce545fe5ceef66e053628a786ddd74a05da86fd",
    "JobDetails": {
        "JobRuns": [
            {
                "Id":
 "jr_690b9f7fc5cb399204bc542c6c956f39934496a5d665a42de891e5b01f59e613",
                "Attempt": 0,
                "TriggerName": "test1_aggregate_failure_649b2432",
                "JobName": "test1_post_failure_4592978",
                "StartedOn": 1595358275.375,
                "LastModifiedOn": 1595358298.785,
                "CompletedOn": 1595358298.785,
                "JobRunState": "FAILED",
                "PredecessorRuns": [],
                "AllocatedCapacity": 0,
                "ExecutionTime": 16,
                "Timeout": 2880,
                "MaxCapacity": 0.0625,
                "LogGroupName": "/aws-glue/python-jobs"
            }
        ]
    }
}
```

c. Obtenez l'ID de nœud à partir de la propriété `UniqueId` de l'objet nœud.

3. Exécutez la commande `resume-workflow-run`. Indiquez le nom du flux de travail, l'ID d'exécution et la liste des nœuds IDs séparés par des espaces, comme indiqué dans l'exemple suivant.

```
aws glue resume-workflow-run --name cloudtrailtest1 --run-id
 wr_a07e55f2087afdd415a404403f644a4265278f68b13ba3da08c71924ebe3c3a8 --node-
ids wnode_ca1f63e918fb855e063aed2f42ec5762ccf71b80082ae2eb5daeb8052442f2f3
 wnode_d1b2563c503078b153142ee76ce545fe5ceef66e053628a786ddd74a05da86fd
```

La commande génère l'ID d'exécution de la (nouvelle) exécution de flux de travail reprise et une liste de nœuds qui seront démarrés.

```
{
    "RunId": "wr_2ada0d3209a262fc1156e4291134b3bd643491bcfb0ceead30bd3e4efac24de9",
    "NodeIds": [
        "wnode_ca1f63e918fb855e063aed2f42ec5762ccf71b80082ae2eb5daeb8052442f2f3"
    ]
}
```

Notez que bien que l'exemple de commande `resume-workflow-run` répertorie deux nœuds à redémarrer, l'exemple de sortie indiquait qu'un seul nœud serait redémarré. C'est parce qu'un nœud était en aval de l'autre nœud, et le nœud en aval serait redémarré de toute façon par le flux normal du flux de travail.

Remarques et limitations pour la reprise des exécutions de flux de travail

Gardez les notes et limitations suivantes à l'esprit lors de la reprise des exécutions du flux de travail.

- Vous pouvez reprendre l'exécution d'un flux de travail uniquement s'il est dans l'état `COMPLETED`.

> **Note**
>
> Même si un ou plusieurs nœuds d'une exécution de flux de travail ne se terminent pas, l'état d'exécution du flux de travail est affiché sous la forme `COMPLETED`. Assurez-vous de vérifier le graphique d'exécution pour découvrir tous les nœuds qui ne se sont pas terminés avec succès.

- Vous pouvez reprendre une exécution de flux de travail à partir de n'importe quel tâche ou nœud d'crawler que l'exécution de flux de travail d'origine a tenté d'exécuter. Vous ne pouvez pas reprendre un flux de travail exécuté à partir d'un nœud déclencheur.
- Le redémarrage d'un nœud ne réinitialise pas son état. Toutes les données partiellement traitées ne sont pas annulées.
- Vous pouvez reprendre plusieurs fois l'exécution d'un flux de travail ayant échoué. Toutefois, une reprise ne peut être reprise qu'une seule fois. Pour des tentatives supplémentaires, reprenez plutôt l'exécution initiale qui a échoué
- Si vous sélectionnez deux nœuds à redémarrer et qu'ils dépendent l'un de l'autre, le nœud en amont est exécuté avant le nœud en aval. En fait, la sélection du nœud en aval est redondante, car elle sera exécutée selon le flux normal du flux de travail.

Obtention et définition des propriétés d'exécution du flux de travail dans AWS Glue

Utilisez les propriétés d'exécution de flux de travail pour partager et gérer l'état entre les tâches de votre flux de travail AWS Glue. Vous pouvez définir les propriétés d'exécution par défaut lorsque vous créez le flux de travail. Ensuite, tandis que vos tâches s'exécutent, elles peuvent récupérer les valeurs des propriétés d'exécution et, le cas échéant, les modifier pour les entrées de tâches qui sont plus loin dans le flux de travail. Lorsqu'une tâche modifie une propriété d'exécution, la nouvelle valeur existe uniquement pour l'exécution du flux de travail. Les propriétés d'exécution par défaut ne sont pas affectées.

Si votre tâche AWS Glue ne fait pas partie d'un flux de travail, ces propriétés ne seront pas définies.

L'exemple de code Python suivant à partir d'une tâche d'extraction, de transformation et de chargement (ETL) montre comment obtenir les propriétés d'exécution des flux de travail.

```
import sys
import boto3
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from awsglue.context import GlueContext
from pyspark.context import SparkContext

glue_client = boto3.client("glue")
args = getResolvedOptions(sys.argv, ['JOB_NAME','WORKFLOW_NAME', 'WORKFLOW_RUN_ID'])
workflow_name = args['WORKFLOW_NAME']
```

```
workflow_run_id = args['WORKFLOW_RUN_ID']
workflow_params = glue_client.get_workflow_run_properties(Name=workflow_name,
                                          RunId=workflow_run_id)["RunProperties"]

target_database = workflow_params['target_database']
target_s3_location = workflow_params['target_s3_location']
```

Le code suivant continue en définissant la propriété d'exécution `target_format` sur `'csv'`.

```
workflow_params['target_format'] = 'csv'
glue_client.put_workflow_run_properties(Name=workflow_name, RunId=workflow_run_id,
 RunProperties=workflow_params)
```

Pour plus d'informations, consultez les ressources suivantes :

- GetWorkflowRunProperties action (Python : get_workflow_run_properties)
- PutWorkflowRunProperties action (Python : put_workflow_run_properties)

Interrogation des flux de travail à l'aide du AWS Glue API

AWS Glue fournit un riche API de gestion des flux de travail. Vous pouvez récupérer la vue statique ou dynamique d'un flux de travail en cours d'exécution à l'aide de l' AWS Glue API. Pour de plus amples informations, veuillez consulter Flux de travail.

Rubriques

- Interrogation des vues statiques
- Interrogation des vues dynamiques

Interrogation des vues statiques

Utilisez l'opération d'API `GetWorkflow` pour obtenir une vue statique qui indique la conception d'un flux de travail. Cette opération renvoie un graphe orienté composé de nœuds et d'arcs, où un nœud représente un déclencheur, une tâche ou un crawler. Les arcs définissent les relations entre les nœuds. Ils sont représentés par des connecteurs (flèches) sur le graphique dans la console AWS Glue.

Vous pouvez également utiliser cette opération avec des bibliothèques de traitement graphique populaires telles que NetworkX, igraph, JGraph T et le Java Universal Network/Graph (JUNG)

Framework. Étant donné que toutes ces bibliothèques représentent les graphiques de la même manière, des transformations minimes sont nécessaires.

La vue statique renvoyée par cette API est la plus grande up-to-date vue selon la dernière définition des déclencheurs associés au flux de travail.

Définition du graphique

Un graphe G de flux de travail est une paire ordonnée (N, E), où N est un ensemble de nœuds et E un ensemble d'arcs. Un nœud est un sommet du graphique identifié par un numéro unique. Un nœud peut être de type déclencheur, tâche ou crawler. Par exemple : `{name:T1, type:Trigger, uniqueId:1}, {name:J1, type:Job, uniqueId:2}`.

Un arc est un 2-tuplet de la forme (`src, dest`), où `src` et `dest` sont les nœuds et il y a un arc dirigé de `src` vers `dest`.

Exemple d'interrogation d'une vue statique

Envisagez un déclencheur conditionnel T, qui déclenche la tâche J2 après la fin de la tâche J1.

```
J1 ---> T ---> J2
```

Nœuds : J1, T, J2

Arcs : (J1, T), (T, J2)

Interrogation des vues dynamiques

Utilisez l'opération d'API`GetWorkflowRun` pour obtenir une vue dynamique d'un flux de travail en cours d'exécution. Cette opération renvoie la même vue statique du graphe en même temps que les métadonnées relatives à l'exécution du flux de travail.

Pour l'exécution, les nœuds représentent les tâches dans l'appel `GetWorkflowRun` ont une liste d'exécutions de tâches lancées dans le cadre de la dernière exécution du flux de travail. Vous pouvez utiliser cette liste pour afficher le statut d'exécution de chaque tâche dans le graphique lui-même. Pour les dépendances en aval qui ne sont pas encore exécutées, ce champ est défini sur `null`. Le graphique d'informations vous permet de connaître l'état actuel d'un flux de travail à tout moment.

La vue dynamique renvoyée par cette API est basée sur la vue statique qui était présente lorsque l'exécution du flux de travail a été démarrée.

Exemple de nœuds d'exécution : `{name:T1, type: Trigger, uniqueId:1}`, `{name:J1, type:Job, uniqueId:2, jobDetails:{jobRuns}}`, `{name:C1, type:Crawler, uniqueId:3, crawlerDetails:{crawls}}`

Exemple 1 : Vue dynamique

L'exemple suivant illustre un simple flux de travail à deux déclencheurs.

- Nœuds : t1, j1, t2, j2
- Arcs : (t1, j1), (j1, t2), (t2, j2)

La réponse `GetWorkflow` contient les éléments suivants.

```
{
    Nodes : [
        {
            "type" : Trigger,
            "name" : "t1",
            "uniqueId" : 1
        },
        {
            "type" : Job,
            "name" : "j1",
            "uniqueId" : 2
        },
        {
            "type" : Trigger,
            "name" : "t2",
            "uniqueId" : 3
        },
        {
            "type" : Job,
            "name" : "j2",
            "uniqueId" : 4
        }
    ],
    Edges : [
        {
            "sourceId" : 1,
            "destinationId" : 2
        },
        {
```

```
            "sourceId" : 2,
            "destinationId" : 3
        },
        {
            "sourceId" : 3,
            "destinationId" : 4
        }
}
```

La réponse `GetWorkflowRun` contient les éléments suivants.

```
{
    Nodes : [
        {
            "type" : Trigger,
            "name" : "t1",
            "uniqueId" : 1,
            "jobDetails" : null,
            "crawlerDetails" : null
        },
        {
            "type" : Job,
            "name" : "j1",
            "uniqueId" : 2,
            "jobDetails" : [
                {
                    "id" : "jr_12334",
                    "jobRunState" : "SUCCEEDED",
                    "errorMessage" : "error string"
                }
            ],
            "crawlerDetails" : null
        },
        {
            "type" : Trigger,
            "name" : "t2",
            "uniqueId" : 3,
            "jobDetails" : null,
            "crawlerDetails" : null
        },
        {
            "type" : Job,
            "name" : "j2",
```

```
            "uniqueId" : 4,
            "jobDetails" : [
                {
                    "id" : "jr_1233sdf4",
                    "jobRunState" : "SUCCEEDED",
                    "errorMessage" : "error string"
                }
            ],
            "crawlerDetails" : null
        }
    ],
    Edges : [
        {
            "sourceId" : 1,
            "destinationId" : 2
        },
        {
            "sourceId" : 2,
            "destinationId" : 3
        },
        {
            "sourceId" : 3,
            "destinationId" : 4
        }
}
```

Exemple 2 : Tâches multiples avec un déclencheur conditionnel

L'exemple suivant illustre un flux de travail avec plusieurs tâches et un déclencheur conditionnel (t3).

```
Consider Flow:
T(t1) ---> J(j1) ---> T(t2) ---> J(j2)
             |                    |
             |                    |
             >+------> T(t3) <-----+
                        |
                        |
                      J(j3)

Graph generated:
Nodes: t1, t2, t3, j1, j2, j3
Edges: (t1, j1), (j1, t2), (t2, j2), (j1, t3), (j2, t3), (t3, j3)
```

Restrictions du plan et du flux de travail dans AWS Glue

Les restrictions suivantes s'appliquent aux modèles et aux flux de travail.

Restrictions du plan

Gardez à l'esprit les restrictions de modèle suivantes :

- Le plan doit être enregistré dans la même AWS région que celle où réside le compartiment Amazon S3.
- Pour partager des plans entre AWS comptes, vous devez accorder les autorisations de lecture sur l'archive ZIP des plans dans Amazon S3. Les clients qui ont une autorisation de lecture sur une archive ZIP de plan peuvent enregistrer le plan dans leur AWS compte et l'utiliser.
- L'ensemble des paramètres de modèle est stocké en tant qu'objet JSON unique. La longueur maximale de cet objet est de 128 Ko.
- La taille maximale non compressée de l'archive ZIP du modèle est de 5 Mo. La taille maximale compressée est de 1 Mo.
- Limitez le nombre total de tâches, de crawlers et de déclencheurs au sein d'un flux de travail à 100 ou moins. Si vous en incluez plus de 100, vous risquez de rencontrer des erreurs lorsque vous tentez de reprendre ou d'arrêter les exécutions du flux de travail.

Restrictions du flux de travail

Gardez à l'esprit les restrictions de flux de travail suivantes. Certains de ces commentaires s'adressent davantage à un utilisateur qui crée des flux de travail manuellement.

- La taille de lot maximale pour un déclencheur d' EventBridge événement Amazon est de 100. La taille maximale de la fenêtre temporelle est de 900 secondes (15 minutes).
- Un déclencheur peut être associé à un seul flux de travail.
- Seul un déclencheur de départ (à la demande ou planifié) est autorisé.
- Si une tâche ou un crawler dans un flux de travail est démarré par un déclencheur qui se trouve en dehors du flux de travail, les déclencheurs intégrés au flux de travail et dépendant de l'achèvement (réussi ou non) de la tâche ou de l'crawler ne se déclenchent pas.
- De même, si une tâche ou un crawler dans un flux de travail comporte des déclencheurs qui dépendent de l'achèvement (avec succès ou non) d'une tâche ou d'un crawler à la fois dans et en dehors du flux de travail, si la tâche ou l'crawler est démarré à partir d'un flux de travail, seuls

les déclencheurs à l'intérieur du flux de travail se déclenchent à l'achèvement de la tâche ou de l'crawler.

Résolution des erreurs de plans dans AWS Glue

Si vous rencontrez des erreurs lorsque vous utilisez des modèles AWS Glue, utilisez les solutions suivantes afin de trouver la source des problèmes et de corriger ceux-ci.

Rubriques

- Erreur : PySpark module manquant
- Error: missing blueprint config file (Erreur : fichier de configuration de modèle manquant)
- Error: missing imported file (Erreur : fichier importé manquant)
- Erreur : non autorisé à effectuer une exécution iamPassRole sur la ressource
- Error: invalid cron schedule (Erreur : calendrier cron non valide)
- Error: a trigger with the same name already exists (Erreur : un déclencheur du même nom existe déjà)
- Erreur : un flux de travail du même nom : foo existe déjà.
- Error: module not found in specified layoutGenerator path (Erreur : module introuvable dans le chemin LayoutGenerator spécifié)
- Error: validation error in Connections field (Erreur : erreur de validation dans le champ Connections [Connexions])

Erreur : PySpark module manquant

AWS Gluerenvoie l'erreur « Erreur inconnue lors de l'exécution de la fonction de génération de mise en page ModuleNotFoundError : aucun module nommé 'pyspark' ».

Lorsque vous décompressez l'archive du modèle, cela peut ressembler à l'un des éléments suivants :

```
$ unzip compaction.zip
Archive:  compaction.zip
   creating: compaction/
  inflating: compaction/blueprint.cfg
  inflating: compaction/layout.py
  inflating: compaction/README.md
  inflating: compaction/compaction.py
```

```
$ unzip compaction.zip
Archive:  compaction.zip
  inflating: blueprint.cfg
  inflating: compaction.py
  inflating: layout.py
  inflating: README.md
```

Dans le premier cas, tous les fichiers liés au modèle ont été placés dans un dossier nommé compactage et il a ensuite été converti en un fichier zip nommé compaction.zip.

Dans le second cas, tous les fichiers requis pour le modèle n'étaient pas inclus dans un dossier et ont été ajoutés en tant que fichiers racine sous le fichier zip compaction.zip.

La création d'un fichier dans l'un des formats ci-dessus est autorisée. Assurez-vous toutefois que `blueprint.cfg` a le chemin d'accès approprié au nom de la fonction dans le script qui génère la mise en page.

Exemples

Dans le cas 1 : `blueprint.cfg` doit disposer de `layoutGenerator` comme suit :

```
layoutGenerator": "compaction.layout.generate_layout"
```

Dans le cas 2 : `blueprint.cfg` doit disposer de `layoutGenerator` comme suit :

```
layoutGenerator": "layout.generate_layout"
```

Si ce chemin n'est pas inclus correctement, vous pourriez voir une erreur comme indiqué. Par exemple, si vous avez la structure de dossiers mentionnée dans le cas 2 et que vous avez `layoutGenerator` comme indiquée dans le cas 1, vous pouvez voir l'erreur ci-dessus.

Error: missing blueprint config file (Erreur : fichier de configuration de modèle manquant)

AWS Gluerenvoie l'erreur « Erreur inconnue lors de l'exécution de la fonction de génération de mise en page FileNotFoundError : [Errno 2] Aucun fichier ou répertoire de ce type : '/ tmp/compaction/ blueprint .cfg' ».

Le blueprint.cfg doit être placé au niveau racine de l'archive ZIP ou dans un dossier portant le même nom que l'archive ZIP.

Lorsque nous extrayons l'archive ZIP de modèle, blueprint.cfg devrait se trouver dans l'un des chemins suivants. S'il ne se trouve pas dans l'un des chemins suivants, vous pouvez voir l'erreur ci-dessus.

```
$ unzip compaction.zip
Archive:  compaction.zip
   creating: compaction/
  inflating: compaction/blueprint.cfg

$ unzip compaction.zip
Archive:  compaction.zip
  inflating: blueprint.cfg
```

Error: missing imported file (Erreur : fichier importé manquant)

AWS Gluerenvoie l'erreur « Erreur inconnue lors de l'exécution de la fonction de génération de mise en page FileNotFoundError : [Errno 2] Aucun fichier ou répertoire de ce type :* *'demo-project/foo.py' ».

Si votre script de génération de mise en page a une fonctionnalité pour lire d'autres fichiers, assurez-vous de donner un chemin complet pour le fichier à importer. Par exemple, le script Conversion.py peut être référencé dans Layout.py. Pour en savoir plus, veuillez consulter la rubrique Exemple de projet de plan.

Erreur : non autorisé à effectuer une exécution iamPassRole sur la ressource

AWS Gluerenvoie l'erreur « User : arn:aws:sts : :123456789012:assumed- » role/AWSGlueServiceRole/GlueSession is not authorized to perform: iam:PassRole on resource: arn:aws:iam::123456789012:role/AWSGlueServiceRole

Si les travaux et les crawlers du flux de travail assument le même rôle que le rôle transmis pour créer le flux de travail à partir du modèle, alors le rôle du modèle doit inclure l'autorisation `iam:PassRole` pour lui-même.

Si les travaux et les crawlers du flux de travail assument un rôle autre que celui transmis pour créer les entités du flux de travail à partir du modèle, le rôle de ce dernier doit inclure l'autorisation `iam:PassRole` sur cet autre rôle plutôt que sur le rôle du modèle.

Pour plus d'informations, veuillez consulter Autorisations pour les rôles de plans.

Error: invalid cron schedule (Erreur : calendrier cron non valide)

AWS Glue renvoie l'erreur « The schedule cron(0 0 * * * *) is invalid » (Le cron horaire [0 0 * * * *] n'est pas valide).

Fournissez une expression cron valide. Pour plus d'informations, consultez Planifications temporelles pour les tâches et les crawlers.

Error: a trigger with the same name already exists (Erreur : un déclencheur du même nom existe déjà)

AWS Glue renvoie l'erreur « Trigger with name 'foo_starting_trigger' already submitted with different configuration » (Trigger avec le nom 'foo_starting_trigger' déjà soumis avec une configuration différente).

Un modèle ne nécessite pas que vous définissiez des déclencheurs dans le script de mise en page pour la création de flux de travail. La création de déclencheur est gérée par la bibliothèque de modèles en fonction des dépendances définies entre deux actions.

La dénomination des déclencheurs est la suivante :

- Pour le déclencheur de démarrage dans le flux de travail, le nom est <workflow_name>_starting_trigger.
- Pour un nœud (tâche/crawler) dans le flux de travail qui dépend de l'achèvement d'un ou de plusieurs nœuds en amont ; AWS Glue définit un déclencheur avec le nom <workflow_name>_<node_name>_trigger.

Cette erreur signifie qu'un déclencheur du même nom existe déjà. Vous pouvez supprimer le déclencheur existant et relancer la création du flux de travail.

> **Note**
>
> La suppression d'un flux de travail ne supprime pas les nœuds du flux de travail. Bien que le flux de travail soit supprimé, il est possible que des déclencheurs soient laissés pour compte. Pour cette raison, vous ne pouvez pas recevoir une erreur « workflow already exists » (le flux de travail existe déjà), mais vous pouvez recevoir une erreur « trigger already exists » (le déclencheur existe déjà) dans le cas où vous créez un flux de travail, le supprimez, puis essayez de le recréer avec le même nom du même modèle.

Erreur : un flux de travail du même nom : foo existe déjà.

Le nom du flux de travail doit être unique. Veuillez essayer avec un nom différent.

Error: module not found in specified layoutGenerator path (Erreur : module introuvable dans le chemin LayoutGenerator spécifié)

AWS Gluerenvoie l'erreur « Erreur inconnue lors de l'exécution de la fonction de génération de mise en page ModuleNotFoundError : aucun module nommé 'crawl_s3_locations' ».

```
layoutGenerator": "crawl_s3_locations.layout.generate_layout"
```

Par exemple, si vous avez le chemin layoutGenerator ci-dessus, lorsque vous décompressez l'archive de modèle, il doit ressembler à ce qui suit :

```
$ unzip crawl_s3_locations.zip
Archive:  crawl_s3_locations.zip
   creating: crawl_s3_locations/
  inflating: crawl_s3_locations/blueprint.cfg
  inflating: crawl_s3_locations/layout.py
  inflating: crawl_s3_locations/README.md
```

Lorsque vous décompressez l'archive, si l'archive du modèle ressemble à ce qui suit, vous pouvez obtenir l'erreur ci-dessus.

```
$ unzip crawl_s3_locations.zip
Archive:  crawl_s3_locations.zip
  inflating: blueprint.cfg
  inflating: layout.py
  inflating: README.md
```

Vous pouvez voir qu'il n'y a pas de dossier nommé `crawl_s3_locations` et lorsque le chemin `layoutGenerator` fait référence au fichier de mise en page via le module `crawl_s3_locations`, vous pouvez obtenir l'erreur ci-dessus.

Error: validation error in Connections field (Erreur : erreur de validation dans le champ Connections [Connexions])

AWS Gluerenvoie l'erreur « Erreur inconnue lors de l'exécution de la fonction de génération de mise en page TypeError : la valeur ['foo'] pour les connexions clés doit être de type <class 'dict'> ! ».

Il s'agit d'une erreur de validation. Le champs `Connections` dans le champ `Job` attend un dictionnaire et à la place une liste de valeurs sont fournies, ce qui provoque l'erreur.

```
User input was list of values
Connections= ['string']

Should be a dict like the following
Connections*=*{'Connections': ['string']}
```

Pour éviter ces erreurs d'exécution lors de la création d'un flux de travail à partir d'un plan, vous pouvez valider les définitions du flux de travail, de la tâche et du crawler comme indiqué dans Tester un plan.

Référez-vous à la syntaxe dans la Référence des classes de plans AWS Glue pour définir la tâche AWS Glue, le crawler et le flux de travail dans le script de mise en page.

Autorisations de personas et de rôles pour les plans AWS Glue

Vous trouverez ci-dessous les personas typiques et les politiques d'autorisation suggérées Gestion des identités et des accès AWS (IAM) pour les personas et les rôles pour les blueprints. AWS Glue

Rubriques

- Personas du plan
- Autorisations pour les personas du plan
- Autorisations des rôles de plans

Personas du plan

Voici les personas généralement impliquées dans le cycle de vie des plans AWS Glue.

Persona	Description
Développeur AWS Glue	Développe, teste et publie des modèles.
administrateur AWS Glue	Inscrit, gère et accorde des autorisations pour les modèles.
Analyste des données	Exécute des modèles pour créer des flux de travail.

Pour de plus amples informations, veuillez consulter the section called "Présentation des plans".

Autorisations pour les personas du plan

Voici les autorisations suggérées pour chaque persona du modèle.

Autorisations de développeur AWS Glue pour les plans

Le développeur AWS Glue doit disposer des autorisations d'écriture sur le compartiment Amazon S3 utilisé pour publier le plan. Souvent, le développeur enregistre le modèle après l'avoir téléchargé. Dans ce cas, le développeur a besoin des autorisations répertoriées dans the section called "Autorisations d'administrateur AWS Glue pour les plans". De plus, si le développeur souhaite tester le modèle après son enregistrement, il a également besoin des autorisations répertoriées dans the section called "Autorisations d'analyste des données pour les plans".

Autorisations d'administrateur AWS Glue pour les plans

La politique suivante accorde des autorisations pour l'enregistrement, l'affichage et la gestion des plans AWS Glue.

> **Important**
>
> Dans la politique suivante, remplacez *<s3-bucket-name>* et *<prefix>* par le chemin Amazon S3 vers les archives ZIP du plan chargées pour vous enregistrer.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:CreateBlueprint",
        "glue:UpdateBlueprint",
        "glue:DeleteBlueprint",
        "glue:GetBlueprint",
        "glue:ListBlueprints",
        "glue:BatchGetBlueprints"
      ],
```

```
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject"
      ],
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/prefix/*"
    }
  ]
}
```

Autorisations d'analyste des données pour les plans

La politique suivante accorde des autorisations pour exécuter les modèles et afficher le flux de travail et ses composants résultants. Elle accorde également à `PassRole` le rôle que AWS Glue endosse pour créer le flux de travail et ses composants.

La politique accorde des autorisations pour n'importe quelle ressource. Si vous souhaitez configurer un accès détaillé à des plans individuels, utilisez le format suivant pour le plan : ARNs

```
arn:aws:glue:<region>:<account-id>:blueprint/<blueprint-name>
```

> **Important**
>
> Dans la politique suivante, remplacez *<account-id>* par un AWS compte valide et remplacez *<role-name>* par le nom du rôle utilisé pour exécuter un plan. Veuillez consulter the section called "Autorisations des rôles de plans" pour connaître les autorisations requises par ce rôle.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
```

```
        "glue:ListBlueprints",
        "glue:GetBlueprint",
        "glue:StartBlueprintRun",
        "glue:GetBlueprintRun",
        "glue:GetBlueprintRuns",
        "glue:GetCrawler",
        "glue:ListTriggers",
        "glue:ListJobs",
        "glue:BatchGetCrawlers",
        "glue:GetTrigger",
        "glue:BatchGetWorkflows",
        "glue:BatchGetTriggers",
        "glue:BatchGetJobs",
        "glue:BatchGetBlueprints",
        "glue:GetWorkflowRun",
        "glue:GetWorkflowRuns",
        "glue:ListCrawlers",
        "glue:ListWorkflows",
        "glue:GetJob",
        "glue:GetWorkflow",
        "glue:StartWorkflowRun"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "iam:PassRole",
      "Resource": "arn:aws:iam::111122223333:role/role-name"
    }
  ]
}
```

Autorisations des rôles de plans

Voici les autorisations suggérées pour le rôle IAM utilisé pour créer un flux de travail à partir d'un modèle. Le rôle doit avoir une relation d'approbation avec `glue.amazonaws.com`.

> **Important**
>
> Dans la politique suivante, remplacez *<account-id>* par un AWS compte valide et remplacez *<role-name>* par le nom du rôle.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:CreateJob",
        "glue:GetCrawler",
        "glue:GetTrigger",
        "glue:DeleteCrawler",
        "glue:CreateTrigger",
        "glue:DeleteTrigger",
        "glue:DeleteJob",
        "glue:CreateWorkflow",
        "glue:DeleteWorkflow",
        "glue:GetJob",
        "glue:GetWorkflow",
        "glue:CreateCrawler"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "iam:PassRole",
      "Resource": "arn:aws:iam::111122223333:role/role-name"
    }
  ]
}
```

> **Note**
>
> Si les tâches et les crawlers du flux de travail endossent un rôle autre que celui-ci, cette politique doit inclure l'autorisation `iam:PassRole` pour cet autre rôle plutôt que pour le rôle du modèle.

Développement des plans dans AWS Glue

Votre organisation peut avoir un jeu de cas d'utilisation ETL similaires qui pourraient bénéficier de la capacité de paramétrage d'un flux de travail unique pour les gérer tous. Pour répondre à ce besoin, AWS Glue vous permet de définir des plans que vous pouvez utiliser pour générer des flux de travail. Un modèle accepte des paramètres, de sorte qu'à partir d'un modèle unique, un analyste de données peut créer différents flux de travail pour gérer des cas d'utilisation ETL similaires. Après avoir créé un modèle, vous pouvez le réutiliser pour différents services, équipes et projets.

Rubriques

- Présentation des plans dans AWS Glue
- Développement des plans dans AWS Glue
- Enregistrement d'un plan dans AWS Glue
- Affichage des plans dans AWS Glue
- Mise à jour d'un plan dans AWS Glue
- Création d'un flux de travail à partir d'un plan dans AWS Glue
- Affichage des exécutions du plan dans AWS Glue

Présentation des plans dans AWS Glue

> **Note**
>
> La fonctionnalité Blueprints n'est actuellement pas disponible dans les régions suivantes de la console AWS Glue : Asie-Pacifique (Jakarta) et Moyen-Orient (Émirats arabes unis).

Les plans AWS Glue offrent un moyen de créer et de partager des flux de travail AWS Glue. Lorsqu'il existe un processus ETL complexe qui pourrait être utilisé pour des cas d'utilisation similaires, plutôt que de créer un flux de travail AWS Glue pour chaque cas d'utilisation, vous pouvez créer un seul plan.

Le modèle spécifie les tâches et les crawlers à inclure dans un flux de travail, et spécifie les paramètres que l'utilisateur du flux de travail fournit lorsqu'il exécute le modèle pour créer un flux de travail. L'utilisation de paramètres permet à un seul modèle de générer des flux de travail pour les différents cas d'utilisation similaires. Pour de plus amples informations sur les flux de travail, veuillez consulter Présentation des flux de travail dans AWS Glue.

Voici quelques exemples de cas d'utilisation pour les modèles :

- Vous souhaitez partitionner un jeu de données existant. Les paramètres d'entrée du modèle sont Amazon Simple Storage Service (Amazon S3) et une liste pour les colonnes de partition.
- Vous souhaitez créer un instantané d'une table Amazon DynamoDB dans un magasin de données SQL comme Amazon Redshift. Les paramètres d'entrée du plan sont le nom de la table DynamoDB et une connexion AWS Glue, qui désigne un cluster Amazon Redshift et une base de données de destination.
- Vous souhaitez convertir les données CSV dans plusieurs chemins Amazon S3 en Parquet. Vous souhaitez que le flux de travail AWS Glue inclue un crawler et une tâche distincts pour chaque chemin. Les paramètres d'entrée sont la base de données de destination dans le catalogue de données AWS Glue et une liste délimitée par des virgules de chemins Amazon S3. Notez que dans ce cas, le nombre d'crawlers et de tâches créés par le flux de travail est variable.

Composants du plan

Un modèle est une archive ZIP qui contient les composants suivants :

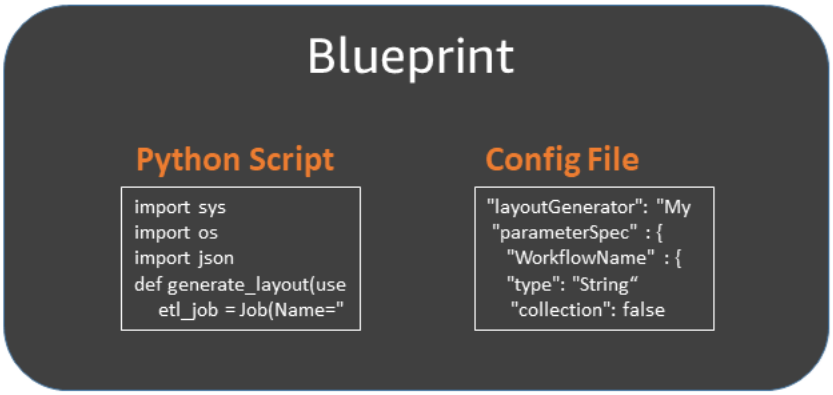
- Un script de générateur de mise en page Python

  Contient une fonction qui spécifie la disposition du flux de travail : les crawlers et les tâches à créer pour le flux de travail, les propriétés de la tâche et de l'crawler et les dépendances entre les tâches et les crawlers. La fonction accepte les paramètres de plan et renvoie une structure de flux de travail (objet JSON) que AWS Glue utilise pour générer le flux de travail. Étant donné que vous utilisez un script Python pour générer le flux de travail, vous pouvez ajouter votre propre logique adaptée à vos cas d'utilisation.
- Un fichier de configuration

  Spécifie le nom complet de la fonction Python qui génère la disposition du flux de travail. Spécifie également les noms, types de données et autres propriétés de tous les paramètres de modèle utilisés par le script.
- (Facultatif) scripts ETL et fichiers de support

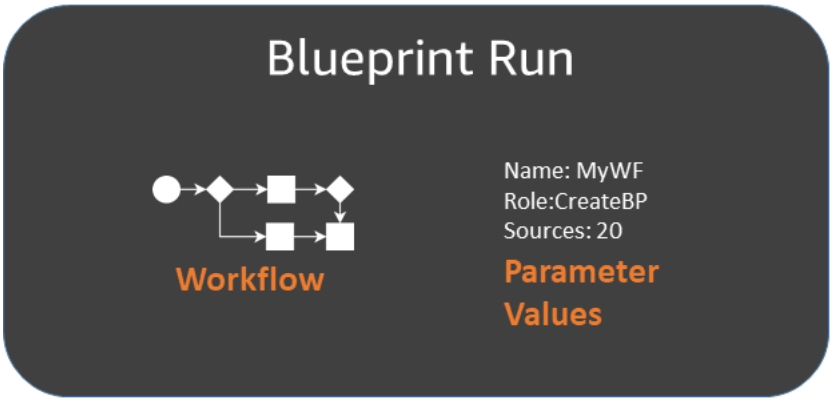
  En tant que cas d'utilisation avancé, vous pouvez paramétrer l'emplacement des scripts ETL que vos tâches utilisent. Vous pouvez inclure des fichiers de script de tâche dans l'archive ZIP et spécifier un paramètre de modèle pour un emplacement Amazon S3 vers lequel les scripts doivent être copiés. Le script du générateur de mise en page peut copier les scripts ETL à l'emplacement

désigné et spécifier cet emplacement en tant que propriété d'emplacement du script de tâche. Vous pouvez également inclure des bibliothèques ou d'autres fichiers de support, à condition que votre script les gère.



Exécutions de modèle

Lorsque vous créez un flux de travail à partir d'un plan, AWS Glue exécute le plan, qui démarre un processus asynchrone pour créer le flux de travail et les tâches, les moteurs d'exploration et les déclencheurs que le flux de travail encapsule. AWS Glue utilise l'exécution du plan pour orchestrer la création du flux de travail et de ses composantes. Vous affichez le statut du processus de création en affichant le statut d'exécution du modèle. L'exécution du modèle stocke également les valeurs que vous avez fournies pour les paramètres du modèle.



Vous pouvez afficher les exécutions du plan à l'aide de la AWS Glue console ou AWS Command Line Interface (AWS CLI). Lors de l'affichage ou du dépannage d'un flux de travail, vous pouvez toujours revenir à l'exécution du modèle pour afficher les valeurs des paramètres de modèle qui ont été utilisées pour créer le flux de travail.

Cycle de vie d'un plan

Les plans sont développés, testés, enregistrés auprès de AWS Glue et exécutés pour créer des flux de travail. Il existe généralement trois personas impliquées dans le cycle de vie du modèle.

Persona	Tâches
Développeur AWS Glue	• Écrit le script de mise en page du flux de travail et crée le fichier de configuration. • Teste le plan localement à l'aide des bibliothèques fournies par le service AWS Glue. • Crée une archive ZIP du script, du fichier de configura tion et des fichiers de prise en charge et publie l'archive dans un emplacement dans Amazon S3. • Ajoute une politique de compartiment au compartiment Amazon S3 qui accorde des autorisations de lecture sur les objets du compartiment au AWS compte de l'AWS Glueadministrateur. • Accorde des autorisations de lecture IAM sur l'archive ZIP dans Amazon S3 à l'administrateur AWS Glue.
administrateur AWS Glue	• Enregistre le plan avec AWS Glue. AWS Glue effectue une copie de l'archive ZIP dans un emplacement Amazon S3 réservé. • Accorde des autorisations IAM sur le modèle aux analystes de données.
Analyste des données	• Exécute le modèle pour créer un flux de travail et fournit des valeurs de paramètres de modèle. Vérifie l'état d'exécution du modèle pour s'assurer que le flux de travail et les composants de flux de travail ont été générés avec succès. • Exécute le flux de travail et en résout les problèmes. Avant d'exécuter le flux de travail, vous pouvez vérifier le flux de travail en affichant le graphique de conception du flux de travail sur la console AWS Glue.

Consultez aussi

- Développement des plans dans AWS Glue
- Création d'un flux de travail à partir d'un plan dans AWS Glue
- Autorisations de personas et de rôles pour les plans AWS Glue

Développement des plans dans AWS Glue

En tant que développeur AWS Glue, vous pouvez créer et publier des plans que les analystes de données peuvent utiliser pour générer des flux de travail.

Rubriques

- Présentation de l'élaboration de plans
- Prérequis pour l'élaboration des plans
- Rédiger le code du plan
- Exemple de projet de plan
- Test d'un plan
- Publication d'un plan
- Référence des classes de plan AWS Glue
- Exemples de plans

Consultez aussi

- Présentation des plans dans AWS Glue

Présentation de l'élaboration de plans

La première étape de votre processus de développement consiste à identifier un cas d'utilisation commun pour lequel un modèle serait utile. Un cas d'utilisation typique implique un problème ETL récurrent qui, selon vous, devrait être résolu de manière générale. Ensuite, concevez un modèle qui implémente le cas d'utilisation généralisé et définissez les paramètres d'entrée du modèle qui ensemble, peuvent définir un cas d'utilisation spécifique à partir du cas d'utilisation généralisé.

Un modèle se compose d'un projet, qui contient un fichier de configuration de paramètres du modèle, de même qu'un script qui définit la structure du flux de travail à générer. La structure définit les tâches et les crawlers (ou entities dans la terminologie du script de modèle) à créer.

Vous ne spécifiez pas directement de déclencheurs dans le script de structure. Au lieu de cela, vous écrivez du code pour spécifier les dépendances entre les tâches et les crawlers que le script crée. AWS Glue génère les déclencheurs en fonction de vos spécifications de dépendances. La sortie du script de structure est un objet de flux de travail, qui contient des spécifications pour toutes les entités de flux de travail.

Vous créez votre objet de flux de travail à l'aide des bibliothèques de plans AWS Glue :

- `awsglue.blueprint.base_resource` - Bibliothèque de ressources de base utilisées par les bibliothèques.
- `awsglue.blueprint.workflow` - Bibliothèque permettant de définir une classe `Workflow`.
- `awsglue.blueprint.job` - Bibliothèque permettant de définir une classe `Job`.
- `awsglue.blueprint.crawler` - Bibliothèque permettant de définir une classe `Crawler`.

Les seules autres bibliothèques qui sont prises en charge pour la génération de structure sont les bibliothèques qui sont disponibles pour le shell Python.

Avant de publier votre modèle, vous pouvez utiliser les méthodes définies dans les bibliothèques de modèles pour le tester localement.

Lorsque vous êtes prêt à mettre le modèle à la disposition des analystes de données, packagez le script, le fichier de configuration des paramètres et tous les fichiers de support, tels que les scripts et les bibliothèques supplémentaires, en une seule ressource déployable. Chargez ensuite la ressource sur Amazon S3 et demandez à un administrateur de l'enregistrer dans AWS Glue.

Pour plus d'informations sur d'autres projets de modèle, veuillez consulter les rubriques Exemple de projet de plan et Exemples de plans.

Prérequis pour l'élaboration des plans

Pour développer des plans, vous devez être familiarisé avec l'utilisation de AWS Glue et l'écriture de scripts pour les tâches ETL Apache Spark ou les tâches shell Python. Vous devez également effectuer les tâches de configuration suivantes.

- Téléchargez quatre bibliothèques AWS Python à utiliser dans vos scripts de mise en page de plan.

- Configurez le AWS SDKs.
- Configurez le AWS CLI.

Télécharger les bibliothèques Python

Téléchargez les bibliothèques suivantes depuis GitHub et installez-les dans votre projet :

- https://github.com/awslabs/aws-glue-blueprint-libs/tree/master/awsglue/blueprint/base_resource.py
- https://github.com/awslabs/aws-glue-blueprint-libs/tree/master/awsglue/blueprint/workflow.py
- https://github.com/awslabs/aws-glue-blueprint-libs/tree/master/awsglue/blueprint/crawler.py
- https://github.com/awslabs/aws-glue-blueprint-libs/tree/master/awsglue/blueprint/job.py

Configuration du SDK AWS Java

Pour le SDK AWS Java, vous devez ajouter un `jar` fichier qui inclut l'API pour les plans.

1. Si ce n'est pas déjà fait, configurez le AWS SDK pour Java.
   - Pour Java 1.x, suivez les instructions de la rubrique Configuration de AWS SDK pour Java dans le Guide du développeur AWS SDK pour Java .
   - Pour Java 2.x, suivez les instructions de la rubrique Configuration de AWS SDK for Java 2.x dans le Guide du développeur AWS SDK for Java 2.x .
2. Téléchargez le `jar` fichier client qui a accès aux plans APIs pour les plans.
   - Pour Java 1.x : s3 ://awsglue-custom-blueprints-preview- artifacts/awsglue-java-sdk-preview/AWSGlueJavaClient -1.11.x.jar
   - Pour Java 2.x : s3 ://awsglue-custom-blueprints-preview- artifacts/awsglue-java-sdk-v2-preview/AwsJavaSdk -Glue-2.0.jar
3. Ajoutez le client `jar` au début du chemin de classe Java pour remplacer le client AWS Glue fourni par le AWS SDK Java.

   ```
   export CLASSPATH=<path-to-preview-client-jar>:$CLASSPATH
   ```

4. (Facultatif) Testez le SDK avec l'application Java suivante. L'application doit afficher une liste vide.

Remplacez `accessKey` et `secretKey` par vos informations d'identification et remplacez `us-east-1` par votre région.

```
import com.amazonaws.auth.AWSCredentials;
import com.amazonaws.auth.AWSCredentialsProvider;
import com.amazonaws.auth.AWSStaticCredentialsProvider;
import com.amazonaws.auth.BasicAWSCredentials;
import com.amazonaws.services.glue.AWSGlue;
import com.amazonaws.services.glue.AWSGlueClientBuilder;
import com.amazonaws.services.glue.model.ListBlueprintsRequest;

public class App{
    public static void main(String[] args) {
        AWSCredentials credentials = new BasicAWSCredentials("accessKey",
 "secretKey");
        AWSCredentialsProvider provider = new
 AWSStaticCredentialsProvider(credentials);
        AWSGlue glue = AWSGlueClientBuilder.standard().withCredentials(provider)
                .withRegion("us-east-1").build();
        ListBlueprintsRequest request = new
 ListBlueprintsRequest().withMaxResults(2);
        System.out.println(glue.listBlueprints(request));
    }
}
```

Configuration du SDK AWS Python

Les étapes suivantes supposent que la version 2.7 ou ultérieure de Python, ou la version 3.9 ou ultérieure, est installée sur votre ordinateur.

1. Téléchargez le fichier Wheel boto3 suivant. Si vous êtes invité à ouvrir ou à enregistrer, enregistrez le fichier. s3 ://awsglue-custom-blueprints-preview- artifacts/aws-python-sdk-preview/boto 3-1.17.31-py2.py3-none-any.whl
2. Téléchargez le fichier botocore wheel suivant : s3 ://awsglue-custom-blueprints-preview--1.20.31-py2.py3-none-any.whl artifacts/aws-python-sdk-preview/botocore
3. Vérifiez votre version Python.

```
python --version
```

4. Selon votre version Python, saisissez les commandes suivantes (pour Linux) :

   - Pour Python 2.7 ou supérieur.

   ```
   python3 -m pip install --user virtualenv
   source env/bin/activate
   ```

   - Pour Python 3.9 ou version ultérieure.

   ```
   python3 -m venv python-sdk-test
   source python-sdk-test/bin/activate
   ```

5. Installez le fichier Wheel botocore.

   ```
   python3 -m pip install <download-directory>/botocore-1.20.31-py2.py3-none-any.whl
   ```

6. Installez le fichier Wheel boto3.

   ```
   python3 -m pip install <download-directory>/boto3-1.17.31-py2.py3-none-any.whl
   ```

7. Configurez vos informations d'identification et région par défaut dans les fichiers `~/.aws/credentials` et `~/.aws/config` suivants. Pour plus d'informations, veuillez consulter la rubrique Configuration de l' AWS CLI dans le Guide de l'utilisateur de l'AWS Command Line Interface .

8. Testez votre configuration (facultatif). Les commandes suivantes doivent renvoyer une liste vide.

   Remplacez `us-east-1` par votre région.

   ```
   $ python
   >>> import boto3
   >>> glue = boto3.client('glue', 'us-east-1')
   >>> glue.list_blueprints()
   ```

Configuration de l'aperçu AWS CLI

1. Si ce n'est pas déjà fait, installez and/or update the AWS Command Line Interface (AWS CLI) sur votre ordinateur. La façon la plus simple de procéder est d'utiliser `pip`, l'utilitaire d'installation de Python :

```
pip install awscli --upgrade --user
```

Vous pouvez trouver des instructions d'installation complètes pour l'installation de AWS CLI ici : Installation de AWS Command Line Interface.

2. Téléchargez le fichier de AWS CLI roue depuis : s3 ://awsglue-custom-blueprints-preview-artifacts/awscli-preview-build/awscli -1.19.31-py2.py3-none-any.whl
3. Installez le fichier AWS CLI Wheel.

```
python3 -m pip install awscli-1.19.31-py2.py3-none-any.whl
```

4. Exécutez la commande `aws configure`. Configurez vos AWS informations d'identification (y compris la clé d'accès et la clé secrète) et votre AWS région. Vous trouverez des informations sur la configuration du AWS CLI ici : Configuration du AWS CLI.
5. Testez le AWS CLI. La commande suivante doit renvoyer une liste vide.

   Remplacez `us-east-1` par votre région.

```
aws glue list-blueprints --region us-east-1
```

Rédiger le code du plan

Chaque projet de modèle que vous créez doit contenir au minimum les fichiers suivants :

- Un script de structure Python qui définit le flux de travail. Le script contient une fonction qui définit les entités (tâches et crawlers) dans un flux de travail, ainsi que les dépendances entre elles.
- Fichier de configuration `blueprint.cfg`, qui définit :
  - Le chemin d'accès complet de la fonction de définition de structure du flux de travail.
  - Les paramètres acceptés par le modèle.

Rubriques

- Création du script de structure du plan
- Créez le fichier de configuration
- Spécifier les paramètres du plan

Création du script de structure du plan

Le script de structure de modèle doit inclure une fonction qui génère les entités dans votre flux de travail. Vous pouvez nommer cette fonction comme vous le souhaitez. AWS Glue utilise le fichier de configuration pour déterminer le nom complet de la fonction.

Votre fonction de structure effectue les opérations suivantes :

- (Facultatif) Instancie la classe `Job` pour créer des objets `Job`, et passe des arguments tels que `Command` et `Role`. Ce sont des propriétés de tâche que vous spécifieriez si vous créiez la tâche à l'aide de la console AWS Glue ou de l'API.
- (Facultatif) Instancie la classe `Crawler` pour créer des objets `Crawler`, et passe les arguments nom, rôle et cible.
- Pour indiquer les dépendances entre les objets (entités de flux de travail), transmet les arguments supplémentaires `DependsOn` et `WaitForDependencies` à `Job()` et `Crawler()`. Ces arguments sont expliqués plus loin dans cette section.
- Instancie la classe `Workflow` pour créer l'objet de flux de travail qui est retourné à AWS Glue, en passant un argument `Name`, un argument `Entities`, et un argument facultatif `OnSchedule`. L'argument `Entities` spécifie tous les tâches et les crawlers à inclure dans le flux de travail. Pour voir comment construire un objet `Entities`, vous trouverez l'exemple de projet plus loin dans cette section.
- Renvoie un objet `Workflow`.

Pour les définitions de classes `Job`, `Crawler` et `Workflow`, veuillez consulter Référence des classes de plan AWS Glue.

La fonction de structure doit accepter les arguments d'entrée suivants.

Argument	Description
`user_params`	Dictionnaire Python des noms et valeurs de paramètres de modèle. Pour de plus amples informations, veuillez consulter Spécifier les paramètres du plan.
`system_params`	Dictionnaire Python contenant deux propriétés : `region` et `accountId` .

Voici un exemple de script générateur de structure dans un fichier nommé `Layout.py` :

```
import argparse
import sys
import os
import json
from awsglue.blueprint.workflow import *
from awsglue.blueprint.job import *
from awsglue.blueprint.crawler import *


def generate_layout(user_params, system_params):

    etl_job = Job(Name="{}_etl_job".format(user_params['WorkflowName']),
                  Command={
                      "Name": "glueetl",
                      "ScriptLocation": user_params['ScriptLocation'],
                      "PythonVersion": "2"
                  },
                  Role=user_params['PassRole'])
    post_process_job = Job(Name="{}_post_process".format(user_params['WorkflowName']),
                           Command={
                               "Name": "pythonshell",
                               "ScriptLocation": user_params['ScriptLocation'],
                               "PythonVersion": "2"
                           },
                           Role=user_params['PassRole'],
                           DependsOn={
                               etl_job: "SUCCEEDED"
                           },
                           WaitForDependencies="AND")
    sample_workflow = Workflow(Name=user_params['WorkflowName'],
                           Entities=Entities(Jobs=[etl_job, post_process_job]))
    return sample_workflow
```

L'exemple de script importe les bibliothèques de modèles requises et inclut une fonction `generate_layout` qui génère un flux de travail avec deux tâches. Ceci est un script très simple. Un script plus complexe pourrait utiliser une logique et des paramètres supplémentaires pour générer un flux de travail avec de nombreux crawlers et tâches, ou même un nombre variable de tâches et d'crawlers.

Utilisation de l' DependsOn argument

L'argument `DependsOn` est une représentation sous forme de dictionnaire d'une dépendance que cette entité a sur d'autres entités dans le flux de travail. Elle se présente sous la forme suivante.

```
DependsOn = {dependency1 : state, dependency2 : state, ...}
```

Les clés de ce dictionnaire représentent la référence d'objet, et non le nom, de l'entité, tandis que les valeurs sont des chaînes qui correspondent à l'état à surveiller. AWS Glue déduit les déclencheurs appropriés. Pour connaître les états valides, veuillez consulter Condition Structure (Structure des conditions).

Par exemple, une tâche peut dépendre de la réussite d'un crawler. Si vous définissez un objet crawler nommé `crawler2` comme suit :

```
crawler2 = Crawler(Name="my_crawler", ...)
```

Ensuite, un objet dépendant de `crawler2` comprendrait un argument de constructeur tel que :

```
DependsOn = {crawler2 : "SUCCEEDED"}
```

Par exemple :

```
job1 = Job(Name="Job1", ..., DependsOn = {crawler2 : "SUCCEEDED", ...})
```

Si `DependsOn` est omis pour une entité, cette entité dépend du déclencheur de démarrage du flux de travail.

Utilisation de l' WaitForDependencies argument

L'argument `WaitForDependencies` définit si une entité de tâche ou d'crawler doit attendre jusqu'à ce que toutes les entités dont elle dépend se terminent ou qu'une seule d'entre elles se termine.

Les valeurs autorisées sont « `AND` » ou « `ANY` ».

Utilisation de l' OnSchedule argument

L'argument `OnSchedule` pour le constructeur de classe `Workflow` est une expression `cron` qui définit la définition de déclenchement de démarrage d'un flux de travail.

Si cet argument est spécifié, AWS Glue crée un déclencheur de planification avec la planification correspondante. Si elle n'est pas spécifiée, le déclencheur de démarrage du flux de travail est un déclencheur à la demande.

Créez le fichier de configuration

Le fichier de configuration de modèle est un fichier obligatoire qui définit le point d'entrée de script pour la génération du flux de travail et les paramètres que le modèle accepte. Le fichier doit être nommé `blueprint.cfg`.

Voici un exemple de fichier de configuration.

```
{
    "layoutGenerator": "DemoBlueprintProject.Layout.generate_layout",
    "parameterSpec" : {
           "WorkflowName" : {
                "type": "String",
                "collection": false
           },
           "WorkerType" : {
                "type": "String",
                "collection": false,
                "allowedValues": ["G1.X", "G2.X"],
                "defaultValue": "G1.X"
           },
           "Dpu" : {
                "type" : "Integer",
                "allowedValues" : [2, 4, 6],
                "defaultValue" : 2
           },
           "DynamoDBTableName": {
                "type": "String",
                "collection" : false
           },
           "ScriptLocation" : {
                "type": "String",
                "collection": false
     }
    }
}
```

La propriété `layoutGenerator` spécifie le nom complet de la fonction dans le script qui génère la structure.

La propriété `parameterSpec` spécifie les paramètres que ce modèle accepte. Pour de plus amples informations, veuillez consulter Spécifier les paramètres du plan.

> **Important**
>
> Votre fichier de configuration doit inclure le nom du flux de travail en tant que paramètre de modèle, ou vous devez générer un nom de flux de travail unique dans votre script de structure.

Spécifier les paramètres du plan

Le fichier de configuration contient les spécifications des paramètres de modèle dans un objet JSON `parameterSpec`. `parameterSpec` contient un ou plusieurs objets paramètres.

```
"parameterSpec": {
    "<parameter_name>": {
      "type": "<parameter-type>",
      "collection": true|false,
      "description": "<parameter-description>",
      "defaultValue": "<default value for the parameter if value not specified>"
      "allowedValues": "<list of allowed values>"
    },
    "<parameter_name>": {
       ...
    }
  }
```

Voici les règles de codage de chaque objet paramètre :

- Le nom du paramètre et `type` sont obligatoires. Toutes les autres propriétés sont facultatives.
- Si vous spécifiez la propriété `defaultValue`, le paramètre est facultatif. Sinon, le paramètre est obligatoire et l'analyste de données qui crée un flux de travail à partir du modèle doit lui fournir une valeur.
- Si vous définissez la propriété `collection` à `true`, le paramètre peut prendre une collection de valeurs. Les collections peuvent être de n'importe quel type de données.
- Si vous spécifiez `allowedValues`, la console AWS Glue affiche une liste déroulante de valeurs que l'analyste de données doit choisir lors de la création d'un flux de travail à partir du plan.

Les valeurs suivantes sont autorisées pour `type` :

Types de données de paramètre	Remarques
`String`	-
`Integer`	-
`Double`	-
`Boolean`	Les valeurs possibles sont `true` et `false`. Génère une case à vérifier sur la page Create a workflow from <blueprint> (Créer un flux de travail à partir du <blueprint>) sur la console AWS Glue.
`S3Uri`	Chemin d'accès Amazon S3, en commençant par `s3://`. Génère un champ de texte et un bouton Browse (Parcourir)sur la page Create a workflow from <blueprint> (Créer un flux de travail à partir du <blueprint>).
`S3Bucket`	Noms du compartiment Amazon S3 uniquement. Génère un sélecteur de compartiment sur la page Create a workflow from <blueprint> (Créer un flux de travail à partir du <blueprint>).
`IAMRoleArn`	Nom de ressource Amazon (ARN) d'un rôle Gestion des identités et des accès AWS (IAM). Génère un sélecteur de rôle sur la page Create a workflow from <blueprint> (Créer un flux de travail à partir du <blueprint>).
`IAMRoleName`	Nom d'un rôle IAM. Génère un sélecteur de rôle sur la page Create a workflow from <blueprint> (Créer un flux de travail à partir du <blueprint>).

Exemple de projet de plan

La conversion de format de données est un cas d'utilisation fréquent d'extraction, de transformation et de chargement (ETL). Dans les charges de travail analytiques typiques, les formats de fichiers basés sur des colonnes comme Parquet ou ORC sont préférés aux formats de texte tels que CSV ou JSON.

Cet exemple de plan vous permet de convertir des données CSV/JSON/etc de. en Parquet pour les fichiers sur Amazon S3.

Ce modèle prend une liste de chemins S3 définis par un paramètre de modèle, convertit les données au format Parquet et les écrit à l'emplacement S3 spécifié par un autre paramètre de modèle. Le script de structure crée un crawler et une tâche pour chaque chemin d'accès. Le script de structure télécharge également le script ETL se trouvant dans `Conversion.py` vers un compartiment S3 spécifié par un autre paramètre de modèle. Le script de structure spécifie ensuite le script téléchargé en tant que script ETL pour chaque tâche. L'archive ZIP du projet contient le script de structure, le script ETL et le fichier de configuration du modèle.

Pour plus d'informations sur d'autres projets de modèle, veuillez consulter la rubrique Exemples de plans.

Ce qui suit est le script de structure, dans le fichier `Layout.py`.

```
from awsglue.blueprint.workflow import *
from awsglue.blueprint.job import *
from awsglue.blueprint.crawler import *
import boto3

s3_client = boto3.client('s3')

# Ingesting all the S3 paths as Glue table in parquet format
def generate_layout(user_params, system_params):
    #Always give the full path for the file
    with open("ConversionBlueprint/Conversion.py", "rb") as f:
        s3_client.upload_fileobj(f, user_params['ScriptsBucket'], "Conversion.py")
    etlScriptLocation = "s3://{}/Conversion.py".format(user_params['ScriptsBucket'])

    crawlers = []
    jobs = []
    workflowName = user_params['WorkflowName']
    for path in user_params['S3Paths']:
      tablePrefix = "source_"
      crawler = Crawler(Name="{}_crawler".format(workflowName),
                        Role=user_params['PassRole'],
                        DatabaseName=user_params['TargetDatabase'],
                        TablePrefix=tablePrefix,
                        Targets= {"S3Targets": [{"Path": path}]})
      crawlers.append(crawler)
      transform_job = Job(Name="{}_transform_job".format(workflowName),
```

```
                        Command={"Name": "glueetl",
                                 "ScriptLocation": etlScriptLocation,
                                 "PythonVersion": "3"},
                        Role=user_params['PassRole'],
                        DefaultArguments={"--database_name":
user_params['TargetDatabase'],
                                          "--table_prefix": tablePrefix,
                                          "--region_name": system_params['region'],
                                          "--output_path":
user_params['TargetS3Location']},
                        DependsOn={crawler: "SUCCEEDED"},
                        WaitForDependencies="AND")
     jobs.append(transform_job)
   conversion_workflow = Workflow(Name=workflowName, Entities=Entities(Jobs=jobs,
Crawlers=crawlers))
   return conversion_workflow
```

Ce qui suit est le fichier de configuration du modèle correspondant `blueprint.cfg`.

```
{
    "layoutGenerator": "ConversionBlueprint.Layout.generate_layout",
    "parameterSpec" : {
        "WorkflowName" : {
            "type": "String",
            "collection": false,
            "description": "Name for the workflow."
        },
        "S3Paths" : {
            "type": "S3Uri",
            "collection": true,
            "description": "List of Amazon S3 paths for data ingestion."
        },
        "PassRole" : {
            "type": "IAMRoleName",
            "collection": false,
            "description": "Choose an IAM role to be used in running the job/crawler"
        },
        "TargetDatabase": {
            "type": "String",
            "collection" : false,
            "description": "Choose a database in the Data Catalog."
        },
        "TargetS3Location": {
```

```
            "type": "S3Uri",
            "collection" : false,
            "description": "Choose an Amazon S3 output path: ex:s3://<target_path>/."
        },
        "ScriptsBucket": {
            "type": "S3Bucket",
            "collection": false,
            "description": "Provide an S3 bucket name(in the same AWS Region) to store
 the scripts."
        }
    }
}
```

Le script suivant dans le fichier `Conversion.py` est le script ETL téléchargé. Notez qu'il conserve le schéma de partitionnement pendant la conversion.

```
import sys
from pyspark.sql.functions import *
from pyspark.context import SparkContext
from awsglue.transforms import *
from awsglue.context import GlueContext
from awsglue.job import Job
from awsglue.utils import getResolvedOptions
import boto3

args = getResolvedOptions(sys.argv, [
    'JOB_NAME',
    'region_name',
    'database_name',
    'table_prefix',
    'output_path'])
databaseName = args['database_name']
tablePrefix = args['table_prefix']
outputPath = args['output_path']

glue = boto3.client('glue', region_name=args['region_name'])

glue_context = GlueContext(SparkContext.getOrCreate())
spark = glue_context.spark_session
job = Job(glue_context)
job.init(args['JOB_NAME'], args)

def get_tables(database_name, table_prefix):
```

```
    tables = []
    paginator = glue.get_paginator('get_tables')
    for page in paginator.paginate(DatabaseName=database_name, Expression=table_prefix
+"*"):
        tables.extend(page['TableList'])
    return tables

for table in get_tables(databaseName, tablePrefix):
    tableName = table['Name']
    partitionList = table['PartitionKeys']
    partitionKeys = []
    for partition in partitionList:
        partitionKeys.append(partition['Name'])

    # Create DynamicFrame from Catalog
    dyf = glue_context.create_dynamic_frame.from_catalog(
        name_space=databaseName,
        table_name=tableName,
        additional_options={
            'useS3ListImplementation': True
        },
        transformation_ctx='dyf'
    )

    # Resolve choice type with make_struct
    dyf = ResolveChoice.apply(
        frame=dyf,
        choice='make_struct',
        transformation_ctx='resolvechoice_' + tableName
    )

    # Drop null fields
    dyf = DropNullFields.apply(
        frame=dyf,
        transformation_ctx="dropnullfields_" + tableName
    )

    # Write DynamicFrame to S3 in glueparquet
    sink = glue_context.getSink(
        connection_type="s3",
        path=outputPath,
        enableUpdateCatalog=True,
        partitionKeys=partitionKeys
    )
```

```
    sink.setFormat("glueparquet")

    sink.setCatalogInfo(
        catalogDatabase=databaseName,
        catalogTableName=tableName[len(tablePrefix):]
    )
    sink.writeFrame(dyf)

job.commit()
```

> **Note**
>
> Seuls deux chemins Amazon S3 peuvent être fournis en entrée pour l'exemple de modèle. La raison en est que les déclencheurs AWS Glue sont limités à invoquer seulement deux actions de crawler.

Test d'un plan

Pendant que vous développez votre code, vous devez effectuer des tests locaux pour vérifier que la structure du flux de travail est correcte.

Les tests locaux ne génèrent pas de tâches, d'crawlers ou de déclencheurs AWS Glue. Au lieu de cela, vous exécutez le script de structure localement et utilisez la méthode `to_json()` et `validate()` pour imprimer des objets et trouver des erreurs. Ces méthodes sont disponibles dans les trois classes définies dans les bibliothèques.

Il existe deux façons de gérer les arguments `user_params` et `system_params` que AWS Glue passe à votre fonction de structure. Votre code de banc de test peut créer un dictionnaire d'exemples de valeurs de paramètre de modèle et le transmettre à la fonction de structure en tant qu'argument `user_params`. Vous pouvez également supprimer les références à `user_params` et remplacez-les par des chaînes codées en dur.

Si votre code utilise les propriétés `region` et `accountId` dans `system_params`, vous pouvez passer votre propre dictionnaire pour `system_params`.

Pour tester un modèle

1. Démarrez un interpréteur Python dans un répertoire avec les bibliothèques, ou chargez les fichiers du modèle et les bibliothèques fournies dans votre environnement de développement intégré (IDE) préféré.
2. Assurez-vous que votre code importe les bibliothèques fournies.
3. Ajouter du code à votre fonction de structure pour appeler `validate()` ou `to_json()` sur n'importe quelle entité ou sur l'objet `Workflow`. Par exemple, si votre code crée un objet `Crawler` nommé `mycrawler`, vous pouvez appeler `validate()` comme suit.

   ```
   mycrawler.validate()
   ```

   Vous pouvez imprimer `mycrawler` comme suit :

   ```
   print(mycrawler.to_json())
   ```

   Si vous appelez `to_json` sur un objet, il n'est pas nécessaire d'appeler également `validate()`, parce que `to_json()` appelle `validate()`.

   Il est plus utile d'appeler ces méthodes sur l'objet de flux de travail. En supposant que votre script nomme l'objet de flux de travail `my_workflow`, validez et imprimez l'objet de flux de travail comme suit.

   ```
   print(my_workflow.to_json())
   ```

   Pour plus d'informations sur `to_json()` et `validate()`, consultez Méthodes de classe.

   Vous pouvez également importer `pprint` et imprimer l'objet de flux de travail, tel que décrit dans l'exemple plus loin dans cette section.
4. Exécutez le code, corrigez les erreurs et supprimez enfin tous les appels à `validate()` ou `to_json()`.

Example

L'exemple suivant montre comment construire un dictionnaire d'exemples de paramètres de modèle et le transmettre en tant qu'argument `user_params` à la fonction de structure

`generate_compaction_workflow`. L'exemple montre également comment imprimer automatiquement l'objet de flux de travail généré.

```
from pprint import pprint
from awsglue.blueprint.workflow import *
from awsglue.blueprint.job import *
from awsglue.blueprint.crawler import *

USER_PARAMS = {"WorkflowName": "compaction_workflow",
               "ScriptLocation": "s3://amzn-s3-demo-bucket/scripts/threaded-
compaction.py",
               "PassRole": "arn:aws:iam::111122223333:role/GlueRole-ETL",
               "DatabaseName": "cloudtrial",
               "TableName": "ct_cloudtrail",
               "CoalesceFactor": 4,
               "MaxThreadWorkers": 200}


def generate_compaction_workflow(user_params: dict, system_params: dict) -> Workflow:
    compaction_job = Job(Name=f"{user_params['WorkflowName']}_etl_job",
                         Command={"Name": "glueetl",
                                  "ScriptLocation": user_params['ScriptLocation'],
                                  "PythonVersion": "3"},
                         Role="arn:aws:iam::111122223333:role/
AWSGlueServiceRoleDefault",
                         DefaultArguments={"DatabaseName": user_params['DatabaseName'],
                                           "TableName": user_params['TableName'],
                                           "CoalesceFactor":
 user_params['CoalesceFactor'],
                                           "max_thread_workers":
 user_params['MaxThreadWorkers']})

    catalog_target = {"CatalogTargets": [{"DatabaseName": user_params['DatabaseName'],
 "Tables": [user_params['TableName']]}]}

    compacted_files_crawler = Crawler(Name=f"{user_params['WorkflowName']}_post_crawl",
                                      Targets = catalog_target,
                                      Role=user_params['PassRole'],
                                      DependsOn={compaction_job: "SUCCEEDED"},
                                      WaitForDependencies="AND",
                                      SchemaChangePolicy={"DeleteBehavior": "LOG"})

    compaction_workflow = Workflow(Name=user_params['WorkflowName'],
```

```
                                    Entities=Entities(Jobs=[compaction_job],

 Crawlers=[compacted_files_crawler]))
    return compaction_workflow

generated = generate_compaction_workflow(user_params=USER_PARAMS, system_params={})
gen_dict = generated.to_json()

pprint(gen_dict)
```

Publication d'un plan

Une fois que vous avez développé un modèle, vous devez le télécharger vers Amazon S3. Vous devez disposer d'autorisations en écriture sur le compartiment Amazon S3 que vous utilisez pour publier le modèle. Vous devez également vous assurer que l'administrateur AWS Glue, qui enregistrera le plan, dispose d'un accès en lecture au compartiment Amazon S3. Pour les politiques d'autorisation Gestion des identités et des accès AWS (IAM) suggérées pour les personas et les rôles pour les AWS Glue plans, voir. Autorisations de personas et de rôles pour les plans AWS Glue

Pour publier un modèle

1. Créez les scripts, les ressources et le fichier de configuration de modèle nécessaires.
2. Ajoutez tous les fichiers à une archive ZIP et téléchargez ce fichier ZIP sur Amazon S3. Utilisez un compartiment S3 qui se trouve dans la même région que celle dans laquelle les utilisateurs vont enregistrer et exécuter le modèle.

   Vous pouvez créer un fichier ZIP à partir de la ligne de commande à l'aide de la commande suivante.

   ```
   zip -r folder.zip folder
   ```

3. Ajoutez une politique de compartiment qui accorde des autorisations de lecture au compte AWS souhaité. Voici un exemple de politique.

   JSON

   ```
   {
     "Version":"2012-10-17",
     "Statement": [
       {
   ```

```
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:root"
      },
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::my-blueprints/*"
    }
  ]
}
```

4. Accorder à l'administrateur AWS Glue, ou à quiconque va enregistrer des plans, l'autorisation IAM `s3:GetObject` d'accès au compartiment Amazon S3. Pour obtenir un exemple de politique d'octroi d'accès aux administrateurs, veuillez consulter la rubrique Autorisations d'administrateur AWS Glue pour les plans.

Une fois que vous avez terminé les tests locaux de votre plan, vous pouvez également tester un plan sur AWS Glue. Pour tester un plan sur AWS Glue, il doit être enregistré. Vous pouvez limiter qui voit le modèle enregistré à l'aide de l'autorisation IAM ou à l'aide de comptes de test distincts.

Voir aussi :

- Enregistrement d'un plan dans AWS Glue

Référence des classes de plan AWS Glue

Les bibliothèques pour les modèles AWS Glue définissent trois classes que vous utilisez dans votre script de structure de flux de travail : `Job`, `Crawler` et `Workflow`.

Rubriques

- Classe Tâche
- Classe de crawler
- Classe de flux de travail
- Méthodes de classe

Classe Tâche

La classe `Job` représente une tâche ETL AWS Glue.

Arguments de constructeur obligatoires

Les arguments de constructeur suivants sont obligatoires pour la classe `Job`.

Nom d'argument	Type	Description
`Name`	`str`	Nom à affecter à la tâche. AWS Glue ajoute un suffixe généré de manière aléatoire au nom pour distinguer la tâche de celles créées par d'autres exécutions de plan.
`Role`	`str`	Amazon Resource Name (ARN) du rôle que la tâche doit assumer lors de son exécution.
`Command`	`dict`	Commande de tâche, comme spécifié dans la rubrique JobCommand structure de la documentation de l'API.

Arguments de constructeur facultatifs

Les arguments de constructeur suivants sont facultatifs pour la classe `Job`.

Nom d'argument	Type	Description
`DependsOn`	`dict`	Liste des entités de flux de travail dont dépend la tâche. Pour de plus amples informations, veuillez consulter Utilisation de l' DependsOn argument.
`WaitForDependencies`	`str`	Indique si la tâche doit attendre jusqu'à ce que toutes les entités dont elle dépend se terminent ou qu'une seule d'entre elles se termine avant de s'exécuter. Pour de plus amples informations, veuillez consulter Utilisation de l' WaitForDependencies argument. Omettre si la tâche ne dépend que d'une seule entité.

Nom d'argument	Type	Description
(Propriétés de la tâche)	-	Toutes les propriétés de tâche répertoriées dans Structure Job dans la documentation de l'API AWS Glue (sauf `CreatedOn` et `LastModifiedOn`).

Classe de crawler

La classe `Crawler` représente un crawler AWS Glue.

Arguments de constructeur obligatoires

Les arguments de constructeur suivants sont obligatoires pour la classe `Crawler`.

Nom d'argument	Type	Description
`Name`	`str`	Nom à affecter à la recherche. AWS Glue ajoute un suffixe généré de manière aléatoire au nom pour distinguer l'crawler de ceux créés par d'autres exécutions de plan.
`Role`	`str`	ARN du rôle que l'crawler doit assumer lors de l'exécution.
`Targets`	`dict`	Collecte de cibles à analyser. Les arguments de constructeur de classe `Targets` sont définis dans la rubrique CrawlerTargets structure de la documentation de l'API. Tous les arguments de constructeur `Targets` sont facultatifs, mais vous devez en passer au moins un.

Arguments de constructeur facultatifs

Les arguments de constructeur suivants sont facultatifs pour la classe `Crawler`.

Nom d'argument	Type	Description
`DependsOn`	`dict`	Liste des entités de flux de travail dont dépend l'crawler. Pour de plus amples informations, veuillez consulter Utilisation de l' DependsOn argument.
`WaitForDependencies`	`str`	Indique si l'crawler doit attendre jusqu'à ce que toutes les entités dont il dépend se terminent ou qu'une seule d'entre elles se termine avant de s'exécuter. Pour de plus amples informations, veuillez consulter Utilisation de l' WaitForDependencies argument. Omettre si l'crawler ne dépend que d'une seule entité.
(Propriétés de l'crawler)	-	Toutes les propriétés de l'crawler répertoriées dans Structure du crawler de la documentation de l'API AWS Glue, avec les exceptions suivantes : • `State` • `CrawlElapsedTime` • `CreationTime` • `LastUpdated` • `LastCrawl` • `Version`

Classe de flux de travail

La classe `Workflow` représente un flux de travail AWS Glue. Le script de mise en page du flux de travail renvoie un objet `Workflow`. AWS Glue crée un flux de travail basé sur cet objet.

Arguments de constructeur obligatoires

Les arguments de constructeur suivants sont obligatoires pour la classe `Workflow`.

Nom d'argument	Type	Description
`Name`	`str`	Nom à affecter au flux de travail.
`Entities`	`Entities`	Ensemble d'entités (tâches et crawlers) à inclure dans le flux de travail. Le constructeur de classe `Entities` accepte un argument `Jobs`, qui est une liste de `Job` et un objet `Crawlers`, qui est une liste d'objets `Crawler`.

Arguments de constructeur facultatifs

Les arguments de constructeur suivants sont facultatifs pour la classe `Workflow`.

Nom d'argument	Type	Description
`Description`	`str`	Consultez Structure de flux de travail.
`DefaultRunProperties`	`dict`	Consultez Structure de flux de travail.
`OnSchedule`	`str`	Une expression `cron`.

Méthodes de classe

Les trois classes comprennent les méthodes suivantes.

validate()

Valide les propriétés de l'objet et, si des erreurs sont détectées, affiche un message et sort. Ne génère aucune sortie s'il n'y a pas d'erreurs. Pour la classe `Workflow`, s'appelle elle-même pour chaque entité du flux de travail.

to_json()

Sérialise l'objet au format JSON. Appelle également `validate()`. Pour la classe `Workflow`, l'objet JSON inclut des listes de tâches et d'crawlers, ainsi qu'une liste de déclencheurs générés par les spécifications de dépendance de la tâche et de l'crawler.

Exemples de plans

Un certain nombre d'exemples de projets de plan sont disponibles dans le répertoire Github de plan AWS Glue. Ces exemples sont à titre de référence seulement et ne sont pas destinés à la production.

Les titres des projets exemples sont :

- Compactage : ce modèle crée une tâche qui compacte les fichiers d'entrée en paquets plus volumineux en fonction de la taille de fichier souhaitée.
- Conversion : ce modèle convertit les fichiers d'entrée dans différents formats de fichiers standard au format Apache Parquet, qui est optimisé pour les charges de travail analytiques.
- Analyse des emplacements Amazon S3 : ce modèle analyse plusieurs emplacements Amazon S3 pour ajouter des tables de métadonnées au catalogue de données.
- Connexion personnalisée au catalogue de données : ce plan accède aux magasins de données à l'aide de connecteurs AWS Glue personnalisés, lit les enregistrements et remplit les définitions des tables dans le catalogue de données AWS Glue en fonction du schéma d'enregistrement.
- Encodage : ce modèle convertit vos fichiers non-UTF en fichiers encodés UTF.
- Partitionnement : ce modèle crée une tâche de partitionnement qui place les fichiers de sortie dans des partitions basées sur des clés de partition spécifiques.
- Importation de données Amazon S3 dans une table DynamoDB : ce modèle importe les données d'Amazon S3 dans une table DynamoDB.
- Table standard à gouverner : ce plan importe une table AWS Glue Data Catalog dans une table Lake Formation.

Enregistrement d'un plan dans AWS Glue

Une fois que le développeur AWS Glue a codé le plan et téléchargé une archive ZIP sur Amazon Simple Storage Service (Amazon S3), un administrateur AWS Glue doit enregistrer ce plan. L'enregistrement du modèle le rend disponible pour utilisation.

Lorsque vous enregistrez un plan, AWS Glue copie l'archive de celui-ci vers un emplacement Amazon S3 réservé. Vous pouvez ensuite supprimer l'archive de l'emplacement de téléchargement.

Pour enregistrer un modèle, vous avez besoin d'autorisations de lecture sur l'emplacement Amazon S3 qui contient l'archive téléchargée. Vous avez également besoin de l'autorisation Gestion des identités et des accès AWS `glue:CreateBlueprint` (IAM). Pour obtenir les autorisations

suggérées pour un administrateur AWS Glue qui doit enregistrer, afficher et gérer des plans, consulter Autorisations d'administrateur AWS Glue pour les plans.

Vous pouvez enregistrer un plan en utilisant la console AWS Glue, l'API AWS Glue ou AWS Command Line Interface (AWS CLI).

Pour enregistrer un modèle (console)

1. Assurez-vous de disposer des autorisations de lecture (`s3:GetObject`) dans l'archive ZIP du modèle dans Amazon S3.
2. Ouvrez la console AWS Glue à l'adresse https://console.aws.amazon.com/glue/.

   Connectez-vous sous un profil utilisateur qui dispose des autorisations nécessaires pour enregistrer un modèle. Basculez vers la même région AWS en tant que compartiment Amazon S3 contenant l'archive ZIP du modèle.
3. Dans le panneau de navigation, sélectionnez Plans. Ensuite, sur la page Plans, sélectionnez Ajouter un plan.
4. Saisissez un nom de modèle et, éventuellement, une description.
5. Pour ZIP archive location (S3) (Emplacement de l'archive ZIP [S3]), saisissez le chemin Amazon S3 de l'archive ZIP du modèle téléchargé. Incluez le nom du fichier d'archive dans le chemin d'accès et commencez le chemin par `s3://`.
6. (Facultatif) Ajoutez une balise ou plusieurs balises.
7. Sélectionnez Add blueprint (Ajouter un modèle).

   La page Plans renvoie des résultats et indique que le statut du plan est à `CREATING`. Cliquez sur le bouton d'actualisation jusqu'à ce que le statut devienne `ACTIVE` ou `FAILED`.
8. Si le statut est `FAILED`, sélectionnez le modèle et dans le menu Actions, sélectionnez View (Afficher).

   La page des détails indique la raison de l'échec. Si le message d'erreur est « Unable to access object at location... » (Impossible d'accéder à l'objet à l'emplacement...) ou « Access denied on object at location... » (Accès refusé sur l'objet à l'emplacement...), passez en revue les exigences suivantes :

   - L'utilisateur sous lequel vous êtes connecté doit disposer d'une autorisation de lecture sur l'archive ZIP du modèle dans Amazon S3.

- Le compartiment Amazon S3 qui contient l'archive ZIP doit avoir une politique de compartiment qui accorde une autorisation de lecture sur l'objet à votre identifiant de AWS compte. Pour de plus amples informations, veuillez consulter Développement des plans dans AWS Glue.
- Le compartiment Amazon S3 que vous utilisez doit se trouver dans la même région que la région à laquelle vous êtes connecté sur la console.

9. Assurez-vous que les analystes de données disposent d'autorisations sur le modèle.

   La politique IAM suggérée pour les analystes des données est illustrée dansAutorisations d'analyste des données pour les plans. Cette politique accorde `glue:GetBlueprint` pour n'importe quelle ressource. Si votre politique est plus précise au niveau des ressources, accordez des autorisations aux analystes des données sur cette ressource qui vient d'être créée.

Pour enregistrer un plan (AWS CLI)

1. Entrez la commande suivante.

```
aws glue create-blueprint --name <blueprint-name> [--description <description>] --
blueprint-location s3://<s3-path>/<archive-filename>
```

2. Pour vérifier l'état du modèle, saisissez la commande suivante. Répétez la commande jusqu'à ce que le statut devienne `ACTIVE` ou `FAILED`.

```
aws glue get-blueprint --name <blueprint-name>
```

   Si le statut est `FAILED` et le message d'erreur est « Unable to access object at location... » (Impossible d'accéder à l'objet à l'emplacement...) ou « Access denied on object at location... » (Accès refusé sur l'objet à l'emplacement...), passez en revue les exigences suivantes :

   - L'utilisateur sous lequel vous êtes connecté doit disposer d'une autorisation de lecture sur l'archive ZIP du modèle dans Amazon S3.
   - Le compartiment Amazon S3 contenant l'archive ZIP doit avoir une politique de compartiment qui accorde une autorisation de lecture sur l'objet à votre identifiant de AWS compte. Pour de plus amples informations, veuillez consulter Publication d'un plan.
   - Le compartiment Amazon S3 que vous utilisez doit se trouver dans la même région que la région à laquelle vous êtes connecté sur la console.

Voir aussi :

- Présentation des plans dans AWS Glue

Affichage des plans dans AWS Glue

Affichez un modèle pour consulter la description, l'état et les spécifications des paramètres du celui-ci, et pour télécharger son archive ZIP.

Vous pouvez afficher un plan à l'aide de la console AWS Glue, de l'API AWS Glue, ou de l' AWS Command Line Interface (AWS CLI).

Pour afficher un modèle (console)

1. Ouvrez la console AWS Glue à l'adresse https://console.aws.amazon.com/glue/.
2. Dans le panneau de navigation, sélectionnez Plans.
3. Sur la page plans, sélectionnez un plan. Ensuite, dans le menu Actions, sélectionnez View (Afficher).

Pour afficher un plan (AWS CLI)

- Saisissez la commande suivante pour afficher uniquement le nom, la description et l'état du modèle. Remplacez *<blueprint-name>* par le nom du plan à afficher.

```
aws glue get-blueprint --name <blueprint-name>
```

Le résultat de la commande se présente comme suit.

```
{
    "Blueprint": {
        "Name": "myDemoBP",
        "CreatedOn": 1587414516.92,
        "LastModifiedOn": 1587428838.671,
        "BlueprintLocation": "s3://amzn-s3-demo-bucket1/demo/
DemoBlueprintProject.zip",
        "Status": "ACTIVE"
    }
```

```
}
```

Saisissez la commande suivante pour afficher également les spécifications des paramètres.

```
aws glue get-blueprint --name <blueprint-name>  --include-parameter-spec
```

Le résultat de la commande se présente comme suit.

```
{
    "Blueprint": {
        "Name": "myDemoBP",
        "CreatedOn": 1587414516.92,
        "LastModifiedOn": 1587428838.671,
        "ParameterSpec": "{\"WorkflowName\":{\"type\":\"String\",\"collection
\":false,\"description\":null,\"defaultValue\":null,\"allowedValues\":null},
\"PassRole\":{\"type\":\"String\",\"collection\":false,\"description\":null,
\"defaultValue\":null,\"allowedValues\":null},\"DynamoDBTableName\":{\"type
\":\"String\",\"collection\":false,\"description\":null,\"defaultValue\":null,
\"allowedValues\":null},\"ScriptLocation\":{\"type\":\"String\",\"collection
\":false,\"description\":null,\"defaultValue\":null,\"allowedValues\":null}}",
        "BlueprintLocation": "s3://awsexamplebucket1/demo/
DemoBlueprintProject.zip",
        "Status": "ACTIVE"
    }
}
```

Ajoutez `--include-blueprint` pour inclure une URL dans le résultat que vous pouvez coller dans votre navigateur pour télécharger l'archive ZIP du plan que AWS Glue a stocké.

Voir aussi :

- Présentation des plans dans AWS Glue

Mise à jour d'un plan dans AWS Glue

Vous pouvez mettre à jour un modèle si vous disposez d'un script de mise en page révisé, d'un ensemble révisé de paramètres de modèle ou de fichiers de support révisés. La mise à jour d'un modèle crée une nouvelle version.

La mise à jour d'un modèle n'affecte pas les flux de travail existants créés à partir du modèle.

Vous pouvez mettre à jour un plan en utilisant la console AWS Glue Glue, l'API Glue AWS Glue ou AWS Command Line Interface (AWS CLI).

La procédure suivante suppose que le développeur AWS Glue a créé et chargé une archive ZIP de plan mise à jour sur Amazon S3.

Pour mettre à jour un plan (console)

1. Assurez-vous de disposer des autorisations de lecture (`s3:GetObject`) dans l'archive ZIP du modèle dans Amazon S3.
2. Ouvrez la console AWS Glue à l'adresse https://console.aws.amazon.com/glue/.

   Connectez-vous sous un profil utilisateur qui dispose des autorisations nécessaires pour mettre à jour un modèle. Basculez vers la même région AWS en tant que compartiment Amazon S3 contenant l'archive ZIP du modèle.
3. Dans le panneau de navigation, sélectionnez Plans.
4. Sur la page plans, sélectionnez un plan, puis dans le menu Actions, sélectionnez Modifier.
5. Dans la page Edit a blueprint (Modifier un modèle), mettez à jour la Description du modèle ou ZIP archive location (S3) (Emplacement de l'archive ZIP [S3]). Veillez à inclure le nom du fichier de l'archive dans le chemin.
6. Choisissez Enregistrer.

   La page plans affiche des résultats indiquant que le statut du plan est `UPDATING`. Cliquez sur le bouton d'actualisation jusqu'à ce que le statut devienne `ACTIVE` ou `FAILED`.
7. Si le statut est `FAILED`, sélectionnez le modèle et dans le menu Actions, sélectionnez View (Afficher).

   La page des détails indique la raison de l'échec. Si le message d'erreur est « Unable to access object at location... » (Impossible d'accéder à l'objet à l'emplacement...) ou « Access denied on object at location... » (Accès refusé sur l'objet à l'emplacement...), passez en revue les exigences suivantes :

   - L'utilisateur sous lequel vous êtes connecté doit disposer d'une autorisation de lecture sur l'archive ZIP du modèle dans Amazon S3.

- Le compartiment Amazon S3 qui contient l'archive ZIP doit avoir une politique de compartiment qui accorde une autorisation de lecture sur l'objet à votre identifiant de AWS compte. Pour de plus amples informations, veuillez consulter Publication d'un plan.
- Le compartiment Amazon S3 que vous utilisez doit se trouver dans la même région que la région à laquelle vous êtes connecté sur la console.

> **Note**
>
> Si la mise à jour échoue, la prochaine exécution de modèle utilise la dernière version du modèle qui a été enregistrée ou mise à jour avec succès.

Pour mettre à jour un modèle (AWS CLI)

1. Entrez la commande suivante.

```
aws glue update-blueprint --name <blueprint-name> [--description <description>] --
blueprint-location s3://<s3-path>/<archive-filename>
```

2. Pour vérifier l'état du modèle, saisissez la commande suivante. Répétez la commande jusqu'à ce que le statut devienne `ACTIVE` ou `FAILED`.

```
aws glue get-blueprint --name <blueprint-name>
```

   Si le statut est `FAILED` et le message d'erreur est « Unable to access object at location... » (Impossible d'accéder à l'objet à l'emplacement...) ou « Access denied on object at location... » (Accès refusé sur l'objet à l'emplacement...), passez en revue les exigences suivantes :

   - L'utilisateur sous lequel vous êtes connecté doit disposer d'une autorisation de lecture sur l'archive ZIP du modèle dans Amazon S3.
   - Le compartiment Amazon S3 contenant l'archive ZIP doit avoir une politique de compartiment qui accorde une autorisation de lecture sur l'objet à votre identifiant de AWS compte. Pour de plus amples informations, veuillez consulter Publication d'un plan.
   - Le compartiment Amazon S3 que vous utilisez doit se trouver dans la même région que la région à laquelle vous êtes connecté sur la console.

Consultez aussi

- Présentation des plans dans AWS Glue

Création d'un flux de travail à partir d'un plan dans AWS Glue

Vous pouvez créer un flux de travail AWS Glue manuellement, en ajoutant un composant à la fois, ou vous pouvez créer un flux de travail à partir d'un plan AWS Glue. AWS Glue comprend des plans pour les cas d'utilisation courants. Vos développeurs AWS Glue peuvent créer des plans supplémentaires.

Important

Limitez le nombre total de tâches, de crawlers et de déclencheurs au sein d'un flux de travail à 100 ou moins. Si vous en incluez plus de 100, vous risquez de rencontrer des erreurs lorsque vous tentez de reprendre ou d'arrêter les exécutions du flux de travail.

Lorsque vous utilisez un modèle, vous pouvez générer rapidement un flux de travail pour un cas d'utilisation spécifique basé sur le cas d'utilisation généralisé défini par le modèle. Vous définissez le cas d'utilisation spécifique en fournissant des valeurs pour les paramètres de modèle. Par exemple, un modèle qui partitionne un jeu de données peut avoir les chemins source et cible Amazon S3 comme paramètres.

AWS Glue crée un flux de travail à partir d'un plan en exécutant celui-ci. L'exécution du modèle directeur enregistre les valeurs des paramètres que vous avez fournies et est utilisée pour suivre la progression et le résultat de la création du flux de travail et de ses composants. Lors du dépannage d'un flux de travail, vous pouvez afficher l'exécution du modèle pour déterminer les valeurs des paramètres de modèle qui ont été utilisées pour créer un flux de travail.

Pour créer et afficher des flux de travail, vous avez besoin de certaines autorisations IAM. Pour consulter un exemple de politique IAM, veuillez consulter Autorisations d'analyste des données pour les plans.

Vous pouvez créer un flux de travail à partir d'un plan en utilisant la console AWS Glue, l'API AWS Glue ou AWS Command Line Interface (AWS CLI).

Pour créer un flux de travail à partir d'un modèle (console)

1. Ouvrez la console AWS Glue à l'adresse https://console.aws.amazon.com/glue/.

   Connectez-vous en tant qu'utilisateur disposant des autorisations pour créer un flux de travail.
2. Dans le panneau de navigation, sélectionnez Plans.
3. Sélectionnez un modèle, puis dans le menu Actions, sélectionnez Create workflow (Créer un flux de travail).
4. Dans la page Create a workflow from <blueprint-name> (Créer un flux de travail à partir de <blueprint-name>), saisissez les informations suivantes :

   Paramètres des modèles

   Ceux-ci varient en fonction de la conception du modèle. Pour toute question concernant les paramètres, veuillez consulter le développeur. Les plans incluent généralement un paramètre pour le nom du flux de travail.

   Rôle IAM

   Le rôle que AWS Glue endosse pour créer le flux de travail et ses composants. Le rôle doit disposer des autorisations pour créer et supprimer des flux de travail, des tâches, des crawlers et des déclencheurs. Pour obtenir un exemple de politique pour le rôle, veuillez consulter Autorisations des rôles de plans.
5. Sélectionnez Soumettre.

   La page Blueprint Details (Détails du modèle) s'affiche et affiche une liste des exécutions de modèle en bas.
6. Dans la liste des exécutions de modèle, vérifiez l'exécution de modèle en tête de liste pour connaître l'état de création du flux de travail.

   L'état initial est `RUNNING`. Cliquez sur le bouton d'actualisation jusqu'à ce que le statut devienne `SUCCEEDED` ou `FAILED`.
7. Effectuez l'une des actions suivantes :

   - Si le statut d'achèvement est `SUCCEEDED`, vous pouvez accéder à la page Workflows (Flux de travail), puis sélectionnez le flux de travail qui vient d'être créé et exécutez-le. Avant d'exécuter le flux de travail, vous pouvez consulter le graphique de conception.
   - Si le statut d'achèvement est `FAILED`, sélectionnez l'exécution du modèle, puis dans la fenêtre Actions, sélectionnez View (Afficher) pour afficher le message d'erreur.

Pour en savoir plus sur les flux de travail et les plans, consultez les rubriques suivantes.

- Présentation des flux de travail dans AWS Glue
- Mise à jour d'un plan dans AWS Glue
- Création et développement d'un flux de travail manuellement dans AWS Glue

Affichage des exécutions du plan dans AWS Glue

Affichez une exécution de modèle pour afficher les informations suivantes :

- Nom du flux de travail qui a été créé.
- Valeurs de paramètre du plan utilisées pour créer le flux de travail.
- Statut de l'opération de création du flux de travail.

Vous pouvez afficher une exécution de plan à l'aide de la console AWS Glue, de l'API AWS Glue, ou de l' AWS Command Line Interface (AWS CLI).

Pour afficher une exécution de modèle (console)

1. Ouvrez la console AWS Glue à l'adresse https://console.aws.amazon.com/glue/.
2. Dans le panneau de navigation, sélectionnez Plans.
3. Sur la page plans, sélectionnez un plan. Ensuite, dans le menu Actions, sélectionnez View (Afficher).
4. Au bas de la page Blueprint Details (Détails des modèles), sélectionnez une exécution de modèle, puis dans le menu Actions, sélectionnez View (Afficher).

Pour afficher une exécution de modèle (AWS CLI)

- Entrez la commande suivante. Remplacez *<blueprint-name>* par le nom du plan. Remplacez *<blueprint-run-id>* par l'ID d'exécution du plan.

```
aws glue get-blueprint-run --blueprint-name <blueprint-name> --run-id <blueprint-run-id>
```

Voir aussi :

- Présentation des plans dans AWS Glue

AWS CloudFormation pour AWS Glue

CloudFormation est un service qui permet de créer de nombreuses AWS ressources. AWS Gluefournit des opérations d'API pour créer des objets dans le AWS Glue Data Catalog. Toutefois, il peut être plus pratique de définir et de créer AWS Glue des objets et d'autres objets de AWS ressources connexes dans un fichier CloudFormation modèle. Ensuite, vous pouvez automatiser le processus de création d'objets.

CloudFormation fournit une syntaxe simplifiée, soit JSON (JavaScript Object Notation) soit YAML (YAML Ain't Markup Language), pour exprimer la création de ressources. AWS Vous pouvez utiliser les modèles CloudFormation pour définir les objets de catalogue de données tels que les bases de données, les tables, les partitions, les crawlers, les classifieurs et les connexions. Vous pouvez également définir des objets ETL tels que les tâches, les déclencheurs et les points de terminaison de développement. Vous créez un modèle qui décrit toutes les AWS ressources que vous souhaitez, et vous vous CloudFormation chargez de les approvisionner et de les configurer pour vous.

Pour plus d'informations, voir Qu'est-ce que c'est AWS CloudFormation ? et Utilisation de AWS CloudFormation modèles dans le guide de AWS CloudFormation l'utilisateur.

Si vous envisagez d'utiliser des CloudFormation modèles compatibles avecAWS Glue, en tant qu'administrateur, vous devez accorder l'accès aux CloudFormation AWS services et aux actions dont il dépend. Pour accorder l'autorisation de créer CloudFormation des ressources, associez la politique suivante aux utilisateurs qui travaillent avec CloudFormation :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cloudformation:*"
      ],
      "Resource": "*"
    }
  ]
}
```

Le tableau suivant répertorie les actions qu'un CloudFormation modèle peut effectuer en votre nom. Il inclut des liens vers des informations sur les types de AWS ressources et leurs types de propriétés que vous pouvez ajouter à un CloudFormation modèle.

AWS Glue ressource	CloudFormation modèle	Exemples AWS Glue
Classifieur	AWS::Glue::Classifier	classifieur Grok, classifieur JSON, classifieur XML
Connexion	AWS::Glue::Connection	Connexion MySQL
crawler	AWS::Glue::Crawler	Crawler Amazon S3, Crawler MySQL
Base de données	AWS::Glue::Database	Base de données vide, Base de données avec tables
Point de terminaison de développement	AWS::Glue::DevEndpoint	Point de terminaison de développement
Integration	AWS::Glue::Integration	Intégration zéro ETL
Propriété de ressource d'intégration	AWS::Glue::IntegrationResourcePropriété	Intégration zéro ETL avec propriété de ressource d'intégration
Tâche	AWS::Glue::Job	Tâche Amazon S3, Tâche JDBC
Transformation de machine learning	AWS::Glue::MLTransform	Transformation de machine learning
Ensemble de règles de qualité des données	AWS::Glue::DataQualityEnsemble de règles	Ensemble de règles de qualité des données, Ensemble de règles de qualité des données avec planificateur EventBridge
Partition	AWS::Glue::Partition	Partitions d'une table
Table	AWS::Glue::Table	Table d'une base de données

AWS Glue ressource	CloudFormation modèle	Exemples AWS Glue
Déclencheur	AWS::Glue::Trigger	Déclencheur à la demande, déclencheur planifié, déclencheur conditionnel

Pour démarrer, utilisez les exemples de modèles suivants et personnalisez-les avec vos propres métadonnées. Utilisez ensuite la CloudFormation console pour créer une CloudFormation pile dans laquelle ajouter des objets AWS Glue et les services associés. La plupart des champs d'un objet AWS Glue sont facultatifs. Ces modèles illustrent les champs obligatoires ou qui sont nécessaires pour un objet AWS Glue opérationnel et fonctionnel.

Un CloudFormation modèle peut être au format JSON ou YAML. Dans ces exemples, YAML est utilisé pour faciliter la lisibilité. Les exemples contiennent des commentaires (#) pour décrire les valeurs qui sont définies dans les modèles.

CloudFormation les modèles peuvent inclure une `Parameters` section. Cette section peut être modifiée dans le texte d'exemple ou lorsque le fichier YAML est soumis à la CloudFormation console pour créer une pile. La `Resources` section du modèle contient la définition AWS Glue et les objets associés. CloudFormation les définitions de syntaxe des modèles peuvent contenir des propriétés qui incluent une syntaxe de propriété plus détaillée. Toutes les propriétés ne sont pas nécessairement obligatoires pour créer un objet AWS Glue. Ces exemples montrent des exemples de valeurs des propriétés communes pour créer un objet AWS Glue.

Exemple de CloudFormation modèle pour une AWS Glue base de données

Une base de données AWS Glue dans le catalogue de données contient les tables de métadonnées. La base de données comprend très peu de propriétés et peut être créée dans le catalogue de données à l'aide d'un CloudFormation modèle. L'exemple de modèle suivant est fourni pour vous aider à démarrer et pour illustrer l'utilisation de CloudFormation stacks avecAWS Glue. La seule ressource créée par l'exemple de modèle est une base de données nommée `cfn-mysampledatabase`. Vous pouvez le modifier en modifiant le texte de l'exemple ou en modifiant la valeur sur la CloudFormation console lorsque vous soumettez le YAML.

L'exemple suivant montre des exemples de valeurs des propriétés communes pour créer une base de données AWS Glue. Pour plus d'informations sur le modèle CloudFormation de base de données pourAWS Glue, consultez AWS::Glue::Database.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CloudFormation template in YAML to demonstrate creating a database named
 mysampledatabase
# The metadata created in the Data Catalog points to the flights public S3 bucket
#
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:
  CFNDatabaseName:
    Type: String
    Default: cfn-mysampledatabse

# Resources section defines metadata for the Data Catalog
Resources:
# Create an AWS Glue database
  CFNDatabaseFlights:
    Type: AWS::Glue::Database
    Properties:
      # The database is created in the Data Catalog for your account
      CatalogId: !Ref AWS::AccountId
      DatabaseInput:
        # The name of the database is defined in the Parameters section above
        Name: !Ref CFNDatabaseName
        Description: Database to hold tables for flights data
        LocationUri: s3://crawler-public-us-east-1/flight/2016/csv/
        #Parameters: Leave AWS database parameters blank
```

Exemple de CloudFormation modèle pour une AWS Glue base de données, une table et une partition

Une table AWS Glue contient les métadonnées qui définissent la structure et l'emplacement des données que vous souhaitez traiter avec vos scripts ETL. Dans une table, vous pouvez définir des partitions pour mettre en parallèle le traitement de vos données. Une partition est un fragment de données que vous avez défini avec une clé. Par exemple, en utilisant un mois en tant que clé, toutes

les données de janvier se trouvent dans la même partition. Dans AWS Glue, les bases de données peuvent contenir des tables et les tables peuvent contenir des partitions.

L'exemple suivant montre comment remplir une base de données, une table et des partitions à l'aide d'un modèle CloudFormation . Le format des données de base est `csv` et délimité par une virgule (,). Comme une base de données doit exister avant qu'elle ne puisse contenir une table et qu'une table doit exister avant que les partitions ne puissent être créées, le modèle utilise l'instruction `DependsOn` pour définir la dépendance de ces objets lorsqu'ils sont créés.

Les valeurs de cet exemple définissent une table qui contient les données de vol à partir d'un compartiment Amazon S3 disponible publiquement. À des fins d'illustration, quelques colonnes de données et une clé de partitionnement sont définies. Quatre partitions sont également définies dans le catalogue de données. Certains champs pour décrire le stockage des données de base sont également affichés dans les champs `StorageDescriptor`.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CloudFormation template in YAML to demonstrate creating a database, a table,
 and partitions
# The metadata created in the Data Catalog points to the flights public S3 bucket
#
# Parameters substituted in the Resources section
# These parameters are names of the resources created in the Data Catalog
Parameters:
  CFNDatabaseName:
    Type: String
    Default: cfn-database-flights-1
  CFNTableName1:
    Type: String
    Default: cfn-manual-table-flights-1
# Resources to create metadata in the Data Catalog
Resources:
###
# Create an AWS Glue database
  CFNDatabaseFlights:
    Type: AWS::Glue::Database
    Properties:
      CatalogId: !Ref AWS::AccountId
      DatabaseInput:
        Name: !Ref CFNDatabaseName
        Description: Database to hold tables for flights data
```

```
###
# Create an AWS Glue table
  CFNTableFlights:
    # Creating the table waits for the database to be created
    DependsOn: CFNDatabaseFlights
    Type: AWS::Glue::Table
    Properties:
      CatalogId: !Ref AWS::AccountId
      DatabaseName: !Ref CFNDatabaseName
      TableInput:
        Name: !Ref CFNTableName1
        Description: Define the first few columns of the flights table
        TableType: EXTERNAL_TABLE
        Parameters: {
    "classification": "csv"
  }
#       ViewExpandedText: String
        PartitionKeys:
        # Data is partitioned by month
        - Name: mon
          Type: bigint
        StorageDescriptor:
          OutputFormat: org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat
          Columns:
          - Name: year
            Type: bigint
          - Name: quarter
            Type: bigint
          - Name: month
            Type: bigint
          - Name: day_of_month
            Type: bigint
          InputFormat: org.apache.hadoop.mapred.TextInputFormat
          Location: s3://crawler-public-us-east-1/flight/2016/csv/
          SerdeInfo:
            Parameters:
              field.delim: ","
            SerializationLibrary: org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
# Partition 1
# Create an AWS Glue partition
  CFNPartitionMon1:
    DependsOn: CFNTableFlights
    Type: AWS::Glue::Partition
    Properties:
```

```
      CatalogId: !Ref AWS::AccountId
      DatabaseName: !Ref CFNDatabaseName
      TableName: !Ref CFNTableName1
      PartitionInput:
        Values:
        - 1
        StorageDescriptor:
          OutputFormat: org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat
          Columns:
          - Name: mon
            Type: bigint
          InputFormat: org.apache.hadoop.mapred.TextInputFormat
          Location: s3://crawler-public-us-east-1/flight/2016/csv/mon=1/
          SerdeInfo:
            Parameters:
              field.delim: ","
            SerializationLibrary: org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
# Partition 2
# Create an AWS Glue partition
  CFNPartitionMon2:
    DependsOn: CFNTableFlights
    Type: AWS::Glue::Partition
    Properties:
      CatalogId: !Ref AWS::AccountId
      DatabaseName: !Ref CFNDatabaseName
      TableName: !Ref CFNTableName1
      PartitionInput:
        Values:
        - 2
        StorageDescriptor:
          OutputFormat: org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat
          Columns:
          - Name: mon
            Type: bigint
          InputFormat: org.apache.hadoop.mapred.TextInputFormat
          Location: s3://crawler-public-us-east-1/flight/2016/csv/mon=2/
          SerdeInfo:
            Parameters:
              field.delim: ","
            SerializationLibrary: org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
# Partition 3
# Create an AWS Glue partition
  CFNPartitionMon3:
    DependsOn: CFNTableFlights
```

```
    Type: AWS::Glue::Partition
    Properties:
      CatalogId: !Ref AWS::AccountId
      DatabaseName: !Ref CFNDatabaseName
      TableName: !Ref CFNTableName1
      PartitionInput:
        Values:
        - 3
        StorageDescriptor:
          OutputFormat: org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat
          Columns:
          - Name: mon
            Type: bigint
          InputFormat: org.apache.hadoop.mapred.TextInputFormat
          Location: s3://crawler-public-us-east-1/flight/2016/csv/mon=3/
          SerdeInfo:
            Parameters:
              field.delim: ","
            SerializationLibrary: org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
# Partition 4
# Create an AWS Glue partition
  CFNPartitionMon4:
    DependsOn: CFNTableFlights
    Type: AWS::Glue::Partition
    Properties:
      CatalogId: !Ref AWS::AccountId
      DatabaseName: !Ref CFNDatabaseName
      TableName: !Ref CFNTableName1
      PartitionInput:
        Values:
        - 4
        StorageDescriptor:
          OutputFormat: org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat
          Columns:
          - Name: mon
            Type: bigint
          InputFormat: org.apache.hadoop.mapred.TextInputFormat
          Location: s3://crawler-public-us-east-1/flight/2016/csv/mon=4/
          SerdeInfo:
            Parameters:
              field.delim: ","
            SerializationLibrary: org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
```

Exemple de CloudFormation modèle pour un AWS Glue classificateur Grok

Un classifieur AWS Glue détermine le schéma de vos données. Un type de classifieur utilise un modèle grok personnalisé pour mettre vos données en correspondance. Si le modèle correspond, le classificateur personnalisé est utilisé pour créer le schéma de votre table et définir la `classification` avec la valeur définie dans la définition du classificateur.

Cet exemple crée un classificateur qui crée un schéma avec une colonne nommée `message` et définit la classification avec la valeur `greedy`.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating a classifier
#
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:

# The name of the classifier to be created
  CFNClassifierName:
    Type: String
    Default: cfn-classifier-grok-one-column-1

#
#
# Resources section defines metadata for the Data Catalog
Resources:
# Create classifier that uses grok pattern to put all data in one column and classifies
 it as "greedy".
  CFNClassifierFlights:
    Type: AWS::Glue::Classifier
    Properties:
      GrokClassifier:
        #Grok classifier that puts all data in one column
        Name: !Ref CFNClassifierName
        Classification: greedy

        GrokPattern: "%{GREEDYDATA:message}"
        #CustomPatterns: none
```

Exemple de CloudFormation modèle pour un classificateur AWS Glue JSON

Un classifieur AWS Glue détermine le schéma de vos données. Un type de classificateur personnalisé utilise une `JsonPath` chaîne définissant les données JSON à classer par le classificateur. AWS Glueprend en charge un sous-ensemble d'opérateurs pour`JsonPath`, comme décrit dans la section Écrire des JsonPath classificateurs personnalisés.

Si le modèle correspond, le classifieur personnalisé est utilisé pour créer le schéma de votre table.

Cet exemple crée un classifieur qui crée un schéma à chaque enregistrement du tableau `Records3` dans un objet.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating a JSON classifier
#
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:

# The name of the classifier to be created
  CFNClassifierName:
    Type: String
    Default: cfn-classifier-json-one-column-1

#
#
# Resources section defines metadata for the Data Catalog
Resources:
# Create classifier that uses a JSON pattern.
  CFNClassifierFlights:
    Type: AWS::Glue::Classifier
    Properties:
      JSONClassifier:
        #JSON classifier
        Name: !Ref CFNClassifierName
        JsonPath: $.Records3[*]
```

Exemple CloudFormation de modèle pour un classificateur AWS Glue XML

Un classifieur AWS Glue détermine le schéma de vos données. Un type de classifieur personnalisé spécifie une balise XML pour désigner l'élément qui contient chaque enregistrement dans un document XML en cours d'analyse. Si le modèle correspond, le classificateur personnalisé est utilisé pour créer le schéma de votre table et définir la `classification` avec la valeur définie dans la définition du classificateur.

Cet exemple crée un classifieur qui crée un schéma avec chaque enregistrement dans la balise `Record` et définit la classification sur `XML`.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating an XML classifier
#
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:

# The name of the classifier to be created
  CFNClassifierName:
    Type: String
    Default: cfn-classifier-xml-one-column-1

#
#
# Resources section defines metadata for the Data Catalog
Resources:
# Create classifier that uses the XML pattern and classifies it as "XML".
  CFNClassifierFlights:
    Type: AWS::Glue::Classifier
    Properties:
      XMLClassifier:
        #XML classifier
        Name: !Ref CFNClassifierName
        Classification: XML
        RowTag: <Records>
```

Exemple de CloudFormation modèle pour un AWS Glue robot d'exploration pour Amazon S3

Un crawler AWS Glue crée les tables de métadonnées dans votre catalogue de données correspondant à vos données. Vous pouvez ensuite utiliser ces définitions de table en tant que sources et cibles dans vos tâches ETL.

Cet exemple crée un crawler, le rôle IAM nécessaire et une base de données AWS Glue dans le catalogue de données. Lorsque cet crawler est exécuté, il endosse le rôle IAM et crée une table dans la base de données pour les données de vol publiques. La table est créée avec le préfixe « `cfn_sample_1_` ». Le rôle IAM créé par ce modèle permet des autorisations globales ; il se peut que vous vouliez créer un rôle personnalisé. Aucun classifieur personnalisé n'est défini par ce classifieur. Les classifieurs AWS Glue intégrés sont utilisées par défaut.

Lorsque vous soumettez cet exemple à la CloudFormation console, vous devez confirmer que vous souhaitez créer le rôle IAM.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating a crawler
#
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:

# The name of the crawler to be created
  CFNCrawlerName:
    Type: String
    Default: cfn-crawler-flights-1
  CFNDatabaseName:
    Type: String
    Default: cfn-database-flights-1
  CFNTablePrefixName:
    Type: String
    Default: cfn_sample_1_
#
#
# Resources section defines metadata for the Data Catalog
Resources:
```

```
#Create IAM Role assumed by the crawler. For demonstration, this role is given all
 permissions.
  CFNRoleFlights:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
        Statement:
          -
            Effect: "Allow"
            Principal:
              Service:
                - "glue.amazonaws.com"
            Action:
              - "sts:AssumeRole"
      Path: "/"
      Policies:
        -
          PolicyName: "root"
          PolicyDocument:
            Version: "2012-10-17"
            Statement:
              -
                Effect: "Allow"
                Action: "*"
                Resource: "*"
 # Create a database to contain tables created by the crawler
  CFNDatabaseFlights:
    Type: AWS::Glue::Database
    Properties:
      CatalogId: !Ref AWS::AccountId
      DatabaseInput:
        Name: !Ref CFNDatabaseName
        Description: "AWS Glue container to hold metadata tables for the flights
crawler"
 #Create a crawler to crawl the flights data on a public S3 bucket
  CFNCrawlerFlights:
    Type: AWS::Glue::Crawler
    Properties:
      Name: !Ref CFNCrawlerName
      Role: !GetAtt CFNRoleFlights.Arn
      #Classifiers: none, use the default classifier
      Description: AWS Glue crawler to crawl flights data
      #Schedule: none, use default run-on-demand
```

```
      DatabaseName: !Ref CFNDatabaseName
      Targets:
        S3Targets:
          # Public S3 bucket with the flights data
          - Path: "s3://crawler-public-us-east-1/flight/2016/csv"
      TablePrefix: !Ref CFNTablePrefixName
      SchemaChangePolicy:
        UpdateBehavior: "UPDATE_IN_DATABASE"
        DeleteBehavior: "LOG"
      Configuration: "{\"Version\":1.0,\"CrawlerOutput\":{\"Partitions\":
{\"AddOrUpdateBehavior\":\"InheritFromTable\"},\"Tables\":{\"AddOrUpdateBehavior\":
\"MergeNewColumns\"}}}"
```

Exemple CloudFormation de modèle pour une AWS Glue connexion

Une connexion AWS Glue dans le catalogue de données contient les informations JDBC et réseau nécessaires pour se connecter à une base de données JDBC. Ces informations sont utilisées lorsque vous vous connectez à une base de données JDBC pour analyser ou exécuter des tâches ETL.

Cet exemple crée une connexion à une base de données MySQL Amazon RDS nommée `devdb`. Lorsque cette connexion est utilisée, un rôle IAM, les informations d'identification de base de données et les valeurs de connexion réseau doivent également être fournies. Consultez les détails des champs nécessaires du modèle.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating a connection
#
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:

# The name of the connection to be created
  CFNConnectionName:
    Type: String
    Default: cfn-connection-mysql-flights-1
  CFNJDBCString:
    Type: String
```

```
    Default: "jdbc:mysql://xxx-mysql.yyyyyyyyyyyyyy.us-east-1.rds.amazonaws.com:3306/
devdb"
  CFNJDBCUser:
    Type: String
    Default: "master"
  CFNJDBCPassword:
    Type: String
    Default: "12345678"
    NoEcho: true
#
#
# Resources section defines metadata for the Data Catalog
Resources:
  CFNConnectionMySQL:
    Type: AWS::Glue::Connection
    Properties:
      CatalogId: !Ref AWS::AccountId
      ConnectionInput:
        Description: "Connect to MySQL database."
        ConnectionType: "JDBC"
        #MatchCriteria: none
        PhysicalConnectionRequirements:
          AvailabilityZone: "us-east-1d"
          SecurityGroupIdList:
           - "sg-7d52b812"
          SubnetId: "subnet-84f326ee"
        ConnectionProperties: {
          "JDBC_CONNECTION_URL": !Ref CFNJDBCString,
          "USERNAME": !Ref CFNJDBCUser,
          "PASSWORD": !Ref CFNJDBCPassword
        }
        Name: !Ref CFNConnectionName
```

Exemple de CloudFormation modèle pour une intégration AWS Glue sans ETL

AWSZero-ETL est un ensemble d'intégrations entièrement gérées qui minimisent le besoin de créer des pipelines de données ETL pour les cas d'utilisation courants d'ingestion et de réplication.

Cet exemple crée une intégration zéro ETL entre la source donnée et la cible.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating a zero-ETL integration in AWS Glue
#
# Parameters section contains names that are substituted in the Resources section
#
Parameters:

  # The name of the zero-ETL integration to be created
  IntegrationName:
    Type: String
  # The ARN for the source of the zero-ETL integration
  SourceArn:
    Type: String
  # The ARN for the target of the zero-ETL integration
  TargetArn:
    Type: String
#
#
Resources:
# Create an AWS Glue zero-ETL integration
  GlueIntegration:
    Type: AWS::Glue::Integration
    Properties:
      IntegrationName: !Ref IntegrationName
      Description: "AWS Glue zero-ETL integration"
      SourceArn: !Ref SourceArn
      TargetArn: !Ref TargetArn
      DataFilter: "include:table1"
      Tags:
        - Key: Purpose
          Value: GlueZeroETLIntegration
```

Exemple de CloudFormation modèle pour une intégration AWS Glue zéro ETL avec les propriétés des ressources d'intégration

Une intégration AWS Glue zéro ETL nécessite que les propriétés des ressources soient définies pour la source et la cible. Pour la source, la seule propriété à définir est le rôle IAM que l'intégration utilisera pour accéder à la AWS Glue connexion ou à la base de données DynamoDB. Pour la cible, les propriétés qui peuvent être configurées incluent le rôle IAM qui sera utilisé pour accéder à la cible, le réseau VPC dans lequel l'intégration doit être créée, le bus d'événements qui sera utilisé

pour configurer les notifications d'événements pour l'intégration et la clé KMS qui sera utilisée pour le chiffrement des données.

L'exemple ci-dessous définit les propriétés des ressources source et cible, puis crée une intégration zéro ETL entre la source et la cible.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate defining the integration resource properties and then
 creating a zero-ETL integration in AWS Glue
#
# Parameters section contains names that are substituted in the Resources section
#
Parameters:
  #The name of the zero-ETL integration to be created
  IntegrationName:
    Type: String
  # The ARN for the target of the zero-ETL integration
  TargetArn:
    Type: String
  # The ARN for the IAM role that will be used to access the target
  TargetRoleArn:
    Type: String
  # The ARN for the source of the zero-ETL integration
  SourceArn:
    Type: String
  # The ARN for the IAM role that will be used to access thesource
  SourceRoleArn:
    Type: String
#
#
Resources:
  # Integration Resource Property for zero-ETL target
  TargetIntegrationResourceProperty:
    Type: AWS::Glue::IntegrationResourceProperty
    Properties:
      ResourceArn: !Ref TargetArn
      TargetProcessingProperties:
        RoleArn: !Ref TargetRoleArn
      Tags:
        - Key: Purpose
          Value: TargetIrpTag
```

```
# Integration Resource Property for zero-ETL source
SourceIntegrationResourceProperty:
  Type: AWS::Glue::IntegrationResourceProperty
  Properties:
    ResourceArn: !Ref SourceArn
    SourceProcessingProperties:
      RoleArn: !Ref SourceRoleArn
    Tags:
      - Key: Purpose
        Value: SourceIRPTag

# Create an AWS Glue zero-ETL integration
GlueIntegration:
  Type: AWS::Glue::Integration
  Properties:
    IntegrationName: !Ref IntegrationName
    Description: "AWS Glue zero-ETL integration"
    SourceArn: !Ref SourceArn
    TargetArn: !Ref TargetArn
    DataFilter: "include:table1"
    Tags:
      - Key: Purpose
        Value: GlueZeroETLIntegration
```

Exemple de CloudFormation modèle pour un AWS Glue crawler pour JDBC

Un analyseur AWS Glue crée les tables de métadonnées dans votre catalogue de données correspondant à vos données. Vous pouvez ensuite utiliser ces définitions de table en tant que sources et cibles dans vos tâches ETL.

Cet exemple crée un crawler, le rôle IAM nécessaire et une base de données AWS Glue dans le catalogue de données. Lorsque cet crawler est exécuté, il endosse le rôle IAM et crée une table dans la base de données pour les données de vol publiques qui ont été stockées dans une base de données MySQL. La table est créée avec le préfixe « `cfn_jdbc_1_` ». Le rôle IAM créé par ce modèle permet des autorisations globales ; il se peut que vous vouliez créer un rôle personnalisé. Aucun classifieur personnalisé ne peur être défini pour les données JDBC. Les classifieurs AWS Glue intégrés sont utilisées par défaut.

Lorsque vous soumettez cet exemple à la CloudFormation console, vous devez confirmer que vous souhaitez créer le rôle IAM.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating a crawler
#
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:

# The name of the crawler to be created
  CFNCrawlerName:
    Type: String
    Default: cfn-crawler-jdbc-flights-1
# The name of the database to be created to contain tables
  CFNDatabaseName:
    Type: String
    Default: cfn-database-jdbc-flights-1
# The prefix for all tables crawled and created
  CFNTablePrefixName:
    Type: String
    Default: cfn_jdbc_1_
# The name of the existing connection to the MySQL database
  CFNConnectionName:
    Type: String
    Default: cfn-connection-mysql-flights-1
# The name of the JDBC path (database/schema/table) with wildcard (%) to crawl
  CFNJDBCPath:
    Type: String
    Default: saldev/%
#
#
# Resources section defines metadata for the Data Catalog
Resources:
#Create IAM Role assumed by the crawler. For demonstration, this role is given all
 permissions.
  CFNRoleFlights:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
```

```
        Statement:
          -
            Effect: "Allow"
            Principal:
              Service:
                - "glue.amazonaws.com"
            Action:
              - "sts:AssumeRole"
      Path: "/"
      Policies:
        -
          PolicyName: "root"
          PolicyDocument:
            Version: "2012-10-17"
            Statement:
              -
                Effect: "Allow"
                Action: "*"
                Resource: "*"
# Create a database to contain tables created by the crawler
 CFNDatabaseFlights:
   Type: AWS::Glue::Database
   Properties:
     CatalogId: !Ref AWS::AccountId
     DatabaseInput:
       Name: !Ref CFNDatabaseName
       Description: "AWS Glue container to hold metadata tables for the flights
crawler"
#Create a crawler to crawl the flights data in MySQL database
 CFNCrawlerFlights:
   Type: AWS::Glue::Crawler
   Properties:
     Name: !Ref CFNCrawlerName
     Role: !GetAtt CFNRoleFlights.Arn
     #Classifiers: none, use the default classifier
     Description: AWS Glue crawler to crawl flights data
     #Schedule: none, use default run-on-demand
     DatabaseName: !Ref CFNDatabaseName
     Targets:
       JdbcTargets:
         # JDBC MySQL database with the flights data
         - ConnectionName: !Ref CFNConnectionName
           Path: !Ref CFNJDBCPath
         #Exclusions: none
```

```
      TablePrefix: !Ref CFNTablePrefixName
      SchemaChangePolicy:
        UpdateBehavior: "UPDATE_IN_DATABASE"
        DeleteBehavior: "LOG"
   Configuration: "{\"Version\":1.0,\"CrawlerOutput\":{\"Partitions\":
{\"AddOrUpdateBehavior\":\"InheritFromTable\"},\"Tables\":{\"AddOrUpdateBehavior\":
\"MergeNewColumns\"}}}"
```

Exemple CloudFormation de modèle de AWS Glue tâche pour Amazon S3 vers Amazon S3

Une tâche AWS Glue dans le catalogue de données contient les valeurs des paramètres qui sont requises pour exécuter un script dans AWS Glue.

Cet exemple crée une tâche qui lit les données de vol à partir d'un compartiment Amazon S3 dans un format `csv` et les écrit dans un fichier Parquet Amazon S3. Le script qui est exécuté par cette tâche doit déjà exister. Vous pouvez générer un script ETL pour votre environnement avec la console AWS Glue. Lorsque la tâche est exécutée, un rôle IAM avec les autorisations appropriées doit également être fourni.

Les valeurs courantes des paramètres sont affichées dans le modèle. Par exemple, `AllocatedCapacity` (DPUs) prend par défaut la valeur 5.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating a job using the public flights S3 table in a
 public bucket
#
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:

# The name of the job to be created
  CFNJobName:
    Type: String
    Default: cfn-job-S3-to-S3-2
# The name of the IAM role that the job assumes. It must have access to data, script,
 temporary directory
  CFNIAMRoleName:
    Type: String
```

```
    Default: AWSGlueServiceRoleGA
# The S3 path where the script for this job is located
  CFNScriptLocation:
    Type: String
    Default: s3://aws-glue-scripts-123456789012-us-east-1/myid/sal-job-test2
#
#
# Resources section defines metadata for the Data Catalog
Resources:
# Create job to run script which accesses flightscsv table and write to S3 file as
 parquet.
# The script already exists and is called by this job
  CFNJobFlights:
    Type: AWS::Glue::Job
    Properties:
      Role: !Ref CFNIAMRoleName
      #DefaultArguments: JSON object
      # If script written in Scala, then set DefaultArguments={'--job-language';
 'scala', '--class': 'your scala class'}
      #Connections:  No connection needed for S3 to S3 job
      #  ConnectionsList
      #MaxRetries: Double
      Description: Job created with CloudFormation
      #LogUri: String
      Command:
        Name: glueetl
        ScriptLocation: !Ref CFNScriptLocation
             # for access to directories use proper IAM role with permission to buckets
 and folders that begin with "aws-glue-"
             # script uses temp directory from job definition if required (temp
 directory not used S3 to S3)
             # script defines target for output as s3://aws-glue-target/sal
      AllocatedCapacity: 5
      ExecutionProperty:
        MaxConcurrentRuns: 1
      Name: !Ref CFNJobName
```

Exemple CloudFormation de modèle de AWS Glue tâche pour JDBC sur Amazon S3

Une tâche AWS Glue dans le catalogue de données contient les valeurs des paramètres qui sont requises pour exécuter un script dans AWS Glue.

Cet exemple crée une tâche qui lit les données de vol à partir d'une base de données JDBC MySQL comme définie par la connexion nommée `cfn-connection-mysql-flights-1` et les écrit dans un fichier Parquet Amazon S3. Le script qui est exécuté par cette tâche doit déjà exister. Vous pouvez générer un script ETL pour votre environnement avec la console AWS Glue. Lorsque la tâche est exécutée, un rôle IAM avec les autorisations appropriées doit également être fourni.

Les valeurs courantes des paramètres sont affichées dans le modèle. Par exemple, `AllocatedCapacity` (DPUs) prend par défaut la valeur 5.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating a job using a MySQL JDBC DB with the flights
 data to an S3 file
#
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:

# The name of the job to be created
  CFNJobName:
    Type: String
    Default: cfn-job-JDBC-to-S3-1
# The name of the IAM role that the job assumes. It must have access to data, script,
 temporary directory
  CFNIAMRoleName:
    Type: String
    Default: AWSGlueServiceRoleGA
# The S3 path where the script for this job is located
  CFNScriptLocation:
    Type: String
    Default: s3://aws-glue-scripts-123456789012-us-east-1/myid/sal-job-dec4a
# The name of the connection used for JDBC data source
  CFNConnectionName:
    Type: String
    Default: cfn-connection-mysql-flights-1
#
#
# Resources section defines metadata for the Data Catalog
Resources:
# Create job to run script which accesses JDBC flights table via a connection and write
 to S3 file as parquet.
# The script already exists and is called by this job
```

```
  CFNJobFlights:
    Type: AWS::Glue::Job
    Properties:
      Role: !Ref CFNIAMRoleName
      #DefaultArguments: JSON object
      # For example, if required by script, set temporary directory as
DefaultArguments={'--TempDir'; 's3://aws-glue-temporary-xyc/sal'}
      Connections:
        Connections:
        - !Ref CFNConnectionName
      #MaxRetries: Double
      Description: Job created with CloudFormation using existing script
      #LogUri: String
      Command:
        Name: glueetl
        ScriptLocation: !Ref CFNScriptLocation
             # for access to directories use proper IAM role with permission to buckets
and folders that begin with "aws-glue-"
             # if required, script defines temp directory as argument TempDir and used
in script like redshift_tmp_dir = args["TempDir"]
             # script defines target for output as s3://aws-glue-target/sal
      AllocatedCapacity: 5
      ExecutionProperty:
        MaxConcurrentRuns: 1
      Name: !Ref CFNJobName
```

Exemple de CloudFormation modèle pour un déclencheur AWS Glue à la demande

Un déclencheur AWS Glue du catalogue de données contient les valeurs des paramètres qui sont requises pour démarrer une tâche lorsque le déclencheur est exécuté. Un déclencheur à la demande s'exécute lorsque vous l'activez.

Cet exemple crée un déclencheur à la demande qui démarre une tâche nommée `cfn-job-S3-to-S3-1`.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating an on-demand trigger
#
```

```
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:
  # The existing job to be started by this trigger
  CFNJobName:
    Type: String
    Default: cfn-job-S3-to-S3-1
  # The name of the trigger to be created
  CFNTriggerName:
    Type: String
    Default: cfn-trigger-ondemand-flights-1
#
# Resources section defines metadata for the Data Catalog
# Sample CFN YAML to demonstrate creating an on-demand trigger for a job
Resources:
# Create trigger to run an existing job (CFNJobName) on an on-demand schedule.
  CFNTriggerSample:
    Type: AWS::Glue::Trigger
    Properties:
      Name:
        Ref: CFNTriggerName
      Description: Trigger created with CloudFormation
      Type: ON_DEMAND
      Actions:
        - JobName: !Ref CFNJobName
        # Arguments: JSON object
      #Schedule:
      #Predicate:
```

Exemple CloudFormation de modèle pour un déclencheur AWS Glue planifié

Un déclencheur AWS Glue du catalogue de données contient les valeurs des paramètres qui sont requises pour démarrer une tâche lorsque le déclencheur est exécuté. Un déclencheur planifié est exécuté lorsqu'il est activé et que le minuteur cron s'affiche.

Cet exemple crée un déclencheur planifié qui démarre une tâche nommée `cfn-job-S3-to-S3-1`. Le minuteur est une expression cron qui permet d'exécuter la tâche toutes les 10 minutes du lundi au vendredi.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating a scheduled trigger
#
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:
  # The existing job to be started by this trigger
  CFNJobName:
    Type: String
    Default: cfn-job-S3-to-S3-1
  # The name of the trigger to be created
  CFNTriggerName:
    Type: String
    Default: cfn-trigger-scheduled-flights-1
#
# Resources section defines metadata for the Data Catalog
# Sample CFN YAML to demonstrate creating a scheduled trigger for a job
#
Resources:
# Create trigger to run an existing job (CFNJobName) on a cron schedule.
  TriggerSample1CFN:
    Type: AWS::Glue::Trigger
    Properties:
      Name:
        Ref: CFNTriggerName
      Description: Trigger created with CloudFormation
      Type: SCHEDULED
      Actions:
        - JobName: !Ref CFNJobName
        # Arguments: JSON object
      # # Run the trigger every 10 minutes on Monday to Friday
      Schedule: cron(0/10 * ? * MON-FRI *)
      #Predicate:
```

Exemple de CloudFormation modèle pour un déclencheur AWS Glue conditionnel

Un déclencheur AWS Glue du catalogue de données contient les valeurs des paramètres qui sont requises pour démarrer une tâche lorsque le déclencheur est exécuté. Un déclencheur conditionnel

est exécuté lorsqu'il est activé et que ses conditions sont satisfaites, comme un travail terminé avec succès.

Cet exemple crée un déclencheur conditionnel qui démarre une tâche nommée `cfn-job-S3-to-S3-1`. Cette tâche commence lorsque la tâche nommée `cfn-job-S3-to-S3-2` se termine correctement.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating a conditional trigger for a job, which starts
 when another job completes
#
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:
  # The existing job to be started by this trigger
  CFNJobName:
    Type: String
    Default: cfn-job-S3-to-S3-1
  # The existing job that when it finishes causes trigger to fire
  CFNJobName2:
    Type: String
    Default: cfn-job-S3-to-S3-2
  # The name of the trigger to be created
  CFNTriggerName:
    Type: String
    Default: cfn-trigger-conditional-1
#
Resources:
# Create trigger to run an existing job (CFNJobName) when another job completes
 (CFNJobName2).
  CFNTriggerSample:
    Type: AWS::Glue::Trigger
    Properties:
      Name:
        Ref: CFNTriggerName
      Description: Trigger created with CloudFormation
      Type: CONDITIONAL
      Actions:
        - JobName: !Ref CFNJobName
        # Arguments: JSON object
      #Schedule: none
```

```
      Predicate:
        #Value for Logical is required if more than 1 job listed in Conditions
        Logical: AND
        Conditions:
          - LogicalOperator: EQUALS
            JobName: !Ref CFNJobName2
            State: SUCCEEDED
```

Exemple de CloudFormation modèle pour un point de terminaison AWS Glue de développement

Une transformation de machine learning AWS Glue est une transformation personnalisée destinée à nettoyer vos données. Il existe actuellement une transformation disponible nommée FindMatches. La FindMatches transformation vous permet d'identifier les enregistrements dupliqués ou correspondants dans votre ensemble de données, même lorsque les enregistrements n'ont pas d'identifiant unique commun et qu'aucun champ ne correspond exactement.

Cet exemple crée une transformation de machine learning. Pour plus d'informations sur les paramètres dont vous avez besoin pour créer une transformation de machine learning, consultez Record correspondant à AWS Lake Formation FindMatches.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating a machine learning transform
#
# Resources section defines metadata for the machine learning transform
Resources:
  MyMLTransform:
    Type: "AWS::Glue::MLTransform"
    Condition: "isGlueMLGARegion"
    Properties:
      Name: !Sub "MyTransform"
      Description: "The bestest transform ever"
      Role: !ImportValue MyMLTransformUserRole
      GlueVersion: "1.0"
      WorkerType: "Standard"
      NumberOfWorkers: 5
      Timeout: 120
      MaxRetries: 1
      InputRecordTables:
```

```
        GlueTables:
          - DatabaseName: !ImportValue MyMLTransformDatabase
            TableName: !ImportValue MyMLTransformTable
      TransformParameters:
        TransformType: "FIND_MATCHES"
        FindMatchesParameters:
          PrimaryKeyColumnName: "testcolumn"
          PrecisionRecallTradeoff: 0.5
          AccuracyCostTradeoff: 0.5
          EnforceProvidedLabels: True
      Tags:
        key1: "value1"
        key2: "value2"
      TransformEncryption:
        TaskRunSecurityConfigurationName: !ImportValue
 MyMLTransformSecurityConfiguration
        MLUserDataEncryption:
          MLUserDataEncryptionMode: "SSE-KMS"
          KmsKeyId: !ImportValue MyMLTransformEncryptionKey
```

Exemple de CloudFormation modèle pour un ensemble de AWS Glue Data Quality règles

Un ensemble de règles relatives à la qualité des AWS Glue données contient des règles qui peuvent être évaluées sur une table du catalogue de données. Une fois que l'ensemble de règles est placé sur votre table cible, vous pouvez accéder au catalogue de données et exécuter une évaluation qui confronte vos données aux règles de l'ensemble de règles. Ces règles peuvent aller de l'évaluation du nombre de lignes à l'évaluation de l'intégrité référentielle de vos données.

L'exemple suivant est un CloudFormation modèle qui crée un ensemble de règles avec une variété de règles sur la table cible spécifiée.

```
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating a DataQualityRuleset
#
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:

  # The name of the ruleset to be created
  RulesetName:
```

```
    Type: String
    Default: "CFNRulesetName"
  RulesetDescription:
    Type: String
    Default: "CFN DataQualityRuleset"
  # Rules that will be associated with this ruleset
  Rules:
    Type: String
    Default: 'Rules = [
        RowCount > 100,
        IsUnique "id",
        IsComplete "nametype"
        ]'
  # Name of database and table within Data Catalog which the ruleset will
  # be applied too
  DatabaseName:
    Type: String
    Default: "ExampleDatabaseName"
  TableName:
    Type: String
    Default: "ExampleTableName"

# Resources section defines metadata for the Data Catalog
Resources:
  # Creates a Data Quality ruleset under specified rules
  DQRuleset:
    Type: AWS::Glue::DataQualityRuleset
    Properties:
      Name: !Ref RulesetName
      Description: !Ref RulesetDescription
      # The String within rules must be formatted in DQDL, a language
      # used specifically to make rules
      Ruleset: !Ref Rules
      # The targeted table must exist within Data Catalog alongside
      # the correct database
      TargetTable:
        DatabaseName: !Ref DatabaseName
        TableName: !Ref TableName
```

Exemple de CloudFormation modèle pour un ensemble de AWS Glue Data Quality règles avec planificateur EventBridge

Un ensemble de règles relatives à la qualité des AWS Glue données contient des règles qui peuvent être évaluées sur une table du catalogue de données. Une fois que l'ensemble de règles est placé sur votre table cible, vous pouvez accéder au catalogue de données et exécuter une évaluation qui confronte vos données aux règles de l'ensemble de règles. Au lieu d'avoir à accéder manuellement au catalogue de données pour évaluer l'ensemble de règles, vous pouvez également ajouter un EventBridge planificateur dans notre CloudFormation modèle pour planifier ces évaluations d'ensembles de règles pour vous à intervalles réguliers.

L'exemple suivant est un CloudFormation modèle qui crée un ensemble de règles de qualité des données et un EventBridge planificateur pour évaluer l'ensemble de règles susmentionné toutes les cinq minutes.

```
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating a DataQualityRuleset
#
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:

  # The name of the ruleset to be created
  RulesetName:
    Type: String
    Default: "CFNRulesetName"
  # Rules that will be associated with this Ruleset
  Rules:
    Type: String
    Default: 'Rules = [
        RowCount > 100,
        IsUnique "id",
        IsComplete "nametype"
        ]'
  # The name of the Schedule to be created
  ScheduleName:
    Type: String
    Default: "ScheduleDQRulsetEvaluation"
  # This expression determines the rate at which the Schedule will evaluate
  # your data using the above ruleset
  ScheduleRate:
```

```
    Type: String
    Default: "rate(5 minutes)"
  # The Request that being sent must match the details of the Data Quality Ruleset
  ScheduleRequest:
    Type: String
    Default: '
        { "DataSource": { "GlueTable": { "DatabaseName": "ExampleDatabaseName",
         "TableName": "ExampleTableName" } },
         "Role": "role/AWSGlueServiceRoleDefault",
          "RulesetNames": [ ""CFNRulesetName"" ] }
        '

# Resources section defines metadata for the Data Catalog
Resources:
  # Creates a Data Quality ruleset under specified rules
  DQRuleset:
    Type: AWS::Glue::DataQualityRuleset
    Properties:
      Name: !Ref RulesetName
      Description: "CFN DataQualityRuleset"
      # The String within rules must be formatted in DQDL, a language
      # used specifically to make rules
      Ruleset: !Ref Rules
      # The targeted table must exist within Data Catalog alongside
      # the correct database
      TargetTable:
        DatabaseName: "ExampleDatabaseName"
        TableName: "ExampleTableName"
  # Create a Scheduler to schedule evaluation runs on the above ruleset
  ScheduleDQEval:
    Type: AWS::Scheduler::Schedule
    Properties:
      Name: !Ref ScheduleName
      Description: "Schedule DataQualityRuleset Evaluations"
      FlexibleTimeWindow:
        Mode: "OFF"
      ScheduleExpression: !Ref ScheduleRate
      ScheduleExpressionTimezone: "America/New_York"
      State: "ENABLED"
      Target:
        # The ARN is the API that will be run, since we want to evaluate our ruleset
        # we want this specific ARN
        Arn: "arn:aws:scheduler:::aws-sdk:glue:startDataQualityRulesetEvaluationRun"
        # Your RoleArn must have approval to schedule
```

```
        RoleArn: "arn:aws:iam::123456789012:role/AWSGlueServiceRoleDefault"
        # This is the Request that is being sent to the Arn
        Input: '
        { "DataSource": { "GlueTable": { "DatabaseName": "sampledb", "TableName":
 "meteorite" } },
         "Role": "role/AWSGlueServiceRoleDefault",
          "RulesetNames": [ "TestCFN" ] }
        '
```

Exemple de CloudFormation modèle pour un point de terminaison AWS Glue de développement

Un point de terminaison de développement AWS Glue est un environnement que vous pouvez utiliser pour développer et tester vos scripts AWS Glue.

Cet exemple crée un point de terminaison de développement avec les valeurs de paramètres réseau minimales requises pour le créer avec succès. Pour plus d'informations sur les paramètres dont vous avez besoin pour configurer un point de terminaison de développement, consultez Configuration du réseau pour le développement de AWS Glue.

Vous fournissez un ARN (Amazon Resource Name) de rôle IAM existant pour créer le point de terminaison de développement. Fournissez une clé publique RSA valide et conservez la clé privée correspondante disponible si vous prévoyez de créer un serveur de bloc-notes sur le point de terminaison de développement.

> **Note**
>
> Dans le cas d'un serveur de bloc-notes que vous créez qui est associé à un point de terminaison de développement, vous le gérez. Par conséquent, si vous supprimez le point de terminaison de développement, pour supprimer le serveur bloc-notes, vous devez supprimer la CloudFormation pile sur la CloudFormation console.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating a development endpoint
#
# Parameters section contains names that are substituted in the Resources section
```

```
# These parameters are the names the resources created in the Data Catalog
Parameters:

# The name of the crawler to be created
  CFNEndpointName:
    Type: String
    Default: cfn-devendpoint-1
  CFNIAMRoleArn:
    Type: String
    Default: arn:aws:iam::123456789012/role/AWSGlueServiceRoleGA
#
#
# Resources section defines metadata for the Data Catalog
Resources:
  CFNDevEndpoint:
    Type: AWS::Glue::DevEndpoint
    Properties:
      EndpointName: !Ref CFNEndpointName
      #ExtraJarsS3Path: String
      #ExtraPythonLibsS3Path: String
      NumberOfNodes: 5
      PublicKey: ssh-rsa public.....key myuserid-key
      RoleArn: !Ref CFNIAMRoleArn
      SecurityGroupIds:
        - sg-64986c0b
      SubnetId: subnet-c67cccac
```

AWS Glue guide de programmation

Un script contient le code qui extrait les données des sources, les transforme et les charge dans des cibles. AWS Glue exécute un script lorsqu'il démarre une tâche.

AWS Glue Les scripts ETL sont codés en Python ou en Scala. Bien que tous les types de tâches puissent être écrits en Python, les tâches AWS Glue pour Spark peuvent également être écrites en Scala. Lorsque vous générez automatiquement la logique du code source de votre tâche dans AWS Glue Studio, un script est créé. Vous pouvez modifier ce script ou fournir votre propre script personnalisé pour effectuer votre travail ETL.

Fournir vos propres scripts personnalisés

Les scripts exécutent les tâches d'extraction, de transformation et de chargement (ETL) AWS Glue. Un script est créé lorsque vous générez automatiquement la logique de code source pour une tâche. Vous pouvez modifier ce script généré ou fournir votre propre script personnalisé.

Pour fournir votre propre script personnalisé AWS Glue, suivez les étapes générales suivantes :

1. Connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/.
2. Choisissez l'onglet Tâches ETL, puis consultez la section Créer une tâche. Choisissez une option d'éditeur de script.
3. Sous This job runs (Cette tâche exécute), choisissez l'une des actions suivantes :

   - Créer un script avec du code standard
   - Charger et modifier un script existant
4. Sur la page Détails de la tâche, choisissez le rôle IAM requis pour l'exécution de votre script personnalisé. Pour de plus amples informations, veuillez consulter Gestion des identités et des accès pour AWS Glue.
5. Choisissez les connexions référencées par votre script. Ces objets sont requis pour la connexion aux magasins de données JDBC nécessaires.

   Une interface réseau Elastic est une interface réseau virtuelle que vous pouvez attacher à une instance dans un cloud privé virtuel (VPC). Choisissez l'interface réseau Elastic requise pour la connexion au magasin de données utilisé dans le script.

6. Fournissez une configuration supplémentaire, y compris des paramètres, spécifiques à votre type de tâche. Pour plus d'informations sur la configuration de votre type de tâche, consultez la section Création de tâches ETL visuelles.
7. Dans l'onglet Script, collez ou écrivez votre script personnalisé.

Utilisez le contenu de cette section pour vous guider dans le processus d'écriture de votre script personnalisé.

Pour plus d'informations sur l'ajout de tâches AWS Glue, consultezCréation de tâches ETL visuelles.

Pour step-by-step obtenir des conseils, consultez le didacticiel Ajouter une tâche dans la AWS Glue console.

Programmation de scripts Spark

AWS Glue facilite l'écriture ou la génération automatique de scripts d'extraction, de transformation et de chargement (ETL), en plus de les tester et de les exécuter. Cette section décrit les extensions pour Apache Spark introduites par AWS Glue et fournit des exemples de codage et d'exécution de scripts ETL en Python et en Scala.

> **Important**
>
> Les différentes versions de AWS Glue sont compatibles avec différentes versions d'Apache Spark. Votre script personnalisé doit être compatible avec la version Apache Spark prise en charge. Pour plus d'informations sur les versions d' AWS Glue, veuillez consulter Glue version job property.

Rubriques

- Tutoriel : Écrire un script AWS Glue for Spark
- Programmez les scripts ETL AWS Glue dans PySpark
- Programmation de scripts ETL AWS Glue dans Scala
- Fonctionnalités et optimisations de la programmation des scripts ETL AWS Glue pour Spark

Tutoriel : Écrire un script AWS Glue for Spark

Ce didacticiel vous présente le processus d'écriture des scripts AWS Glue. Vous pouvez exécuter des scripts selon un calendrier avec des tâches, ou de manière interactive avec des sessions interactives. Pour plus d'informations sur les tâches, consultez Création de tâches ETL visuelles. Pour plus d'informations sur les sessions interactives, consultez the section called "Présentation de séances interactives AWS Glue".

L'éditeur visuel AWS Glue Studio propose une interface graphique sans code pour créer des tâches AWS Glue. AWS Les scripts Glue soutiennent les tâches visuelles. Ils vous donnent accès à l'ensemble étendu d'outils disponibles pour travailler avec les programmes Apache Spark. Vous pouvez accéder aux bibliothèques natives Spark APIs et AWS Glue qui facilitent les flux de travail d'extraction, de transformation et de chargement (ETL) à partir d'un script AWS Glue.

Dans ce didacticiel, vous allez extraire, transformer et charger un jeu de données de tickets de stationnement. Le script qui effectue ce travail est identique en termes de forme et de fonction à celui généré dans Making ETL easy with AWS Glue Studio sur le blog AWS Big Data, qui présente l'éditeur visuel AWS Glue Studio. En exécutant ce script dans une tâche, vous pouvez le comparer à des tâches visuelles et voir comment fonctionnent les scripts AWS Glue ETL. Cela vous prépare à utiliser des fonctionnalités supplémentaires qui ne sont pas encore disponibles dans les tâches visuels.

Vous utilisez le langage et les bibliothèques Python dans ce didacticiel. Des fonctionnalités similaires sont disponibles dans Scala. Après avoir suivi ce didacticiel, vous devriez être en mesure de générer et d'inspecter un exemple de script Scala pour comprendre comment exécuter le processus d'écriture de script Scala AWS Glue ETL.

Conditions préalables

Ce didacticiel nécessite la configuration suivante :

- Les mêmes prérequis que dans le billet de blog de AWS Glue Studio, qui vous demande d'exécuter un CloudFormation modèle.

  Ce modèle utilise le catalogue de données AWS Glue pour gérer le jeu de données des tickets de stationnement disponible dans`s3://aws-bigdata-blog/artifacts/gluestudio/`. Il crée les ressources suivantes, qui seront référencées par la suite :

- Rôle AWS Glue Studio – Rôle IAM à exécuter les tâchesAWS Glue

- Compartiment Amazon S3 AWS Glue Studio – Nom du compartiment Amazon S3 pour stocker les fichiers liés aux blogs
- Billets YYZDB AWS Glue Studio – Base de données du catalogue de données AWS Glue
- AWS Glue StudioTableTickets— Table du catalogue de données à utiliser comme source
- AWS Glue StudioTableTrials— Table du catalogue de données à utiliser comme source
- AWS Glue StudioParkingTicketCount — Table du catalogue de données à utiliser comme destination
- Le script généré dans le billet de blog de AWS Glue Studio. Si des modifications sont apportées au billet de blog, le script est également disponible dans le texte suivant.

Génération d'un exemple de script

Vous pouvez utiliser l'éditeur visuel AWS Glue Studio comme un puissant outil de génération de code pour créer un échafaudage pour le script que vous souhaitez écrire. Vous allez utiliser cet outil pour créer un exemple de script.

Si vous voulez ignorer ces étapes, le script est fourni.

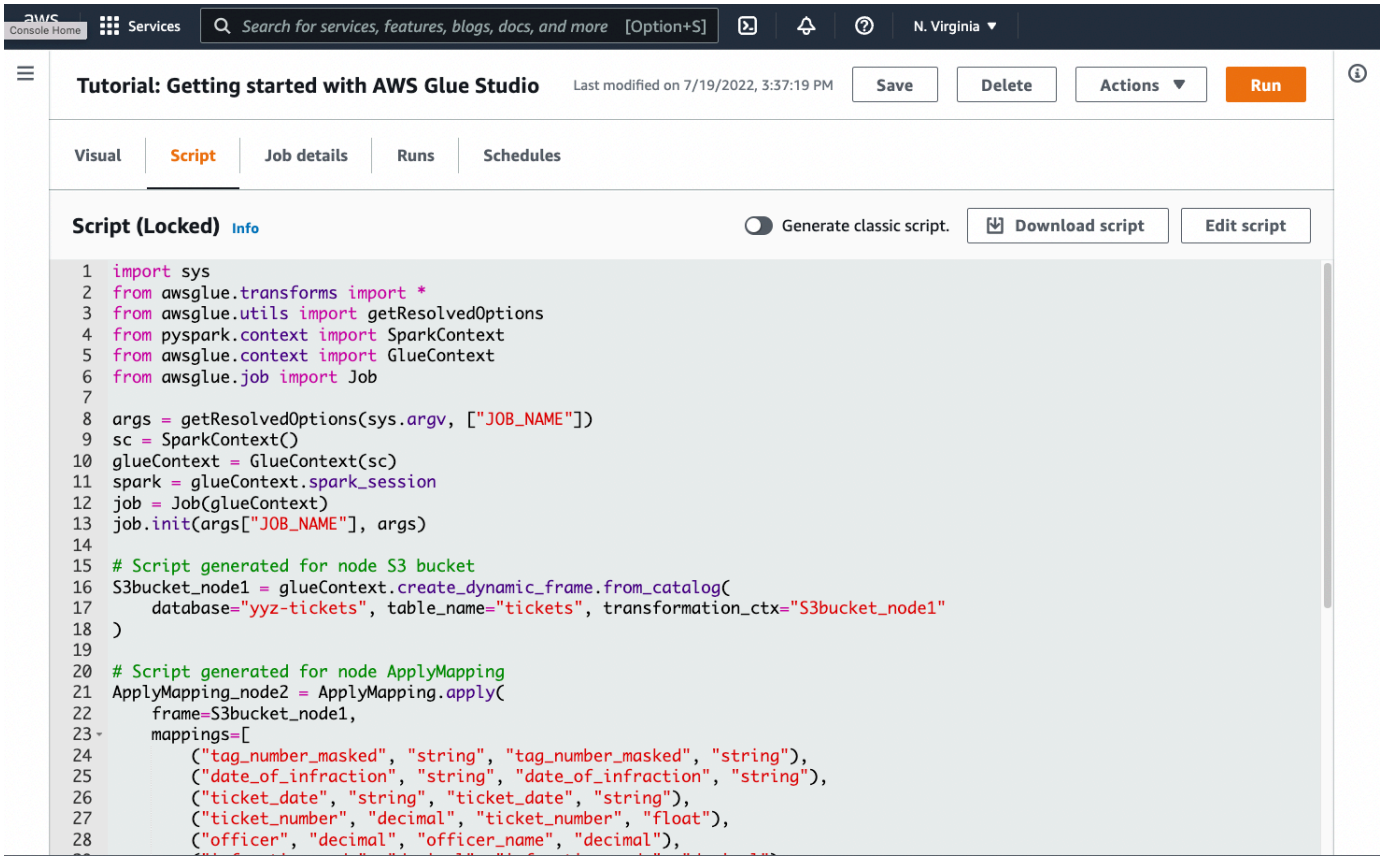
Exemple de script de didacticiel

```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job

args = getResolvedOptions(sys.argv, ["JOB_NAME"])
sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
job = Job(glueContext)
job.init(args["JOB_NAME"], args)

# Script generated for node S3 bucket
S3bucket_node1 = glueContext.create_dynamic_frame.from_catalog(
    database="yyz-tickets", table_name="tickets", transformation_ctx="S3bucket_node1"
)
```

```
# Script generated for node ApplyMapping
ApplyMapping_node2 = ApplyMapping.apply(
    frame=S3bucket_node1,
    mappings=[
        ("tag_number_masked", "string", "tag_number_masked", "string"),
        ("date_of_infraction", "string", "date_of_infraction", "string"),
        ("ticket_date", "string", "ticket_date", "string"),
        ("ticket_number", "decimal", "ticket_number", "float"),
        ("officer", "decimal", "officer_name", "decimal"),
        ("infraction_code", "decimal", "infraction_code", "decimal"),
        ("infraction_description", "string", "infraction_description", "string"),
        ("set_fine_amount", "decimal", "set_fine_amount", "float"),
        ("time_of_infraction", "decimal", "time_of_infraction", "decimal"),
    ],
    transformation_ctx="ApplyMapping_node2",
)

# Script generated for node S3 bucket
S3bucket_node3 = glueContext.write_dynamic_frame.from_options(
    frame=ApplyMapping_node2,
    connection_type="s3",
    format="glueparquet",
    connection_options={"path": "s3://DOC-EXAMPLE-BUCKET", "partitionKeys": []},
    format_options={"compression": "gzip"},
    transformation_ctx="S3bucket_node3",
)

job.commit()
```

Générer un exemple de script

1. Terminez le didacticiel AWS Glue Studio. Pour terminer ce didacticiel, voir Création d'une tâche dans AWS Glue Studio à partir d'un exemple de tâche.
2. Accédez à l'onglet Scriptsur la page de la tâche, comme illustré dans la capture d'écran suivante :



3. Copiez l'intégralité du contenu de l'onglet Script. En définissant le langage de script dans Job details (Détails de la tâche), vous pouvez alterner entre la génération de code Python ou Scala.

Étape 1. Créer une tâche et coller votre script

Au cours de cette étape, vous allez créer une tâche AWS Glue dans le AWS Management Console. Cela permet de configurer une configuration qui permet à AWS Glue d'exécuter votre script. Elle crée également un espace dans lequel vous pouvez stocker et modifier votre script.

Pour créer une tâche

1. Dans le AWS Management Console, accédez à la page d'accueil de AWS Glue.
2. Dans le panneau de navigation, choisissez Tâches.
3. Choisissez Spark script editor (Éditeur de script Spark) dans Create job (Créer une tâche), puis Create (Créer).
4. Facultatif - Collez le texte intégral de votre script dans le volet Script. Vous pouvez également suivre le didacticiel.

Étape 2. Importer des bibliothèques AWS Glue

Vous devez configurer votre script pour interagir avec le code et la configuration qui sont définis en dehors du script. Ce travail est réalisé dans les coulisses de AWS Glue Studio.

Dans cette étape, vous effectuez les actions suivantes.

- Importer et initialiser un objet `GlueContext`. Il s'agit de l'importation la plus importante, du point de vue de l'écriture de scripts. Elle vous présente les méthodes standards pour définir les jeux de données source et cible, qui sont le point de départ de tout script ETL. Pour en savoir plus sur la classe `GlueContext`, consultez GlueContext classe.
- Initialisation d'un `SparkContext` et d'une `SparkSession`. Ils vous permettent de configurer le moteur Spark disponible dans le job AWS Glue. Vous n'aurez pas besoin de les utiliser directement dans les scripts d'introduction de AWS Glue.
- Appelez `getResolvedOptions` pour préparer vos arguments de tâche à utiliser dans le script. Pour plus d'informations sur la résolution des paramètres de la tâche, consultez the section called "getResolvedOptions".
- Initialisation d'une `Job`. L'`Job`objet définit la configuration et suit l'état des différentes fonctionnalités optionnelles de AWS Glue. Votre script peut s'exécuter sans objet `Job`, mais il est recommandé de l'initialiser afin d'éviter toute confusion si ces fonctionnalités sont intégrées ultérieurement.

  L'une de ces fonctionnalités concerne les signets de tâche, que vous pouvez éventuellement configurer dans ce didacticiel. Vous pouvez en savoir plus sur les signets de tâche dans la section suivante, the section called "Facultatif : activer les signets de tâche".

Dans cette procédure, vous écrivez le code suivant. Ce code fait partie de l'exemple de script généré.

```
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job

args = getResolvedOptions(sys.argv, ["JOB_NAME"])
sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
job = Job(glueContext)
```

```
job.init(args["JOB_NAME"], args)
```

Pour importer des bibliothèques AWS Glue

- Copiez cette section de code et collez-la dans l'éditeur Script.

> **Note**
>
> Vous pourriez considérer la copie de code comme une mauvaise pratique d'ingénierie. Dans ce didacticiel, nous vous le suggérons pour vous encourager à nommer systématiquement vos variables principales dans tous les scripts AWS Glue ETL.

Étape 3. Extraire des données d'une source

Dans tout processus ETL, vous devez d'abord définir le jeu de données source que vous voulez modifier. Dans l'éditeur visuel AWS Glue Studio, vous fournissez ces informations en créant un nœud Source.

Au cours de cette étape, vous devez fournir à la méthode `create_dynamic_frame.from_catalog` une `database` et un `table_name` pour extraire des données d'une source configurée dans le catalogue de données AWS Glue.

À l'étape précédente, vous avez initialisé un objet `GlueContext`. Vous utilisez cet objet pour rechercher les méthodes qui sont utilisées pour configurer les sources, telles que `create_dynamic_frame.from_catalog`.

Dans cette procédure, vous écrivez le code suivant à l'aide de `create_dynamic_frame.from_catalog`. Ce code fait partie de l'exemple de script généré.

```
S3bucket_node1 = glueContext.create_dynamic_frame.from_catalog(
    database="yyz-tickets", table_name="tickets", transformation_ctx="S3bucket_node1"
    )
```

Extraire des données d'une source

1. Consultez la documentation pour trouver une méthode permettant d'`GlueContext`extraire des données d'une source définie dans le AWS Glue Data Catalog. Ces méthodes sont documentées dans the section called "GlueContext". Choisissez la méthode create_dynamic_frame.from_catalog. Appelez cette méthode sur `glueContext`.

2. Consultez la documentation pour `create_dynamic_frame.from_catalog`. Cette méthode nécessite les paramètres `database` et `table_name`. Fournissez les paramètres nécessaires à `create_dynamic_frame.from_catalog`.

   Le catalogue de données AWS Glue stocke des informations sur l'emplacement et le format de vos données sources. Il a été configuré dans la section des prérequis. Vous n'avez pas besoin de fournir ces informations directement à votre script.

3. Facultatif – fournissez le paramètre `transformation_ctx` à la méthode afin de prendre en charge les signets de tâche. Vous pouvez en savoir plus sur les signets de tâche dans la section suivante, the section called "Facultatif : activer les signets de tâche".

> **Note**
>
> Méthodes courantes d'extraction de données
> the section called "create_dynamic_frame_from_catalog"est utilisé pour se connecter aux tables du catalogue de données AWS Glue.
> Si vous devez fournir directement à votre tâche une configuration qui décrit la structure et l'emplacement de votre source, consultez la méthode the section called "create_dynamic_frame_from_options". Vous devrez fournir des paramètres plus détaillés décrivant vos données que lorsque vous utilisez `create_dynamic_frame.from_catalog`.
> Reportez-vous à la documentation supplémentaire sur `format_options` et `connection_parameters` pour identifier les paramètres requis. Pour savoir comment fournir à votre script des informations concernant le format de vos données sources, consultez the section called "Options de format de données". Pour savoir comment fournir à votre script des informations concernant l'emplacement de vos données sources, consultez the section called "Paramètres de connexion".
> Si vous lisez des informations provenant d'une source de streaming, vous fournissez des informations sources à votre tâche via les méthodes the section called "create_data_frame_from_catalog" ou the section called "create_data_frame_from_options". Notez que ces méthodes renvoient Apache Spark `DataFrames`.
> Notre code généré appelle `create_dynamic_frame.from_catalog`, tandis que la documentation de référence fait référence à `create_dynamic_frame_from_catalog`. Ces méthodes appellent finalement le même code et sont incluses afin que vous puissiez écrire du code plus propre. Vous pouvez le vérifier en consultant la source de notre wrapper Python, disponible sur aws-glue-libs.

Étape 4 : Transformez les données avec AWS Glue

Après avoir extrait les données sources dans un processus ETL, vous devez décrire la manière dont vous souhaitez modifier vos données. Vous fournissez ces informations en créant un nœud Transform dans l'éditeur visuel AWS Glue Studio.

À cette étape, fournissez la méthode `ApplyMapping` avec une carte des noms et des types de champs actuels et souhaités pour transformer votre `DynamicFrame`.

Vous effectuez les transformations suivantes.

- Supprimez les quatre clés `location` et `province`.
- Modifiez le nom de `officer` en `officer_name`.
- Modifiez le type de `ticket_number` et `set_fine_amount` en `float`.

`create_dynamic_frame.from_catalog` vous fournit un objet `DynamicFrame`. A `DynamicFrame` représente un ensemble de données dans AWS Glue. AWS Les transformations à la colle sont des opérations qui changent`DynamicFrames`.

> **Note**
>
> Qu'est-ce qu'un `DynamicFrame` ?
> Un `DynamicFrame` est une abstraction qui vous permet de connecter un jeu de données à une description des noms et types d'entrées dans les données. Dans Apache Spark, il existe une abstraction similaire appelée DataFrame a. Pour une explication DataFrames, consultez le guide Spark SQL.
> Avec `DynamicFrames`, vous pouvez décrire les schémas de jeux de données de manière dynamique. Prenons l'exemple d'un ensemble de données avec une colonne de prix, où certaines entrées stockent le prix sous forme de chaîne, tandis que d'autres le stockent sous forme de double. AWS Glue calcule un schéma on-the-fly : elle crée un enregistrement autodescriptif pour chaque ligne.
> Les champs incohérents (comme le prix) sont explicitement représentés par un type (`ChoiceType`) dans le schéma du cadre. Vous pouvez corriger vos champs incohérents en les supprimant avec `DropFields` ou en les résolvant avec `ResolveChoice`. Il s'agit de transformations disponibles sur le `DynamicFrame`. Vous pouvez ensuite réécrire vos données dans votre lac de données avec `writeDynamicFrame`.

Vous pouvez appeler un grand nombre des mêmes transformations à partir des méthodes sur la classe `DynamicFrame`, ce qui peut conduire à des scripts plus lisibles. Pour plus d'informations sur `DynamicFrame`, consultez the section called "DynamicFrame".

Dans cette procédure, vous écrivez le code suivant à l'aide de `ApplyMapping`. Ce code fait partie de l'exemple de script généré.

```
ApplyMapping_node2 = ApplyMapping.apply(
    frame=S3bucket_node1,
    mappings=[
        ("tag_number_masked", "string", "tag_number_masked", "string"),
        ("date_of_infraction", "string", "date_of_infraction", "string"),
        ("ticket_date", "string", "ticket_date", "string"),
        ("ticket_number", "decimal", "ticket_number", "float"),
        ("officer", "decimal", "officer_name", "decimal"),
        ("infraction_code", "decimal", "infraction_code", "decimal"),
        ("infraction_description", "string", "infraction_description", "string"),
        ("set_fine_amount", "decimal", "set_fine_amount", "float"),
        ("time_of_infraction", "decimal", "time_of_infraction", "decimal"),
    ],
    transformation_ctx="ApplyMapping_node2",
)
```

Pour transformer les données avec AWS Glue

1. Consultez la documentation pour identifier une transformation permettant de modifier et de supprimer des champs. Pour en savoir plus, consultez the section called "GlueTransform". Choisissez la transformation `ApplyMapping`. Pour plus d'informations sur `ApplyMapping`, consultez the section called "ApplyMapping". Appelez `apply` sur l'objet de transformation `ApplyMapping`.

   > **Note**
   >
   > Qu'est-ce qu'`ApplyMapping` ?
   > `ApplyMapping` prend un `DynamicFrame` et le transforme. Il prend une liste de tuples qui représentent des transformations sur des champs – un « mappage ». Les deux premiers éléments du tuple, un nom et un type de champ, sont utilisés pour identifier un champ du cadre. Les deux autres paramètres sont également un nom et un type de champ.

ApplyMapping convertit le champ source en nom cible et saisissez-en un nouveau`DynamicFrame`, qu'il renvoie. Les champs qui ne sont pas fournis sont supprimés dans la valeur de retour.
Plutôt que d'appeler `apply`, vous pouvez appeler la même transformation avec la méthode `apply_mapping` sur le `DynamicFrame`, pour créer un code plus fluide et lisible. Pour de plus amples informations, veuillez consulter the section called "apply_mapping".

2. Consultez la documentation pour `ApplyMapping` afin d'identifier les paramètres requis. Consultez the section called "ApplyMapping". Vous constaterez que cette méthode nécessite les paramètres `frame` et `mappings`. Fournissez les paramètres nécessaires à `ApplyMapping`.
3. Facultatif – fournissez `transformation_ctx` à la méthode afin de prendre en charge les signets de tâche. Vous pouvez en savoir plus sur les signets de tâche dans la section suivante, the section called "Facultatif : activer les signets de tâche".

> **Note**
>
> Fonctionnalité Apache Spark
> Nous proposons des transformations pour rationaliser les flux de travail ETL au sein de votre tâche. Vous avez également accès aux bibliothèques qui sont disponibles dans un programme Spark de votre tâche, conçues à des fins plus générales. Pour les utiliser, vous effectuez une conversion entre `DynamicFrame` et `DataFrame`.
> Vous pouvez créer un `DataFrame` avec the section called "toDF". Vous pouvez ensuite utiliser les méthodes disponibles sur le DataFrame pour transformer votre ensemble de données. Pour plus d'informations sur ces méthodes, consultez DataFrame. Vous pouvez ensuite effectuer une conversion à l'the section called "fromDF"envers en utilisant les opérations AWS Glue pour charger votre cadre sur une cible.

Étape 5. Charger des données dans une cible

Après avoir transformé vos données, vous les stockez généralement dans un endroit différent de la source. Vous effectuez cette opération en créant un nœud cible dans l'éditeur visuel AWS Glue Studio.

Au cours de cette étape, vous fournissez à la méthode des valeurs `write_dynamic_frame.from_options`, `connection_type`, `connection_options`, `format` et `format_options` pour charger des données dans un compartiment cible dans Amazon S3.

À l'étape 1, vous avez initialisé un objet `GlueContext`. Dans AWS Glue, vous trouverez ici les méthodes utilisées pour configurer les cibles, un peu comme les sources.

Dans cette procédure, vous écrivez le code suivant à l'aide de `write_dynamic_frame.from_options`. Ce code fait partie de l'exemple de script généré.

```
S3bucket_node3 = glueContext.write_dynamic_frame.from_options(
    frame=ApplyMapping_node2,
    connection_type="s3",
    format="glueparquet",
    connection_options={"path": "s3://amzn-s3-demo-bucket", "partitionKeys": []},
    format_options={"compression": "gzip"},
    transformation_ctx="S3bucket_node3",
    )
```

Charger des données dans une cible

1. Consultez la documentation pour trouver une méthode pour charger des données dans un compartiment Amazon S3 cible. Ces méthodes sont documentées dans the section called "GlueContext". Choisissez la méthode the section called "write_dynamic_frame_from_options". Appelez cette méthode sur `glueContext`.

   > **Note**
   >
   > Méthodes courantes pour le chargement de données
   > `write_dynamic_frame.from_options` est la méthode la plus courante pour charger des données. Il prend en charge toutes les cibles disponibles dans AWS Glue.
   > Si vous écrivez sur une cible JDBC définie dans une connexion AWS Glue, utilisez cette méthode. the section called "write_dynamic_frame_from_jdbc_conf"
   > AWS Les connexions Glue stockent des informations sur la manière de se connecter à une source de données. Il n'est donc plus nécessaire de fournir ces informations dans `connection_options`. Cependant, vous devez toujours utiliser `connection_options` pour fournir `dbtable`.
   > `write_dynamic_frame.from_catalog` n'est pas une méthode courante pour charger des données. Cette méthode met à jour le catalogue de données AWS Glue

sans mettre à jour le jeu de données sous-jacent et est utilisée en combinaison avec d'autres processus qui modifient le jeu de données sous-jacent. Pour de plus amples informations, veuillez consulter the section called "Mise à jour du schéma et ajout de nouvelles partitions".

2. Consultez la documentation pour the section called "write_dynamic_frame_from_options". Cette méthode nécessite `frame`, `connection_type`, `format`, `connection_options` et `format_options`. Appelez cette méthode sur `glueContext`.

   a. Reportez-vous à la documentation supplémentaire sur `format_options` et `format` pour identifier les paramètres dont vous avez besoin. Pour une explication des formats de données, consultez the section called "Options de format de données".

   b. Reportez-vous à la documentation supplémentaire sur `connection_type` et `connection_options` pour identifier les paramètres dont vous avez besoin. Pour une explication des connexions, consultez the section called "Paramètres de connexion".

   c. Fournissez les paramètres nécessaires à `write_dynamic_frame.from_options`. Cette méthode a une configuration similaire à `create_dynamic_frame.from_options`.

3. Facultatif – fournissez `transformation_ctx` à `write_dynamic_frame.from_options` afin de prendre en charge les signets de tâche. Vous pouvez en savoir plus sur les signets de tâche dans la section suivante, the section called "Facultatif : activer les signets de tâche".

Étape 6. Valider l'objet **Job**

À l'étape 1, vous avez initialisé un objet `Job`. Vous devrez peut-être finaliser son cycle de vie manuellement à la fin de votre script si certaines fonctionnalités facultatives en ont besoin pour fonctionner correctement, par exemple lorsque vous utilisez les signets de tâche. Ce travail est réalisé dans les coulisses de AWS Glue Studio.

Dans cette étape, appelez la méthode `commit` sur l'objet `Job`.

Dans cette procédure, vous écrivez le code suivant. Ce code fait partie de l'exemple de script généré.

```
job.commit()
```

Pour valider l'objet **Job**

1. Si vous ne l'avez pas encore fait, suivez les étapes facultatives décrites dans les sections précédentes pour inclure `transformation_ctx`.

2. Appelez `commit`.

Facultatif : activer les signets de tâche

À chaque étape précédente, il vous a été demandé de définir les paramètres `transformation_ctx`. Ceci est lié à une fonctionnalité appelée signets de tâche.

Avec les signets de tâche, vous pouvez gagner du temps et de l'argent grâce à des tâches qui s'exécutent de manière récurrente, par rapport à des jeux de données où il est possible de suivre facilement le travail précédent. Les signets de tâches suivent la progression d'une transformation AWS Glue dans un ensemble de données par rapport aux exécutions précédentes. En suivant la fin des séries précédentes, AWS Glue peut limiter son travail aux lignes qu'elle n'a jamais traitées auparavant. Pour plus d'informations sur les signets de tâche, consultez the section called "Suivi des données traitées à l'aide de signets de tâche".

Pour activer les favoris des signets de tâche, ajoutez d'abord le `transformation_ctx` des déclarations sur les fonctions que nous fournissons, comme illustré dans les exemples précédents. L'état du signet de tâche est conservé au fil des exécutions. Les paramètres `transformation_ctx` sont des clés utilisées pour accéder à cet état. Seules, ces instructions ne servent à rien. Vous devez également activer la fonctionnalité dans la configuration de votre tâche.

Au cours de cette procédure, vous activez les signets de tâche à l'aide de la AWS Management Console.

Définir les signets de tâche

1. Accédez à la section Détails de la tâche de votre tâche correspondante.
2. Définissez Job bookmark (Signet de tâche) sur Enable (Activer).

Étape 7. Exécution de votre code en tant que tâche

Dans cette étape, vous exécutez votre tâche pour vérifier que vous avez bien suivi ce didacticiel. Cela se fait en cliquant sur un bouton, comme dans l'éditeur visuel AWS Glue Studio.

Exécuter votre code en tant que tâche

1. Choisissez Untitled job (Tâche sans titre) dans la barre de titre pour modifier et définir le nom de votre tâche.

2. Accédez à l'onglet Job details (Informations de la tâche). Attribuez à votre tâche un rôle IAM. Vous pouvez utiliser celui créé par le CloudFormation modèle dans les prérequis du didacticiel AWS Glue Studio. Si vous avez terminé ce didacticiel, il devrait être disponible sous `AWS Glue StudioRole`.
3. Choisissez Save (Enregistrer) pour enregistrer votre script.
4. Choisissez Run (Exécuter) pour exécuter votre tâche.
5. Accédez à l'onglet Runs (Exécutions) pour vérifier que votre tâche est terminée.
6. Naviguez vers*`amzn-s3-demo-bucket`*, la cible pour`write_dynamic_frame.from_options`. Vérifiez que le résultat correspond à vos attentes.

Pour plus d'informations sur la configuration et la gestion des tâches, consultez the section called "Fournir vos propres scripts personnalisés".

En savoir plus

Les bibliothèques et méthodes Apache Spark sont disponibles dans les scripts AWS Glue. Vous pouvez consulter la documentation Spark pour comprendre ce que vous pouvez faire avec les bibliothèques incluses. Pour de plus amples informations, consultez la section des exemples du référentiel Spark source.

AWS Glue 2.0+ inclut plusieurs bibliothèques Python courantes par défaut. Il existe également des mécanismes pour charger vos propres dépendances dans une tâche AWS Glue dans un environnement Scala ou Python. Pour plus d'informations sur les dépendances Python, consultez the section called "Bibliothèques Python".

Pour plus d'exemples d'utilisation des fonctionnalités de AWS Glue en Python, consultezthe section called "Exemples Python". Les tâches Scala et Python ont des fonctionnalités identiques, donc nos exemples Python devraient vous donner quelques idées sur la façon d'effectuer une tâche similaire dans Scala.

Programmez les scripts ETL AWS Glue dans PySpark

Vous pouvez trouver des exemples de code Python et des utilitaires pour AWS Glue dans le référentiel d'échantillons AWS Glue sur le GitHub site Web.

Utiliser Python avec AWS Glue

AWS Glue prend en charge une extension du dialecte PySpark Python pour les tâches d'extraction, de transformation et de chargement (ETL) par script. Cette section décrit comment utiliser Python dans les scripts ETL et avec l'API AWS Glue.

- Configuration d'utilisation de Python avec AWS Glue
- Appeler AWS Glue APIs en Python
- Utiliser les bibliothèques Python avec AWS Glue
- Exemples de code Python AWS Glue

AWS PySpark Extensions Glue

AWS Glue a créé les extensions suivantes pour le dialecte PySpark Python.

- Accès aux paramètres à l'aide de `getResolvedOptions`
- PySpark types d'extensions
- DynamicFrame classe
- DynamicFrameCollection classe
- DynamicFrameWriter classe
- DynamicFrameReader classe
- GlueContext classe

AWS Glue PySpark transforme

AWS Glue a créé les classes de transformation suivantes à utiliser dans les opérations PySpark ETL.

- GlueTransform classe de base
- ApplyMapping classe
- DropFields classe
- DropNullFields classe
- ErrorsAsDynamicFrame classe
- FillMissingValues classe

- Classe Filter
- FindIncrementalMatches classe
- FindMatches classe
- FlatMap classe
- Classe Join
- Classe Map
- MapToCollection classe
- mergeDynamicFrame
- Classe Relationalize
- RenameField classe
- ResolveChoice classe
- SelectFields classe
- SelectFromCollection classe
- Classe Spigot
- SplitFields classe
- SplitRows classe
- Classe Unbox
- UnnestFrame classe

Configuration d'utilisation de Python avec AWS Glue

Utilisez Python pour développer vos scripts ETL pour les tâches Spark. Les versions Python prises en charge pour les tâches ETL dépendent de la version AWS Glue de la tâche. Pour plus d'informations sur les versions AWS Glue, consultez Glue version job property.

Pour configurer votre système pour l'utilisation de Python avec AWS Glue

Suivez ces étapes pour installer Python et pour pouvoir invoquer le AWS Glue APIs.

1. Si vous n'avez pas déjà Python installé, téléchargez et installez-le à partir de la page de téléchargement Python.org .
2. Installez le AWS Command Line Interface (AWS CLI) comme indiqué dans la documentation de la AWS CLI.

Le n' AWS CLI est pas directement nécessaire pour utiliser Python. Cependant, son installation et sa configuration constituent un moyen pratique de le configurer à l' AWS aide des informations d'identification de votre compte et de vérifier qu'elles fonctionnent.

3. Installez le AWS SDK pour Python (Boto 3), comme indiqué dans le guide de démarrage rapide de Boto3.

   APIs Les ressources Boto 3 ne sont pas encore disponibles pourAWS Glue. Actuellement, seul le client Boto 3 APIs peut être utilisé.

   Pour en savoir plus sur Boto 3, consultez Mise en route du kit SDK AWS for Python (Boto3).

Vous pouvez trouver des exemples de code Python et des utilitaires AWS Glue dans le référentiel d'AWS Glueexemples du site GitHub Web.

Appeler AWS Glue APIs en Python

Notez que les ressources Boto 3 ne APIs sont pas encore disponibles pourAWS Glue. Actuellement, seul le client Boto 3 APIs peut être utilisé.

Noms d'API AWS Glue dans Python

AWS Les noms des API Glue en Java et dans d'autres langages de programmation sont généralement CamelCased. Cependant, lors d'un appel à partir de Python, ces noms génériques sont modifiés en minuscules, avec les différentes parties du nom séparées par des traits de soulignement afin de les rendre plus proches du langage Python. Dans la documentation de AWS Glue API référence, ces noms pythoniques sont listés entre parenthèses après les noms génériques CamelCased .

Cependant, même si les noms d'API AWS Glue eux-mêmes sont transformés en minuscules, leurs noms de paramètres restent en majuscule. Il est important de s'en souvenir, car les paramètres doivent être transmis par leur nom lors de l'appel AWS Glue APIs, comme décrit dans la section suivante.

Transmission de paramètres Python et accès à ces paramètres dans AWS Glue

Dans les appels Python à AWS Glue APIs, il est préférable de transmettre les paramètres explicitement par leur nom. Par exemple :

```
job = glue.create_job(Name='sample', Role='Glue_DefaultRole',
```

```
                      Command={'Name': 'glueetl',
                               'ScriptLocation': 's3://my_script_bucket/scripts/
my_etl_script.py'})
```

Il est utile de comprendre que Python crée un dictionnaire des name/value tuples que vous spécifiez comme arguments d'un script ETL dans un Structure Job ouJobRun structure. Boto 3 les transmet à AWS Glue au format JSON par le biais d'un appel d'API REST. Cela signifie que vous ne pouvez pas vous fier à l'ordre des arguments lorsque vous y accédez dans votre script.

Par exemple, supposons que vous lanciez un `JobRun` dans un gestionnaire Lambda Python, et que vous souhaitiez spécifier plusieurs paramètres. Votre code peut alors se présenter comme suit :

```
from datetime import datetime, timedelta

client = boto3.client('glue')

def lambda_handler(event, context):
  last_hour_date_time = datetime.now() - timedelta(hours = 1)
  day_partition_value = last_hour_date_time.strftime("%Y-%m-%d")
  hour_partition_value = last_hour_date_time.strftime("%-H")

  response = client.start_job_run(
              JobName = 'my_test_Job',
              Arguments = {
                '--day_partition_key':   'partition_0',
                '--hour_partition_key':  'partition_1',
                '--day_partition_value':  day_partition_value,
                '--hour_partition_value': hour_partition_value } )
```

Pour accéder à ces paramètres de manière fiable dans votre script ETL, spécifiez-les par nom à l'aide de la fonction AWS Glue `getResolvedOptions` et accédez-y depuis le dictionnaire obtenu :

```
import sys
from awsglue.utils import getResolvedOptions

args = getResolvedOptions(sys.argv,
                          ['JOB_NAME',
                           'day_partition_key',
                           'hour_partition_key',
                           'day_partition_value',
                           'hour_partition_value'])
print "The day partition key is: ", args['day_partition_key']
```

```
print "and the day partition value is: ", args['day_partition_value']
```

Si vous souhaitez transmettre un argument qui est une chaîne JSON imbriquée, pour préserver la valeur du paramètre lorsqu'elle est transmise à votre tâche ETL AWS Glue, vous devez coder la chaîne de paramètre avant de démarrer l'exécution de la tâche, puis décoder la chaîne de paramètre avant de la référencer votre script de tâche. Prenons l'exemple de la la chaîne d'arguments suivante :

```
glue_client.start_job_run(JobName = "gluejobname", Arguments={
"--my_curly_braces_string": '{"a": {"b": {"c": [{"d": {"e": 42}}]}}}'
})
```

Pour transférer ce paramètre correctement, vous devez coder l'argument sous la forme d'une chaîne codée en Base64.

```
import base64
...
sample_string='{"a": {"b": {"c": [{"d": {"e": 42}}]}}}'
sample_string_bytes = sample_string.encode("ascii")

base64_bytes = base64.b64encode(sample_string_bytes)
base64_string = base64_bytes.decode("ascii")
...
glue_client.start_job_run(JobName = "gluejobname", Arguments={
"--my_curly_braces_string": base64_bytes})
...
sample_string_bytes = base64.b64decode(base64_bytes)
sample_string = sample_string_bytes.decode("ascii")
print(f"Decoded string: {sample_string}")
...
```

Exemple : Créer et exécuter une tâche

L'exemple suivant montre comment appeler le à AWS Glue APIs l'aide de Python pour créer et exécuter une tâche ETL.

Pour créer et exécuter une tâche

1. Créez une instance du client AWS Glue :

   ```
   import boto3
   glue = boto3.client(service_name='glue', region_name='us-east-1',
   ```

```
                endpoint_url='https://glue.us-east-1.amazonaws.com')
```

2. Crée une tâche. Vous devez utiliser `glueetl` comme nom de la commande ETL, comme indiqué dans le code suivant :

```
myJob = glue.create_job(Name='sample', Role='Glue_DefaultRole',
                        Command={'Name': 'glueetl',
                                 'ScriptLocation': 's3://my_script_bucket/
scripts/my_etl_script.py'})
```

3. Démarrez une nouvelle exécution de la tâche que vous avez créée dans l'étape précédente :

```
myNewJobRun = glue.start_job_run(JobName=myJob['Name'])
```

4. Obtenez l'état de la tâche :

```
status = glue.get_job_run(JobName=myJob['Name'], RunId=myNewJobRun['JobRunId'])
```

5. Affichez l'état actuel de l'exécution de la tâche :

```
print(status['JobRun']['JobRunState'])
```

Utiliser les bibliothèques Python avec AWS Glue

Vous pouvez installer des modules et bibliothèques Python supplémentaires à utiliser avec AWS Glue ETL. Pour AWS Glue 2.0 et versions ultérieures, AWS Glue utilise le Python Package Installer (pip3) pour installer les modules supplémentaires utilisés par AWS Glue ETL. AWS Glue propose plusieurs options pour intégrer les modules Python supplémentaires à votre environnement de travail AWS Glue. Vous pouvez utiliser ce `--additional-python-modules` paramètre pour intégrer de nouveaux modules à l'aide de fichiers zip contenant des roues Python groupées (également appelées « zip of wheels », disponibles pour AWS Glue 5.0 et versions ultérieures), des fichiers de roues Python individuels, des fichiers d'exigences (requirements.txt, disponible pour AWS Glue 5.0 et versions ultérieures) ou une liste de modules Python séparés par des virgules. Il peut également être utilisé pour modifier la version des modules python fournis dans l'environnement AWS Glue (voir Modules Python déjà fournis dans AWS Glue pour plus de détails).

Rubriques

- Installation de modules Python supplémentaires avec pip dans AWS Glue 2.0 ou version ultérieure
- Y compris des fichiers Python dotés de fonctionnalités PySpark natives

- Scripts de programmation utilisant des transformations visuelles
- Compression de bibliothèques pour intégration
- Chargement de bibliothèques Python dans les blocs-notes AWS Glue Studio
- Chargement de bibliothèques Python dans un endpoint de développement dans AWS Glue 0.9/1.0
- Utilisation de bibliothèques Python dans une tâche ou JobRun
- Analyse proactive des dépendances Python
- Modules Python déjà fournis dans AWS Glue
- Annexe A : Création d'un artefact Zip of Wheels
- Annexe B : Détails de l'environnement AWS Glue

Installation de modules Python supplémentaires avec pip dans AWS Glue 2.0 ou version ultérieure

AWS Glue utilise le Python Package Installer (pip3) pour installer des modules supplémentaires qui seront utilisés par AWS Glue ETL. Vous pouvez utiliser le paramètre `--additional-python-modules` avec une liste de modules Python séparés par des virgules pour ajouter un nouveau module ou modifier la version d'un module existant. Vous pouvez installer des artefacts de roue intégrés via un zip de roues ou un artefact de roue autonome en téléchargeant le fichier sur Amazon S3, puis en incluant le chemin d'accès à l'objet Amazon S3 dans votre liste de modules. Pour plus d'informations sur la définition des paramètres des tâches, consultez la section Utilisation des paramètres des tâches dans les tâches AWS Glue.

Vous pouvez transmettre des options supplémentaires à pip3 à l'aide du paramètre `--python-modules-installer-option`. Par exemple, vous pouvez passer `--only-binary` pour forcer pip à installer uniquement des artefacts prédéfinis pour les packages spécifiés par. `--additional-python-modules` Pour plus d'exemples, voir Création de modules Python à partir d'une roue pour les charges de travail Spark ETL à l'aide de AWS Glue 2.0.

Meilleures pratiques pour la gestion des dépendances en Python

Pour les charges de travail de production, AWS Glue recommande de regrouper toutes vos dépendances Python sous forme de fichiers de roue dans un seul artefact zip. Cette approche permet de :

- Exécution déterministe : contrôle précis des versions de package installées
- Fiabilité : aucune dépendance à l'égard des référentiels de packages externes lors de l'exécution de la tâche

- Performances : opération de téléchargement unique au lieu de plusieurs appels réseau
- Installation hors ligne : fonctionne dans des environnements VPC privés sans accès à Internet

Importantes considérations

Dans le cadre du modèle de responsabilitéAWS partagée, vous êtes responsable de la gestion des modules Python supplémentaires, des bibliothèques et de leurs dépendances. Cela inclut notamment les éléments suivants :

- Mises à jour de sécurité : mise à jour régulière des packages pour corriger les failles de sécurité
- Compatibilité des versions : assurez-vous que les packages sont compatibles avec votre version de AWS Glue
- Tests : vérifier que vos dépendances packagées fonctionnent correctement dans l'environnement Glue

Si vos dépendances sont minimes, vous pouvez envisager d'utiliser des fichiers de roue individuels à la place.

(Recommandé) Installation de bibliothèques Python supplémentaires dans AWS Glue 5.0 ou version ultérieure à l'aide de Zip of Wheels

AWS Glue 5.0 et versions ultérieures permettent de regrouper plusieurs fichiers de roues dans un seul artefact zip contenant des roues Python groupées pour une gestion des dépendances plus fiable et déterministe. Pour utiliser cette approche, créez un fichier zip contenant toutes vos dépendances relatives aux roues et leurs dépendances transitives avec le `.gluewheels.zip` suffixe, chargez-le sur Amazon S3 et référencez-le à l'aide du `--additional-python-modules` paramètre. Assurez-vous d'ajouter `--no-index` au paramètre de `--python-modules-installer-option` tâche. Avec cette configuration, le fichier zip of wheels agit essentiellement comme un index local permettant à pip de résoudre les dépendances au moment de l'exécution. Cela élimine les dépendances vis-à-vis des référentiels de packages externes tels que PyPI lors de l'exécution des tâches, offrant ainsi une stabilité et une cohérence accrues pour les charges de travail de production. Par exemple :

```
--additional-python-modules s3://amzn-s3-demo-bucket/path/to/zip-of-
wheels-1.0.0.gluewheels.zip
--python-modules-installer-option --no-index
```

Pour obtenir des instructions sur la création d'un fichier zip de roues, voirAnnexe A : Création d'un artefact Zip of Wheels.

Installation de bibliothèques Python supplémentaires à l'aide de Wheel

AWS Glue prend en charge l'installation de packages Python personnalisés à l'aide de fichiers wheel (.whl) stockés dans Amazon S3. Pour inclure des fichiers de roues dans vos tâches AWS Glue, fournissez une liste séparée par des virgules de vos fichiers de roues stockés dans s3 dans le paramètre de `--additional-python-modules` tâche. Par exemple :

```
--additional-python-modules s3://amzn-s3-demo-bucket/path/to/package-1.0.0-py3-none-any.whl,s3://your-bucket/path/to/another-package-2.1.0-cp311-cp311-linux_x86_64.whl
```

Cette approche est également utile lorsque vous avez besoin de distributions personnalisées ou de packages avec des dépendances natives précompilés pour le système d'exploitation approprié. Pour plus d'exemples, voir Création de modules Python à partir d'une roue pour les charges de travail Spark ETL à l'aide de AWS Glue 2.0.

Installation de bibliothèques Python supplémentaires dans AWS Glue 5.0 ou version ultérieure à l'aide de requirements.txt

Dans AWS Glue 5.0+, vous pouvez fournir le standard de facto `requirements.txt` pour gérer les dépendances des bibliothèques Python. Pour ce faire, indiquez les deux paramètres de tâche suivants :

- Clé : `--python-modules-installer-option`

  Valeur : `-r`
- Clé : `--additional-python-modules`

  Valeur : `s3://path_to_requirements.txt`

AWS Les nœuds Glue 5.0 chargent initialement les bibliothèques python spécifiées dans`requirements.txt`.

Voici un exemple de requirements.txt :

```
awswrangler==3.9.1
elasticsearch==8.15.1
```

```
PyAthena==3.9.0
PyMySQL==1.1.1
PyYAML==6.0.2
pyodbc==5.2.0
pyorc==0.9.0
redshift-connector==2.1.3
scipy==1.14.1
scikit-learn==1.5.2
SQLAlchemy==2.0.36
```

Important

Utilisez cette option avec prudence, en particulier pour les charges de travail de production. Extraire les dépendances de PyPI au moment de l'exécution est très risqué car vous ne pouvez pas être sûr de l'artefact auquel correspond pip. L'utilisation de versions de bibliothèques non épinglées est particulièrement risquée car elle extrait la dernière version des modules python, ce qui peut introduire des modifications importantes ou introduire un module python incompatible. Cela pourrait entraîner un échec de la tâche en raison de l'échec de l'installation de Python dans l'environnement de travail AWS Glue. Bien que l'épinglage de la version de la bibliothèque augmente la stabilité, la résolution des pip n'est toujours pas totalement déterministe, de sorte que des problèmes similaires peuvent survenir. À titre de bonne pratique, AWS Glue recommande d'utiliser des objets figés tels que des fermetures de roues ou des limes individuelles (voir the section called "(Recommandé) Installation de bibliothèques Python supplémentaires dans AWS Glue 5.0 ou version ultérieure à l'aide de Zip of Wheels" pour plus de détails).

Important

Si vous n'identifiez pas les versions de vos dépendances transitives, une dépendance principale peut générer des versions de dépendances transitives incompatibles. La meilleure pratique consiste à épingler toutes les versions de bibliothèque pour améliorer la cohérence des tâches AWS Glue. Mieux encore, AWS Glue recommande de regrouper vos dépendances dans un fichier zip of wheels afin de garantir une cohérence et une fiabilité maximales pour vos charges de travail de production.

Installation de bibliothèques Python supplémentaires, configurées directement sous forme de liste séparée par des virgules

Pour mettre à jour ou ajouter un nouveau module Python, AWS Glue permet de transmettre des `--additional-python-modules` paramètres avec une liste de modules Python séparés par des virgules sous forme de valeurs. Par exemple, pour mettre à jour ou ajouter un module scikit-learn, utilisez la clé/valeur suivante : `"--additional-python-modules", "scikit-learn==0.21.3"`. Vous avez deux options pour configurer directement les modules Python.

- Module Python épinglé

  ```
  "--additional-python-modules", "scikit-learn==0.21.3,ephem==4.1.6"
  ```

- Module Python non épinglé : (non recommandé pour les charges de travail de production)

  ```
  "--additional-python-modules", "scikit-learn>==0.20.0,ephem>=4.0.0"
  ```

  OU

  ```
  "--additional-python-modules", "scikit-learn,ephem"
  ```

> **Important**
>
> Utilisez cette option avec prudence, en particulier pour les charges de travail de production. Extraire les dépendances de PyPI au moment de l'exécution est très risqué car vous ne pouvez pas être sûr de l'artefact auquel correspond pip. L'utilisation de versions de bibliothèques non épinglées est particulièrement risquée car elle extrait la dernière version des modules python, ce qui peut introduire des modifications importantes ou introduire un module python incompatible. Cela pourrait entraîner un échec de la tâche en raison de l'échec de l'installation de Python dans l'environnement de travail AWS Glue. Bien que l'épinglage de la version de la bibliothèque augmente la stabilité, la résolution des pip n'est toujours pas totalement déterministe, de sorte que des problèmes similaires peuvent survenir. À titre de bonne pratique, AWS Glue recommande d'utiliser des objets figés tels que des fermetures de roues ou des limes individuelles (voir the section called "(Recommandé) Installation de bibliothèques Python supplémentaires dans AWS Glue 5.0 ou version ultérieure à l'aide de Zip of Wheels" pour plus de détails).

> **Important**
>
> Si vous n'identifiez pas les versions de vos dépendances transitives, une dépendance principale peut générer des versions de dépendances transitives incompatibles. La meilleure pratique consiste à épingler toutes les versions de bibliothèque pour améliorer la cohérence des tâches AWS Glue. Mieux encore, AWS Glue recommande de regrouper vos dépendances dans un fichier zip of wheels afin de garantir une cohérence et une fiabilité maximales pour vos charges de travail de production.

Y compris des fichiers Python dotés de fonctionnalités PySpark natives

AWS Glue est utilisé PySpark pour inclure des fichiers Python dans les tâches AWS Glue ETL. Vous aurez envie d'utiliser `--additional-python-modules` pour gérer vos dépendances lorsqu'elles sont disponibles. Vous pouvez utiliser le paramètre de tâche `--extra-py-files` pour inclure des fichiers Python. Les dépendances doivent être hébergées dans Amazon S3, et la valeur de l'argument doit être une liste de chemins non-espacés Amazon S3 délimités par des virgules. Cette fonctionnalité se comporte comme la gestion des dépendances Python que vous utiliseriez avec Spark. Pour plus d'informations sur la gestion des dépendances Python dans Spark, consultez la page Utilisation des fonctionnalités PySpark natives dans la documentation d'Apache Spark. `--extra-py-files`est utile dans les cas où votre code supplémentaire n'est pas empaqueté ou lorsque vous migrez un programme Spark avec une chaîne d'outils existante pour gérer les dépendances. Pour que vos outils de dépendance soient gérables, vous devez regrouper vos dépendances avant de les soumettre.

Scripts de programmation utilisant des transformations visuelles

Lorsque vous créez une tâche AWS Glue à l'aide de l'interface visuelle de AWS Glue Studio, vous pouvez transformer vos données à l'aide de nœuds de transformation de données gérés et de transformations visuelles personnalisées. Pour plus d'informations sur les nœuds de transformation de données gérés, consultez the section called “Transformation de données avec des transformations gérées par AWS Glue”. Pour plus d'informations sur les transformations visuelles personnalisées, consultez the section called “ Transformation des données avec des transformations visuelles personnalisées ”. Les scripts utilisant des transformations visuelles ne peuvent être générés que lorsque le langage de votre tâche est configuré pour utiliser Python.

Lors de la génération d'une tâche AWS Glue à l'aide de transformations visuelles, AWS Glue Studio inclut ces transformations dans l'environnement d'exécution en utilisant le `--extra-py-files`

paramètre de configuration de la tâche. Pour de plus amples informations sur la définition des paramètres de la tâche, consultez the section called "Paramètres du job". Lorsque vous apportez des modifications à un script généré ou à un environnement d'exécution, vous devez conserver cette configuration de tâche pour que votre script s'exécute correctement.

Compression de bibliothèques pour intégration

Sauf si elle est comprise dans un seul fichier `.py`, une bibliothèque doit être packagée dans une archive `.zip`. Le répertoire du package doit être à la racine de l'archive et contenir un fichier `__init__.py` pour le package. Python sera alors en mesure d'importer le package normalement.

Si votre bibliothèque se compose d'un seul module Python dans un fichier `.py`, vous n'avez pas besoin de la mettre dans un fichier `.zip`.

Chargement de bibliothèques Python dans les blocs-notes AWS Glue Studio

Pour spécifier les bibliothèques Python dans les blocs-notes AWS Glue Studio, consultez la section Installation de modules Python supplémentaires.

Chargement de bibliothèques Python dans un endpoint de développement dans AWS Glue 0.9/1.0

Si vous utilisez différents ensembles de bibliothèques pour différents scripts ETL, vous pouvez configurer un point de terminaison de développement distinct pour chaque ensemble, ou écraser le(s) fichier(s) `.zip` de bibliothèque que votre point de terminaison de développement charge à chaque fois que vous basculez d'un script à un autre.

Vous pouvez utiliser la console pour spécifier un ou plusieurs fichiers .zip de bibliothèque pour un point de terminaison de développement lorsque vous créez celui-ci. Après avoir attribué un nom et un rôle IAM, sélectionnez Script Libraries and job parameters (optional) (Bibliothèques de scripts et paramètres de tâches [facultatif]) et saisissez le chemin d'accès Amazon S3 complet à votre fichier `.zip` de bibliothèque dans la zone Python library path (Chemin de bibliothèque Python). Par exemple :

```
s3://bucket/prefix/site-packages.zip
```

Si vous le souhaitez, vous pouvez spécifier plusieurs chemins complets pour les fichiers, en les séparant par des virgules, mais sans espace, comme l'exemple suivant :

```
s3://bucket/prefix/lib_A.zip,s3://bucket_B/prefix/lib_X.zip
```

Si vous mettez à jour ces fichiers `.zip` ultérieurement, vous pouvez utiliser la console pour les importer de nouveau dans votre point de terminaison de développement. Naviguez vers le point de terminaison de développement concerné, cochez la case située en regard, puis choisissez Update ETL libraries (Mettre à jour des bibliothèques ETL) dans le menu Action.

De la même manière, vous pouvez spécifier des fichiers de bibliothèque à l'aide de la AWS Glue APIs. Lorsque vous créez un point de terminaison de développement en appelant CreateDevEndpoint action (Python : create_dev_endpoint), vous pouvez spécifier un ou plusieurs chemins complets de bibliothèques dans le paramètre `ExtraPythonLibsS3Path`, dans un appel ressemblant à ceci :

```
dep = glue.create_dev_endpoint(
             EndpointName="testDevEndpoint",
             RoleArn="arn:aws:iam::123456789012",
             SecurityGroupIds="sg-7f5ad1ff",
             SubnetId="subnet-c12fdba4",
             PublicKey="ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQCtp04H/y...",
             NumberOfNodes=3,
             ExtraPythonLibsS3Path="s3://bucket/prefix/lib_A.zip,s3://bucket_B/prefix/
lib_X.zip")
```

Lorsque vous mettez à jour un point de terminaison de développement, vous pouvez également mettre à jour les bibliothèques chargées par ce point de terminaison en utilisant un objet DevEndpointCustomLibraries et en définissant le paramètre `UpdateEtlLibraries` sur `True` lors de l'appel de UpdateDevEndpoint (update_dev_endpoint).

Utilisation de bibliothèques Python dans une tâche ou JobRun

Lorsque vous créez un objet Job sur la console, vous pouvez spécifier un ou plusieurs fichiers .zip de bibliothèque en sélectionnant Script Libraries and job parameters (optional) (Bibliothèques de scripts et paramètres de tâches [facultatif]) et en entrant le ou les chemins d'accès Amazon S3 complets aux bibliothèques, comme vous le feriez lors de la création d'un point de terminaison de développement :

```
s3://bucket/prefix/lib_A.zip,s3://bucket_B/prefix/lib_X.zip
```

Si vous appelez CreateJob (créer_job), vous pouvez spécifier un ou plusieurs chemins d'accès complets aux bibliothèques par défaut en utilisant les paramètres par défaut `--extra-py-files`, comme suit :

```
job = glue.create_job(Name='sampleJob',
                      Role='Glue_DefaultRole',
                      Command={'Name': 'glueetl',
                               'ScriptLocation': 's3://my_script_bucket/scripts/
my_etl_script.py'},
                      DefaultArguments={'--extra-py-files': 's3://bucket/prefix/
lib_A.zip,s3://bucket_B/prefix/lib_X.zip'})
```

Ensuite, lorsque vous démarrez une JobRun, vous pouvez remplacer le paramètre de bibliothèque par défaut par un autre :

```
runId = glue.start_job_run(JobName='sampleJob',
                           Arguments={'--extra-py-files': 's3://bucket/prefix/
lib_B.zip'})
```

Analyse proactive des dépendances Python

Pour identifier de manière proactive les problèmes de dépendance potentiels avant le déploiement sur AWS Glue, vous pouvez utiliser l'outil d'analyse des dépendances pour valider vos packages Python par rapport à votre environnement AWS Glue cible.

AWS fournit un outil d'analyse de dépendance Python open source spécialement conçu pour les environnements AWS Glue. Cet outil est disponible dans le référentiel d'échantillons AWS Glue et peut être utilisé localement pour valider vos dépendances avant le déploiement.

Cette analyse permet de garantir que vos dépendances respectent la pratique recommandée qui consiste à épingler toutes les versions de bibliothèque pour des déploiements de production cohérents. Pour plus de détails, veuillez consulter le fichier README de l'outil.

Utilisation de l'analyseur AWS de dépendance Glue

L'analyseur de dépendance AWS Glue Python permet d'identifier les dépendances non épinglées et les conflits de versions en simulant l'installation de pip avec des contraintes spécifiques à la plate-forme qui correspondent à votre environnement Glue cible. AWS

```
# Analyze a single Glue job
python glue_dependency_analyzer.py -j my-glue-job

# Analyze multiple jobs with specific AWS configuration
python glue_dependency_analyzer.py -j job1 -j job2 --aws-profile production --aws-
region us-west-2
```

L'outil signalera les éléments suivants :

- Dépendances non épinglées susceptibles d'installer différentes versions au cours de l'exécution des tâches
- Conflits de version entre les packages
- Dépendances non disponibles pour votre environnement AWS Glue cible

Analyse et correction des échecs de tâches dus aux dépendances Python avec Amazon Q Developer

Amazon Q Developer est un assistant conversationnel basé sur l'intelligence artificielle générative (IA) qui peut vous aider à comprendre, créer, étendre et exploiter AWS des applications. Vous pouvez le télécharger en suivant les instructions du Guide de démarrage d'Amazon Q.

Amazon Q Developer peut être utilisé pour analyser et corriger les échecs de tâches dus à la dépendance Python. Nous vous suggérons d'utiliser l'invite suivante en remplaçant l'espace réservé <Job-Name> de la tâche par le nom de votre tâche Glue.

```
I have an AWS Glue job named <Job-Name> that has failed due to Python module
 installation conflicts. Please assist in diagnosing and resolving this issue using the
 following systematic approach. Proceed once sufficient information is available.

Objective: Implement a fix that addresses the root cause module while minimizing
 disruption to the existing working environment.

Step 1: Root Cause Analysis
• Retrieve the most recent failed job run ID for the specified Glue job
• Extract error logs from CloudWatch Logs using the job run ID as a log stream prefix
• Analyze the logs to identify:
  • The recently added or modified Python module that triggered the dependency conflict
  • The specific dependency chain causing the installation failure
  • Version compatibility conflicts between required and existing modules

Step 2: Baseline Configuration Identification
• Locate the last successful job run ID prior to the dependency failure
• Document the Python module versions that were functioning correctly in that baseline
 run
• Establish the compatible version constraints for conflicting dependencies

Step 3: Targeted Resolution Implementation
• Apply pinning by updating the job's additional_python_modules parameter
```

```
• Pin only the root cause module and its directly conflicting dependencies to
 compatible versions, and do not remove python modules unless necessary
• Preserve flexibility for non-conflicting modules by avoiding unnecessary version
 constraints
• Deploy the configuration changes with minimal changes to the existing configuration
 and execute a validation test run. Do not change the Glue versions.

Implementation Example:
Scenario: Recently added pandas==2.0.0 to additional_python_modules
Error: numpy version conflict (pandas 2.0.0 requires numpy>=1.21, but existing job code
 requires numpy<1.20)
Resolution: Update additional_python_modules to "pandas==1.5.3,numpy==1.19.5"
Rationale: Use pandas 1.5.3 (compatible with numpy 1.19.5) and pin numpy to last known
 working version

Expected Outcome: Restore job functionality with minimal configuration changes while
 maintaining system stability.
```

L'invite demande à Q d'effectuer les opérations suivantes :

1. Récupérer l'ID d'exécution de la dernière tâche ayant échoué
2. Rechercher les journaux et les détails associés
3. Rechercher des exécutions de tâches réussies pour détecter tout package Python modifié
4. Apporter des corrections aux configurations et déclencher une autre exécution test

Modules Python déjà fournis dans AWS Glue

Pour modifier la version de ces modules fournis, fournissez de nouvelles versions avec le paramètre de tâche `--additional-python-modules`.

AWS Glue version 5.1

AWS La version 5.1 de Glue inclut les modules Python suivants prêts à l'emploi :

- aiobotocore==2,25,1
- aiohappyeballs==2.6.1
- aiohttp==3.13.2
- aioitertools==0.12.0
- aiosignal ==1.4.0

- appdirs==1.4.4
- attrs = 25,4,0
- boto3==1.40.61
- botocore==1,40,61
- certifi==2025.10.5
- normaliseur de jeux de caractères ==3.4.4
- chorégraphe ==1.2.0
- contourpy==1.3.3
- cycleur ==0,12.1
- distlib ==0,4.0
- filelock==3.20.0
- fonttools==4.60.1
- liste congelée ==1.8.0
- fsspec==2025.10.0
- idna=3.11
- iniconfig==2.3.0
- mespath=1.0.1
- kaleido==1.2.0
- kiwisolver ==1.4.9
- logistro==2.0.1
- matplotlib==3.10.7
- multidict==6.7.0
- narvals ==2.10.2
- numpy==2.3.4
- orjson==3.11.4
- emballage = 25,0
- pandas==2.3.3
- oreiller==12.0.0
- pip=24,0

- platformdirs==4.5.0
- plutly=6.4.0
- pluggy==1.6.0
- propcache==0.4.1
- pyarrow=22.0.0
- Pygments==2.19.2
- pyparsing==3.2.5
- pytest-timeout=2.4.0
- pytest=8.4.2
- python-dateutil==2.9.0.post0
- pytz=2025,2
- demandes ==2.32.5
- s3fs==2025,1,0
- transfert s3 = 0,14.0
- seaborn==0.13.2
- outils de configuration ==79.0.1
- simplejson==3.20.2
- six = 1,17,0
- ténacité ==9.1.2
- typing_extensions==4.15.0
- tzdata=2025,2
- urllib3==2.5.0
- uv = 0,9,7
- virtualenv = 20,35,4
- wrapt==1.17.3
- fil = 1,2.0

AWS Glue version 5.0

AWS La version 5.0 de Glue inclut les modules Python suivants prêts à l'emploi :

- aiobotocore==2,13.1
- aiohappyeyeballs==2.3.5
- aiohttp==3.10.1
- aioitertools==0.11.0
- aiosignal==1.3.1
- appdirs==1.4.4
- attrs = 24,2,0
- boto3==1.34.131
- botocore==1,34,131
- certifi==2024.7.4
- normaliseur de jeux de caractères ==3.3.2
- contourpy==1.2.1
- cycleur ==0,12.1
- fonttools==4.53.1
- liste congelée ==1.4.1
- fsspec==2024.6.1
- idna==2.10
- jmespath==0.10.0
- kaleido==0.2.1
- kiwisolver ==1.4.5
- matplotlib==3.9.0
- multidict==6,0.5
- numpy==1,26,4
- emballage==24,1
- pandas==2.2.2
- oreiller ==10.4.0
- pip==23.0.1
- plotly==5.23.0
- pyarrow=17,0,0

- pyparsing==3.1.2
- python-dateutil==2.9.0.post0
- pytz=2024,1
- demandes ==2.32.2
- s3fs==2024,6,1
- transfert s3 ==0.10.2
- seaborn==0.13.2
- outils de configuration ==59.6.0
- six==1.16.0
- tenacity==9.0.0
- tzdata==2024.1
- urllib3==1.25.10
- virtualenv==20.4.0
- wrapt==1,16,0
- yarl==1,9.4

AWS Glue version 4.0

AWS La version 4.0 de Glue inclut les modules Python suivants prêts à l'emploi :

- aiobotocore==2.4.1
- aiohttp==3.8.3
- aioitertools==0.11.0
- aiosignal==1.3.1
- async-timeout==4.0.2
- asynctest==0.13.0
- attrs==22.2.0
- avro-python3==1.10.2
- boto3==1.24.70
- botocore==1.27.59

- certifi==2021.5.30
- chardet==3.0.4
- charset-normalizer==2.1.1
- click==8.1.3
- cycler==0.10.0
- Cython==0.29.32
- fsspec==2021.8.1
- idna==2.10
- importlib-metadata==5.0.0
- jmespath==0.10.0
- joblib==1.0.1
- kaleido==0.2.1
- kiwisolver==1.4.4
- matplotlib==3.4.3
- mpmath==1.2.1
- multidict==6.0.4
- nltk==3.7
- numpy==1.23.5
- packaging==23.0
- pandas==1.5.1
- patsy==0.5.1
- Pillow==9.4.0
- pip==23.0.1
- plotly==5.16.0
- pmdarima==2.0.1
- ptvsd==4.3.2
- pyarrow==10.0.0
- pydevd==2.5.0
- pyhocon==0.3.58

- PyMySQL==1.0.2
- pyparsing==2.4.7
- python-dateutil==2.8.2
- pytz==2021.1
- PyYAML==6.0.1
- regex==2022.10.31
- requests==2.23.0
- s3fs==2022.11.0
- s3transfer==0.6.0
- scikit-learn==1.1.3
- scipy==1.9.3
- setuptools==49.1.3
- six==1.16.0
- statsmodels==0.13.5
- subprocess32==3.5.4
- sympy==1.8
- tbats==1.1.0
- threadpoolctl==3.1.0
- tqdm==4.64.1
- typing_extensions==4.4.0
- urllib3==1.25.11
- wheel==0.37.0
- wrapt==1.14.1
- yarl==1.8.2
- zipp==3.10.0

AWS Glue version 3.0

AWS La version 3.0 de Glue inclut les modules Python suivants prêts à l'emploi :,

- aiobotocore==1.4.2
- aiohttp==3.8.3
- aioitertools==0.11.0
- aiosignal==1.3.1
- async-timeout==4.0.2
- asynctest==0.13.0
- attrs==22.2.0
- avro-python3==1.10.2
- boto3==1.18.50
- botocore==1.21.50
- certifi==2021.5.30
- chardet==3.0.4
- charset-normalizer==2.1.1
- click==8.1.3
- cycler==0.10.0
- Cython==0.29.4
- docutils==0.17.1
- enum34==1.1.10
- frozenlist==1.3.3
- fsspec==2021.8.1
- idna==2.10
- importlib-metadata==6.0.0
- jmespath==0.10.0
- joblib==1.0.1
- kiwisolver==1.3.2
- matplotlib==3.4.3
- mpmath==1.2.1
- multidict==6.0.4

- nltk==3.6.3
- numpy==1.19.5
- packaging==23.0
- pandas==1.3.2
- patsy==0.5.1
- Pillow==9.4.0
- pip==23.0
- pmdarima==1.8.2
- ptvsd==4.3.2
- pyarrow==5.0.0
- pydevd==2.5.0
- pyhocon==0.3.58
- PyMySQL==1.0.2
- pyparsing==2.4.7
- python-dateutil==2.8.2
- pytz==2021.1
- PyYAML==5.4.1
- regex==2022.10.31
- requests==2.23.0
- s3fs==2021.8.1
- s3transfer==0.5.0
- scikit-learn==0.24.2
- scipy==1.7.1
- six==1.16.0
- Spark==1.0
- statsmodels==0.12.2
- subprocess32==3.5.4
- sympy==1.8
- tbats==1.1.0

- threadpoolctl==3.1.0
- tqdm==4.64.1
- typing_extensions==4.4.0
- urllib3==1.25.11
- wheel==0.37.0
- wrapt==1.14.1
- yarl==1.8.2
- zipp==3.12.0

AWS Glue version 2.0

AWS La version 2.0 de Glue inclut les modules Python suivants prêts à l'emploi :

- avro-python3==1.10.0
- awscli==1.27.60
- boto3==1.12.4
- botocore==1.15.4
- certifi==2019.11.28
- chardet==3.0.4
- click==8.1.3
- colorama==0.4.4
- cycler==0.10.0
- Cython==0.29.15
- docutils==0.15.2
- enum34==1.1.9
- fsspec==0.6.2
- idna==2.9
- importlib-metadata==6.0.0
- jmespath==0.9.4
- joblib==0.14.1

- kiwisolver==1.1.0
- matplotlib==3.1.3
- mpmath==1.1.0
- nltk==3.5
- numpy==1.18.1
- pandas==1.0.1
- patsy==0.5.1
- pmdarima==1.5.3
- ptvsd==4.3.2
- pyarrow==0.16.0
- pyasn1==0.4.8
- pydevd==1.9.0
- pyhocon==0.3.54
- PyMySQL==0,9.3
- pyparsing==2.4.6
- python-dateutil==2.8.1
- pytz==2019.3
- PyYAML==5.3.1
- regex==2022.10.31
- requests==2.23.0
- rsa==4.7.2
- s3fs==0.4.0
- s3transfer==0.3.3
- scikit-learn==0.22.1
- scipy==1.4.1
- setuptools==45.2.0
- six==1.14.0
- Spark==1.0
- statsmodels==0.11.1

- subprocess32==3.5.4
- sympy==1.5.1
- tbats==1.0.9
- tqdm==4.64.1
- typing-extensions==4.4.0
- urllib3==1.25.8
- wheel==0.35.1
- zipp==3.12.0

Annexe A : Création d'un artefact Zip of Wheels

Nous montrons par exemple comment créer un artefact « zip of wheels ». L'exemple illustré télécharge les packages `cryptography` `scipy` dans un artefact ZIP of wheels et copie le zip of wheels vers un emplacement Amazon S3.

1. Vous devez exécuter les commandes pour créer le zip of wheels dans un environnement Amazon Linux similaire à celui de Glue. Consultez Annexe B : Détails de l'environnement AWS Glue. Glue 5.1 est utilisé AL2023 avec Python en version 3.11. Créez un Dockerfile qui créera cet environnement :

```
FROM --platform=linux/amd64 public.ecr.aws/amazonlinux/amazonlinux:2023-minimal

# Install Python 3.11, pip, and zip utility
RUN dnf install -y python3.11 pip zip && \
    dnf clean all

WORKDIR /build
```

2. Création d'un fichier requirements.txt

```
cryptography
scipy
```

3. Créez et lancez un conteneur docker

```
# Build docker image
docker build --platform linux/amd64 -t glue-wheel-builder .
```

```
# Spin up container
docker run --platform linux/amd64 -v $(pwd)/requirements.txt:/input/
requirements.txt:ro -v $(pwd):/output -it glue-wheel-builder bash
```

4. Exécutez les commandes suivantes dans l'image du docker

```
# Create a directory for the wheels
mkdir wheels

# Copy requirements.txt into wheels directory
cp /input/requirements.txt wheels/

# Download the wheels with the correct platform and Python version
pip3 download \
    -r wheels/requirements.txt \
    --dest wheels/ \
    --platform manylinux2014_x86_64 \
    --python-version 311 \
    --only-binary=:all:

# Package the wheels into a zip archive with the .gluewheels.zip suffix
zip -r mylibraries-1.0.0.gluewheels.zip wheels/

# Copy zip to output
cp mylibraries-1.0.0.gluewheels.zip /output/

# Exit the container
exit
```

5. Téléchargez Zip of Wheels sur l'emplacement Amazon S3

```
aws s3 cp mylibraries-1.0.0.gluewheels.zip s3://amzn-s3-demo-bucket/example-prefix/
```

6. Nettoyage optionnel

```
rm mylibraries-1.0.0.gluewheels.zip
rm Dockerfile
rm requirements.txt
```

7. Exécutez la tâche Glue avec les arguments de tâche suivants :

```
--additional-python-modules s3://amzn-s3-demo-bucket/example-prefix/
mylibraries-1.0.0.gluewheels.zip
```

```
--python-modules-installer-option --no-index
```

Annexe B : Détails de l'environnement AWS Glue

Compatibilité des versions de Glue et méthodes d'installation

AWS Version Glue	Version de Python	Image de base	version glibc	Tags de plateforme compatibles
5.1	3,11	Amazon Linux 2023 (AL2023)	2,34	manylinux_2_34_x86_64 manylinux_2_28_x86_64 manylinux2014_x86_64
5.0	3,11	Amazon Linux 2023 (AL2023)	2,34	manylinux_2_34_x86_64 manylinux_2_28_x86_64 manylinux2014_x86_64
4.0	3,10	Amazon Linux (2AL2)	2,26	manylinux2014_x86_64
3.0	3.7	Amazon Linux (2AL2)	2,26	manylinux2014_x86_64
2.0	3.7	AMI Amazon Linux (AL1)	2,17	manylinux2014_x86_64

Dans le cadre du modèle de responsabilitéAWS partagée, vous êtes responsable de la gestion des modules Python supplémentaires, des bibliothèques et de leurs dépendances que vous utilisez avec vos tâches AWS Glue ETL. Cela inclut l'application de mises à jour et de correctifs de sécurité.

AWS Glue ne prend pas en charge la compilation de code natif dans l'environnement de travail. Cependant, les tâches AWS Glue s'exécutent dans un environnement Linux géré par Amazon. Vous pouvez peut-être fournir vos dépendances natives sous une forme compilée via un fichier Wheel Python. Reportez-vous au tableau ci-dessus pour plus d'informations sur la compatibilité des versions de AWS Glue.

> **Important**
>
> L'utilisation de dépendances incompatibles peut entraîner des problèmes d'exécution, en particulier pour les bibliothèques dont les extensions natives doivent correspondre à l'architecture et aux bibliothèques système de l'environnement cible. Chaque version de AWS Glue fonctionne sur une version spécifique de Python avec des bibliothèques et des configurations système préinstallées.

Exemples de code Python AWS Glue

- Exemple de code : Données de jonction et de mise en relation
- Exemple de code : préparation des données à l'aide ResolveChoice de Lambda et ApplyMapping

Exemple de code : Données de jonction et de mise en relation

Cet exemple utilise un ensemble de données qui a été téléchargé sur http://everypolitician.org/ dans le compartiment `sample-dataset` d'Amazon Simple Storage Service (Amazon S3) : `s3://awsglue-datasets/examples/us-legislators/all`. Cet ensemble de données contient des données au format JSON concernant les législateurs américains et les fonctions qu'ils ont occupées à la Chambre des représentants et au Sénat. Il a été légèrement modifié et il a été mis à la disposition des utilisateurs dans un compartiment Amazon S3 public aux fins de ce didacticiel.

Le code source de cet exemple se trouve dans le `join_and_relationalize.py` fichier du référentiel AWS Glue d'échantillons du site GitHub Web.

Grâce à ces données, ce didacticiel vous montre comment effectuer les opérations suivantes :

- Utilisez un AWS Glue robot pour classer les objets stockés dans un compartiment Amazon S3 public et enregistrez leurs schémas dans le catalogue de données AWS Glue.
- Examiner les métadonnées et les schémas de la table résultant de l'analyse ;
- Écrire un script Extract-transform-load (ETL) en Python qui utilise les métadonnées dans Data Catalog pour effectuer les actions suivantes :
  - Joindre les données dans différents fichiers sources en une seule table de données (c'est-à-dire, dénormaliser les données).
  - Filtrer la table jointe en tables distinctes par type de législateur.

- Écrire les données résultantes en vue de séparer les fichiers Apache Parquet à des fins d'analyse ultérieure.

La méthode préférée pour déboguer Python ou PySpark des scripts pendant l'exécution AWS consiste à utiliser Notebooks on AWS Glue Studio.

Étape 1 : analyser les données dans le compartiment Amazon S3

1. Connectez-vous à et ouvrez AWS Management Console la AWS Glue console à l'adresse https://console.aws.amazon.com/glue/.
2. En suivant les étapes décritesConfiguration d'un robot, créez un nouveau robot d'exploration capable d'explorer le `s3://awsglue-datasets/examples/us-legislators/all` jeu de données dans une base de données nommée `legislators` dans le catalogue de données AWS Glue. Les exemples de données sont déjà dans ce compartiment Amazon S3 public.
3. Exécutez le nouvel crawler, puis activez la base de données `legislators`.

   L'crawler crée les tables de métadonnées suivantes :

   - `persons_json`
   - `memberships_json`
   - `organizations_json`
   - `events_json`
   - `areas_json`
   - `countries_r_json`

   Il s'agit d'un ensemble de tables semi-normalisé contenant les législateurs et leurs carrières.

Étape 2 : Ajouter le script Boilerplate au bloc-notes de point de terminaison de développement

Collez le script Boilerplate suivant dans le bloc-notes du point de terminaison de développement pour importer les bibliothèques AWS Glue dont vous avez besoin, et configurez un `GlueContext` :

```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
```

```
from awsglue.context import GlueContext
from awsglue.job import Job

glueContext = GlueContext(SparkContext.getOrCreate())
```

Étape 3 : examiner les schémas à partir des données de Data Catalog

Ensuite, vous pouvez facilement créer, examiner un DynamicFrame à partir du catalogue de données AWS Glue et examiner les schémas des données. Par exemple, pour afficher le schéma de la table `persons_json`, ajoutez les éléments suivants à votre bloc-notes :

```
persons = glueContext.create_dynamic_frame.from_catalog(
             database="legislators",
             table_name="persons_json")
print "Count: ", persons.count()
persons.printSchema()
```

Voici le résultat des appels d'impression :

```
Count:  1961
root
|-- family_name: string
|-- name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- gender: string
|-- image: string
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- name: string
|    |    |-- lang: string
|-- sort_name: string
|-- images: array
```

```
|    |-- element: struct
|    |    |-- url: string
|-- given_name: string
|-- birth_date: string
|-- id: string
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
|-- death_date: string
```

Chaque personne contenue dans la table est un membre de certains des organismes du Congrès des Etats-Unis.

Pour afficher le schéma de la table `memberships_json`, tapez ce qui suit :

```
memberships = glueContext.create_dynamic_frame.from_catalog(
                 database="legislators",
                 table_name="memberships_json")
print "Count: ", memberships.count()
memberships.printSchema()
```

La sortie est la suivante :

```
Count:  10439
root
|-- area_id: string
|-- on_behalf_of_id: string
|-- organization_id: string
|-- role: string
|-- person_id: string
|-- legislative_period_id: string
|-- start_date: string
|-- end_date: string
```

Les `organizations` sont des partis et les deux chambres du Congrès, le Sénat et la Chambre des représentants. Pour afficher le schéma de la table `organizations_json`, tapez ce qui suit :

```
orgs = glueContext.create_dynamic_frame.from_catalog(
```

```
           database="legislators",
           table_name="organizations_json")
print "Count: ", orgs.count()
orgs.printSchema()
```

La sortie est la suivante :

```
Count:  13
root
|-- classification: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- image: string
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- lang: string
|    |    |-- note: string
|    |    |-- name: string
|-- id: string
|-- name: string
|-- seats: int
|-- type: string
```

Étape 4 : Filtrer les données

Ensuite, conservez uniquement les champs de votre choix, et renommez `id` en `org_id`. L'ensemble de données est suffisamment petit pour que vous puissiez l'afficher entièrement.

`toDF()` convertit `DynamicFrame` en `DataFrame` Apache Spark, afin que vous puissiez appliquer les transformations qui existent déjà dans Apache Spark SQL :

```
orgs = orgs.drop_fields(['other_names',
                         'identifiers']).rename_field(
                             'id', 'org_id').rename_field(
```

```
                                'name', 'org_name')
orgs.toDF().show()
```

Le résultat est présenté ci-dessous :

```
+--------------+--------------------+--------------------+--------------------+-----
+-----------+--------------------+
|classification|              org_id|            org_name|               links|seats|
     type|               image|
+--------------+--------------------+--------------------+--------------------+-----
+-----------+--------------------+
|         party|            party/al|                  AL|                null| null|
     null|                null|
|         party|      party/democrat|            Democrat|[[website,http://...| null|
     null|https://upload.wi...|
|         party|party/democrat-li...|    Democrat-Liberal|[[website,http://...| null|
     null|                null|
|   legislature|d56acebe-8fdc-47b...|House of Represen...|                null|  435|
lower house|                null|
|         party|   party/independent|         Independent|                null| null|
     null|                null|
|         party|party/new_progres...|     New Progressive|[[website,http://...| null|
     null|https://upload.wi...|
|         party|party/popular_dem...|    Popular Democrat|[[website,http://...| null|
     null|                null|
|         party|    party/republican|          Republican|[[website,http://...| null|
     null|https://upload.wi...|
|         party|party/republican-...|Republican-Conser...|[[website,http://...| null|
     null|                null|
|         party|      party/democrat|            Democrat|[[website,http://...| null|
     null|https://upload.wi...|
|         party|   party/independent|         Independent|                null| null|
     null|                null|
|         party|    party/republican|          Republican|[[website,http://...| null|
     null|https://upload.wi...|
|   legislature|8fa6c3d2-71dc-478...|              Senate|                null|  100|
upper house|                null|
+--------------+--------------------+--------------------+--------------------+-----
+-----------+--------------------+
```

Tapez la commande suivante pour afficher les `organizations` qui apparaissent dans `memberships` :

```
memberships.select_fields(['organization_id']).toDF().distinct().show()
```

Le résultat est présenté ci-dessous :

```
+--------------------+
|     organization_id|
+--------------------+
|d56acebe-8fdc-47b...|
|8fa6c3d2-71dc-478...|
+--------------------+
```

Étape 5 : Synthèse

À présent, utilisez AWS Glue pour joindre ces tables relationnelles et créer une table complète des carrières des `memberships` des législateurs et de leurs `organizations` correspondantes.

1. Commencez par joindre `persons` et `memberships` à `id` et `person_id`.
2. Ensuite, joignez le résultat avec `orgs` à `org_id` et `organization_id`.
3. Ensuite, déplacez les champs redondants, `person_id` et `org_id`.

Vous pouvez effectuer toutes ces opérations en une seule ligne de code (étendu) :

```
l_history = Join.apply(orgs,
                       Join.apply(persons, memberships, 'id', 'person_id'),
                       'org_id', 'organization_id').drop_fields(['person_id',
 'org_id'])
print "Count: ", l_history.count()
l_history.printSchema()
```

La sortie est la suivante :

```
Count:  10439
root
|-- role: string
|-- seats: int
```

```
|-- org_name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- type: string
|-- sort_name: string
|-- area_id: string
|-- images: array
|    |-- element: struct
|    |    |-- url: string
|-- on_behalf_of_id: string
|-- other_names: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- name: string
|    |    |-- lang: string
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
|-- name: string
|-- birth_date: string
|-- organization_id: string
|-- gender: string
|-- classification: string
|-- death_date: string
|-- legislative_period_id: string
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- image: string
|-- given_name: string
|-- family_name: string
|-- id: string
|-- start_date: string
|-- end_date: string
```

Vous disposez à présent de la table finale que vous pouvez utiliser pour l'analyse. Vous pouvez l'écrire dans un format compact et efficace pour les analyses, notamment Parquet, sur lesquelles vous pouvez exécuter SQL dans AWS Glue, Amazon Athena ou Amazon Redshift Spectrum.

L'appel suivant enregistre la table dans plusieurs fichiers afin de prendre en charge des lectures parallèles rapides lors de l'exécution d'une analyse ultérieure :

```
glueContext.write_dynamic_frame.from_options(frame = l_history,
          connection_type = "s3",
          connection_options = {"path": "s3://glue-sample-target/output-dir/
legislator_history"},
          format = "parquet")
```

Pour réunir toutes les données d'historique en un seul fichier, vous devez les convertir dans un cadre de données, les repartitionner et les écrire :

```
s_history = l_history.toDF().repartition(1)
s_history.write.parquet('s3://glue-sample-target/output-dir/legislator_single')
```

Sinon, si vous souhaitez les distinguer en fonction du Sénat et de la Chambre des représentants :

```
l_history.toDF().write.parquet('s3://glue-sample-target/output-dir/legislator_part',
                               partitionBy=['org_name'])
```

Étape 6 : Transformer les données pour les bases de données relationnelles

AWS Glue permet d'écrire les données dans des bases de données relationnelles comme Amazon Redshift, même avec des données semi-structurées. Ainsi, vous obtenez une `relationalize` de transformation, qui aplatit `DynamicFrames`, quelle que soit la complexité des objets du cadre.

Dans cet exemple, à l'aide de `l_history` `DynamicFrame`, transmettez le nom d'une table racine (`hist_root`) et un chemin de travail temporaire `relationalize`. Cela renvoie une `DynamicFrameCollection`. Vous pouvez ensuite répertorier les noms des `DynamicFrames` de cet ensemble :

```
dfc = l_history.relationalize("hist_root", "s3://glue-sample-target/temp-dir/")
dfc.keys()
```

Voici la sortie de l'appel `keys` :

```
[u'hist_root', u'hist_root_contact_details', u'hist_root_links',
 u'hist_root_other_names', u'hist_root_images', u'hist_root_identifiers']
```

`Relationalize` a décomposé la table d'historique en six nouvelles tables : une table racine qui contient un enregistrement pour chaque objet dans le `DynamicFrame` et des tables auxiliaires pour les tableaux. La gestion de tableaux dans les bases de données relationnelles est souvent sous-optimale, en particulier lorsque ces tableaux deviennent volumineux. Décomposer les tableaux en différentes tables permet d'accélérer considérablement les requêtes.

Ensuite, consultez la décomposition en examinant `contact_details` :

```
l_history.select_fields('contact_details').printSchema()
dfc.select('hist_root_contact_details').toDF().where("id = 10 or id =
 75").orderBy(['id','index']).show()
```

Voici la sortie de l'appel `show` :

```
root
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
+---+-----+-----------------------+-------------------------+
| id|index|contact_details.val.type|contact_details.val.value|
+---+-----+-----------------------+-------------------------+
| 10|    0|                    fax|                         |
| 10|    1|                       |             202-225-1314|
| 10|    2|                  phone|                         |
| 10|    3|                       |             202-225-3772|
| 10|    4|                twitter|                         |
| 10|    5|                       |          MikeRossUpdates|
| 75|    0|                    fax|                         |
| 75|    1|                       |             202-225-7856|
| 75|    2|                  phone|                         |
| 75|    3|                       |             202-225-2711|
| 75|    4|                twitter|                         |
| 75|    5|                       |                SenCapito|
+---+-----+-----------------------+-------------------------+
```

Le champ `contact_details` était un tableau de structures dans le `DynamicFrame` d'origine. Chaque élément de ces tableaux est une ligne distincte dans la table auxiliaire, indexée par `index`. L'`id` ici est une clé étrangère dans la table `hist_root` avec la clé `contact_details` :

```
dfc.select('hist_root').toDF().where(
    "contact_details = 10 or contact_details = 75").select(
       ['id', 'given_name', 'family_name', 'contact_details']).show()
```

En voici la sortie :

```
+--------------------+----------+-----------+---------------+
|                  id|given_name|family_name|contact_details|
+--------------------+----------+-----------+---------------+
|f4fc30ee-7b42-432...|      Mike|       Ross|             10|
|e3c60f34-7d1b-4c0...|   Shelley|     Capito|             75|
+--------------------+----------+-----------+---------------+
```

Notez dans ces commandes que `toDF()`, puis une expression `where` sont utilisés pour filtrer les lignes à afficher.

Ainsi, la jonction de la table `hist_root` avec les tables auxiliaire vous permet d'effectuer les actions suivantes :

- Charger des données dans les bases de données sans prise en charge par un tableau ;
- Interroger chaque élément individuel dans un tableau à l'aide de SQL.

Stockez et accédez en toute sécurité à vos informations d'identification Amazon Redshift avec une connexion AWS Glue. Pour plus d'informations sur la création de votre propre connexion, consultez Connexion aux données.

Vous êtes maintenant prêt à écrire vos données dans une connexion en parcourant le `DynamicFrames` un par un :

```
for df_name in dfc.keys():
  m_df = dfc.select(df_name)
  print "Writing to table: ", df_name
```

```
 glueContext.write_dynamic_frame.from_jdbc_conf(frame = m_df, connection settings
here)
```

Les paramètres de connexion varient en fonction de votre type de base de données relationnelle :

- Pour obtenir des instructions sur l'écriture sur Amazon Redshift, consultez the section called "Connexions Redshift".
- Pour d'autres bases de données, consultez Types et options de connexion pour ETL dans AWS Glue pour Spark.

Conclusion

En général, AWS Glue est très flexible. Il vous permet de réaliser, en quelques lignes de code, ce qui prendrait normalement plusieurs jours pour écrire. Vous pouvez trouver l'intégralité des scripts source-to-target ETL dans le fichier Python `join_and_relationalize.py` dans les AWS Glueexemples ci-dessous GitHub.

Exemple de code : préparation des données à l'aide ResolveChoice de Lambda et ApplyMapping

L'ensemble de données utilisé dans cet exemple comprend les données de paiement des prestataires de soins médicaux téléchargées à partir de deux ensembles de données Data.cms.gov : « Résumé des fournisseurs potentiels du système de paiement pour les patients hospitalisés pour les 100 principaux groupes liés au diagnostic - FY2011 » et « Données sur les frais d'hospitalisation pour l'exercice 2011 ». Après le téléchargement des données, nous avons modifié le jeu de données de manière à introduire quelques enregistrements erronés à la fin du fichier. Ce fichier modifié est situé dans un compartiment Amazon S3 public à l'adresse `s3://awsglue-datasets/examples/medicare/Medicare_Hospital_Provider.csv`.

Le code source de cet exemple se trouve dans le `data_cleaning_and_lambda.py` fichier du GitHub référentiel AWS Glued'exemples.

La méthode préférée pour déboguer Python ou PySpark des scripts pendant l'exécution AWS consiste à utiliser Notebooks on AWS Glue Studio.

Étape 1 : analyser les données dans le compartiment Amazon S3

1. Connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/.
2. En suivant le processus décrit dansConfiguration d'un robot, créez un nouveau robot d'exploration capable d'explorer le `s3://awsglue-datasets/examples/medicare/`

`Medicare_Hospital_Provider.csv` fichier et de placer les métadonnées obtenues dans une base de données nommée `payments` dans le catalogue de données AWS Glue.

3. Exécutez le nouvel crawler, puis activez la base de données `payments`. Vous devriez constater que l'crawler a créé un tableau de métadonnées nommé `medicare` dans la base de données après avoir lu le début du fichier pour déterminer son format et son délimiteur.

   Le schéma de la nouvelle table `medicare` se présente comme suit :

```
Column  name                              Data type
================================================
drg definition                              string
provider id                                 bigint
provider name                               string
provider street address                     string
provider city                               string
provider state                              string
provider zip code                           bigint
hospital referral region description        string
total discharges                            bigint
average covered charges                     string
average total payments                      string
average medicare payments                   string
```

Étape 2 : Ajouter le script Boilerplate au bloc-notes de point de terminaison de développement

Collez le script Boilerplate suivant dans le bloc-notes du point de terminaison de développement pour importer les bibliothèques AWS Glue dont vous avez besoin, et configurez un `GlueContext` :

```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job

glueContext = GlueContext(SparkContext.getOrCreate())
```

Étape 3 : Comparer les analyses des différents schémas

Ensuite, vous pouvez voir si le schéma qui a été reconnu par un `DataFrame` Apache Spark est le même que celui que votre crawler AWS Glue a enregistré. Exécutez ce code :

```
medicare = spark.read.format(
   "com.databricks.spark.csv").option(
   "header", "true").option(
   "inferSchema", "true").load(
   's3://awsglue-datasets/examples/medicare/Medicare_Hospital_Provider.csv')
medicare.printSchema()
```

Voici la sortie de l'appel `printSchema` :

```
root
 |-- DRG Definition: string (nullable = true)
 |-- Provider Id: string (nullable = true)
 |-- Provider Name: string (nullable = true)
 |-- Provider Street Address: string (nullable = true)
 |-- Provider City: string (nullable = true)
 |-- Provider State: string (nullable = true)
 |-- Provider Zip Code: integer (nullable = true)
 |-- Hospital Referral Region Description: string (nullable = true)
 |--  Total Discharges : integer (nullable = true)
 |--  Average Covered Charges : string (nullable = true)
 |--  Average Total Payments : string (nullable = true)
 |-- Average Medicare Payments: string (nullable = true)
```

Ensuite, examinez le schéma généré par un `DynamicFrame` AWS Glue :

```
medicare_dynamicframe = glueContext.create_dynamic_frame.from_catalog(
       database = "payments",
       table_name = "medicare")
medicare_dynamicframe.printSchema()
```

La sortie de `printSchema` est la suivante :

```
root
```

```
 |-- drg definition: string
 |-- provider id: choice
 |    |-- long
 |    |-- string
 |-- provider name: string
 |-- provider street address: string
 |-- provider city: string
 |-- provider state: string
 |-- provider zip code: long
 |-- hospital referral region description: string
 |-- total discharges: long
 |-- average covered charges: string
 |-- average total payments: string
 |-- average medicare payments: string
```

Le `DynamicFrame` génère un schéma dans lequel `provider id` peut être un type `long` ou `string`. Le schéma `DataFrame` répertorie `Provider Id` comme étant un type `string` et le `provider id` répertorie Data Catalog comme étant un type `bigint`.

Lequel est correct ? Il existe deux enregistrements à la fin du fichier (sur un total de 160 000 enregistrements) avec des valeurs `string` dans cette colonne. Il s'agit des enregistrements erronés qui ont été introduits pour illustrer un problème.

Pour résoudre ce type de problème, `DynamicFrame` AWS Glue présente le concept de type choice. Dans ce cas, `DynamicFrame` montre que les valeurs `long` et `string` peuvent apparaître dans cette colonne. L'crawler AWS Glue a manqué les valeurs `string` car il seulement a pris en compte un préfixe de 2 Mo de données. Le `DataFrame` Apache Spark a pris en compte la totalité de l'ensemble de données, mais il a été contraint d'affecter le type le plus général à la colonne, à savoir `string`. En fait, Spark a souvent recours au cas le plus général pour des types ou des variations complexes avec lesquels il n'est pas familier.

Pour interroger la colonne `provider id`, commencez par résoudre le type de choix. Vous pouvez utiliser la méthode de transformation `resolveChoice` dans votre `DynamicFrame` pour convertir ces valeurs `string` en valeurs `long` avec une option `cast:long` :

```
medicare_res = medicare_dynamicframe.resolveChoice(specs = [('provider
 id','cast:long')])
medicare_res.printSchema()
```

La sortie `printSchema` est désormais :

```
root
 |-- drg definition: string
 |-- provider id: long
 |-- provider name: string
 |-- provider street address: string
 |-- provider city: string
 |-- provider state: string
 |-- provider zip code: long
 |-- hospital referral region description: string
 |-- total discharges: long
 |-- average covered charges: string
 |-- average total payments: string
 |-- average medicare payments: string
```

Lorsque la valeur `string` ne peut pas être convertie, AWS Glue insère une valeur `null`.

Une autre option consiste à convertir le type de choix en `struct`, qui conserve les valeurs des deux types.

Ensuite, regardez les lignes qui étaient anormales :

```
medicare_res.toDF().where("'provider id' is NULL").show()
```

Les informations suivantes s'affichent :

```
+--------------------+-----------+--------------+-----------------------+-------------
+-------------+----------------+-------------------------------------+---------------
+-----------------------+----------------------+------------------------+
|      drg definition|provider id|  provider name|provider street address|provider
 city|provider state|provider zip code|hospital referral region description|total
 discharges|average covered charges|average total payments|average medicare payments|
+--------------------+-----------+--------------+-----------------------+-------------
+-------------+----------------+-------------------------------------+---------------
+-----------------------+----------------------+------------------------+
|948 - SIGNS & SYM...|       null|            INC|       1050 DIVISION ST|
 MAUSTON|            WI|            53948|                        WI - Madison|
      12|             $11961.41|             $4619.00|                $3775.33|
|948 - SIGNS & SYM...|       null| INC- ST JOSEPH|     5000 W CHAMBERS ST|
 MILWAUKEE|            WI|            53210|                        WI - Milwaukee|
        14|             $10514.28|             $5562.50|                $4522.78|
```

```
+--------------------+----------+--------------+----------------------+------------
+-------------+----------------+-------------------------------------+---------------
+----------------------+----------------------+------------------------+
```

Maintenant, supprimez les deux enregistrements incorrects, comme suit :

```
medicare_dataframe = medicare_res.toDF()
medicare_dataframe = medicare_dataframe.where("'provider id' is NOT NULL")
```

Étape 4 : Mapper les données et utiliser les fonctions Lambda Apache Spark

AWS Glue ne prend pas encore directement en charge les fonctions Lambda, également connues sous le nom de fonctions définies par l'utilisateur. Mais vous pouvez toujours convertir un `DynamicFrame` vers et depuis un `DataFrame` Apache Spark pour tirer avantage de la fonction Spark en plus des fonctions spécifiques de `DynamicFrames`.

Ensuite, convertissez les informations de paiement en chiffres, de sorte que les moteurs analytiques tels qu'Amazon Redshift ou Amazon Athena puissent analyser les chiffres plus rapidement :

```
from pyspark.sql.functions import udf
from pyspark.sql.types import StringType

chop_f = udf(lambda x: x[1:], StringType())
medicare_dataframe = medicare_dataframe.withColumn(
        "ACC", chop_f(
            medicare_dataframe["average covered charges"])).withColumn(
                "ATP", chop_f(
                    medicare_dataframe["average total payments"])).withColumn(
                        "AMP", chop_f(
                            medicare_dataframe["average medicare payments"]))
medicare_dataframe.select(['ACC', 'ATP', 'AMP']).show()
```

La sortie de l'appel `show` est la suivante :

```
+--------+-------+-------+
|     ACC|    ATP|    AMP|
+--------+-------+-------+
|32963.07|5777.24|4763.73|
|15131.85|5787.57|4976.71|
|37560.37|5434.95|4453.79|
|13998.28|5417.56|4129.16|
|31633.27|5658.33|4851.44|
```

```
|16920.79|6653.80|5374.14|
|11977.13|5834.74|4761.41|
|35841.09|8031.12|5858.50|
|28523.39|6113.38|5228.40|
|75233.38|5541.05|4386.94|
|67327.92|5461.57|4493.57|
|39607.28|5356.28|4408.20|
|22862.23|5374.65|4186.02|
|31110.85|5366.23|4376.23|
|25411.33|5282.93|4383.73|
| 9234.51|5676.55|4509.11|
|15895.85|5930.11|3972.85|
|19721.16|6192.54|5179.38|
|10710.88|4968.00|3898.88|
|51343.75|5996.00|4962.45|
+--------+-------+-------+
only showing top 20 rows
```

Cela reste toujours des chaînes dans les données. Nous pouvons utiliser la puissante méthode de transformation `apply_mapping` pour supprimer, renommer, analyser et imbriquer les données afin que d'autres systèmes et langages de programmation puissent facilement y accéder :

```
from awsglue.dynamicframe import DynamicFrame
medicare_tmp_dyf = DynamicFrame.fromDF(medicare_dataframe, glueContext, "nested")
medicare_nest_dyf = medicare_tmp_dyf.apply_mapping([('drg definition', 'string', 'drg',
 'string'),
                ('provider id', 'long', 'provider.id', 'long'),
                ('provider name', 'string', 'provider.name', 'string'),
                ('provider city', 'string', 'provider.city', 'string'),
                ('provider state', 'string', 'provider.state', 'string'),
                ('provider zip code', 'long', 'provider.zip', 'long'),
                ('hospital referral region description', 'string','rr', 'string'),
                ('ACC', 'string', 'charges.covered', 'double'),
                ('ATP', 'string', 'charges.total_pay', 'double'),
                ('AMP', 'string', 'charges.medicare_pay', 'double')])
medicare_nest_dyf.printSchema()
```

La sortie `printSchema` est la suivante :

```
root
 |-- drg: string
 |-- provider: struct
```

```
 |    |-- id: long
 |    |-- name: string
 |    |-- city: string
 |    |-- state: string
 |    |-- zip: long
 |-- rr: string
 |-- charges: struct
 |    |-- covered: double
 |    |-- total_pay: double
 |    |-- medicare_pay: double
```

Vous pouvez reconvertir les données en `DataFrame` Spark, comme suit :

```
medicare_nest_dyf.toDF().show()
```

La sortie est la suivante :

```
+--------------------+--------------------+--------------+--------------------+
|                 drg|            provider|            rr|             charges|
+--------------------+--------------------+--------------+--------------------+
|039 - EXTRACRANIA...|[10001,SOUTHEAST ...|    AL - Dothan|[32963.07,5777.24...|
|039 - EXTRACRANIA...|[10005,MARSHALL M...|AL - Birmingham|[15131.85,5787.57...|
|039 - EXTRACRANIA...|[10006,ELIZA COFF...|AL - Birmingham|[37560.37,5434.95...|
|039 - EXTRACRANIA...|[10011,ST VINCENT...|AL - Birmingham|[13998.28,5417.56...|
|039 - EXTRACRANIA...|[10016,SHELBY BAP...|AL - Birmingham|[31633.27,5658.33...|
|039 - EXTRACRANIA...|[10023,BAPTIST ME...|AL - Montgomery|[16920.79,6653.8,...|
|039 - EXTRACRANIA...|[10029,EAST ALABA...|AL - Birmingham|[11977.13,5834.74...|
|039 - EXTRACRANIA...|[10033,UNIVERSITY...|AL - Birmingham|[35841.09,8031.12...|
|039 - EXTRACRANIA...|[10039,HUNTSVILLE...|AL - Huntsville|[28523.39,6113.38...|
|039 - EXTRACRANIA...|[10040,GADSDEN RE...|AL - Birmingham|[75233.38,5541.05...|
|039 - EXTRACRANIA...|[10046,RIVERVIEW ...|AL - Birmingham|[67327.92,5461.57...|
|039 - EXTRACRANIA...|[10055,FLOWERS HO...|    AL - Dothan|[39607.28,5356.28...|
|039 - EXTRACRANIA...|[10056,ST VINCENT...|AL - Birmingham|[22862.23,5374.65...|
|039 - EXTRACRANIA...|[10078,NORTHEAST ...|AL - Birmingham|[31110.85,5366.23...|
|039 - EXTRACRANIA...|[10083,SOUTH BALD...|    AL - Mobile|[25411.33,5282.93...|
|039 - EXTRACRANIA...|[10085,DECATUR GE...|AL - Huntsville|[9234.51,5676.55,...|
|039 - EXTRACRANIA...|[10090,PROVIDENCE...|    AL - Mobile|[15895.85,5930.11...|
|039 - EXTRACRANIA...|[10092,D C H REGI...|AL - Tuscaloosa|[19721.16,6192.54...|
|039 - EXTRACRANIA...|[10100,THOMAS HOS...|    AL - Mobile|[10710.88,4968.0,...|
|039 - EXTRACRANIA...|[10103,BAPTIST ME...|AL - Birmingham|[51343.75,5996.0,...|
+--------------------+--------------------+--------------+--------------------+
only showing top 20 rows
```

Étape 5 : Écrire les données dans Apache Parquet

AWS Glue permet d'écrire facilement des données dans un format tel que Apache Parquet, que les bases de données relationnelles peuvent utiliser de manière efficace :

```
glueContext.write_dynamic_frame.from_options(
       frame = medicare_nest_dyf,
       connection_type = "s3",
       connection_options = {"path": "s3://glue-sample-target/output-dir/
medicare_parquet"},
       format = "parquet")
```

AWS Référence des PySpark extensions Glue

AWS Glue a créé les extensions suivantes pour le dialecte PySpark Python.

- Accès aux paramètres à l'aide de `getResolvedOptions`
- PySpark types d'extensions
- DynamicFrame classe
- DynamicFrameCollection classe
- DynamicFrameWriter classe
- DynamicFrameReader classe
- GlueContext classe

Accès aux paramètres à l'aide de **`getResolvedOptions`**

La fonction utilitaire AWS Glue `getResolvedOptions(args, options)` vous donne accès aux arguments transmis à votre script lorsque vous exécutez une tâche. Pour utiliser cette fonction, commencez par l'importer à partir du module AWS Glue `utils`, avec le module `sys` :

```
import sys
from awsglue.utils import getResolvedOptions
```

`getResolvedOptions(args, options)`

- `args` - Liste des arguments contenus dans `sys.argv`.
- `options` - Tableau Python des noms d'arguments que vous souhaitez extraire.

Example Récupération des arguments transmis à un JobRun

Supposons que vous ayez créé un JobRun dans un script, peut-être dans une fonction Lambda :

```
response = client.start_job_run(
             JobName = 'my_test_Job',
             Arguments = {
               '--day_partition_key':   'partition_0',
               '--hour_partition_key':  'partition_1',
               '--day_partition_value':  day_partition_value,
               '--hour_partition_value': hour_partition_value } )
```

Pour extraire les arguments transmis, vous pouvez utiliser la fonction `getResolvedOptions` comme suit :

```
import sys
from awsglue.utils import getResolvedOptions

args = getResolvedOptions(sys.argv,
                          ['JOB_NAME',
                           'day_partition_key',
                           'hour_partition_key',
                           'day_partition_value',
                           'hour_partition_value'])
print "The day-partition key is: ", args['day_partition_key']
print "and the day-partition value is: ", args['day_partition_value']
```

Notez que chaque argument est défini avec deux traits d'union au début, puis est référencé dans le script sans trait d'union. Les arguments utilisent uniquement des traits de soulignement, pas des traits d'union. Vos arguments doivent suivre cette convention pour être résolus.

PySpark types d'extensions

Les types utilisés par les AWS Glue PySpark extensions.

DataType

Classe de base pour les autres types de AWS Glue.

`__init__(properties={})`

- `properties` – Propriétés des types de données (facultatif).

`typeName(cls)`

Renvoie le type de la classe de type AWS Glue (c'est-à-dire, le nom de classe avec « Type » supprimé de la fin).

- `cls` – Instance de classe AWS Glue dérivée de `DataType`.

`jsonValue( )`

Renvoie un objet JSON qui contient le type de données et les propriétés de la classe :

```
{
  "dataType": typeName,
  "properties": properties
}
```

AtomicType et dérivés simples

Hérite de la classe DataType et l'étend, et sert de classe de base pour tous les types de données atomiques AWS Glue .

`fromJsonValue(cls, json_value)`

Initialise une instance de classe avec les valeurs d'un objet JSON.

- `cls` – Instance de classe de type AWS Glue.
- `json_value` – Objet JSON à partir duquel charger les paires clé-valeur.

Les types suivants sont des dérivés simples de la classe AtomicType :

- `BinaryType` – Données binaires.
- `BooleanType` – Valeurs booléennes.
- `ByteType` – Valeur byte.
- `DateType` – Valeur datetime.
- `DoubleType` – Valeur double à virgule flottante.

- `IntegerType` – Une valeur d'entier.
- `LongType` - Entier long.
- `NullType` – Valeur null.
- `ShortType` – Entier court.
- `StringType` – Chaîne texte.
- `TimestampType` – Valeur timestamp (généralement en secondes à partir du 01/01/1970).
- `UnknownType` – Valeur de type non identifié.

DecimalType(AtomicType)

Hérite depuis la classe AtomicType et l'étend pour représenter un nombre décimal (nombre exprimé en décimales, par opposition aux nombres binaires de base 2).

`__init__(precision=10, scale=2, properties={})`

- `precision` – Nombre de chiffres dans le nombre décimal (facultatif ; la valeur par défaut est 10).
- `scale` – Nombre de chiffres à droite du point décimal (facultatif ; la valeur par défaut est 2).
- `properties` – Propriétés du nombre décimal (facultatif).

EnumType(AtomicType)

Hérite de la classe AtomicType et l'étend pour représenter une énumération des options valides.

`__init__(options)`

- `options` – Liste des options énumérées.

Types de collections

- ArrayType(DataType)
- ChoiceType(DataType)
- MapType(DataType)
- Field(Object)
- StructType(DataType)
- EntityType(DataType)

ArrayType(DataType)

`__init__(elementType=UnknownType(), properties={})`

- `elementType`— Le type des éléments du tableau (facultatif ; la valeur par défaut est UnknownType).
- `properties` – Propriétés du tableau (facultatif).

ChoiceType(DataType)

`__init__(choices=[], properties={})`

- `choices` – Liste des choix possibles (facultatif).
- `properties` – Propriétés de ces options (facultatif).

`add(new_choice)`

Ajoute un nouveau choix à la liste des choix possibles.

- `new_choice` – Choix à ajouter à la liste des choix possibles.

`merge(new_choices)`

Fusionne une liste de nouveaux choix avec la liste de choix existantes.

- `new_choices` – Liste des nouveaux choix à fusionner avec les choix existants.

MapType(DataType)

`__init__(valueType=UnknownType, properties={})`

- `valueType`— Le type de valeurs sur la carte (facultatif ; la valeur par défaut est UnknownType).
- `properties` – Propriétés de la map (facultatif).

Field(Object)

Crée un objet champ hors d'un objet qui dérive de DataType.

`__init__(name, dataType, properties={})`

- `name` – Nom à attribuer au champ.
- `dataType` – Objet à partir duquel créer un champ.
- `properties` – Propriétés du champ (facultatif).

StructType(DataType)

Définit une structure de données (`struct`).

`__init__(fields=[], properties={})`

- `fields` – Liste des champs (de type `Field`) à inclure dans la structure (facultatif).
- `properties` – Propriétés de la structure (facultatif).

`add(field)`

- `field` – Objet de type `Field` à ajouter à la structure.

`hasField(field)`

Renvoie `True` si la structure dispose d'un champ du même nom, ou `False` si ce n'est pas le cas.

- `field` – Nom de champ ou objet de type `Field` dont le nom est utilisé.

`getField(field)`

- `field` – nom de champ ou objet de type `Field` dont le nom est utilisé. Si la structure a un champ du même nom, il est retourné.

EntityType(DataType)

`__init__(entity, base_type, properties)`

Cette classe n'a pas encore été implémentée.

Autres types

- DataSource(objet)
- DataSink(objet)

DataSource(objet)

`__init__(j_source, sql_ctx, name)`

- `j_source` – Source de données.
- `sql_ctx` – Contexte SQL.
- `name` – Nom de la source de données.

`setFormat(format, **options)`

- `format` – Format à définir pour la source de données.
- `options` – Collection d'options à définir pour la source de données. Pour plus d'informations sur ces options de format, consultez the section called "Options de format de données".

`getFrame()`

Renvoie un `DynamicFrame` pour la source de données.

DataSink(objet)

`__init__(j_sink, sql_ctx)`

- `j_sink` – Récepteur à créer.
- `sql_ctx` – Contexte SQL pour le récepteur de données.

`setFormat(format, **options)`

- `format` – Format à définir pour le récepteur de données.
- `options` – Collection d'options à définir pour le récepteur de données. Pour plus d'informations sur ces options de format, consultez the section called "Options de format de données".

`setAccumulableSize(size)`

- `size` – Taille cumulable à définir, en octets.

`writeFrame(dynamic_frame, info="")`

- `dynamic_frame` – Objet `DynamicFrame` à écrire.
- `info` – Informations sur l'objet `DynamicFrame` (facultatif).

`write(dynamic_frame_or_dfc, info="")`

Écrit un `DynamicFrame` ou un `DynamicFrameCollection`.

- `dynamic_frame_or_dfc` – Objet `DynamicFrame` ou `DynamicFrameCollection` à écrire.
- `info` – Informations sur le `DynamicFrame` ou `DynamicFrames` à écrire (facultatif).

DynamicFrame classe

L'une des principales abstractions dans Apache Spark est le `DataFrame` SparkSQL, ce qui est similaire à la construction `DataFrame` présente dans R et Pandas. A `DataFrame` est similaire à une table et prend en charge les opérations de style fonctionnel (map/reduce/filter/etc.) et les opérations SQL (sélection, projet, agrégation).

`DataFrames`Les sont puissants et largement utilisés, mais ils présentent des limitations au niveau des opérations d'extraction, de transformation et de chargement (opérations ETL). Cela signifie notamment qu'un schéma doit être spécifié avant tout chargement de données. Pour faire face à ce problème, SparkSQL effectue deux passages sur les données : un premier passage pour déduire le schéma et un second pour charger les données. Cependant, cette inférence est limitée et ne prend pas en compte la désorganisation des données. Par exemple, un même champ peut avoir différents types dans différents enregistrements. L'opération est souvent abandonnée par Apache Spark, qui indique que le type est `string` en tenant compte du texte du champ d'origine. Cette information peut être erronée et vous pouvez souhaiter exercer un meilleur contrôle sur la résolution des écarts entre les schémas. Et pour les ensembles de données volumineux, un passage supplémentaire sur les données source peut s'avérer hors de prix.

Pour remédier à ces limites, AWS Glue introduit leDynamicFrame. Un DynamicFrame est similaire à un DataFrame, mais chaque enregistrement y est auto-descriptif : initialement, aucun schéma n'est donc requis. AWS GlueCalcule plutôt un schéma on-the-fly lorsque cela est nécessaire et code explicitement les incohérences du schéma à l'aide d'un type de choix (ou d'union). Vous pouvez résoudre ces incohérences afin de rendre vos ensembles de données compatibles avec les magasins de données qui nécessitent un schéma fixe.

De même, un DynamicRecord représente un enregistrement logique au sein dans un DynamicFrame. Il est comparable à une ligne dans un DataFrame Spark, à la différence qu'il est auto-descriptif et peut être utilisé pour les données qui ne sont pas conformes à un schéma fixe. Lorsque vous utilisez AWS Glue with PySpark, vous ne manipulez généralement pas de manière indépendanteDynamicRecords. Vous allez plutôt transformer le jeu de données par le biais de son DynamicFrame.

Vous pouvez convertir DynamicFrames vers et depuis DataFrames après la résolution des incohérences de schémas.

– construction –

- __init__
- fromDF
- toDF

__init__

__init__(jdf, glue_ctx, name)

- jdf – Référence à la trame de données dans la machine virtuelle Java (JVM).
- glue_ctx – Un objet GlueContext classe.
- name – Nom de chaîne facultatif, vide par défaut.

fromDF

fromDF(dataframe, glue_ctx, name)

Convertit un DataFrame en DynamicFrame via la conversion de champs DataFrame en champs DynamicRecord. Renvoie un nouveau DynamicFrame.

Un `DynamicRecord` représente un enregistrement logique dans un `DynamicFrame`. Il est comparable à une ligne dans un `DataFrame` Spark, à la différence qu'il est auto-descriptif et peut être utilisé pour les données qui ne sont pas conformes à un schéma fixe.

Cette fonction s'attend à ce que les colonnes dont les noms sont dupliqués dans votre `DataFrame` aient déjà été résolues.

- `dataframe` – `DataFrame` Apache Spark SQL à convertir (obligatoire).
- `glue_ctx` – Objet GlueContext classe qui spécifie le contexte pour cette transformation (obligatoire).
- `name`— Le nom du résultat `DynamicFrame` (facultatif depuis AWS Glue 3.0).

toDF

`toDF(options)`

Convertit un `DynamicFrame` en `DataFrame` Apache Spark via la conversion de `DynamicRecords` en champs `DataFrame`. Renvoie un nouveau `DataFrame`.

Un `DynamicRecord` représente un enregistrement logique dans un `DynamicFrame`. Il est comparable à une ligne dans un `DataFrame` Spark, à la différence qu'il est auto-descriptif et peut être utilisé pour les données qui ne sont pas conformes à un schéma fixe.

- `options`— Liste d'`ResolveOption`objets indiquant comment résoudre les types de choix lors de la conversion. Ce paramètre est utilisé pour gérer les incohérences du schéma, et non pour les options de format telles que l'analyse CSV.

  Pour l'analyse CSV et les autres options de format, spécifiez-les dans la `from_options` méthode lors de sa création DynamicFrame, et non dans la `toDF` méthode.

  Voici un exemple de la bonne façon de gérer les options de format CSV :

```
from awsglue.context import GlueContext
from awsglue.dynamicframe import DynamicFrame
from pyspark.context import SparkContext

sc = SparkContext()
glueContext = GlueContext(sc)

# Correct: Specify format options in from_options
```

```
csv_dyf = glueContext.create_dynamic_frame.from_options(
    connection_type="s3",
    connection_options={"paths": ["s3://my-bucket/path/to/csv/"]},
    format="csv",
    format_options={
        "withHeader": True,
        "separator": ",",
        "inferSchema": True
    }
)

# Convert to DataFrame (no format options needed here)
csv_df = csv_dyf.toDF()
```

Le `options` paramètre in `toDF` est spécifiquement destiné à la résolution des types de choix. Spécifiez le type de cible si vous choisissez le type d'action `Project` et `Cast`. Voici quelques exemples :

```
>>>toDF([ResolveOption("a.b.c", "KeepAsStruct")])
>>>toDF([ResolveOption("a.b.c", "Project", DoubleType())])
```

– informations –

- count
- schéma
- printSchema
- show
- repartition
- coalesce

count

`count( )` – Renvoie le nombre de lignes dans le `DataFrame` sous-jacent.

schéma

`schema( )` – Renvoie le schéma de ce `DynamicFrame` ou, s'il n'est pas disponible, le schéma du `DataFrame` sous-jacent.

Pour plus d'informations sur les types de `DynamicFrame` qui composent ce schéma, consultez the section called "Types".

printSchema

`printSchema( )` – Imprime le schéma du `DataFrame` sous-jacent.

show

`show(num_rows)` – Imprime un nombre de lignes spécifié du `DataFrame` sous-jacent.

repartition

`repartition(numPartitions)` – renvoie un nouvel objet `DynamicFrame` avec des partitions `numPartitions`.

coalesce

`coalesce(numPartitions)` – renvoie un nouvel objet `DynamicFrame` avec des partitions `numPartitions`.

– transformations –

- apply_mapping
- drop_fields
- filtre
- joindre
- map
- mergeDynamicFrame
- relationalize
- rename_field
- resolveChoice
- select_fields
- spigot
- split_fields
- split_rows
- unbox

- the section called "union"
- unnest
- unnest_ddb_json
- write

apply_mapping

`apply_mapping(mappings, transformation_ctx="", info="", stageThreshold=0, totalThreshold=0)`

Applique un mappage déclaratif à une image `DynamicFrame` et renvoie une nouvelle image `DynamicFrame` sur laquelle ces mappages sont appliqués. Les champs non spécifiés sont omis dans le nouveau`DynamicFrame`.

- `mappings`— Une liste de tuples de mappage (obligatoire). Chacun étant composé comme suit : (colonne source, type source, colonne cible, type cible).

  Si la colonne source comporte un point « `.` » dans le nom, vous devez l'entourer d'accents graves « ``` ». Par exemple, pour mapper `this.old.name` (chaîne) à `thisNewName`, vous devez utiliser le tuple suivant :

  ```
  ("`this.old.name`", "string", "thisNewName", "string")
  ```

- `transformation_ctx` – Chaîne unique utilisée pour identifier les informations sur l'état (facultatif).
- `info` – Chaîne à associer avec le signalement des erreurs pour cette transformation (facultatif).
- `stageThreshold` – Nombre d'erreurs rencontrées pendant cette transformation et qui doit entraîner l'erreur générée par le processus (facultatif) La valeur par défaut est zéro, ce qui indique que le processus ne doit pas générer d'erreur.
- `totalThreshold` – Nombre d'erreurs rencontrées jusqu'à cette transformation, incluse, et qui doit entraîner l'arrêt du processus (facultatif). La valeur par défaut est zéro, ce qui indique que le processus ne doit pas générer d'erreur.

Exemple : Utilisez apply_mapping afin de renommer des champs ainsi que de modifier ses types

L'exemple de code suivant montre la `apply_mapping`méthode d'utilisation relative au renommage des champs sélectionnés ainsi qu' à la modification des types de champs.

> **Note**
>
> Pour accéder au jeu de données utilisé dans cet exemple, voir Exemple de code : Données de jonction et de mise en relation et suivez les instructions de la sectionÉtape 1 : analyser les données dans le compartiment Amazon S3.

```
# Example: Use apply_mapping to reshape source data into
# the desired column names and types as a new DynamicFrame

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Create a DynamicFrame and view its schema
persons = glueContext.create_dynamic_frame.from_catalog(
    database="legislators", table_name="persons_json"
)
print("Schema for the persons DynamicFrame:")
persons.printSchema()

# Select and rename fields, change field type
print("Schema for the persons_mapped DynamicFrame, created with apply_mapping:")
persons_mapped = persons.apply_mapping(
    [
        ("family_name", "String", "last_name", "String"),
        ("name", "String", "first_name", "String"),
        ("birth_date", "String", "date_of_birth", "Date"),
    ]
)
persons_mapped.printSchema()
```

Output

```
Schema for the persons DynamicFrame:
root
|-- family_name: string
|-- name: string
```

```
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- gender: string
|-- image: string
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- lang: string
|    |    |-- note: string
|    |    |-- name: string
|-- sort_name: string
|-- images: array
|    |-- element: struct
|    |    |-- url: string
|-- given_name: string
|-- birth_date: string
|-- id: string
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
|-- death_date: string

Schema for the persons_mapped DynamicFrame, created with apply_mapping:
root
|-- last_name: string
|-- first_name: string
|-- date_of_birth: date
```

drop_fields

`drop_fields(paths, transformation_ctx="", info="", stageThreshold=0, totalThreshold=0)`

Appelle la transformation FlatMap classe afin de supprimer des champs d'un `DynamicFrame`. Renvoie un nouveau `DynamicFrame` avec les champs supprimés spécifiés.

- `paths` - Une liste de chaînes. Chacune contient le chemin d'accès complet à un nœud de champ que vous souhaitez supprimer. Vous pouvez utiliser la notation par points pour spécifier des champs imbriqués. Par exemple, si le champ`first` est un enfant du champ `name`dans l'arborescence, vous spécifiez`"name.first"`pour le chemin d'accès.

  Si le nom d'un nœud de champ contient un `.` littéral, vous devez l'entourer d'accents graves (`` ` ``).
- `transformation_ctx` – Chaîne unique utilisée pour identifier les informations sur l'état (facultatif).
- `info` – Chaîne à associer avec le signalement des erreurs pour cette transformation (facultatif).
- `stageThreshold` – Nombre d'erreurs rencontrées pendant cette transformation et qui doit entraîner l'erreur générée par le processus (facultatif) La valeur par défaut est zéro, ce qui indique que le processus ne doit pas générer d'erreur.
- `totalThreshold` – Nombre d'erreurs rencontrées jusqu'à cette transformation, incluse, et qui doit entraîner l'arrêt du processus (facultatif). La valeur par défaut est zéro, ce qui indique que le processus ne doit pas générer d'erreur.

Exemple : utilisez supprimer les chams pour supprimer les champs d'un**`DynamicFrame`**

Cet exemple de code utilise le`drop_fields`méthode pour supprimer les champs de niveau supérieur et imbriqués sélectionnés d'un`DynamicFrame`.

Exemple de jeu de données

L'exemple utilise l'ensemble de données suivant qui est représenté par`EXAMPLE-FRIENDS-DATA`table dans le code :

```
{"name": "Sally", "age": 23, "location": {"state": "WY", "county": "Fremont"},
 "friends": []}
{"name": "Varun", "age": 34, "location": {"state": "NE", "county": "Douglas"},
 "friends": [{"name": "Arjun", "age": 3}]}
{"name": "George", "age": 52, "location": {"state": "NY"}, "friends": [{"name":
 "Fred"}, {"name": "Amy", "age": 15}]}
{"name": "Haruki", "age": 21, "location": {"state": "AK", "county": "Denali"}}
{"name": "Sheila", "age": 63, "friends": [{"name": "Nancy", "age": 22}]}
```

Exemple de code

```
# Example: Use drop_fields to remove top-level and nested fields from a DynamicFrame.
```

```
# Replace MY-EXAMPLE-DATABASE with your Glue Data Catalog database name.
# Replace EXAMPLE-FRIENDS-DATA with your table name.

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Create a DynamicFrame from Glue Data Catalog
glue_source_database = "MY-EXAMPLE-DATABASE"
glue_source_table = "EXAMPLE-FRIENDS-DATA"

friends = glueContext.create_dynamic_frame.from_catalog(
    database=glue_source_database, table_name=glue_source_table
)
print("Schema for friends DynamicFrame before calling drop_fields:")
friends.printSchema()

# Remove location.county, remove friends.age, remove age
friends = friends.drop_fields(paths=["age", "location.county", "friends.age"])
print("Schema for friends DynamicFrame after removing age, county, and friend age:")
friends.printSchema()
```

Output

```
Schema for friends DynamicFrame before calling drop_fields:
root
|-- name: string
|-- age: int
|-- location: struct
|    |-- state: string
|    |-- county: string
|-- friends: array
|    |-- element: struct
|    |    |-- name: string
|    |    |-- age: int

Schema for friends DynamicFrame after removing age, county, and friend age:
root
|-- name: string
|-- location: struct
```

```
|    |-- state: string
|-- friends: array
|    |-- element: struct
|    |    |-- name: string
```

filtre

`filter(f, transformation_ctx="", info="", stageThreshold=0, totalThreshold=0)`

Renvoie un nouveau `DynamicFrame` contien tout `DynamicRecords` de l'entrée `DynamicFrame` qui correspondent à la fonction de prédicat `f`.

- `f` – Fonction de prédicat à appliquer au paramètre `DynamicFrame`. La fonction doit prendre un `DynamicRecord` en tant qu'argument et renvoyer la valeur True si le `DynamicRecord` répond aux critères du filtre, ou la valeur False si ce n'est pas le cas (obligatoire).

  Un `DynamicRecord` représente un enregistrement logique dans un `DynamicFrame`. Il est comparable à une ligne dans un Spark `DataFrame`, à la différence qu'il est auto-descriptif et peut être utilisé pour les données qui ne sont pas conformes à un schéma fixe.
- `transformation_ctx` – Chaîne unique utilisée pour identifier les informations sur l'état (facultatif).
- `info` – Chaîne à associer avec le signalement des erreurs pour cette transformation (facultatif).
- `stageThreshold` – Nombre d'erreurs rencontrées pendant cette transformation et qui doit entraîner l'erreur générée par le processus (facultatif) La valeur par défaut est zéro, ce qui indique que le processus ne doit pas générer d'erreur.
- `totalThreshold` – Nombre d'erreurs rencontrées jusqu'à cette transformation, incluse, et qui doit entraîner l'arrêt du processus (facultatif). La valeur par défaut est zéro, ce qui indique que le processus ne doit pas générer d'erreur.

Exemple : Utiliser un filtre pour obtenir une sélection de champs filtrée

Cet exemple utilise le`filter` Méthode de création d'un nouveau`DynamicFrame`qui inclut une sélection filtrée d'un autre `DynamicFrame` des champs.

Comme le `map` Méthode, `filter` prend une fonction comme argument qui est appliquée à chaque enregistrement de l'original`DynamicFrame`. La fonction prend un enregistrement en entrée et

renvoie une valeur booléenne. Si la valeur de retour est vraie, l'enregistrement est inclus dans le résultat`DynamicFrame`. Si c'est faux, le dossier est omis.

> **Note**
>
> Pour accéder au jeu de données utilisé dans cet exemple, voir Exemple de code : préparation des données à l'aide ResolveChoice de Lambda et ApplyMapping et suivez les instructions de la sectionÉtape 1 : analyser les données dans le compartiment Amazon S3.

```
# Example: Use filter to create a new DynamicFrame
# with a filtered selection of records

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Create DynamicFrame from Glue Data Catalog
medicare = glueContext.create_dynamic_frame.from_options(
    "s3",
    {
        "paths": [
            "s3://awsglue-datasets/examples/medicare/Medicare_Hospital_Provider.csv"
        ]
    },
    "csv",
    {"withHeader": True},
)

# Create filtered DynamicFrame with custom lambda
# to filter records by Provider State and Provider City
sac_or_mon = medicare.filter(
    f=lambda x: x["Provider State"] in ["CA", "AL"]
    and x["Provider City"] in ["SACRAMENTO", "MONTGOMERY"]
)

# Compare record counts
print("Unfiltered record count: ", medicare.count())
```

```
print("Filtered record count:  ", sac_or_mon.count())
```

Output

```
Unfiltered record count:  163065
Filtered record count:   564
```

joindre

join(paths1, paths2, frame2, transformation_ctx="", info="", stageThreshold=0, totalThreshold=0)

Effectue une jointure d'égalité avec un autre `DynamicFrame` et renvoie le `DynamicFrame` obtenu.

- `paths1` – Liste des clés dans cette trame à joindre.
- `paths2` – Liste des clés dans l'autre trame à joindre.
- `frame2` – Autre `DynamicFrame` à joindre.
- `transformation_ctx` – Chaîne unique utilisée pour identifier les informations sur l'état (facultatif).
- `info` – Chaîne à associer avec le signalement des erreurs pour cette transformation (facultatif).
- `stageThreshold` – Nombre d'erreurs rencontrées pendant cette transformation et qui doit entraîner l'erreur générée par le processus (facultatif) La valeur par défaut est zéro, ce qui indique que le processus ne doit pas générer d'erreur.
- `totalThreshold` – Nombre d'erreurs rencontrées jusqu'à cette transformation, incluse, et qui doit entraîner l'arrêt du processus (facultatif). La valeur par défaut est zéro, ce qui indique que le processus ne doit pas générer d'erreur.

Exemple : Utiliser la jointure pour combiner**DynamicFrames**

Cet exemple utilise la `join` méthode pour effectuer une jointure sur trois`DynamicFrames`. AWS Glue effectue la jointure en fonction des clés de champ que vous fournissez. Le résultat `DynamicFrame` contient les lignes des deux cadres d'origine où les clés spécifiées correspondent.

Notez que le`join`transform conserve tous les champs intacts. Cela signifie que les champs que vous spécifiez comme correspondants apparaissent dans le résultat DynamicFrame, même s'ils sont redondants et contiennent les mêmes clés. Dans cet exemple, nous utilisons`drop_fields`pour supprimer ces clés redondantes après la jointure.

> **Note**
>
> Pour accéder au jeu de données utilisé dans cet exemple, voir Exemple de code : Données de jonction et de mise en relation et suivez les instructions de la sectionÉtape 1 : analyser les données dans le compartiment Amazon S3.

```
# Example: Use join to combine data from three DynamicFrames

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Load DynamicFrames from Glue Data Catalog
persons = glueContext.create_dynamic_frame.from_catalog(
    database="legislators", table_name="persons_json"
)
memberships = glueContext.create_dynamic_frame.from_catalog(
    database="legislators", table_name="memberships_json"
)
orgs = glueContext.create_dynamic_frame.from_catalog(
    database="legislators", table_name="organizations_json"
)
print("Schema for the persons DynamicFrame:")
persons.printSchema()
print("Schema for the memberships DynamicFrame:")
memberships.printSchema()
print("Schema for the orgs DynamicFrame:")
orgs.printSchema()

# Join persons and memberships by ID
persons_memberships = persons.join(
    paths1=["id"], paths2=["person_id"], frame2=memberships
)

# Rename and drop fields from orgs
# to prevent field name collisions with persons_memberships
orgs = (
    orgs.drop_fields(["other_names", "identifiers"])
```

```
    .rename_field("id", "org_id")
    .rename_field("name", "org_name")
)

# Create final join of all three DynamicFrames
legislators_combined = orgs.join(
    paths1=["org_id"], paths2=["organization_id"], frame2=persons_memberships
).drop_fields(["person_id", "org_id"])

# Inspect the schema for the joined data
print("Schema for the new legislators_combined DynamicFrame:")
legislators_combined.printSchema()
```

Output

```
Schema for the persons DynamicFrame:
root
|-- family_name: string
|-- name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- gender: string
|-- image: string
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- lang: string
|    |    |-- note: string
|    |    |-- name: string
|-- sort_name: string
|-- images: array
|    |-- element: struct
|    |    |-- url: string
|-- given_name: string
|-- birth_date: string
|-- id: string
|-- contact_details: array
|    |-- element: struct
```

```
|    |    |-- type: string
|    |    |-- value: string
|-- death_date: string

Schema for the memberships DynamicFrame:
root
|-- area_id: string
|-- on_behalf_of_id: string
|-- organization_id: string
|-- role: string
|-- person_id: string
|-- legislative_period_id: string
|-- start_date: string
|-- end_date: string

Schema for the orgs DynamicFrame:
root
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- lang: string
|    |    |-- note: string
|    |    |-- name: string
|-- id: string
|-- classification: string
|-- name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- image: string
|-- seats: int
|-- type: string

Schema for the new legislators_combined DynamicFrame:
root
|-- role: string
|-- seats: int
|-- org_name: string
|-- links: array
|    |-- element: struct
```

```
|    |    |-- note: string
|    |    |-- url: string
|-- type: string
|-- sort_name: string
|-- area_id: string
|-- images: array
|    |-- element: struct
|    |    |-- url: string
|-- on_behalf_of_id: string
|-- other_names: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- name: string
|    |    |-- lang: string
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
|-- name: string
|-- birth_date: string
|-- organization_id: string
|-- gender: string
|-- classification: string
|-- legislative_period_id: string
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- image: string
|-- given_name: string
|-- start_date: string
|-- family_name: string
|-- id: string
|-- death_date: string
|-- end_date: string
```

map

`map(f, transformation_ctx="", info="", stageThreshold=0, totalThreshold=0)`

Renvoie un nouveau `DynamicFrame` qui résulte de l'application de la fonction de mappage spécifiée à tous les enregistrements dans le `DynamicFrame` d'origine.

- `f` – Fonction de mappage à appliquer à tous les enregistrements dans le paramètre `DynamicFrame`. La fonction doit prendre un `DynamicRecord` comme argument et renvoyer un nouveau `DynamicRecord` (obligatoire).

  Un `DynamicRecord` représente un enregistrement logique dans un `DynamicFrame`. Il est comparable à une ligne dans un `DataFrame` Apache Spark, à la différence qu'il est auto-descriptif et peut être utilisé pour les données qui ne sont pas conformes à un schéma fixe.
- `transformation_ctx` – Chaîne unique utilisée pour identifier les informations sur l'état (facultatif).
- `info` – Chaîne associée à des erreurs dans la transformation (facultatif).
- `stageThreshold` – nombre maximal d'erreurs qui peuvent avoir lieu dans la transformation avant qu'elle ne soit arrêtée.(facultatif) La valeur par défaut est zéro.
- `totalThreshold` – nombre maximal d'erreurs pouvant se produire globalement avant que le processus de traitement des erreurs ne soit arrêté.(facultatif) La valeur par défaut est zéro.

Exemple : utilisez la carte pour appliquer une fonction à chaque enregistrement dans un **`DynamicFrame`**

Cet exemple illustre comment utiliser `map` méthode pour appliquer une fonction à chaque enregistrement d'un `DynamicFrame`. Plus précisément, cet exemple applique une fonction appelée `MergeAddress` à chaque enregistrement afin de fusionner plusieurs champs d'adresse en un seul `struct` type.

> **Note**
>
> Pour accéder au jeu de données utilisé dans cet exemple, voir Exemple de code : préparation des données à l'aide ResolveChoice de Lambda et ApplyMapping et suivez les instructions de la sectionÉtape 1 : analyser les données dans le compartiment Amazon S3.

```
# Example: Use map to combine fields in all records
# of a DynamicFrame

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
```

```
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Create a DynamicFrame and view its schema
medicare = glueContext.create_dynamic_frame.from_options(
        "s3",
        {"paths": ["s3://awsglue-datasets/examples/medicare/
Medicare_Hospital_Provider.csv"]},
        "csv",
        {"withHeader": True})
print("Schema for medicare DynamicFrame:")
medicare.printSchema()

# Define a function to supply to the map transform
# that merges address fields into a single field
def MergeAddress(rec):
  rec["Address"] = {}
  rec["Address"]["Street"] = rec["Provider Street Address"]
  rec["Address"]["City"] = rec["Provider City"]
  rec["Address"]["State"] = rec["Provider State"]
  rec["Address"]["Zip.Code"] = rec["Provider Zip Code"]
  rec["Address"]["Array"] = [rec["Provider Street Address"], rec["Provider City"],
 rec["Provider State"], rec["Provider Zip Code"]]
  del rec["Provider Street Address"]
  del rec["Provider City"]
  del rec["Provider State"]
  del rec["Provider Zip Code"]
  return rec


# Use map to apply MergeAddress to every record
mapped_medicare = medicare.map(f = MergeAddress)
print("Schema for mapped_medicare DynamicFrame:")
mapped_medicare.printSchema()
```

Output

```
Schema for medicare DynamicFrame:
root
|-- DRG Definition: string
|-- Provider Id: string
|-- Provider Name: string
|-- Provider Street Address: string
```

```
|-- Provider City: string
|-- Provider State: string
|-- Provider Zip Code: string
|-- Hospital Referral Region Description: string
|-- Total Discharges: string
|-- Average Covered Charges: string
|-- Average Total Payments: string
|-- Average Medicare Payments: string

Schema for mapped_medicare DynamicFrame:
root
|-- Average Total Payments: string
|-- Average Covered Charges: string
|-- DRG Definition: string
|-- Average Medicare Payments: string
|-- Hospital Referral Region Description: string
|-- Address: struct
|    |-- Zip.Code: string
|    |-- City: string
|    |-- Array: array
|    |    |-- element: string
|    |-- State: string
|    |-- Street: string
|-- Provider Id: string
|-- Total Discharges: string
|-- Provider Name: string
```

mergeDynamicFrame

`mergeDynamicFrame(stage_dynamic_frame, primary_keys, transformation_ctx = "", options = {}, info = "", stageThreshold = 0, totalThreshold = 0)`

Fusionne cette trame `DynamicFrame` avec une trame `DynamicFrame` intermédiaire basée sur les clés primaires spécifiées pour identifier les enregistrements. Les registres en double (registres avec les mêmes clés primaires) ne sont pas dédupliqués. Si aucun enregistrement ne correspond dans la trame intermédiaire, tous les enregistrements (y compris les doublons) sont conservés dans la source. Si la trame intermédiaire contient des enregistrements correspondants, les enregistrements de la trame intermédiaire remplacent ceux de la source dans AWS Glue.

- `stage_dynamic_frame` – trame `DynamicFrame` intermédiaire à fusionner.
- `primary_keys` – liste des champs de clé primaire permettant de faire correspondre les enregistrements des trames dynamiques source et intermédiaire.

- `transformation_ctx` – chaîne unique utilisée pour récupérer les métadonnées relatives à la transformation en cours (facultatif).
- `options` – chaîne de paires nom-valeur JSON qui fournissent des informations supplémentaires pour cette transformation. Cet argument n'est actuellement pas utilisé.
- `info` – A `String`. Toute chaîne à associer à des erreurs dans cette transformation.
- `stageThreshold` – A `Long`. Nombre d'erreurs identifiées dans la transformation donnée et à corriger lors du traitement.
- `totalThreshold` – A `Long`. Nombre total d'erreurs identifiées dans la transformation donnée et qui doivent être corrigées lors du traitement.

La méthode renvoie une nouvelle image `DynamicFrame` obtenue en fusionnant `DynamicFrame` avec l'image `DynamicFrame` intermédiaire.

La trame `DynamicFrame` renvoyée contient l'enregistrement A dans les cas suivants :

- Si `A` se trouve à la fois dans la trame source et la trame intermédiaire, c'est la valeur `A` de la trame intermédiaire qui est renvoyée.
- Si `A` se trouve dans la table source et si `A.primaryKeys` ne se trouve pas dans l'image `stagingDynamicFrame`, `A` n'est pas mis à jour dans la table intermédiaire.

L'image source et l'image intermédiaire n'ont pas besoin d'avoir le même schéma.

Exemple : mergeDynamicFrame à utiliser pour fusionner deux **`DynamicFrames`** en fonction d'une clé primaire

L'exemple de code suivant montre comment utiliser la méthode `mergeDynamicFrame` pour fusionner une image `DynamicFrame` avec une image `DynamicFrame` « intermédiaire », sur la base de la clé primaire `id`.

Exemple de jeu de données

L'exemple utilise deux images `DynamicFrames` d'une collection `DynamicFrameCollection` appelée `split_rows_collection`. Vous trouverez ci-dessous la liste des clés de la collection `split_rows_collection`.

```
dict_keys(['high', 'low'])
```

Exemple de code

```
# Example: Use mergeDynamicFrame to merge DynamicFrames
# based on a set of specified primary keys

from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.transforms import SelectFromCollection

# Inspect the original DynamicFrames
frame_low = SelectFromCollection.apply(dfc=split_rows_collection, key="low")
print("Inspect the DynamicFrame that contains rows where ID < 10")
frame_low.toDF().show()

frame_high = SelectFromCollection.apply(dfc=split_rows_collection, key="high")
print("Inspect the DynamicFrame that contains rows where ID > 10")
frame_high.toDF().show()

# Merge the DynamicFrames based on the "id" primary key
merged_high_low = frame_high.mergeDynamicFrame(
    stage_dynamic_frame=frame_low, primary_keys=["id"]
)

# View the results where the ID is 1 or 20
print("Inspect the merged DynamicFrame that contains the combined rows")
merged_high_low.toDF().where("id = 1 or id= 20").orderBy("id").show()
```

Output

```
Inspect the DynamicFrame that contains rows where ID < 10
+---+-----+-----------------------+-------------------------+
| id|index|contact_details.val.type|contact_details.val.value|
+---+-----+-----------------------+-------------------------+
|  1|    0|                    fax|             202-225-3307|
|  1|    1|                  phone|             202-225-5731|
|  2|    0|                    fax|             202-225-3307|
|  2|    1|                  phone|             202-225-5731|
|  3|    0|                    fax|             202-225-3307|
|  3|    1|                  phone|             202-225-5731|
|  4|    0|                    fax|             202-225-3307|
|  4|    1|                  phone|             202-225-5731|
|  5|    0|                    fax|             202-225-3307|
|  5|    1|                  phone|             202-225-5731|
|  6|    0|                    fax|             202-225-3307|
|  6|    1|                  phone|             202-225-5731|
```

```
|  7|    0|                    fax|            202-225-3307|
|  7|    1|                  phone|            202-225-5731|
|  8|    0|                    fax|            202-225-3307|
|  8|    1|                  phone|            202-225-5731|
|  9|    0|                    fax|            202-225-3307|
|  9|    1|                  phone|            202-225-5731|
| 10|    0|                    fax|            202-225-6328|
| 10|    1|                  phone|            202-225-4576|
+---+-----+----------------------+------------------------+
only showing top 20 rows

Inspect the DynamicFrame that contains rows where ID > 10
+---+-----+----------------------+------------------------+
| id|index|contact_details.val.type|contact_details.val.value|
+---+-----+----------------------+------------------------+
| 11|    0|                    fax|            202-225-6328|
| 11|    1|                  phone|            202-225-4576|
| 11|    2|                twitter|          RepTrentFranks|
| 12|    0|                    fax|            202-225-6328|
| 12|    1|                  phone|            202-225-4576|
| 12|    2|                twitter|          RepTrentFranks|
| 13|    0|                    fax|            202-225-6328|
| 13|    1|                  phone|            202-225-4576|
| 13|    2|                twitter|          RepTrentFranks|
| 14|    0|                    fax|            202-225-6328|
| 14|    1|                  phone|            202-225-4576|
| 14|    2|                twitter|          RepTrentFranks|
| 15|    0|                    fax|            202-225-6328|
| 15|    1|                  phone|            202-225-4576|
| 15|    2|                twitter|          RepTrentFranks|
| 16|    0|                    fax|            202-225-6328|
| 16|    1|                  phone|            202-225-4576|
| 16|    2|                twitter|          RepTrentFranks|
| 17|    0|                    fax|            202-225-6328|
| 17|    1|                  phone|            202-225-4576|
+---+-----+----------------------+------------------------+
only showing top 20 rows

Inspect the merged DynamicFrame that contains the combined rows
+---+-----+----------------------+------------------------+
| id|index|contact_details.val.type|contact_details.val.value|
+---+-----+----------------------+------------------------+
|  1|    0|                    fax|            202-225-3307|
|  1|    1|                  phone|            202-225-5731|
```

```
| 20|    0|                     fax|             202-225-5604|
| 20|    1|                   phone|             202-225-6536|
| 20|    2|                 twitter|                USRepLong|
+---+-----+------------------------+------------------------+
```

relationalize

relationalize(root_table_name, staging_path, options, transformation_ctx="", info="", stageThreshold=0, totalThreshold=0)

Convertit une image `DynamicFrame` sous une forme adaptée à une base de données relationnelle. La mise en relation d'une image `DynamicFrame` est particulièrement utile lorsque vous souhaitez déplacer des données d'un environnement NoSQL tel que DynamoDB vers une base de données relationnelle telle que MySQL.

La transformation génère une liste des images en désimbriquant les colonnes imbriquées et en faisant pivoter les colonnes de tableau. Vous pouvez joindre les colonnes du tableau ayant été pivotées à l'aide de la clé générée au cours de la phase de désimbrication.

- `root_table_name` – Nom de la table racine.
- `staging_path` – Chemin d'accès auquel la méthode peut stocker des partitions des tables dynamiques au format CSV format (facultatif). Les tables dynamiques sont lues à partir de ce chemin.
- `options` – Dictionnaire des paramètres facultatifs.
- `transformation_ctx` – Chaîne unique utilisée pour identifier les informations sur l'état (facultatif).
- `info` – Chaîne à associer avec le signalement des erreurs pour cette transformation (facultatif).
- `stageThreshold` – Nombre d'erreurs rencontrées pendant cette transformation et qui doit entraîner l'erreur générée par le processus (facultatif) La valeur par défaut est zéro, ce qui indique que le processus ne doit pas générer d'erreur.
- `totalThreshold` – Nombre d'erreurs rencontrées jusqu'à cette transformation, incluse, et qui doit entraîner l'arrêt du processus (facultatif). La valeur par défaut est zéro, ce qui indique que le processus ne doit pas générer d'erreur.

Exemple : utiliser la méthode relationalize pour aplatir un schéma imbriqué dans une image **DynamicFrame**

Cet exemple de code utilise la méthode `relationalize` pour aplatir un schéma imbriqué sous une forme adaptée à une base de données relationnelle.

Exemple de jeu de données

L'exemple utilise une image `DynamicFrame` appelée `legislators_combined` avec le schéma suivant. `legislators_combined` possède plusieurs champs imbriqués tels que `links`, `images` et `contact_details`, qui seront aplatis par la transformation `relationalize`.

```
root
|-- role: string
|-- seats: int
|-- org_name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- type: string
|-- sort_name: string
|-- area_id: string
|-- images: array
|    |-- element: struct
|    |    |-- url: string
|-- on_behalf_of_id: string
|-- other_names: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- name: string
|    |    |-- lang: string
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
|-- name: string
|-- birth_date: string
|-- organization_id: string
|-- gender: string
|-- classification: string
|-- legislative_period_id: string
|-- identifiers: array
```

```
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- image: string
|-- given_name: string
|-- start_date: string
|-- family_name: string
|-- id: string
|-- death_date: string
|-- end_date: string
```

Exemple de code

```
# Example: Use relationalize to flatten
# a nested schema into a format that fits
# into a relational database.
# Replace DOC-EXAMPLE-S3-BUCKET/tmpDir with your own location.

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Apply relationalize and inspect new tables
legislators_relationalized = legislators_combined.relationalize(
    "l_root", "s3://DOC-EXAMPLE-BUCKET/tmpDir"
)
legislators_relationalized.keys()

# Compare the schema of the contact_details
# nested field to the new relationalized table that
# represents it
legislators_combined.select_fields("contact_details").printSchema()
legislators_relationalized.select("l_root_contact_details").toDF().where(
    "id = 10 or id = 75"
).orderBy(["id", "index"]).show()
```

Output

La sortie suivante vous permet de comparer le schéma du champ imbriqué appelé `contact_details` à la table créée par la transformation `relationalize`. Notez que les

enregistrements de la table renvoient vers la table principale à l'aide d'une clé étrangère appelée `id` et d'une colonne `index` représentant les positions du tableau.

```
dict_keys(['l_root', 'l_root_images', 'l_root_links', 'l_root_other_names',
 'l_root_contact_details', 'l_root_identifiers'])

root
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string

+---+-----+-----------------------+-----------------------+
| id|index|contact_details.val.type|contact_details.val.value|
+---+-----+-----------------------+-----------------------+
| 10|    0|                    fax|           202-225-4160|
| 10|    1|                  phone|           202-225-3436|
| 75|    0|                    fax|           202-225-6791|
| 75|    1|                  phone|           202-225-2861|
| 75|    2|                twitter|             RepSamFarr|
+---+-----+-----------------------+-----------------------+
```

rename_field

`rename_field(oldName, newName, transformation_ctx="", info="", stageThreshold=0, totalThreshold=0)`

Renomme un champ dans ce `DynamicFrame` et renvoie un nouveau `DynamicFrame` avec le champ renommé.

- `oldName` – Chemin d'accès complet au nœud que vous souhaitez renommer.

  Si l'ancien nom contient des points, `RenameField` ne fonctionne pas à moins que vous ne le délimitiez avec des apostrophes inverses (`). Par exemple, pour remplacer `this.old.name` par `thisNewName`, vous devez appeler rename_field comme suit.

  ```
  newDyF = oldDyF.rename_field("`this.old.name`", "thisNewName")
  ```

- `newName` – Nouveau nom, sous forme de chemin d'accès complet.
- `transformation_ctx` – Chaîne unique utilisée pour identifier les informations sur l'état (facultatif).

- `info` – Chaîne à associer avec le signalement des erreurs pour cette transformation (facultatif).
- `stageThreshold` – Nombre d'erreurs rencontrées pendant cette transformation et qui doit entraîner l'erreur générée par le processus (facultatif) La valeur par défaut est zéro, ce qui indique que le processus ne doit pas générer d'erreur.
- `totalThreshold` – Nombre d'erreurs rencontrées jusqu'à cette transformation, incluse, et qui doit entraîner l'arrêt du processus (facultatif). La valeur par défaut est zéro, ce qui indique que le processus ne doit pas générer d'erreur.

Exemple : utiliser la méthode rename_field pour renommer des champs dans une image **`DynamicFrame`**

Cet exemple de code utilise la méthode `rename_field` pour renommer les champs dans une image `DynamicFrame`. Notez que l'exemple utilise le chaînage de méthode pour renommer plusieurs champs en même temps.

> **Note**
>
> Pour accéder au jeu de données utilisé dans cet exemple, voir Exemple de code : Données de jonction et de mise en relation et suivez les instructions de la sectionÉtape 1 : analyser les données dans le compartiment Amazon S3.

Exemple de code

```
# Example: Use rename_field to rename fields
# in a DynamicFrame

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Inspect the original orgs schema
orgs = glueContext.create_dynamic_frame.from_catalog(
    database="legislators", table_name="organizations_json"
)
print("Original orgs schema: ")
```

```
orgs.printSchema()

# Rename fields and view the new schema
orgs = orgs.rename_field("id", "org_id").rename_field("name", "org_name")
print("New orgs schema with renamed fields: ")
orgs.printSchema()
```

Output

```
Original orgs schema:
root
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- lang: string
|    |    |-- note: string
|    |    |-- name: string
|-- id: string
|-- classification: string
|-- name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- image: string
|-- seats: int
|-- type: string

New orgs schema with renamed fields:
root
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- lang: string
|    |    |-- note: string
|    |    |-- name: string
|-- classification: string
```

```
|-- org_id: string
|-- org_name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- image: string
|-- seats: int
|-- type: string
```

resolveChoice

`resolveChoice(specs = None, choice = "" , database = None , table_name = None , transformation_ctx="", info="", stageThreshold=0, totalThreshold=0, catalog_id = None)`

Résout un type de choix au sein de ce `DynamicFrame` et renvoie le nouveau `DynamicFrame`.

- `specs` – Liste d'ambiguïtés spécifiques à résoudre, apparaissant sous forme de tuple: `(field_path, action)`.

  Il existe deux façons d'utiliser `resolveChoice`. La première consiste à utiliser l'argument `specs` pour spécifier une séquence de champs spécifiques et la façon de les résoudre. L'autre mode pour `resolveChoice` consiste à utiliser l'argument `choice` afin de spécifier une seule résolution pour tous les `ChoiceTypes`.

  Les valeurs pour `specs` sont spécifiées en tant que tuples composés de paires `(field_path, action)`. La valeur `field_path` identifie un élément ambigu spécifique, et la valeur `action` identifie la résolution correspondante. Les actions possibles sont les suivantes :

  - `cast:`*`type`* – tente de convertir toutes les valeurs vers le type spécifié. Par exemple : `cast:int`.
  - `make_cols` – Convertit chaque type distinct en une colonne avec le nom *`columnName_type`*. Résout une ambiguïté potentielle en aplatissant les données. Par exemple, si `columnA` peut être un `int` ou un `string`, la résolution consisterait à produire deux colonnes nommées `columnA_int` et `columnA_string` dans le `DynamicFrame` obtenu.
  - `make_struct` – résout une ambigüité potentielle en utilisant un `struct` pour représenter les données. Par exemple, si des données d'une colonne peuvent être de type `int` ou `string`,

l'action `make_struct` produit une colonne de structures dans l'image `DynamicFrame` obtenue. Chaque structure contient à la fois des données de type `int` et des données de type `string`.

- `project:`*`type`* – résout une ambiguïté potentielle en projetant toutes les données sur l'un des types de données possibles. Par exemple, si des données d'une colonne peuvent être un `int` ou un `string`, une action `project:string` produit une colonne dans le `DynamicFrame` obtenu, où toutes les valeurs `int` ont été converties en chaînes.

Si le `field_path` identifie un tableau, placez des crochets vides après le nom du tableau pour éviter toute ambiguïté. Par exemple, supposons que vous travailliez avec les données structurées comme suit :

```
"myList": [
  { "price": 100.00 },
  { "price": "$100.00" }
]
```

Vous pouvez sélectionner la version numérique plutôt que la version chaîne du prix en définissant `field_path` sur `"myList[].price"` et `action` sur `"cast:double"`.

> **Note**
>
> Vous ne pouvez utiliser qu'un seul des paramètres `specs` et `choice`. Si le paramètre `specs` n'est pas `None`, alors le paramètre `choice` doit être une chaîne vide. Inversement, si le paramètre `choice` n'est pas une chaîne vide, alors le paramètre `specs` doit être `None`.

- `choice` – spécifie une résolution unique pour tous les `ChoiceTypes`. Vous pouvez utiliser cela lorsque la liste complète des `ChoiceTypes` est inconnue avant l'exécution. En plus des actions répertoriées précédemment pour `specs`, cet argument prend également en charge l'action suivante :
  - `match_catalog` – tente de convertir chaque `ChoiceType` dans le type correspondant de table Data Catalog spécifiée.
- `database` – base de données Data Catalog à utiliser avec l'action `match_catalog`.
- `table_name` – table Data Catalog à utiliser avec l'action `match_catalog`.
- `transformation_ctx` – Chaîne unique utilisée pour identifier les informations sur l'état (facultatif).

- `info` – Chaîne à associer avec le signalement des erreurs pour cette transformation (facultatif).
- `stageThreshold` – Nombre d'erreurs rencontrées pendant cette transformation et qui doit entraîner l'erreur générée par le processus (facultatif) La valeur par défaut est zéro, ce qui indique que le processus ne doit pas générer d'erreur.
- `totalThreshold` – Nombre d'erreurs rencontrées jusqu'à cette transformation incluse, et qui doit entraîner l'arrêt du processus (facultatif). La valeur par défaut est zéro, ce qui indique que le processus ne doit pas être arrêté dans ce cas.
- `catalog_id` – ID du catalogue Data Catalog auquel vous accédez (ID du compte Data Catalog). Lorsque cette option est définie sur `None` (valeur par défaut), l'ID de catalogue du compte appelant est utilisé.

Exemple : utiliser la méthode resolveChoice pour gérer une colonne contenant plusieurs types

Cet exemple de code utilise la méthode `resolveChoice` pour spécifier comment gérer une colonne `DynamicFrame` contenant des valeurs de plusieurs types. L'exemple montre deux méthodes courantes pour gérer une colonne de types différents :

- Convertir la colonne en un seul type de données.
- Conserver tous les types dans des colonnes distinctes.

Exemple de jeu de données

> **Note**
>
> Pour accéder au jeu de données utilisé dans cet exemple, voir Exemple de code : préparation des données à l'aide ResolveChoice de Lambda et ApplyMapping et suivez les instructions de la sectionÉtape 1 : analyser les données dans le compartiment Amazon S3.

L'exemple utilise une image `DynamicFrame` appelée `medicare` avec le schéma suivant :

```
root
|-- drg definition: string
|-- provider id: choice
|    |-- long
|    |-- string
|-- provider name: string
```

```
|-- provider street address: string
|-- provider city: string
|-- provider state: string
|-- provider zip code: long
|-- hospital referral region description: string
|-- total discharges: long
|-- average covered charges: string
|-- average total payments: string
|-- average medicare payments: string
```

Exemple de code

```
# Example: Use resolveChoice to handle
# a column that contains multiple types

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Load the input data and inspect the "provider id" column
medicare = glueContext.create_dynamic_frame.from_catalog(
    database="payments", table_name="medicare_hospital_provider_csv"
)
print("Inspect the provider id column:")
medicare.toDF().select("provider id").show()

# Cast provider id to type long
medicare_resolved_long = medicare.resolveChoice(specs=[("provider id", "cast:long")])
print("Schema after casting provider id to type long:")
medicare_resolved_long.printSchema()
medicare_resolved_long.toDF().select("provider id").show()

# Create separate columns
# for each provider id type
medicare_resolved_cols = medicare.resolveChoice(choice="make_cols")
print("Schema after creating separate columns for each type:")
medicare_resolved_cols.printSchema()
medicare_resolved_cols.toDF().select("provider id_long", "provider id_string").show()
```

Output

```
Inspect the 'provider id' column:
+-----------+
|provider id|
+-----------+
|   [10001,]|
|   [10005,]|
|   [10006,]|
|   [10011,]|
|   [10016,]|
|   [10023,]|
|   [10029,]|
|   [10033,]|
|   [10039,]|
|   [10040,]|
|   [10046,]|
|   [10055,]|
|   [10056,]|
|   [10078,]|
|   [10083,]|
|   [10085,]|
|   [10090,]|
|   [10092,]|
|   [10100,]|
|   [10103,]|
+-----------+
only showing top 20 rows

Schema after casting 'provider id' to type long:
root
|-- drg definition: string
|-- provider id: long
|-- provider name: string
|-- provider street address: string
|-- provider city: string
|-- provider state: string
|-- provider zip code: long
|-- hospital referral region description: string
|-- total discharges: long
|-- average covered charges: string
|-- average total payments: string
|-- average medicare payments: string
```

```
+-----------+
|provider id|
+-----------+
|      10001|
|      10005|
|      10006|
|      10011|
|      10016|
|      10023|
|      10029|
|      10033|
|      10039|
|      10040|
|      10046|
|      10055|
|      10056|
|      10078|
|      10083|
|      10085|
|      10090|
|      10092|
|      10100|
|      10103|
+-----------+
only showing top 20 rows

Schema after creating separate columns for each type:
root
|-- drg definition: string
|-- provider id_string: string
|-- provider id_long: long
|-- provider name: string
|-- provider street address: string
|-- provider city: string
|-- provider state: string
|-- provider zip code: long
|-- hospital referral region description: string
|-- total discharges: long
|-- average covered charges: string
|-- average total payments: string
|-- average medicare payments: string

+----------------+------------------+
|provider id_long|provider id_string|
```

```
+----------------+------------------+
|           10001|              null|
|           10005|              null|
|           10006|              null|
|           10011|              null|
|           10016|              null|
|           10023|              null|
|           10029|              null|
|           10033|              null|
|           10039|              null|
|           10040|              null|
|           10046|              null|
|           10055|              null|
|           10056|              null|
|           10078|              null|
|           10083|              null|
|           10085|              null|
|           10090|              null|
|           10092|              null|
|           10100|              null|
|           10103|              null|
+----------------+------------------+
only showing top 20 rows
```

select_fields

`select_fields(paths, transformation_ctx="", info="", stageThreshold=0, totalThreshold=0)`

Renvoie un nouveau `DynamicFrame` contenant les champs sélectionnés.

- `paths` - Une liste de chaînes. Chaque chaîne est un chemin d'accès au nœud de niveau supérieur que vous souhaitez sélectionner.
- `transformation_ctx` – Chaîne unique utilisée pour identifier les informations sur l'état (facultatif).
- `info` – Chaîne à associer avec le signalement des erreurs pour cette transformation (facultatif).
- `stageThreshold` – Nombre d'erreurs rencontrées pendant cette transformation et qui doit entraîner l'erreur générée par le processus (facultatif) La valeur par défaut est zéro, ce qui indique que le processus ne doit pas générer d'erreur.

- `totalThreshold` – Nombre d'erreurs rencontrées jusqu'à cette transformation, incluse, et qui doit entraîner l'arrêt du processus (facultatif). La valeur par défaut est zéro, ce qui indique que le processus ne doit pas générer d'erreur.

Exemple : utilisez select_fields pour créer un nouveau **`DynamicFrame`** avec les champs sélectionnés

L'exemple de code suivant illustre comment utiliser `select_fields` Méthode de création d'un nouveau `DynamicFrame` avec une liste de champs sélectionnés parmi un `DynamicFrame`.

> **Note**
>
> Pour accéder au jeu de données utilisé dans cet exemple, voir Exemple de code : Données de jonction et de mise en relation et suivez les instructions de la sectionÉtape 1 : analyser les données dans le compartiment Amazon S3.

```
# Example: Use select_fields to select specific fields from a DynamicFrame

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Create a DynamicFrame and view its schema
persons = glueContext.create_dynamic_frame.from_catalog(
    database="legislators", table_name="persons_json"
)
print("Schema for the persons DynamicFrame:")
persons.printSchema()

# Create a new DynamicFrame with chosen fields
names = persons.select_fields(paths=["family_name", "given_name"])
print("Schema for the names DynamicFrame, created with select_fields:")
names.printSchema()
names.toDF().show()
```

Output

```
Schema for the persons DynamicFrame:
root
|-- family_name: string
|-- name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- gender: string
|-- image: string
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- lang: string
|    |    |-- note: string
|    |    |-- name: string
|-- sort_name: string
|-- images: array
|    |-- element: struct
|    |    |-- url: string
|-- given_name: string
|-- birth_date: string
|-- id: string
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
|-- death_date: string

Schema for the names DynamicFrame:
root
|-- family_name: string
|-- given_name: string

+-----------+----------+
|family_name|given_name|
+-----------+----------+
|    Collins|   Michael|
|   Huizenga|      Bill|
```

```
|    Clawson|    Curtis|
|    Solomon|    Gerald|
|     Rigell|    Edward|
|      Crapo|   Michael|
|      Hutto|      Earl|
|      Ertel|     Allen|
|     Minish|    Joseph|
|    Andrews|    Robert|
|     Walden|      Greg|
|      Kazen|   Abraham|
|     Turner|   Michael|
|      Kolbe|     James|
|  Lowenthal|      Alan|
|    Capuano|   Michael|
|   Schrader|      Kurt|
|     Nadler|   Jerrold|
|     Graves|       Tom|
|   McMillan|      John|
+-----------+----------+
only showing top 20 rows
```

simplify_ddb_json

`simplify_ddb_json(): DynamicFrame`

Simplifie les colonnes imbriquées dans un `DynamicFrame` qui se trouvent spécifiquement dans la structure JSON DynamoDB, et renvoie une nouvelle version simplifiée du `DynamicFrame`. S'il existe plusieurs types ou un type Carte dans un type Liste, les éléments de la liste ne seront pas simplifiés. Notez qu'il s'agit d'un type spécifique de transformation qui se comporte différemment de la transformation `unnest` normale et nécessite que les données soient déjà dans la structure JSON DynamoDB. Pour plus d'informations, consultez JSON DynamoDB.

Par exemple, le schéma d'une lecture d'exportation avec la structure JSON DynamoDB pourrait ressembler à ce qui suit :

```
root
|-- Item: struct
|    |-- parentMap: struct
|    |    |-- M: struct
|    |    |    |-- childMap: struct
|    |    |    |    |-- M: struct
|    |    |    |    |    |-- appName: struct
```

```
|    |    |    |    |    |    |-- S: string
|    |    |    |    |    |-- packageName: struct
|    |    |    |    |    |    |-- S: string
|    |    |    |    |    |-- updatedAt: struct
|    |    |    |    |    |    |-- N: string
|    |-- strings: struct
|    |    |-- SS: array
|    |    |    |-- element: string
|    |-- numbers: struct
|    |    |-- NS: array
|    |    |    |-- element: string
|    |-- binaries: struct
|    |    |-- BS: array
|    |    |    |-- element: string
|    |-- isDDBJson: struct
|    |    |-- BOOL: boolean
|    |-- nullValue: struct
|    |    |-- NULL: boolean
```

La transformation `simplify_ddb_json()` convertirait ceci en :

```
root
|-- parentMap: struct
|    |-- childMap: struct
|    |    |-- appName: string
|    |    |-- packageName: string
|    |    |-- updatedAt: string
|-- strings: array
|    |-- element: string
|-- numbers: array
|    |-- element: string
|-- binaries: array
|    |-- element: string
|-- isDDBJson: boolean
|-- nullValue: null
```

Exemple : utilisez simplify_ddb_json pour invoquer une simplification JSON DynamoDB.

Cet exemple de code utilise la `simplify_ddb_json` méthode permettant d'utiliser le connecteur d'exportation AWS Glue DynamoDB, d'invoquer un code JSON Simplify DynamoDB et d'imprimer le nombre de partitions.

Exemple de code

```
from pyspark.context import SparkContext
from awsglue.context import GlueContext

sc = SparkContext()
glueContext = GlueContext(sc)

dynamicFrame = glueContext.create_dynamic_frame.from_options(
    connection_type = "dynamodb",
    connection_options = {
        'dynamodb.export': 'ddb',
        'dynamodb.tableArn': '<table arn>',
        'dynamodb.s3.bucket': '<bucket name>',
        'dynamodb.s3.prefix': '<bucket prefix>',
        'dynamodb.s3.bucketOwner': '<account_id of bucket>'
    }
)
simplified = dynamicFrame.simplify_ddb_json()
print(simplified.getNumPartitions())
```

spigot

`spigot(path, options={})`

Écrit des exemples d'enregistrement sur une destination spécifiée pour vous aider à vérifier les transformations effectuées par votre tâche.

- `path` – Chemin d'accès de la destination sur laquelle écrire (obligatoire).
- `options` – Paires clé-valeur spécifiant des options (facultatif). L'option `"topk"` indique que les premiers enregistrements k doivent être écrits. L'option `"prob"` indique la probabilité (sous forme de décimale) de choisir un enregistrement donné. Vous pouvez l'utiliser pour sélectionner les enregistrements à écrire.
- `transformation_ctx` – Chaîne unique utilisée pour identifier les informations sur l'état (facultatif).

Exemple : utiliser la méthode spigot pour écrire des exemples de champs d'une image **`DynamicFrame`** vers Amazon S3

Cet exemple de code utilise la méthode `spigot` pour écrire des exemples d'enregistrements dans un compartiment Amazon S3 après avoir appliqué la transformation `select_fields`.

Exemple de jeu de données

> **Note**
>
> Pour accéder au jeu de données utilisé dans cet exemple, voir Exemple de code : Données de jonction et de mise en relation et suivez les instructions de la sectionÉtape 1 : analyser les données dans le compartiment Amazon S3.

L'exemple utilise une image `DynamicFrame` appelée `persons` avec le schéma suivant :

```
root
|-- family_name: string
|-- name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- gender: string
|-- image: string
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- lang: string
|    |    |-- note: string
|    |    |-- name: string
|-- sort_name: string
|-- images: array
|    |-- element: struct
|    |    |-- url: string
|-- given_name: string
|-- birth_date: string
|-- id: string
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
|-- death_date: string
```

Exemple de code

```
# Example: Use spigot to write sample records
# to a destination during a transformation
# from pyspark.context import SparkContext.
# Replace DOC-EXAMPLE-BUCKET with your own location.

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Load table data into a DynamicFrame
persons = glueContext.create_dynamic_frame.from_catalog(
    database="legislators", table_name="persons_json"
)

# Perform the select_fields on the DynamicFrame
persons = persons.select_fields(paths=["family_name", "given_name", "birth_date"])

# Use spigot to write a sample of the transformed data
# (the first 10 records)
spigot_output = persons.spigot(
    path="s3://DOC-EXAMPLE-BUCKET", options={"topk": 10}
)
# Example: Use spigot to write sample records
# to a destination during a transformation
# from pyspark.context import SparkContext.
# Replace DOC-EXAMPLE-BUCKET with your own location.

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Load table data into a DynamicFrame
persons = glueContext.create_dynamic_frame.from_catalog(
    database="legislators", table_name="persons_json"
)
```

```
# Perform the select_fields on the DynamicFrame
persons = persons.select_fields(paths=["family_name", "given_name", "birth_date"])

# Use spigot to write a sample of the transformed data
# (the first 10 records)
spigot_output = persons.spigot(
    path="s3://DOC-EXAMPLE-BUCKET", options={"topk": 10}
)
```

Output

Voici un exemple des données que `spigot` écrit dans Amazon S3. Étant donné que l'exemple de code spécifie `options={"topk": 10}`, les exemples de données contiennent les 10 premiers enregistrements.

```
{"family_name":"Collins","given_name":"Michael","birth_date":"1944-10-15"}
{"family_name":"Huizenga","given_name":"Bill","birth_date":"1969-01-31"}
{"family_name":"Clawson","given_name":"Curtis","birth_date":"1959-09-28"}
{"family_name":"Solomon","given_name":"Gerald","birth_date":"1930-08-14"}
{"family_name":"Rigell","given_name":"Edward","birth_date":"1960-05-28"}
{"family_name":"Crapo","given_name":"Michael","birth_date":"1951-05-20"}
{"family_name":"Hutto","given_name":"Earl","birth_date":"1926-05-12"}
{"family_name":"Ertel","given_name":"Allen","birth_date":"1937-11-07"}
{"family_name":"Minish","given_name":"Joseph","birth_date":"1916-09-01"}
{"family_name":"Andrews","given_name":"Robert","birth_date":"1957-08-04"}
```

split_fields

`split_fields(paths, name1, name2, transformation_ctx="", info="", stageThreshold=0, totalThreshold=0)`

Renvoie une nouvelle collection `DynamicFrameCollection` qui contient deux images `DynamicFrames`. La première image `DynamicFrame` contient tous les nœuds qui ont été fractionnés, et la seconde contient les nœuds qui restent.

- `paths` – Liste de chaînes, chacune étant un chemin d'accès complet à un nœud que vous souhaitez fractionner en un nouveau `DynamicFrame`.
- `name1` – Chaîne de nom pour le `DynamicFrame` fractionné.
- `name2` – Chaîne de nom pour le `DynamicFrame` qui reste après le fractionnement des nœuds spécifiés.

- `transformation_ctx` – Chaîne unique utilisée pour identifier les informations sur l'état (facultatif).
- `info` – Chaîne à associer avec le signalement des erreurs pour cette transformation (facultatif).
- `stageThreshold` – Nombre d'erreurs rencontrées pendant cette transformation et qui doit entraîner l'erreur générée par le processus (facultatif) La valeur par défaut est zéro, ce qui indique que le processus ne doit pas générer d'erreur.
- `totalThreshold` – Nombre d'erreurs rencontrées jusqu'à cette transformation, incluse, et qui doit entraîner l'arrêt du processus (facultatif). La valeur par défaut est zéro, ce qui indique que le processus ne doit pas générer d'erreur.

Exemple : utiliser la méthode split_fields pour fractionner les champs sélectionnés en une image **`DynamicFrame`** distincte

Cet exemple de code utilise la méthode `split_fields` pour fractionner une liste de champs spécifiés en une image `DynamicFrame` distincte.

Exemple de jeu de données

L'exemple utilise une image `DynamicFrame` appelée `l_root_contact_details` qui provient d'une collection nommée `legislators_relationalized`.

`l_root_contact_details` contient le schéma et les entrées suivantes.

```
root
|-- id: long
|-- index: int
|-- contact_details.val.type: string
|-- contact_details.val.value: string

+---+-----+-----------------------+-----------------------+
| id|index|contact_details.val.type|contact_details.val.value|
+---+-----+-----------------------+-----------------------+
|  1|    0|                  phone|             202-225-5265|
|  1|    1|                twitter|              kathyhochul|
|  2|    0|                  phone|             202-225-3252|
|  2|    1|                twitter|            repjackyrosen|
|  3|    0|                    fax|             202-225-1314|
|  3|    1|                  phone|             202-225-3772|
...
```

Exemple de code

```
# Example: Use split_fields to split selected
# fields into a separate DynamicFrame

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Load the input DynamicFrame and inspect its schema
frame_to_split = legislators_relationalized.select("l_root_contact_details")
print("Inspect the input DynamicFrame schema:")
frame_to_split.printSchema()

# Split id and index fields into a separate DynamicFrame
split_fields_collection = frame_to_split.split_fields(["id", "index"], "left", "right")

# Inspect the resulting DynamicFrames
print("Inspect the schemas of the DynamicFrames created with split_fields:")
split_fields_collection.select("left").printSchema()
split_fields_collection.select("right").printSchema()
```

Output

```
Inspect the input DynamicFrame's schema:
root
|-- id: long
|-- index: int
|-- contact_details.val.type: string
|-- contact_details.val.value: string

Inspect the schemas of the DynamicFrames created with split_fields:
root
|-- id: long
|-- index: int

root
|-- contact_details.val.type: string
|-- contact_details.val.value: string
```

split_rows

`split_rows(comparison_dict, name1, name2, transformation_ctx="", info="", stageThreshold=0, totalThreshold=0)`

Scinde une ou plusieurs lignes d'un objet `DynamicFrame` en un nouveau `DynamicFrame`.

La méthode renvoie une nouvelle collection `DynamicFrameCollection` qui contient deux images `DynamicFrames`. La première image `DynamicFrame` contient toutes les lignes qui ont été fractionnées, et la seconde contient les lignes qui restent.

- `comparison_dict` – Dictionnaire dans lequel la clé est un chemin d'accès à une colonne et la valeur est un autre dictionnaire permettant de mapper les comparateurs aux valeurs auxquelles les valeurs de la colonne sont comparées. Par exemple, `{"age": {">": 10, "<": 20}}` fractionne toutes les lignes dont la valeur de la colonne d'âge est supérieure à 10 et inférieure à 20.
- `name1` – Chaîne de nom pour le `DynamicFrame` fractionné.
- `name2` – Chaîne de nom pour le `DynamicFrame` qui reste après le fractionnement des nœuds spécifiés.
- `transformation_ctx` – Chaîne unique utilisée pour identifier les informations sur l'état (facultatif).
- `info` – Chaîne à associer avec le signalement des erreurs pour cette transformation (facultatif).
- `stageThreshold` – Nombre d'erreurs rencontrées pendant cette transformation et qui doit entraîner l'erreur générée par le processus (facultatif) La valeur par défaut est zéro, ce qui indique que le processus ne doit pas générer d'erreur.
- `totalThreshold` – Nombre d'erreurs rencontrées jusqu'à cette transformation, incluse, et qui doit entraîner l'arrêt du processus (facultatif). La valeur par défaut est zéro, ce qui indique que le processus ne doit pas générer d'erreur.

Exemple : utiliser la méthode split_rows pour fractionner des lignes en une image **`DynamicFrame`**

Cet exemple de code utilise la méthode `split_rows` pour fractionner des lignes en une image `DynamicFrame` en se basant sur la valeur du champ `id`.

Exemple de jeu de données

L'exemple utilise une image `DynamicFrame` appelée `l_root_contact_details` qui est sélectionnée à partir d'une collection nommée `legislators_relationalized`.

`l_root_contact_details` contient le schéma et les entrées suivantes.

```
root
|-- id: long
|-- index: int
|-- contact_details.val.type: string
|-- contact_details.val.value: string

+---+-----+-----------------------+-----------------------+
| id|index|contact_details.val.type|contact_details.val.value|
+---+-----+-----------------------+-----------------------+
|  1|    0|                  phone|             202-225-5265|
|  1|    1|                twitter|              kathyhochul|
|  2|    0|                  phone|             202-225-3252|
|  2|    1|                twitter|            repjackyrosen|
|  3|    0|                    fax|             202-225-1314|
|  3|    1|                  phone|             202-225-3772|
|  3|    2|                twitter|          MikeRossUpdates|
|  4|    0|                    fax|             202-225-1314|
|  4|    1|                  phone|             202-225-3772|
|  4|    2|                twitter|          MikeRossUpdates|
|  5|    0|                    fax|             202-225-1314|
|  5|    1|                  phone|             202-225-3772|
|  5|    2|                twitter|          MikeRossUpdates|
|  6|    0|                    fax|             202-225-1314|
|  6|    1|                  phone|             202-225-3772|
|  6|    2|                twitter|          MikeRossUpdates|
|  7|    0|                    fax|             202-225-1314|
|  7|    1|                  phone|             202-225-3772|
|  7|    2|                twitter|          MikeRossUpdates|
|  8|    0|                    fax|             202-225-1314|
+---+-----+-----------------------+-----------------------+
```

Exemple de code

```
# Example: Use split_rows to split up
# rows in a DynamicFrame based on value

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
```

```
glueContext = GlueContext(sc)

# Retrieve the DynamicFrame to split
frame_to_split = legislators_relationalized.select("l_root_contact_details")

# Split up rows by ID
split_rows_collection = frame_to_split.split_rows({"id": {">": 10}}, "high", "low")

# Inspect the resulting DynamicFrames
print("Inspect the DynamicFrame that contains IDs < 10")
split_rows_collection.select("low").toDF().show()
print("Inspect the DynamicFrame that contains IDs > 10")
split_rows_collection.select("high").toDF().show()
```

Output

```
Inspect the DynamicFrame that contains IDs < 10
+---+-----+-----------------------+-----------------------+
| id|index|contact_details.val.type|contact_details.val.value|
+---+-----+-----------------------+-----------------------+
|  1|    0|                  phone|             202-225-5265|
|  1|    1|                twitter|              kathyhochul|
|  2|    0|                  phone|             202-225-3252|
|  2|    1|                twitter|            repjackyrosen|
|  3|    0|                    fax|             202-225-1314|
|  3|    1|                  phone|             202-225-3772|
|  3|    2|                twitter|          MikeRossUpdates|
|  4|    0|                    fax|             202-225-1314|
|  4|    1|                  phone|             202-225-3772|
|  4|    2|                twitter|          MikeRossUpdates|
|  5|    0|                    fax|             202-225-1314|
|  5|    1|                  phone|             202-225-3772|
|  5|    2|                twitter|          MikeRossUpdates|
|  6|    0|                    fax|             202-225-1314|
|  6|    1|                  phone|             202-225-3772|
|  6|    2|                twitter|          MikeRossUpdates|
|  7|    0|                    fax|             202-225-1314|
|  7|    1|                  phone|             202-225-3772|
|  7|    2|                twitter|          MikeRossUpdates|
|  8|    0|                    fax|             202-225-1314|
+---+-----+-----------------------+-----------------------+
only showing top 20 rows
```

```
Inspect the DynamicFrame that contains IDs > 10
+---+-----+-----------------------+-----------------------+
| id|index|contact_details.val.type|contact_details.val.value|
+---+-----+-----------------------+-----------------------+
| 11|    0|                  phone|             202-225-5476|
| 11|    1|                twitter|            RepDavidYoung|
| 12|    0|                  phone|             202-225-4035|
| 12|    1|                twitter|           RepStephMurphy|
| 13|    0|                    fax|             202-226-0774|
| 13|    1|                  phone|             202-225-6335|
| 14|    0|                    fax|             202-226-0774|
| 14|    1|                  phone|             202-225-6335|
| 15|    0|                    fax|             202-226-0774|
| 15|    1|                  phone|             202-225-6335|
| 16|    0|                    fax|             202-226-0774|
| 16|    1|                  phone|             202-225-6335|
| 17|    0|                    fax|             202-226-0774|
| 17|    1|                  phone|             202-225-6335|
| 18|    0|                    fax|             202-226-0774|
| 18|    1|                  phone|             202-225-6335|
| 19|    0|                    fax|             202-226-0774|
| 19|    1|                  phone|             202-225-6335|
| 20|    0|                    fax|             202-226-0774|
| 20|    1|                  phone|             202-225-6335|
+---+-----+-----------------------+-----------------------+
only showing top 20 rows
```

unbox

`unbox(path, format, transformation_ctx="", info="", stageThreshold=0, totalThreshold=0, **options)`

Effectue une opération unbox sur un champ de chaîne (ou le reformate) dans une image `DynamicFrame` et renvoie une nouvelle image `DynamicFrame` contenant les enregistrements `DynamicRecords` sur lesquels l'opération unbox a été effectuée.

Un `DynamicRecord` représente un enregistrement logique dans un `DynamicFrame`. Il est comparable à une ligne dans un `DataFrame` Apache Spark, à la différence qu'il est auto-descriptif et peut être utilisé pour les données qui ne sont pas conformes à un schéma fixe.

- `path` – Chemin d'accès complet au nœud de chaîne pour lequel vous souhaitez effectuer une opération unbox.

- `format` – spécification de format (facultatif). Utilisée pour une connexion Amazon S3 ou AWS Glue prenant en charge plusieurs formats. Consultez Options de format pour les entrées et sorties dans AWS Glue pour Spark pour connaître les formats pris en charge.
- `transformation_ctx` – Chaîne unique utilisée pour identifier les informations sur l'état (facultatif).
- `info` – Chaîne à associer avec le signalement des erreurs pour cette transformation (facultatif).
- `stageThreshold` – Nombre d'erreurs rencontrées pendant cette transformation et qui doit entraîner l'erreur générée par le processus (facultatif) La valeur par défaut est zéro, ce qui indique que le processus ne doit pas générer d'erreur.
- `totalThreshold` – Nombre d'erreurs rencontrées jusqu'à cette transformation, incluse, et qui doit entraîner l'arrêt du processus (facultatif). La valeur par défaut est zéro, ce qui indique que le processus ne doit pas générer d'erreur.
- `options` – Un ou plusieurs des éléments suivants :
  - `separator` – Chaîne contenant le caractère de séparation.
  - `escaper` – Chaîne contenant le caractère d'échappement.
  - `skipFirst` – Valeur booléenne indiquant s'il faut ignorer la première instance.
  - `withSchema` : une chaîne contenant une représentation JSON du schéma du nœud. Le format de la représentation JSON d'un schéma est défini par la sortie de `StructType.json()`.
  - `withHeader` – Valeur booléenne indiquant si un en-tête est inclus.

Exemple : utiliser la méthode unbox pour effectuer une opération unbox sur un champ de chaîne en un champ struct

Cet exemple de code utilise la méthode `unbox` pour effectuer une opération unbox sur un champ de chaîne dans une image `DynamicFrame` (ou la reformater) en un champ de type struct.

Exemple de jeu de données

L'exemple utilise une image `DynamicFrame` appelée `mapped_with_string` avec le schéma et les entrées suivantes.

Notez le champ nommé `AddressString`. Il s'agit du champ sur lequel l'exemple effectue une opération unbox en un champ struct.

```
root
|-- Average Total Payments: string
```

```
|-- AddressString: string
|-- Average Covered Charges: string
|-- DRG Definition: string
|-- Average Medicare Payments: string
|-- Hospital Referral Region Description: string
|-- Address: struct
|    |-- Zip.Code: string
|    |-- City: string
|    |-- Array: array
|    |    |-- element: string
|    |-- State: string
|    |-- Street: string
|-- Provider Id: string
|-- Total Discharges: string
|-- Provider Name: string

+----------------------+--------------------+-----------------------
+--------------------+------------------------+-----------------------------------
+--------------------+-----------+---------------+--------------------+
|Average Total Payments|       AddressString|Average Covered Charges|      DRG
 Definition|Average Medicare Payments|Hospital Referral Region Description|
 Address|Provider Id|Total Discharges|       Provider Name|
+----------------------+--------------------+-----------------------
+--------------------+------------------------+-----------------------------------
+--------------------+-----------+---------------+--------------------+
|              $5777.24|{"Street": "1108 ...|             $32963.07|039 -
 EXTRACRANIA...|                $4763.73|                         AL - Dothan|[36301,
 DOTHAN, [...|      10001|             91|SOUTHEAST ALABAMA...|
|              $5787.57|{"Street": "2505 ...|             $15131.85|039 -
 EXTRACRANIA...|                $4976.71|                     AL - Birmingham|[35957,
 BOAZ, [25...|      10005|             14|MARSHALL MEDICAL ...|
|              $5434.95|{"Street": "205 M...|             $37560.37|039 -
 EXTRACRANIA...|                $4453.79|                     AL - Birmingham|[35631,
 FLORENCE,...|      10006|             24|ELIZA COFFEE MEMO...|
|              $5417.56|{"Street": "50 ME...|             $13998.28|039 -
 EXTRACRANIA...|                $4129.16|                     AL - Birmingham|[35235,
 BIRMINGHA...|      10011|             25|   ST VINCENT'S EAST|
...
```

Exemple de code

```
# Example: Use unbox to unbox a string field
```

```
# into a struct in a DynamicFrame

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

unboxed = mapped_with_string.unbox("AddressString", "json")
unboxed.printSchema()
unboxed.toDF().show()
```

Output

```
root
|-- Average Total Payments: string
|-- AddressString: struct
|    |-- Street: string
|    |-- City: string
|    |-- State: string
|    |-- Zip.Code: string
|    |-- Array: array
|    |    |-- element: string
|-- Average Covered Charges: string
|-- DRG Definition: string
|-- Average Medicare Payments: string
|-- Hospital Referral Region Description: string
|-- Address: struct
|    |-- Zip.Code: string
|    |-- City: string
|    |-- Array: array
|    |    |-- element: string
|    |-- State: string
|    |-- Street: string
|-- Provider Id: string
|-- Total Discharges: string
|-- Provider Name: string

+----------------------+--------------------+-----------------------
+--------------------+-------------------------+-------------------------------------
+--------------------+-----------+----------------+--------------------+
```

```
|Average Total Payments|       AddressString|Average Covered Charges|      DRG
 Definition|Average Medicare Payments|Hospital Referral Region Description|
 Address|Provider Id|Total Discharges|       Provider Name|
+----------------------+--------------------+-----------------------
+--------------------+------------------------+-----------------------------------
+--------------------+-----------+---------------+--------------------+
|              $5777.24|[1108 ROSS CLARK ...|             $32963.07|039 -
 EXTRACRANIA...|                $4763.73|                        AL - Dothan|[36301,
 DOTHAN, [...|      10001|             91|SOUTHEAST ALABAMA...|
|              $5787.57|[2505 U S HIGHWAY...|             $15131.85|039 -
 EXTRACRANIA...|                $4976.71|                    AL - Birmingham|[35957,
 BOAZ, [25...|      10005|             14|MARSHALL MEDICAL ...|
|              $5434.95|[205 MARENGO STRE...|             $37560.37|039 -
 EXTRACRANIA...|                $4453.79|                    AL - Birmingham|[35631,
 FLORENCE,...|      10006|             24|ELIZA COFFEE MEMO...|
|              $5417.56|[50 MEDICAL PARK ...|             $13998.28|039 -
 EXTRACRANIA...|                $4129.16|                    AL - Birmingham|[35235,
 BIRMINGHA...|      10011|             25|   ST VINCENT'S EAST|
|              $5658.33|[1000 FIRST STREE...|             $31633.27|039 -
 EXTRACRANIA...|                $4851.44|                    AL - Birmingham|[35007,
 ALABASTER...|      10016|             18|SHELBY BAPTIST ME...|
|              $6653.80|[2105 EAST SOUTH ...|             $16920.79|039 -
 EXTRACRANIA...|                $5374.14|                    AL - Montgomery|[36116,
 MONTGOMER...|      10023|             67|BAPTIST MEDICAL C...|
|              $5834.74|[2000 PEPPERELL P...|             $11977.13|039 -
 EXTRACRANIA...|                $4761.41|                    AL - Birmingham|[36801,
 OPELIKA, ...|      10029|             51|EAST ALABAMA MEDI...|
|              $8031.12|[619 SOUTH 19TH S...|             $35841.09|039 -
 EXTRACRANIA...|                $5858.50|                    AL - Birmingham|[35233,
 BIRMINGHA...|      10033|             32|UNIVERSITY OF ALA...|
|              $6113.38|[101 SIVLEY RD, H...|             $28523.39|039 -
 EXTRACRANIA...|                $5228.40|                    AL - Huntsville|[35801,
 HUNTSVILL...|      10039|            135| HUNTSVILLE HOSPITAL|
|              $5541.05|[1007 GOODYEAR AV...|             $75233.38|039 -
 EXTRACRANIA...|                $4386.94|                    AL - Birmingham|[35903,
 GADSDEN, ...|      10040|             34|GADSDEN REGIONAL ...|
|              $5461.57|[600 SOUTH THIRD ...|             $67327.92|039 -
 EXTRACRANIA...|                $4493.57|                    AL - Birmingham|[35901,
 GADSDEN, ...|      10046|             14|RIVERVIEW REGIONA...|
|              $5356.28|[4370 WEST MAIN S...|             $39607.28|039 -
 EXTRACRANIA...|                $4408.20|                        AL - Dothan|[36305,
 DOTHAN, [...|      10055|             45|    FLOWERS HOSPITAL|
```

```
|              $5374.65|[810 ST VINCENT'S...|              $22862.23|039 -
 EXTRACRANIA...|                 $4186.02|                     AL - Birmingham|[35205,
 BIRMINGHA...|      10056|              43|ST VINCENT'S BIRM...|
|              $5366.23|[400 EAST 10TH ST...|              $31110.85|039 -
 EXTRACRANIA...|                 $4376.23|                     AL - Birmingham|[36207,
 ANNISTON,...|      10078|              21|NORTHEAST ALABAMA...|
|              $5282.93|[1613 NORTH MCKEN...|              $25411.33|039 -
 EXTRACRANIA...|                 $4383.73|                        AL - Mobile|[36535,
 FOLEY, [1...|      10083|              15|SOUTH BALDWIN REG...|
|              $5676.55|[1201 7TH STREET ...|               $9234.51|039 -
 EXTRACRANIA...|                 $4509.11|                     AL - Huntsville|[35609,
 DECATUR, ...|      10085|              27|DECATUR GENERAL H...|
|              $5930.11|[6801 AIRPORT BOU...|              $15895.85|039 -
 EXTRACRANIA...|                 $3972.85|                        AL - Mobile|[36608,
 MOBILE, [...|      10090|              27| PROVIDENCE HOSPITAL|
|              $6192.54|[809 UNIVERSITY B...|              $19721.16|039 -
 EXTRACRANIA...|                 $5179.38|                     AL - Tuscaloosa|[35401,
 TUSCALOOS...|      10092|              31|D C H REGIONAL ME...|
|              $4968.00|[750 MORPHY AVENU...|              $10710.88|039 -
 EXTRACRANIA...|                 $3898.88|                        AL - Mobile|[36532,
 FAIRHOPE,...|      10100|              18|     THOMAS HOSPITAL|
|              $5996.00|[701 PRINCETON AV...|              $51343.75|039 -
 EXTRACRANIA...|                 $4962.45|                     AL - Birmingham|[35211,
 BIRMINGHA...|      10103|              33|BAPTIST MEDICAL C...|
+----------------------+--------------------+-----------------------
+--------------------+------------------------+-----------------------------------
+--------------------+-----------+---------------+--------------------+
only showing top 20 rows
```

union

union(frame1, frame2, transformation_ctx = "", info = "", stageThreshold = 0, totalThreshold = 0)

Union 2 DynamicFrames. Renvoie DynamicFrame tous les enregistrements des deux entrées DynamicFrames. Cette transformation peut renvoyer des résultats différents à partir de l'union de deux DataFrames avec des données équivalentes. Si vous avez besoin du comportement DataFrame syndical de Spark, pensez à utiliser`toDF`.

- `frame1`— D'abord DynamicFrame à se syndiquer.
- `frame2`— Après DynamicFrame l'union.

- `transformation_ctx` : (facultatif) chaîne unique utilisée pour identifier les informations sur l'état/les statistiques.
- `info` : (facultatif) chaîne à associer à des erreurs dans la transformation.
- `stageThreshold` : (facultatif) nombre maximum d'erreurs dans la transformation jusqu'à ce que le traitement entraîne une erreur
- `totalThreshold` : (facultatif) nombre maximum d'erreurs au total jusqu'à ce que le traitement entraîne une erreur.

unnest

`unnest(transformation_ctx="", info="", stageThreshold=0, totalThreshold=0)`

Désimbrique les objets imbriqués dans une image `DynamicFrame`. Ils deviennent des objets de niveau supérieur et l'opération renvoie une nouvelle image `DynamicFrame` non imbriquée.

- `transformation_ctx` – Chaîne unique utilisée pour identifier les informations sur l'état (facultatif).
- `info` – Chaîne à associer avec le signalement des erreurs pour cette transformation (facultatif).
- `stageThreshold` – Nombre d'erreurs rencontrées pendant cette transformation et qui doit entraîner l'erreur générée par le processus (facultatif) La valeur par défaut est zéro, ce qui indique que le processus ne doit pas générer d'erreur.
- `totalThreshold` – Nombre d'erreurs rencontrées jusqu'à cette transformation, incluse, et qui doit entraîner l'arrêt du processus (facultatif). La valeur par défaut est zéro, ce qui indique que le processus ne doit pas générer d'erreur.

Exemple : utiliser la méthode unnest pour transformer des champs imbriqués en champs de niveau supérieur

Cet exemple de code utilise la méthode `unnest` pour aplatir tous les champs imbriqués dans une image `DynamicFrame` en champs de niveau supérieur.

Exemple de jeu de données

L'exemple utilise une image `DynamicFrame` appelée `mapped_medicare` avec le schéma suivant. Notez que le champ `Address` est le seul qui contient des données imbriquées.

```
root
|-- Average Total Payments: string
```

```
|-- Average Covered Charges: string
|-- DRG Definition: string
|-- Average Medicare Payments: string
|-- Hospital Referral Region Description: string
|-- Address: struct
|    |-- Zip.Code: string
|    |-- City: string
|    |-- Array: array
|    |    |-- element: string
|    |-- State: string
|    |-- Street: string
|-- Provider Id: string
|-- Total Discharges: string
|-- Provider Name: string
```

Exemple de code

```
# Example: Use unnest to unnest nested
# objects in a DynamicFrame

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Unnest all nested fields
unnested = mapped_medicare.unnest()
unnested.printSchema()
```

Output

```
root
|-- Average Total Payments: string
|-- Average Covered Charges: string
|-- DRG Definition: string
|-- Average Medicare Payments: string
|-- Hospital Referral Region Description: string
|-- Address.Zip.Code: string
|-- Address.City: string
|-- Address.Array: array
|    |-- element: string
```

```
|-- Address.State: string
|-- Address.Street: string
|-- Provider Id: string
|-- Total Discharges: string
|-- Provider Name: string
```

unnest_ddb_json

Supprime l'imbrication des colonnes imbriquées dans un `DynamicFrame` qui se trouvent spécifiquement dans la structure JSON DynamoDB, et renvoie une nouvelle version non imbriquée `DynamicFrame`. Les colonnes d'un tableau de types de structure ne seront pas non-imbriquées. Notez qu'il s'agit d'un type spécifique de transformation non imbriquée qui se comporte différemment de la transformation `unnest` normale et nécessite que les données soient déjà dans la structure JSON DynamoDB. Pour plus d'informations, consultez JSON DynamoDB.

`unnest_ddb_json(transformation_ctx="", info="", stageThreshold=0, totalThreshold=0)`

- `transformation_ctx` – Chaîne unique utilisée pour identifier les informations sur l'état (facultatif).
- `info` – Chaîne à associer avec le signalement des erreurs pour cette transformation (facultatif).
- `stageThreshold` – Nombre d'erreurs rencontrées pendant cette transformation et qui doit entraîner l'arrêt du processus (facultatif : zéro par défaut, qui indique que le processus ne doit pas être arrêté dans ce cas).
- `totalThreshold` – Nombre d'erreurs rencontrées jusqu'à cette transformation, incluse, et qui doit entraîner l'arrêt du processus (facultatif : zéro par défaut, qui indique que le processus ne doit pas être arrêté dans ce cas).

Par exemple, le schéma d'une lecture d'exportation avec la structure JSON DynamoDB pourrait ressembler à ce qui suit :

```
root
|-- Item: struct
|    |-- ColA: struct
|    |    |-- S: string
|    |-- ColB: struct
|    |    |-- S: string
|    |-- ColC: struct
```

```
|    |    |-- N: string
|    |-- ColD: struct
|    |    |-- L: array
|    |    |    |-- element: null
```

La transformation `unnest_ddb_json()` convertirait ceci en :

```
root
|-- ColA: string
|-- ColB: string
|-- ColC: string
|-- ColD: array
|    |-- element: null
```

L'exemple de code suivant montre comment utiliser le connecteur d'exportation AWS Glue DynamoDB, appeler un DynamoDB JSON unnest et imprimer le nombre de partitions :

```
import sys
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job
from awsglue.utils import getResolvedOptions

args = getResolvedOptions(sys.argv, ["JOB_NAME"])
glue_context= GlueContext(SparkContext.getOrCreate())
job = Job(glue_context)
job.init(args["JOB_NAME"], args)

dynamicFrame = glue_context.create_dynamic_frame.from_options(
    connection_type="dynamodb",
    connection_options={
        "dynamodb.export": "ddb",
        "dynamodb.tableArn": "<test_source>",
        "dynamodb.s3.bucket": "<bucket name>",
        "dynamodb.s3.prefix": "<bucket prefix>",
        "dynamodb.s3.bucketOwner": "<account_id>",
    }
)
unnested = dynamicFrame.unnest_ddb_json()
print(unnested.getNumPartitions())

job.commit()
```

write

`write(connection_type, connection_options, format, format_options, accumulator_size)`

Permet d'obtenir un DataSink(objet) du type de connexion spécifié à partir du GlueContext classe de ce type `DynamicFrame`, et l'utilise pour formater et écrire le contenu de ce `DynamicFrame`. Renvoie le nouveau `DynamicFrame` formaté et écrit comme spécifié.

- `connection_type` – type de connexion à utiliser. Les valeurs valides sont `s3`, `mysql`, `postgresql`, `redshift`, `sqlserver` et `oracle`.
- `connection_options` – option de connexion à utiliser (facultatif). Pour un `connection_type` de `s3`, un chemin Amazon S3 est défini.

```
connection_options = {"path": "s3://aws-glue-target/temp"}
```

Pour les connexions JDBC, plusieurs propriétés doivent être définies. Notez que le nom de base de données doit être inclus dans l'URL. Il peut éventuellement être inclus dans les options de connexion.

> **Warning**
>
> Il n'est pas recommandé de stocker des mots de passe dans votre script. Envisagez de les utiliser `boto3` pour les extraire AWS Secrets Manager du catalogue de données AWS Glue.

```
connection_options = {"url": "jdbc-url/database", "user": "username",
 "password": passwordVariable,"dbtable": "table-name", "redshiftTmpDir": "s3-tempdir-
path"}
```

- `format` – spécification de format (facultatif). Utilisée pour une connexion Amazon Simple Storage Service (Amazon S3) ou une connexion AWS Glue qui prend en charge plusieurs formats. Consultez Options de format pour les entrées et sorties dans AWS Glue pour Spark pour connaître les formats pris en charge.
- `format_options` – options de format pour le format spécifié. Consultez Options de format pour les entrées et sorties dans AWS Glue pour Spark pour connaître les formats pris en charge.
- `accumulator_size` : la taille accumulable à utiliser, en octets (facultatif).

– erreurs –

- assertErrorThreshold
- errorsAsDynamiccadre
- errorsCount
- stageErrorsCount

assertErrorThreshold

`assertErrorThreshold( )` – Assertion pour les erreurs dans les transformations ayant créé ce `DynamicFrame`. Renvoie un `Exception` du `DataFrame` sous-jacent.

errorsAsDynamiccadre

`errorsAsDynamicFrame( )` – Renvoie un `DynamicFrame` dans lequel des enregistrements d'erreur sont imbriqués.

Exemple : utilisez errorsAsDynamic Frame pour afficher les enregistrements d'erreurs

L'exemple de code suivant illustre comment utiliser `errorsAsDynamicFrame` méthode pour afficher un enregistrement d'erreur pour un `DynamicFrame`.

Exemple de jeu de données

L'exemple utilise l'ensemble de données suivant que vous pouvez charger sur Amazon S3 au format JSON. Notez que le deuxième enregistrement est mal formé. Les données mal formées interrompent généralement l'analyse des fichiers lorsque vous utilisez SparkSQL. Cependant, `DynamicFrame` reconnaît les problèmes de malformation et transforme les lignes mal formées en enregistrements d'erreurs que vous pouvez gérer individuellement.

```
{"id": 1, "name": "george", "surname": "washington", "height": 178}
{"id": 2, "name": "benjamin", "surname": "franklin",
{"id": 3, "name": "alexander", "surname": "hamilton", "height": 171}
{"id": 4, "name": "john", "surname": "jay", "height": 190}
```

Exemple de code

```
# Example: Use errorsAsDynamicFrame to view error records.
# Replace s3://DOC-EXAMPLE-S3-BUCKET/error_data.json with your location.
```

```
from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Create errors DynamicFrame, view schema
errors = glueContext.create_dynamic_frame.from_options(
    "s3", {"paths": ["s3://DOC-EXAMPLE-S3-BUCKET/error_data.json"]}, "json"
)
print("Schema of errors DynamicFrame:")
errors.printSchema()

# Show that errors only contains valid entries from the dataset
print("errors contains only valid records from the input dataset (2 of 4 records)")
errors.toDF().show()

# View errors
print("Errors count:", str(errors.errorsCount()))
print("Errors:")
errors.errorsAsDynamicFrame().toDF().show()

# View error fields and error data
error_record = errors.errorsAsDynamicFrame().toDF().head()

error_fields = error_record["error"]
print("Error fields: ")
print(error_fields.asDict().keys())

print("\nError record data:")
for key in error_fields.asDict().keys():
    print("\n", key, ": ", str(error_fields[key]))
```

Output

```
Schema of errors DynamicFrame:
root
|-- id: int
|-- name: string
|-- surname: string
|-- height: int
```

```
errors contains only valid records from the input dataset (2 of 4 records)
+---+------+----------+------+
| id|  name|   surname|height|
+---+------+----------+------+
|  1|george|washington|   178|
|  4|  john|       jay|   190|
+---+------+----------+------+

Errors count: 1
Errors:
+--------------------+
|               error|
+--------------------+
|[[  File "/tmp/20...|
+--------------------+

Error fields:
dict_keys(['callsite', 'msg', 'stackTrace', 'input', 'bytesread', 'source',
 'dynamicRecord'])

Error record data:

 callsite :  Row(site='  File "/tmp/2060612586885849088", line 549, in <module>\n
 sys.exit(main())\n  File "/tmp/2060612586885849088", line 523, in main\n    response
 = handler(content)\n  File "/tmp/2060612586885849088", line 197, in execute_request
\n    result = node.execute()\n  File "/tmp/2060612586885849088", line 103, in
 execute\n    exec(code, global_dict)\n  File "<stdin>", line 10, in <module>\n
 File "/opt/amazon/lib/python3.6/site-packages/awsglue/dynamicframe.py", line 625, in
 from_options\n    format_options, transformation_ctx, push_down_predicate, **kwargs)\n
  File "/opt/amazon/lib/python3.6/site-packages/awsglue/context.py", line 233, in
 create_dynamic_frame_from_options\n    source.setFormat(format, **format_options)\n',
 info='')

 msg :  error in jackson reader

 stackTrace :  com.fasterxml.jackson.core.JsonParseException: Unexpected character
 ('{' (code 123)): was expecting either valid name character (for unquoted name) or
 double-quote (for quoted) to start field name
 at [Source: com.amazonaws.services.glue.readers.BufferedStream@73492578; line: 3,
 column: 2]
 at com.fasterxml.jackson.core.JsonParser._constructError(JsonParser.java:1581)
 at
 com.fasterxml.jackson.core.base.ParserMinimalBase._reportError(ParserMinimalBase.java:533)
```

```
 at
 com.fasterxml.jackson.core.base.ParserMinimalBase._reportUnexpectedChar(ParserMinimalBase.java
 at
 com.fasterxml.jackson.core.json.UTF8StreamJsonParser._handleOddName(UTF8StreamJsonParser.java:
 at
 com.fasterxml.jackson.core.json.UTF8StreamJsonParser._parseName(UTF8StreamJsonParser.java:1650
 at
 com.fasterxml.jackson.core.json.UTF8StreamJsonParser.nextToken(UTF8StreamJsonParser.java:740)
 at com.amazonaws.services.glue.readers.JacksonReader$$anonfun$hasNextGoodToken
$1.apply(JacksonReader.scala:57)
 at com.amazonaws.services.glue.readers.JacksonReader$$anonfun$hasNextGoodToken
$1.apply(JacksonReader.scala:57)
 at scala.collection.Iterator$$anon$9.next(Iterator.scala:162)
 at scala.collection.Iterator$$anon$16.hasNext(Iterator.scala:599)
 at scala.collection.Iterator$$anon$16.hasNext(Iterator.scala:598)
 at scala.collection.Iterator$class.foreach(Iterator.scala:891)
 at scala.collection.AbstractIterator.foreach(Iterator.scala:1334)
 at com.amazonaws.services.glue.readers.JacksonReader$$anonfun
$1.apply(JacksonReader.scala:120)
 at com.amazonaws.services.glue.readers.JacksonReader$$anonfun
$1.apply(JacksonReader.scala:116)
 at
 com.amazonaws.services.glue.DynamicRecordBuilder.handleErr(DynamicRecordBuilder.scala:209)
 at
 com.amazonaws.services.glue.DynamicRecordBuilder.handleErrorWithException(DynamicRecordBuilder
 at
 com.amazonaws.services.glue.readers.JacksonReader.nextFailSafe(JacksonReader.scala:116)
 at com.amazonaws.services.glue.readers.JacksonReader.next(JacksonReader.scala:109)
 at com.amazonaws.services.glue.readers.JSONReader.next(JSONReader.scala:247)
 at
 com.amazonaws.services.glue.hadoop.TapeHadoopRecordReaderSplittable.nextKeyValue(TapeHadoopRec
 at org.apache.spark.rdd.NewHadoopRDD$$anon$1.hasNext(NewHadoopRDD.scala:230)
 at org.apache.spark.InterruptibleIterator.hasNext(InterruptibleIterator.scala:37)
 at scala.collection.Iterator$$anon$11.hasNext(Iterator.scala:409)
 at scala.collection.Iterator$$anon$11.hasNext(Iterator.scala:409)
 at scala.collection.Iterator$$anon$13.hasNext(Iterator.scala:462)
 at scala.collection.Iterator$$anon$11.hasNext(Iterator.scala:409)
 at scala.collection.Iterator$$anon$11.hasNext(Iterator.scala:409)
 at scala.collection.Iterator$$anon$13.hasNext(Iterator.scala:462)
 at scala.collection.Iterator$$anon$11.hasNext(Iterator.scala:409)
 at scala.collection.Iterator$$anon$11.hasNext(Iterator.scala:409)
 at org.apache.spark.sql.execution.SparkPlan$$anonfun$2.apply(SparkPlan.scala:255)
 at org.apache.spark.sql.execution.SparkPlan$$anonfun$2.apply(SparkPlan.scala:247)
```

```
 at org.apache.spark.rdd.RDD$$anonfun$mapPartitionsInternal$1$$anonfun$apply
$24.apply(RDD.scala:836)
 at org.apache.spark.rdd.RDD$$anonfun$mapPartitionsInternal$1$$anonfun$apply
$24.apply(RDD.scala:836)
 at org.apache.spark.rdd.MapPartitionsRDD.compute(MapPartitionsRDD.scala:52)
 at org.apache.spark.rdd.RDD.computeOrReadCheckpoint(RDD.scala:324)
 at org.apache.spark.rdd.RDD.iterator(RDD.scala:288)
 at org.apache.spark.rdd.MapPartitionsRDD.compute(MapPartitionsRDD.scala:52)
 at org.apache.spark.rdd.RDD.computeOrReadCheckpoint(RDD.scala:324)
 at org.apache.spark.rdd.RDD.iterator(RDD.scala:288)
 at org.apache.spark.scheduler.ResultTask.runTask(ResultTask.scala:90)
 at org.apache.spark.scheduler.Task.run(Task.scala:121)
 at org.apache.spark.executor.Executor$TaskRunner$$anonfun$10.apply(Executor.scala:408)
 at org.apache.spark.util.Utils$.tryWithSafeFinally(Utils.scala:1360)
 at org.apache.spark.executor.Executor$TaskRunner.run(Executor.scala:414)
 at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1149)
 at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:624)
 at java.lang.Thread.run(Thread.java:750)


 input :

 bytesread :  252

 source :

 dynamicRecord :  Row(id=2, name='benjamin', surname='franklin')
```

DynamicFrame Exemples complets

Les exemples suivants montrent différentes manières de créer et d'utiliser des scénarios DynamicFrames autres que les scénarios de catalogue Glue de base.

Chargement depuis PostgreSQL avec une requête SQL SELECT

Cet exemple montre comment charger des données depuis une base de données PostgreSQL à l'aide d'une requête SQL SELECT personnalisée :

```
from awsglue.context import GlueContext
from pyspark.context import SparkContext

sc = SparkContext()
```

```
glueContext = GlueContext(sc)

# Load specific data from PostgreSQL with custom query
postgres_dyf = glueContext.create_dynamic_frame.from_options(
    connection_type="postgresql",
    connection_options={
        "url": "jdbc:postgresql://your-postgres-host:5432/your-database",
        "user": "your-username",
        "password": "your-password",
        "dbtable": "(SELECT customer_id, customer_name, email FROM customers WHERE
 active = true) AS filtered_customers"
    }
)
```

Chargement de colonnes spécifiques pour éviter l'analyse complète du tableau

Cet exemple montre comment charger uniquement des colonnes spécifiques à partir d'une grande table de base de données :

```
from awsglue.context import GlueContext
from pyspark.context import SparkContext

sc = SparkContext()
glueContext = GlueContext(sc)

# Load only specific columns from a large table
selected_columns_dyf = glueContext.create_dynamic_frame.from_options(
    connection_type="mysql",
    connection_options={
        "url": "jdbc:mysql://your-mysql-host:3306/your-database",
        "user": "your-username",
        "password": "your-password",
        "dbtable": "(SELECT order_id, customer_id FROM large_orders_table) AS
 selected_data"
    }
)

# Alternative approach using column selection in query
efficient_load_dyf = glueContext.create_dynamic_frame.from_options(
    connection_type="postgresql",
    connection_options={
        "url": "jdbc:postgresql://your-postgres-host:5432/your-database",
        "user": "your-username",
```

```
        "password": "your-password",
        "query": "SELECT product_id, product_name FROM products WHERE category =
 'electronics'"
    }
)
```

Filtrage au niveau des lignes via des connexions JDBC

Cet exemple montre comment utiliser le filtrage au niveau des lignes pour charger uniquement des lignes spécifiques d'une table de base de données :

```
from awsglue.context import GlueContext
from pyspark.context import SparkContext

sc = SparkContext()
glueContext = GlueContext(sc)

# Load filtered rows using WHERE clause
filtered_rows_dyf = glueContext.create_dynamic_frame.from_options(
    connection_type="postgresql",
    connection_options={
        "url": "jdbc:postgresql://your-postgres-host:5432/your-database",
        "user": "your-username",
        "password": "your-password",
        "dbtable": "(SELECT * FROM transactions WHERE transaction_date >= '2024-01-01'
 AND amount > 100) AS recent_large_transactions"
    }
)

# Using partitionColumn for parallel loading with filtering
partitioned_load_dyf = glueContext.create_dynamic_frame.from_options(
    connection_type="mysql",
    connection_options={
        "url": "jdbc:mysql://your-mysql-host:3306/your-database",
        "user": "your-username",
        "password": "your-password",
        "dbtable": "sales_data",
        "partitionColumn": "sale_date",
        "lowerBound": "2024-01-01",
        "upperBound": "2024-12-31",
        "numPartitions": "10"
    }
)
```

Création DynamicFrame à partir de données Python en mémoire

Cet exemple montre comment créer un fichier à DynamicFrame partir de listes, de tuples ou de dictionnaires Python :

```
from awsglue.context import GlueContext
from awsglue.dynamicframe import DynamicFrame
from pyspark.context import SparkContext
from pyspark.sql import Row

sc = SparkContext()
glueContext = GlueContext(sc)

# Method 1: From list of tuples
data_tuples = [
    ("John", "Doe", 30, "Engineer"),
    ("Jane", "Smith", 25, "Designer"),
    ("Bob", "Johnson", 35, "Manager")
]

# Convert to RDD of Rows
rdd = sc.parallelize([Row(first_name=row[0], last_name=row[1], age=row[2], job=row[3])
 for row in data_tuples])
df = glueContext.spark_session.createDataFrame(rdd)
dyf_from_tuples = DynamicFrame.fromDF(df, glueContext, "employees_from_tuples")

# Method 2: From list of dictionaries
data_dicts = [
    {"product_id": 1, "product_name": "Laptop", "price": 999.99, "category":
 "Electronics"},
    {"product_id": 2, "product_name": "Book", "price": 19.99, "category": "Education"},
    {"product_id": 3, "product_name": "Chair", "price": 149.99, "category":
 "Furniture"}
]

df_from_dicts = glueContext.spark_session.createDataFrame(data_dicts)
dyf_from_dicts = DynamicFrame.fromDF(df_from_dicts, glueContext, "products_from_dicts")

# Method 3: From nested data structures
nested_data = [
    {
        "customer_id": 1,
        "customer_info": {
            "name": "Alice Brown",
```

```
            "email": "alice@example.com"
        },
        "orders": [
            {"order_id": 101, "amount": 250.00},
            {"order_id": 102, "amount": 175.50}
        ]
    }
]

df_nested = glueContext.spark_session.createDataFrame(nested_data)
dyf_nested = DynamicFrame.fromDF(df_nested, glueContext, "customers_with_orders")
```

Optimisation des performances pour les grands ensembles de données

Lorsque vous travaillez avec de grands ensembles de données, considérez les techniques d'optimisation des performances suivantes :

```
# Use partitioning for parallel reads
large_table_dyf = glueContext.create_dynamic_frame.from_options(
    connection_type="postgresql",
    connection_options={
        "url": "jdbc:postgresql://your-postgres-host:5432/your-database",
        "user": "your-username",
        "password": "your-password",
        "dbtable": "large_table",
        "partitionColumn": "id",
        "lowerBound": "1",
        "upperBound": "1000000",
        "numPartitions": "20"
    }
)

# Use pushdown predicates to filter at source
filtered_dyf = glueContext.create_dynamic_frame.from_options(
    connection_type="mysql",
    connection_options={
        "url": "jdbc:mysql://your-mysql-host:3306/your-database",
        "user": "your-username",
        "password": "your-password",
        "dbtable": "transactions"
    },
    push_down_predicate="transaction_date >= '2024-01-01'"
)
```

errorsCount

`errorsCount( )` – Renvoie le nombre total d'erreurs dans un `DynamicFrame`.

stageErrorsCount

`stageErrorsCount` – Renvoie le nombre d'erreurs survenues lors du processus de génération de ce `DynamicFrame`.

DynamicFrameCollection classe

Un `DynamicFrameCollection` est un dictionnaire d'objets DynamicFrame classe dans lequel les clés sont les noms des `DynamicFrames` et les valeurs, les objets `DynamicFrame`.

__init__

`__init__(dynamic_frames, glue_ctx)`

- `dynamic_frames` – Dictionnaire d'objets DynamicFrame classe.
- `glue_ctx` – Un objet GlueContext classe.

Clés

`keys( )` – Renvoie une liste des clés de cette collection, qui comprend généralement les noms des valeurs `DynamicFrame` correspondantes.

Valeurs

`values(key)` – Renvoie une liste des valeurs `DynamicFrame` de cette collection.

Select

`select(key)`

Renvoie le `DynamicFrame` correspondant à la clé spécifiée (il s'agit généralement du nom du `DynamicFrame`).

- `key` - Clé du `DynamicFrameCollection`, qui représente généralement le nom d'un `DynamicFrame`.

Map

`map(callable, transformation_ctx="")`

Utilise une fonction transmise pour créer et renvoyer un nouveau `DynamicFrameCollection` basé sur le `DynamicFrames` de cette collection.

- `callable` - Fonction qui utilise un `DynamicFrame` et le contexte de transformation en tant que paramètres et renvoie un `DynamicFrame`.
- `transformation_ctx` - Contexte de transformation utilisé par l'élément appelable (facultatif).

FlatMap

`flatmap(f, transformation_ctx="")`

Utilise une fonction transmise pour créer et renvoyer un nouveau `DynamicFrameCollection` basé sur le `DynamicFrames` de cette collection.

- `f` - Fonction qui utilise un `DynamicFrame` en tant que paramètre et renvoie un `DynamicFrame` ou un `DynamicFrameCollection`.
- `transformation_ctx` - Contexte de transformation utilisé par la fonction(facultatif).

DynamicFrameWriter classe

Méthodes

- __init__
- from_options
- from_catalog
- from_jdbc_conf

__init__

`__init__(glue_context)`

- `glue_context` – GlueContext classe à utiliser.

from_options

`from_options(frame, connection_type, connection_options={}, format=None, format_options={}, transformation_ctx="")`

Écrit un `DynamicFrame` à l'aide de la connexion et du format spécifiés.

- `frame` – Objet `DynamicFrame` à écrire.
- `connection_type` – type de connexion. Les valeurs valides sont `s3`, `mysql`, `postgresql`, `redshift`, `sqlserver` et `oracle`.
- `connection_options` – Options de connexion, telles que le chemin et la table de base de données (facultatif). Pour un `connection_type` de `s3`, un chemin Amazon S3 est défini.

```
connection_options = {"path": "s3://aws-glue-target/temp"}
```

Pour les connexions JDBC, plusieurs propriétés doivent être définies. Notez que le nom de base de données doit être inclus dans l'URL. Il peut éventuellement être inclus dans les options de connexion.

> **Warning**
>
> Il n'est pas recommandé de stocker des mots de passe dans votre script. Envisagez de les utiliser `boto3` pour les récupérer depuis AWS Secrets Manager le catalogue de données AWS Glue.

```
connection_options = {"url": "jdbc-url/database", "user": "username",
 "password": passwordVariable,"dbtable": "table-name", "redshiftTmpDir": "s3-tempdir-path"}
```

La propriété `dbtable` est le nom de la table JDBC. Pour les magasins de données JDBC qui prennent en charge les schémas dans une base de données, spécifiez `schema.table-name`. Si aucun schéma n'est fourni, c'est le schéma « public » par défaut qui est utilisé.

Pour plus d'informations, consultez Types et options de connexion pour ETL dans AWS Glue pour Spark.

- `format` – spécification de format (facultatif). Utilisée pour une connexion Amazon Simple Storage Service (Amazon S3) ou une connexion AWS Glue qui prend en charge plusieurs formats.

Consultez Options de format pour les entrées et sorties dans AWS Glue pour Spark pour connaître les formats pris en charge.

- `format_options` – options de format pour le format spécifié. Consultez Options de format pour les entrées et sorties dans AWS Glue pour Spark pour connaître les formats pris en charge.
- `transformation_ctx` – Contexte de transformation à utiliser (facultatif).

from_catalog

`from_catalog(frame, name_space, table_name, redshift_tmp_dir="", transformation_ctx="")`

Écrit un `DynamicFrame` à l'aide de la base de données et du nom de table du catalogue spécifié.

- `frame` – Objet `DynamicFrame` à écrire.
- `name_space` – Base de données à utiliser.
- `table_name` – `table_name` à utiliser.
- `redshift_tmp_dir` – répertoire Amazon Redshift temporaire à utiliser (facultatif).
- `transformation_ctx` – Contexte de transformation à utiliser (facultatif).
- `additional_options` — Options supplémentaires fournies à AWS Glue.

  Pour écrire sur des tables régies Lake Formation, vous pouvez utiliser ces options supplémentaires :

  - `transactionId`— (Chaîne) ID de transaction auquel effectuer l'écriture dans la table régie. Cette transaction ne peut pas être déjà validée ou interrompue, sinon l'écriture échouera.
  - `callDeleteObjectsOnCancel` – (booléen, facultatif) Si la valeur est définie sur `true` (par défaut), AWS Glue appelle automatiquement l'API `DeleteObjectsOnCancel` une fois l'objet écrit sur Amazon S3. Pour plus d'informations, consultez DeleteObjectsOnCancel dans le Guide du développeur AWS Lake Formation .

  Example Exemple : écriture dans une table régie dans Lake Formation

```
txId = glueContext.start_transaction(read_only=False)
glueContext.write_dynamic_frame.from_catalog(
    frame=dyf,
    database = db,
    table_name = tbl,
    transformation_ctx = "datasource0",
```

```
    additional_options={"transactionId":txId})
...
glueContext.commit_transaction(txId)
```

from_jdbc_conf

`from_jdbc_conf(frame, catalog_connection, connection_options={}, redshift_tmp_dir = "", transformation_ctx="")`

Écrit un `DynamicFrame` à l'aide des informations de connexion JDBC spécifiées.

- `frame` – Objet `DynamicFrame` à écrire.
- `catalog_connection` – Connexion de catalogue à utiliser.
- `connection_options` – Options de connexion, telles que le chemin et la table de base de données (facultatif).
- `redshift_tmp_dir` – répertoire Amazon Redshift temporaire à utiliser (facultatif).
- `transformation_ctx` – Contexte de transformation à utiliser (facultatif).

Exemple pour write_dynamic_frame

Cet exemple écrit la sortie localement en utilisant un `connection_type` de S3 avec un argument de chemin POSIX dans `connection_options`, ce qui permet d'écrire dans le stockage local.

```
glueContext.write_dynamic_frame.from_options(\
frame = dyf_splitFields,\
connection_options = {'path': '/home/glue/GlueLocalOutput/'},\
connection_type = 's3',\
format = 'json')
```

DynamicFrameReader classe

– méthodes –

- __init__
- from_rdd
- from_options
- from_catalog

__init__

`__init__(glue_context)`

- `glue_context` – GlueContext classe à utiliser.

from_rdd

`from_rdd(data, name, schema=None, sampleRatio=None)`

Lit un `DynamicFrame` à partir d'un RDD (Resilient Distributed Dataset).

- `data` – Ensemble de données à partir duquel lire.
- `name` – Nom à partir duquel lire.
- `schema` – Schéma à lire (facultatif).
- `sampleRatio` – Exemple de ratio (facultatif).

from_options

`from_options(connection_type, connection_options={}, format=None, format_options={}, transformation_ctx="")`

Lit un `DynamicFrame` à l'aide de la connexion et du format spécifiés.

- `connection_type` – type de connexion. Les valeurs valides sont les suivantes : `s3`, `mysql`, `postgresql`, `redshift`, `sqlserver`, `oracle`, `dynamodb` et `snowflake`.
- `connection_options` – Options de connexion, telles que le chemin et la table de base de données (facultatif). Pour plus d'informations, consultez Types de connexion et options pour l'ETL dans AWS Glue for Spark. Pour un `connection_type` de `s3`, les chemins Amazon S3 sont définis dans un tableau.

  ```
  connection_options = {"paths": [ "s3://amzn-s3-demo-bucket/object_a", "s3://amzn-s3-demo-bucket/object_b"]}
  ```

  Pour les connexions JDBC, plusieurs propriétés doivent être définies. Notez que le nom de base de données doit être inclus dans l'URL. Il peut éventuellement être inclus dans les options de connexion.

> **Warning**
>
> Il n'est pas recommandé de stocker des mots de passe dans votre script. Envisagez de les utiliser `boto3` pour les extraire AWS Secrets Manager du catalogue de données AWS Glue.

```
connection_options = {"url": "jdbc-url/database", "user": "username",
 "password": passwordVariable,"dbtable": "table-name", "redshiftTmpDir": "s3-tempdir-path"}
```

Pour une connexion JDBC qui effectue des lectures parallèles, vous pouvez définir l'option hashfield. Par exemple :

```
connection_options = {"url": "jdbc-url/database", "user": "username",
 "password": passwordVariable,"dbtable": "table-name", "redshiftTmpDir": "s3-tempdir-path" , "hashfield": "month"}
```

Pour de plus amples informations, veuillez consulter Lecture en parallèle à partir de tables JDBC.

- `format` – spécification de format (facultatif). Utilisée pour une connexion Amazon Simple Storage Service (Amazon S3) ou une connexion AWS Glue qui prend en charge plusieurs formats. Consultez Options de format pour les entrées et sorties dans AWS Glue pour Spark pour connaître les formats pris en charge.
- `format_options` – options de format pour le format spécifié. Consultez Options de format pour les entrées et sorties dans AWS Glue pour Spark pour connaître les formats pris en charge.
- `transformation_ctx` – Contexte de transformation à utiliser (facultatif).
- `push_down_predicate` – filtre les partitions sans avoir à répertorier ni lire tous les fichiers de votre jeu de données. Pour plus d'informations, consultez Pre-Filtering Using Pushdown Predicates.

from_catalog

`from_catalog(database, table_name, redshift_tmp_dir="", transformation_ctx="", push_down_predicate="", additional_options={})`

Lit un `DynamicFrame` à l'aide de l'espace de noms et du nom de table du catalogue spécifié.

- `database` – Base de données à partir de laquelle lire.
- `table_name` – Nom de la table à partir de laquelle lire.
- `redshift_tmp_dir` – répertoire Amazon Redshift temporaire à utiliser (facultatif s'il ne lit pas les données provenant de Redshift).
- `transformation_ctx` – Contexte de transformation à utiliser (facultatif).
- `push_down_predicate` – filtre les partitions sans avoir à répertorier ni lire tous les fichiers de votre jeu de données. Pour de plus amples informations, veuillez consulter Préfiltrage à l'aide des prédicats pushdown.
- `additional_options` — Options supplémentaires fournies à AWS Glue.
  - Pour utiliser une connexion JDBC qui effectue des lectures parallèles, vous pouvez définir les options `hashfield`, `hashexpression` ou `hashpartitions`. Par exemple :

    ```
    additional_options = {"hashfield": "month"}
    ```

    Pour de plus amples informations, veuillez consulter Lecture en parallèle à partir de tables JDBC.
  - Pour passer une expression de catalogue à filtrer en fonction des colonnes d'index, vous pouvez voir l'option `catalogPartitionPredicate`.

    `catalogPartitionPredicate` – vous pouvez passer une expression de catalogue à filtrer en fonction des colonnes d'index. Cela envoie le filtrage du côté serveur. Pour en savoir plus, consultez AWS Glue Indexes de partition. Notez que `push_down_predicate` et `catalogPartitionPredicate` utilisent des syntaxes différentes. Le premier utilise la syntaxe standard SQL Spark et le dernier utilise l'analyseur JSQL.

    Pour de plus amples informations, veuillez consulter Gestion des partitions pour la sortie ETL dans AWS Glue.

GlueContext classe

Encapsule l'SparkContextobjet Apache Spark et fournit ainsi des mécanismes permettant d'interagir avec la plateforme Apache Spark.

__init__

`__init__(sparkContext)`

- `sparkContext` – Contexte Apache Spark à utiliser.

Création

- __init__
- getSource
- create_dynamic_frame_from_rdd
- create_dynamic_frame_from_catalog
- create_dynamic_frame_from_options
- create_sample_dynamic_frame_from_catalog
- create_sample_dynamic_frame_from_options
- add_ingestion_time_columns
- create_data_frame_from_catalog
- create_data_frame_from_options
- forEachBatch

getSource

`getSource(connection_type, transformation_ctx = "", **options)`

Crée un objet `DataSource` qui peut être utilisé pour lire `DynamicFrames` à partir de sources externes.

- `connection_type` – type de connexion à utiliser, tel qu'Amazon Simple Storage Service (Amazon S3), Amazon Redshift et JDBC. Les valeurs valides sont les suivantes : `s3`, `mysql`, `postgresql`, `redshift`, `sqlserver`, `oracle` et dynamodb.
- `transformation_ctx` – Contexte de transformation à utiliser (facultatif).
- `options` – Ensemble de paires nom-valeur facultatives. Pour plus d'informations, consultez Types et options de connexion pour ETL dans AWS Glue pour Spark.

Voici un exemple d'utilisation de : `getSource`.

```
>>> data_source = context.getSource("file", paths=["/in/path"])
>>> data_source.setFormat("json")
>>> myFrame = data_source.getFrame()
```

create_dynamic_frame_from_rdd

`create_dynamic_frame_from_rdd(data, name, schema=None, sample_ratio=None, transformation_ctx="")`

Renvoie un objet `DynamicFrame` créé à partir d'un RDD (Resilient Distributed Dataset) Apache Spark.

- `data` –- La source de données à utiliser.
- `name` – Nom des données à utiliser.
- `schema` – Schéma à utiliser (facultatif).
- `sample_ratio` – Exemple de ratio à utiliser (facultatif).
- `transformation_ctx` – Contexte de transformation à utiliser (facultatif).

create_dynamic_frame_from_catalog

`create_dynamic_frame_from_catalog(database, table_name, redshift_tmp_dir, transformation_ctx = "", push_down_predicate= "", additional_options = {}, catalog_id = None)`

Renvoie un objet `DynamicFrame` créé à l'aide d'un nom de table et de base de données Data Catalog. Lorsque vous utilisez cette méthode, vous fournissez des `format_options` propriétés de table sur la table AWS Glue Data Catalog spécifiée et d'autres options via l'`additional_options`argument.

- `Database` – Base de données à partir de laquelle lire.
- `table_name` – Nom de la table à partir de laquelle lire.
- `redshift_tmp_dir` – répertoire Amazon Redshift temporaire à utiliser (facultatif).
- `transformation_ctx` – Contexte de transformation à utiliser (facultatif).
- `push_down_predicate` – filtre les partitions sans avoir à répertorier ni lire tous les fichiers de votre jeu de données. Pour connaître les sources prises en charge et les limites, consultez Optimisation des lectures avec pushdown dans AWS Glue ETL. Pour de plus amples informations, veuillez consulter Préfiltrage à l'aide des prédicats pushdown.
- `additional_options` — Ensemble de paires nom-valeur facultatives. Les options possibles comprennent celles répertoriées dans Types et options de connexion pour ETL

dans AWS Glue pour Spark, sauf pour endpointUrl, streamName, bootstrap.servers, security.protocol, topicName, classification et delimiter. Est une autre option prise en charge catalogPartitionPredicate:

catalogPartitionPredicate – Vous pouvez passer une expression de catalogue à filtrer en fonction des colonnes d'index. Cela envoie le filtrage du côté serveur. Pour en savoir plus, consultez AWS Glue Indexes de partition. Notez que push_down_predicate et catalogPartitionPredicate utilisent des syntaxes différentes. Le premier utilise la syntaxe standard SQL Spark et le dernier utilise l'analyseur JSQL.

- catalog_id – ID du catalogue (ID du compte) Data Catalog auquel vous accédez. Lorsque la valeur est Aucun, l'ID de compte par défaut de l'appelant est utilisé.

create_dynamic_frame_from_options

create_dynamic_frame_from_options(connection_type, connection_options={}, format=None, format_options={}, transformation_ctx = "")

Renvoie un DynamicFrame créé avec la connexion et le format spécifiés.

- connection_type – type de connexion, tel qu'Amazon S3, Amazon Redshift et JDBC. Les valeurs valides sont les suivantes : s3, mysql, postgresql, redshift, sqlserver, oracle et dynamodb.
- connection_options – options de connexion, telles que les chemins et la table de base de données (facultatif). Pour un connection_type de s3, une liste de chemins Amazon S3 est définie.

```
connection_options = {"paths": ["s3://aws-glue-target/temp"]}
```

Pour les connexions JDBC, plusieurs propriétés doivent être définies. Notez que le nom de base de données doit être inclus dans l'URL. Il peut éventuellement être inclus dans les options de connexion.

> **Warning**
>
> Il n'est pas recommandé de stocker des mots de passe dans votre script. Envisagez de les utiliser boto3 pour les récupérer depuis AWS Secrets Manager le catalogue de données AWS Glue.

```
connection_options = {"url": "jdbc-url/database", "user": "username",
 "password": passwordVariable,"dbtable": "table-name", "redshiftTmpDir": "s3-tempdir-path"}
```

La propriété `dbtable` est le nom de la table JDBC. Pour les magasins de données JDBC qui prennent en charge les schémas dans une base de données, spécifiez `schema.table-name`. Si aucun schéma n'est fourni, c'est le schéma « public » par défaut qui est utilisé.

Pour de plus amples informations, veuillez consulter Types et options de connexion pour ETL dans AWS Glue pour Spark.

- `format` : une spécification de format. Utilisée pour une connexion Amazon S3 ou AWS Glue prenant en charge plusieurs formats. Consultez Options de format pour les entrées et sorties dans AWS Glue pour Spark pour connaître les formats pris en charge.
- `format_options` – options de format pour le format spécifié. Consultez Options de format pour les entrées et sorties dans AWS Glue pour Spark pour connaître les formats pris en charge.
- `transformation_ctx` – Contexte de transformation à utiliser (facultatif).
- `push_down_predicate` – filtre les partitions sans avoir à répertorier ni lire tous les fichiers de votre jeu de données. Pour connaître les sources prises en charge et les limites, consultez Optimisation des lectures avec pushdown dans AWS Glue ETL. Pour plus d'informations, consultez Pre-Filtering Using Pushdown Predicates.

create_sample_dynamic_frame_from_catalog

`create_sample_dynamic_frame_from_catalog(database, table_name, num, redshift_tmp_dir, transformation_ctx = "", push_down_predicate= "", additional_options = {}, sample_options = {}, catalog_id = None)`

Retourne un objet `DynamicFrame` créé à l'aide d'un nom de tableau et de base de données Data Catalog. Le `DynamicFrame` ne contient que les premiers enregistrements `num` provenant d'une source de données.

- `database` – Base de données à partir de laquelle lire.
- `table_name` – Nom de la table à partir de laquelle lire.
- `num` – Nombre maximal d'enregistrements dans la trame dynamique d'échantillon renvoyée.
- `redshift_tmp_dir` – répertoire Amazon Redshift temporaire à utiliser (facultatif).

- `transformation_ctx` – Contexte de transformation à utiliser (facultatif).
- `push_down_predicate` – filtre les partitions sans avoir à répertorier ni lire tous les fichiers de votre jeu de données. Pour plus d'informations, consultez Préfiltrage à l'aide des prédicats pushdown.
- `additional_options` — Ensemble de paires nom-valeur facultatives. Les options possibles comprennent celles répertoriées dans Types et options de connexion pour ETL dans AWS Glue pour Spark, sauf pour `endpointUrl`, `streamName`, `bootstrap.servers`, `security.protocol`, `topicName`, `classification` et `delimiter`.
- `sample_options` — Paramètres permettant de contrôler le comportement d'échantillonnage (facultatif). Paramètres disponibles actuels pour les sources Amazon S3 :
  - `maxSamplePartitions` — Nombre maximal de partitions que l'échantillonnage lira. Valeur par défaut : 10
  - `maxSampleFilesPerPartition` — Nombre maximal de fichiers que l'échantillonnage lira dans une partition. Valeur par défaut : 10.

    Ces paramètres permettent de réduire le temps consommé par la liste des fichiers. Par exemple, supposons que le jeu de données comporte 1 000 partitions et que chaque partition comporte 10 fichiers. Si vous définissez `maxSamplePartitions` = 10, et `maxSampleFilesPerPartition` = 10, au lieu de répertorier les 10 000 fichiers, l'échantillonnage ne listera et ne lira que les 10 premières partitions avec les 10 premiers fichiers de chacun : 10*10 = 100 au total.
- `catalog_id` – ID du catalogue Data Catalog auquel vous accédez (ID du compte Data Catalog). Valeur définie sur `None` par défaut. `None` correspond par défaut à l'ID de catalogue du compte ayant initié l'appel dans le service.

create_sample_dynamic_frame_from_options

`create_sample_dynamic_frame_from_options(connection_type, connection_options={}, num, sample_options={}, format=None, format_options={}, transformation_ctx = "")`

Renvoie un `DynamicFrame` créé avec la connexion et le format spécifiés. Le `DynamicFrame` ne contient que les premiers enregistrements `num` provenant d'une source de données.

- `connection_type` – type de connexion, tel qu'Amazon S3, Amazon Redshift et JDBC. Les valeurs valides sont les suivantes : `s3`, `mysql`, `postgresql`, `redshift`, `sqlserver`, `oracle` et `dynamodb`.
- `connection_options` – options de connexion, telles que les chemins et la table de base de données (facultatif). Pour plus d'informations, consultez Types et options de connexion pour ETL dans AWS Glue pour Spark.
- `num` – Nombre maximal d'enregistrements dans la trame dynamique d'échantillon renvoyée.
- `sample_options` — Paramètres permettant de contrôler le comportement d'échantillonnage (facultatif). Paramètres disponibles actuels pour les sources Amazon S3 :
  - `maxSamplePartitions` — Nombre maximal de partitions que l'échantillonnage lira. Valeur par défaut : 10
  - `maxSampleFilesPerPartition` — Nombre maximal de fichiers que l'échantillonnage lira dans une partition. Valeur par défaut : 10.

    Ces paramètres permettent de réduire le temps consommé par la liste des fichiers. Par exemple, supposons que le jeu de données comporte 1 000 partitions et que chaque partition comporte 10 fichiers. Si vous définissez `maxSamplePartitions` = 10, et `maxSampleFilesPerPartition` = 10, au lieu de répertorier les 10 000 fichiers, l'échantillonnage ne listera et ne lira que les 10 premières partitions avec les 10 premiers fichiers de chacun : 10*10 = 100 au total.
- `format` : une spécification de format. Utilisée pour une connexion Amazon S3 ou AWS Glue prenant en charge plusieurs formats. Consultez Options de format pour les entrées et sorties dans AWS Glue pour Spark pour connaître les formats pris en charge.
- `format_options` – options de format pour le format spécifié. Consultez Options de format pour les entrées et sorties dans AWS Glue pour Spark pour connaître les formats pris en charge.
- `transformation_ctx` – Contexte de transformation à utiliser (facultatif).
- `push_down_predicate` – filtre les partitions sans avoir à répertorier ni lire tous les fichiers de votre jeu de données. Pour plus d'informations, consultez Préfiltrage à l'aide des prédicats pushdown.

add_ingestion_time_columns

`add_ingestion_time_columns(dataFrame, timeGranularity = "")`

Ajoute des colonnes de temps d'ingestion, telles que `ingest_year`, `ingest_month`, `ingest_day`, `ingest_hour`, `ingest_minute` à l'entrée `DataFrame`. Cette fonction est automatiquement générée dans le script généré par AWS Glue lorsque vous spécifiez une table Data Catalog avec Amazon S3 comme cible. Cette fonction met automatiquement à jour la partition avec les colonnes de temps d'ingestion sur la table de sortie. Cela permet aux données de sortie d'être automatiquement partitionnées à l'heure d'ingestion sans nécessiter de colonnes d'heure d'ingestion explicites dans les données d'entrée.

- `dataFrame` – `dataFrame` auquel ajouter les colonnes de temps d'ingestion.
- `timeGranularity` — granularité des colonnes de temps. Les valeurs valides sont « `day` », « `hour` » et « `minute` ». Par exemple, si « `hour` » est transmis à la fonction, les colonnes de temps « `ingest_year` », « `ingest_month` », « `ingest_day` » et « `ingest_hour` » seront ajoutées à l'original « `dataFrame` ».

Renvoie le bloc de données après l'ajout des colonnes de granularité temporelle.

Exemple :

```
dynamic_frame = DynamicFrame.fromDF(glueContext.add_ingestion_time_columns(dataFrame,
 "hour"))
```

create_data_frame_from_catalog

`create_data_frame_from_catalog(database, table_name, transformation_ctx = "", additional_options = {})`

Renvoie un objet `DataFrame` créé à l'aide d'un nom de table et de base de données Data Catalog.

- `database` – base de données Data Catalog à lire.
- `table_name` – nom de la table Data Catalogà partir de laquelle lire.
- `transformation_ctx` – Contexte de transformation à utiliser (facultatif).
- `additional_options` – Ensemble de paires nom-valeur facultatives. Les options possibles incluent celles répertoriées dans Types et options de connexion pour ETL dans AWS Glue pour

Spark pour les sources de streaming, telles que `startingPosition`, `maxFetchTimeInMs` et `startingOffsets`.

- `useSparkDataSource`— Lorsque ce paramètre est défini sur true, force AWS Glue à utiliser l'API native de source de données Spark pour lire le tableau. L'API Spark Data Source prend en charge les formats suivants : AVRO, binaire, CSV, JSON, ORC, Parquet et texte. Dans une table Data Catalog, vous spécifiez le format à l'aide de la propriété `classification`. Pour en savoir plus sur l'API Spark Data Source, consultez la documentation officielle d'Apache Spark.

  L'utilisation de `create_data_frame_from_catalog` avec `useSparkDataSource` offre les avantages suivants :

  - Renvoie directement un `DataFrame` et fournit une alternative à `create_dynamic_frame.from_catalog().toDF()`.
  - Prend en charge le contrôle des autorisations AWS Lake Formation au niveau de la table pour les formats natifs.
  - Prend en charge la lecture des formats de lacs de données sans contrôle des autorisations AWS Lake Formation au niveau de la table. Pour de plus amples informations, veuillez consulter Utilisation de frameworks de lacs de données avec des tâches AWS Glue ETL.

  Lorsque vous l'activez`useSparkDataSource`, vous pouvez également ajouter n'importe laquelle des options de source de données Spark `additional_options` selon vos besoins. AWS Glue transmet ces options directement au lecteur Spark.

- `useCatalogSchema`— Lorsqu'il est défini sur true, AWS Glue applique le schéma du catalogue de données au résultat`DataFrame`. Sinon, le lecteur déduit le schéma à partir des données. Lorsque vous activez `useCatalogSchema`, vous devez également définir `useSparkDataSource` sur true.

Limites

Lorsque vous utilisez l'option `useSparkDataSource`, tenez compte des restrictions suivantes :

- Lorsque vous l'utilisez`useSparkDataSource`, AWS Glue en crée une nouvelle `DataFrame` dans une session Spark distincte, différente de la session Spark d'origine.
- Le filtrage des DataFrame partitions Spark ne fonctionne pas avec les fonctionnalités de AWS Glue suivantes.
  - Signets de tâche
  - Exclusion des classes de stockage Amazon S3

- Prédicats de partition de catalogue

Pour utiliser le filtrage des partitions avec ces fonctionnalités, vous pouvez utiliser le prédicat AWS Glue pushdown. Pour de plus amples informations, veuillez consulter Préfiltrage à l'aide des prédicats pushdown. Le filtrage sur les colonnes non partitionnées n'est pas concerné.

L'exemple de script suivant illustre un filtrage de partition inadéquat à l'aide de l'option `excludeStorageClasses`.

```
// Incorrect partition filtering using Spark filter with excludeStorageClasses
read_df = glueContext.create_data_frame.from_catalog(
    database=database_name,
    table_name=table_name,
    additional_options = {
      "useSparkDataSource": True,
      "excludeStorageClasses" : ["GLACIER", "DEEP_ARCHIVE"]
    }
)

//  Suppose year and month are partition keys.
//  Filtering on year and month won't work, the filtered_df will still
//  contain data with other year/month values.
filtered_df = read_df.filter("year == '2017 and month == '04' and 'state == 'CA'")
```

L'exemple de script suivant illustre l'utilisation adéquate d'un prédicat pushdown dans le cadre d'un filtrage de partition à l'aide de l'option `excludeStorageClasses`.

```
// Correct partition filtering using the AWS Glue pushdown predicate
// with excludeStorageClasses
read_df = glueContext.create_data_frame.from_catalog(
    database=database_name,
    table_name=table_name,
    //  Use AWS Glue pushdown predicate to perform partition filtering
    push_down_predicate = "(year=='2017' and month=='04')"
    additional_options = {
      "useSparkDataSource": True,
      "excludeStorageClasses" : ["GLACIER", "DEEP_ARCHIVE"]
    }
)

//  Use Spark filter only on non-partitioned columns
```

```
filtered_df = read_df.filter("state == 'CA'")
```

Exemple : Création d'une table CSV à l'aide du lecteur de source de données Spark

```
//  Read a CSV table with '\t' as separator
read_df = glueContext.create_data_frame.from_catalog(
    database=<database_name>,
    table_name=<table_name>,
    additional_options = {"useSparkDataSource": True,  "sep": '\t'}
)
```

create_data_frame_from_options

`create_data_frame_from_options(connection_type, connection_options={}, format=None, format_options={}, transformation_ctx = "")`

Cette API est désormais obsolète. Utilisez plutôt les API `getSource()`. Renvoie un `DataFrame` créé avec la connexion et le format spécifiés. Utilisez cette fonction uniquement avec les sources de streaming AWS Glue.

- `connection_type` – type de connexion en streaming. Les valeurs valides sont `kinesis` et `kafka`.
- `connection_options` – options de connexion, qui sont différentes pour Kinesis et Kafka. Vous trouverez la liste de toutes les options de connexion pour chaque source de données de streaming sur la page Types et options de connexion pour ETL dans AWS Glue pour Spark. Notez les différences suivantes dans les options de connexion en streaming :
  - Les sources de streaming Kinesis nécessitent `streamARN`, `startingPosition`, `inferSchema` et `classification`.
  - Les sources de streaming Kafka nécessitent `connectionName`, `topicName`, `startingOffsets`, `inferSchema` et `classification`.
- `format` : une spécification de format. Utilisée pour une connexion Amazon S3 ou AWS Glue prenant en charge plusieurs formats. Pour plus d'informations sur les formats pris en charge, consultez Options de format pour les entrées et sorties dans AWS Glue pour Spark.
- `format_options` – options de format pour le format spécifié. Pour de plus amples informations sur les options de formats pris en charge, veuillez consulter Options de format pour les entrées et sorties dans AWS Glue pour Spark.
- `transformation_ctx` – Contexte de transformation à utiliser (facultatif).

Exemple pour la source de streaming Amazon Kinesis :

```
kinesis_options =
   { "streamARN": "arn:aws:kinesis:us-east-2:777788889999:stream/fromOptionsStream",
     "startingPosition": "TRIM_HORIZON",
     "inferSchema": "true",
     "classification": "json"
   }
data_frame_datasource0 =
 glueContext.create_data_frame.from_options(connection_type="kinesis",
 connection_options=kinesis_options)
```

Exemple pour la source de streaming Kafka :

```
kafka_options =
    { "connectionName": "ConfluentKafka",
      "topicName": "kafka-auth-topic",
      "startingOffsets": "earliest",
      "inferSchema": "true",
      "classification": "json"
    }
data_frame_datasource0 =
 glueContext.create_data_frame.from_options(connection_type="kafka",
 connection_options=kafka_options)
```

forEachBatch

`forEachBatch(frame, batch_function, options)`

S'applique à `batch_function` transmis à chaque micro-lot lu à partir de la source de streaming.

- `frame`— Le DataFrame contenant le microlot actuel.
- `batch_function` — fonction qui sera appliquée à chaque micro-lot.
- `options` — collection de paires clé-valeur qui contient des informations sur le traitement de micro-lots. Les options suivantes sont requises :
  - `windowSize` — durée de traitement de chaque lot.
  - `checkpointLocation` — emplacement dans lequel les points de contrôle sont stockés pour la tâche ETL en streaming.
  - `batchMaxRetries` – nombre maximum de nouvelles tentatives pour ce lot en cas d'échec. La valeur par défaut est 3. Cette option n'est configurable que pour Glue version 2.0 et ultérieure.

Exemple :

```
glueContext.forEachBatch(
    frame = data_frame_datasource0,
    batch_function = processBatch,
    options = {
        "windowSize": "100 seconds",
        "checkpointLocation": "s3://kafka-auth-dataplane/confluent-test/output/
checkpoint/"
    }
)

def processBatch(data_frame, batchId):
    if (data_frame.count() > 0):
        datasource0 = DynamicFrame.fromDF(
          glueContext.add_ingestion_time_columns(data_frame, "hour"),
          glueContext, "from_data_frame"
        )
        additionalOptions_datasink1 = {"enableUpdateCatalog": True}
        additionalOptions_datasink1["partitionKeys"] = ["ingest_yr", "ingest_mo",
 "ingest_day"]
        datasink1 = glueContext.write_dynamic_frame.from_catalog(
          frame = datasource0,
          database = "tempdb",
          table_name = "kafka-auth-table-output",
          transformation_ctx = "datasink1",
          additional_options = additionalOptions_datasink1
        )
```

Utilisation des jeux de données dans Amazon S3

- purge_table
- purge_s3_path
- transition_table
- transition_s3_path

purge_table

`purge_table(catalog_id=None, database="", table_name="", options={}, transformation_ctx="")`

Supprime les fichiers d'Amazon S3 pour la base de données et la table spécifiés pour le catalogue. Si tous les fichiers d'une partition sont supprimés, cette partition est également éliminée du catalogue. Nous ne prenons pas en charge l'action purge_table sur les tables enregistrées auprès de Lake Formation.

Pour pouvoir récupérer des objets supprimés, vous pouvez activer la gestion des versions d'objet au niveau du compartiment Amazon S3. Lorsqu'un objet est supprimé d'un compartiment pour lequel la gestion des versions d'objet n'est pas activée, il ne peut pas être récupéré. Pour plus d'informations sur la récupération d'objets supprimés dans un compartiment pour lequel la gestion des versions est activée, consultez Comment récupérer un objet Amazon S3 qui a été supprimé ? dans le Centre de connaissances AWS Support .

- `catalog_id` – ID du catalogue Data Catalog auquel vous accédez (ID du compte Data Catalog). Valeur définie sur `None` par défaut. `None` correspond par défaut à l'ID de catalogue du compte ayant initié l'appel dans le service.
- `database` – Base de données à utiliser.
- `table_name` – nom de la table à utiliser.
- `options` – options destinées au filtrage des fichiers à supprimer et à la génération des fichiers manifestes.
  - `retentionPeriod` – spécifie une période de conservation des fichiers, en nombre d'heures. Les dossiers dont la date est postérieure à cette période sont conservés. Par défaut, elle est réglée sur 168 heures (7 jours).
  - `partitionPredicate` – les partitions satisfaisant à ce prédicat sont supprimées. Les fichiers qui se situent dans la période de conservation pour ces partitions ne sont pas supprimés. Valeur définie sur `""` – vide par défaut.
  - `excludeStorageClasses` – les fichiers avec classe de stockage dans le jeu `excludeStorageClasses` ne sont pas supprimés. La valeur par défaut est `Set()` – jeu vide.
  - `manifestFilePath` – chemin facultatif pour la génération du fichier manifeste. Tous les fichiers qui ont été purgés avec succès sont enregistrés dans `Success.csv`, et ceux qui ont échoué dans `Failed.csv`

- `transformation_ctx` – Contexte de transformation à utiliser (facultatif). Utilisé dans le chemin du fichier manifeste.

Example

```
glueContext.purge_table("database", "table", {"partitionPredicate": "(month=='march')",
 "retentionPeriod": 1, "excludeStorageClasses": ["STANDARD_IA"], "manifestFilePath":
 "s3://bucketmanifest/"})
```

purge_s3_path

`purge_s3_path(s3_path, options={}, transformation_ctx="")`

Supprime les fichiers du chemin Amazon S3 spécifié de manière récursive.

Pour pouvoir récupérer des objets supprimés, vous pouvez activer la gestion des versions d'objet au niveau du compartiment Amazon S3. Lorsqu'un objet est supprimé d'un compartiment pour lequel la gestion des versions d'objet n'est pas activée, il ne peut pas être récupéré. Pour plus d'informations sur la façon de récupérer des objets supprimés dans un compartiment avec gestion des versions, consultez Comment puis-je récupérer un objet Amazon S3 qui a été supprimé ? dans le Support Knowledge Center.

- `s3_path` – chemin Amazon S3 des fichiers à supprimer dans le format `s3://<bucket>/<prefix>/`
- `options` – options destinées au filtrage des fichiers à supprimer et à la génération des fichiers manifestes.
  - `retentionPeriod` – spécifie une période de conservation des fichiers, en nombre d'heures. Les dossiers dont la date est postérieure à cette période sont conservés. Par défaut, elle est réglée sur 168 heures (7 jours).
  - `excludeStorageClasses` – les fichiers avec classe de stockage dans le jeu `excludeStorageClasses` ne sont pas supprimés. La valeur par défaut est `Set()` – jeu vide.
  - `manifestFilePath` – chemin facultatif pour la génération du fichier manifeste. Tous les fichiers qui ont été purgés avec succès sont enregistrés dans `Success.csv`, et ceux qui ont échoué dans `Failed.csv`
- `transformation_ctx` – Contexte de transformation à utiliser (facultatif). Utilisé dans le chemin du fichier manifeste.

Example

```
glueContext.purge_s3_path("s3://bucket/path/", {"retentionPeriod": 1,
 "excludeStorageClasses": ["STANDARD_IA"], "manifestFilePath": "s3://bucketmanifest/"})
```

transition_table

`transition_table(database, table_name, transition_to, options={}, transformation_ctx="", catalog_id=None)`

Transition de la classe de stockage des fichiers stockés sur Amazon S3 pour la base de données et la table spécifiés pour le catalogue.

Vous pouvez effectuer une transition entre deux classes de stockage. Pour les classes de stockage `GLACIER` et `DEEP_ARCHIVE`, vous pouvez effectuer la transition vers ces classes. Cependant, vous devez utiliser `S3 RESTORE` pour effectuer la transition des classes de stockage `GLACIER` et `DEEP_ARCHIVE`.

Si vous exécutez des tâches ETL AWS Glue qui lisent des fichiers ou des partitions à partir d'Amazon S3, vous pouvez exclure certains types de classe de stockage Amazon S3. Pour plus d'informations, consultez Exclusion des classes de stockage Amazon S3.

- `database` – Base de données à utiliser.
- `table_name` – nom de la table à utiliser.
- `transition_to` – classe de stockage Amazon S3 vers laquelle effectuer la transition.
- `options` – options destinées au filtrage des fichiers à supprimer et à la génération des fichiers manifestes.
  - `retentionPeriod` – spécifie une période de conservation des fichiers, en nombre d'heures. Les dossiers dont la date est postérieure à cette période sont conservés. Par défaut, elle est réglée sur 168 heures (7 jours).
  - `partitionPredicate` – une transition est effectuée pour les partitions satisfaisant à ce prédicat. Aucune transition n'a lieu pour les fichiers qui se trouvent dans la période de conservation. Valeur définie sur `""` – vide par défaut.
  - `excludeStorageClasses` – aucune transition n'a lieu pour les fichiers avec classe de stockage dans le jeu `excludeStorageClasses`. La valeur par défaut est `Set()` – jeu vide.
  - `manifestFilePath` – chemin facultatif pour la génération du fichier manifeste. Tous les fichiers dont la transition a été effectuée avec succès sont enregistrés dans `Success.csv`, et ceux qui ont échoué dans `Failed.csv`

  - `accountId` – ID de compte Amazon Web Services permettant d'exécuter la transformation de la transition. Obligatoire pour cette transformation.
  - `roleArn`— Le AWS rôle d'exécuter la transformation de transition. Obligatoire pour cette transformation.
- `transformation_ctx` – Contexte de transformation à utiliser (facultatif). Utilisé dans le chemin du fichier manifeste.
- `catalog_id` – ID du catalogue Data Catalog auquel vous accédez (ID du compte Data Catalog). Valeur définie sur None par défaut. None correspond par défaut à l'ID de catalogue du compte ayant initié l'appel dans le service.

Example

```
glueContext.transition_table("database", "table", "STANDARD_IA", {"retentionPeriod": 1,
 "excludeStorageClasses": ["STANDARD_IA"], "manifestFilePath": "s3://bucketmanifest/",
 "accountId": "12345678901", "roleArn": "arn:aws:iam::123456789012:user/example-
username"})
```

transition_s3_path

`transition_s3_path(s3_path, transition_to, options={}, transformation_ctx="")`

Transition récursive de la classe de stockage des fichiers dans le chemin Amazon S3 spécifié.

Vous pouvez effectuer une transition entre deux classes de stockage. Pour les classes de stockage `GLACIER` et `DEEP_ARCHIVE`, vous pouvez effectuer la transition vers ces classes. Cependant, vous devez utiliser `S3 RESTORE` pour effectuer la transition des classes de stockage `GLACIER` et `DEEP_ARCHIVE`.

Si vous exécutez des tâches ETL AWS Glue qui lisent des fichiers ou des partitions à partir d'Amazon S3, vous pouvez exclure certains types de classe de stockage Amazon S3. Pour plus d'informations, consultez Exclusion des classes de stockage Amazon S3.

- `s3_path` – chemin Amazon S3 des fichiers à passer au format `s3://<bucket>/<prefix>/`
- `transition_to` – classe de stockage Amazon S3 vers laquelle effectuer la transition.
- `options` – options destinées au filtrage des fichiers à supprimer et à la génération des fichiers manifestes.

- `retentionPeriod` – spécifie une période de conservation des fichiers, en nombre d'heures. Les dossiers dont la date est postérieure à cette période sont conservés. Par défaut, elle est réglée sur 168 heures (7 jours).
- `partitionPredicate` – une transition est effectuée pour les partitions satisfaisant à ce prédicat. Aucune transition n'a lieu pour les fichiers qui se trouvent dans la période de conservation. Valeur définie sur `""` – vide par défaut.
- `excludeStorageClasses` – aucune transition n'a lieu pour les fichiers avec classe de stockage dans le jeu `excludeStorageClasses`. La valeur par défaut est `Set()` – jeu vide.
- `manifestFilePath` – chemin facultatif pour la génération du fichier manifeste. Tous les fichiers dont la transition a été effectuée avec succès sont enregistrés dans `Success.csv`, et ceux qui ont échoué dans `Failed.csv`
- `accountId` – ID de compte Amazon Web Services permettant d'exécuter la transformation de la transition. Obligatoire pour cette transformation.
- `roleArn`— Le AWS rôle d'exécuter la transformation de transition. Obligatoire pour cette transformation.

- `transformation_ctx` – Contexte de transformation à utiliser (facultatif). Utilisé dans le chemin du fichier manifeste.

Example

```
glueContext.transition_s3_path("s3://bucket/prefix/", "STANDARD_IA",
 {"retentionPeriod": 1, "excludeStorageClasses": ["STANDARD_IA"],
 "manifestFilePath": "s3://bucketmanifest/", "accountId": "12345678901", "roleArn":
 "arn:aws:iam::123456789012:user/example-username"})
```

Extraction

- extract_jdbc_conf

extract_jdbc_conf

`extract_jdbc_conf(connection_name, catalog_id = None)`

Retourne un `dict` dont les clés contiennent les propriétés de configuration de l'objet de connexion AWS Glue dans le catalogue de données.

- `user` – Nom d'utilisateur de la base de données.

- `password` – Le mot de passe de la base de données.
- `vendor` – Spécifie un fournisseur (`mysql`, `postgresql`, `oracle`, `sqlserver`, etc.).
- `enforceSSL` – Chaîne booléenne indiquant si une connexion sécurisée est requise.
- `customJDBCCert` – Utilise un certificat client spécifique à partir du chemin Amazon S3 indiqué.
- `skipCustomJDBCCertValidation` – Chaîne booléenne indiquant si le `customJDBCCert` doit être validé par une autorité de certification.
- `customJDBCCertString` – Informations supplémentaires sur le certificat personnalisé, spécifique au type de pilote.
- `url` – (Obsolète) URL JDBC avec uniquement le protocole, le serveur et le port.
- `fullUrl` – URL JDBC telle qu'elle a été saisie lors de la création de la connexion (disponible dans la version AWS Glue 3.0 ou ultérieure).

Exemple de récupération de configurations JDBC :

```
jdbc_conf = glueContext.extract_jdbc_conf(connection_name="your_glue_connection_name")
print(jdbc_conf)
>>> {'enforceSSL': 'false', 'skipCustomJDBCCertValidation': 'false', 'url':
 'jdbc:mysql://myserver:3306', 'fullUrl': 'jdbc:mysql://myserver:3306/mydb',
 'customJDBCCertString': '', 'user': 'admin', 'customJDBCCert': '', 'password': '1234',
 'vendor': 'mysql'}
```

Transactions

- start_transaction
- commit_transaction
- cancel_transaction

start_transaction

`start_transaction(read_only)`

Démarrez une nouvelle transaction. Appelle en interne l'API Démarrer la transaction Lake Formation.

- `read_only` – Valeur booléenne indiquant si cette transaction doit être en lecture seule ou en lecture et en écriture. Les écritures effectuées à l'aide d'un ID de transaction en lecture seule seront rejetées. Les transactions en lecture seule n'ont pas besoin d'être validées.

Retourne l'ID de transaction.

commit_transaction

`commit_transaction(transaction_id, wait_for_commit = True)`

Tentative de validation de la transaction spécifiée.`commit_transaction` peut être renvoyé avant la fin de la validation de la transaction. Appelle en interne la Lake Formation commitTransaction API.

- `transaction_id` – (Chaîne) La transaction à valider.
- `wait_for_commit` – (Booléen) Détermine si le `commit_transaction` retourne immédiatement. La valeur par défaut est True. Si elle est false,`commit_transaction` interroge et attend que la transaction soit validée. Le temps d'attente est limité à 1 minute en utilisant le backoff exponentiel avec un maximum de 6 tentatives de nouvelle tentative.

Renvoie une valeur de type Boolean pour indiquer si la validation est effectuée ou non.

cancel_transaction

`cancel_transaction(transaction_id)`

Tentative d'annulation de la transaction spécifiée. Retourne une exception `TransactionCommittedException` si la transaction a déjà été validée. Appelle en interne l'CancelTransactionAPI Lake Formation.

- `transaction_id` – (Chaîne) La transaction à annuler.

Ecriture

- getSink
- write_dynamic_frame_from_options
- write_from_options
- write_dynamic_frame_from_catalog
- write_data_frame_from_catalog
- write_dynamic_frame_from_jdbc_conf
- write_from_jdbc_conf

getSink

`getSink(connection_type, format = None, transformation_ctx = "", **options)`

Obtient un objet `DataSink` qui peut être utilisé pour écrire `DynamicFrames` sur des sources externes. Vérifiez d'abord le `format` SparkSQL pour être sûr d'obtenir le récepteur escompté.

- `connection_type` – type de connexion à utiliser, telle qu'Amazon S3, Amazon Redshift et JDBC. Les valeurs valides sont les suivantes : `s3`, `mysql`, `postgresql`, `redshift`, `sqlserver`, `oracle`, `kinesis` et `kafka`.
- `format` – Format SparkSQL à utiliser (facultatif).
- `transformation_ctx` – Contexte de transformation à utiliser (facultatif).
- `options` –Ensemble de paires nom-valeur utilisées pour spécifier les options de connexion. Certaines des valeurs possibles sont les suivantes :
  - `user` et `password` : Pour autorisation
  - `url` : Point de terminaison du magasin de données
  - `dbtable` :Nom de la table cible
  - `bulkSize`: Degré de parallélisme pour les opérations d'insertion

Les options que vous pouvez spécifier dépendent du type de connexion. Pour obtenir d'autres conseils et d'autres exemples, veuillez consulter Types et options de connexion pour ETL dans AWS Glue pour Spark.

Exemple :

```
>>> data_sink = context.getSink("s3")
>>> data_sink.setFormat("json"),
>>> data_sink.writeFrame(myFrame)
```

write_dynamic_frame_from_options

`write_dynamic_frame_from_options(frame, connection_type, connection_options={}, format=None, format_options={}, transformation_ctx = "")`

Écrit et retourne un `DynamicFrame` à l'aide de la connexion et du format spécifiés.

- `frame` – Objet `DynamicFrame` à écrire.

- `connection_type` – type de connexion, tel qu'Amazon S3, Amazon Redshift et JDBC. Les valeurs valides sont les suivantes : `s3`, `mysql`, `postgresql`, `redshift`, `sqlserver`, `oracle`, `kinesis` et `kafka`.
- `connection_options` – Options de connexion, telles que le chemin et la table de base de données (facultatif). Pour un `connection_type` de `s3`, un chemin Amazon S3 est défini.

```
connection_options = {"path": "s3://aws-glue-target/temp"}
```

Pour les connexions JDBC, plusieurs propriétés doivent être définies. Notez que le nom de base de données doit être inclus dans l'URL. Il peut éventuellement être inclus dans les options de connexion.

> **Warning**
>
> Il n'est pas recommandé de stocker des mots de passe dans votre script. Envisagez de les utiliser `boto3` pour les récupérer depuis AWS Secrets Manager le catalogue de données AWS Glue.

```
connection_options = {"url": "jdbc-url/database", "user": "username",
 "password": passwordVariable,"dbtable": "table-name", "redshiftTmpDir": "s3-tempdir-
path"}
```

La propriété `dbtable` est le nom de la table JDBC. Pour les magasins de données JDBC qui prennent en charge les schémas dans une base de données, spécifiez `schema.table-name`. Si aucun schéma n'est fourni, c'est le schéma « public » par défaut qui est utilisé.

Pour de plus amples informations, veuillez consulter Types et options de connexion pour ETL dans AWS Glue pour Spark.

- `format` : une spécification de format. Utilisée pour une connexion Amazon S3 ou AWS Glue prenant en charge plusieurs formats. Consultez Options de format pour les entrées et sorties dans AWS Glue pour Spark pour connaître les formats pris en charge.
- `format_options` – options de format pour le format spécifié. Consultez Options de format pour les entrées et sorties dans AWS Glue pour Spark pour connaître les formats pris en charge.
- `transformation_ctx` – Contexte de transformation à utiliser (facultatif).

write_from_options

`write_from_options(frame_or_dfc, connection_type, connection_options={}, format={}, format_options={}, transformation_ctx = "")`

Écrit et renvoie un `DynamicFrame` ou `DynamicFrameCollection` qui est créé avec la connexion et les informations de format spécifiées.

- `frame_or_dfc` – Objet `DynamicFrame` ou `DynamicFrameCollection` à écrire.
- `connection_type` – type de connexion, tel qu'Amazon S3, Amazon Redshift et JDBC. Les valeurs valides sont `s3`, `mysql`, `postgresql`, `redshift`, `sqlserver` et `oracle`.
- `connection_options` – Options de connexion, telles que le chemin et la table de base de données (facultatif). Pour un `connection_type` de `s3`, un chemin Amazon S3 est défini.

```
connection_options = {"path": "s3://aws-glue-target/temp"}
```

Pour les connexions JDBC, plusieurs propriétés doivent être définies. Notez que le nom de base de données doit être inclus dans l'URL. Il peut éventuellement être inclus dans les options de connexion.

> **Warning**
>
> Il n'est pas recommandé de stocker des mots de passe dans votre script. Envisagez de les utiliser `boto3` pour les récupérer depuis AWS Secrets Manager le catalogue de données AWS Glue.

```
connection_options = {"url": "jdbc-url/database", "user": "username",
 "password": passwordVariable,"dbtable": "table-name", "redshiftTmpDir": "s3-tempdir-path"}
```

La propriété `dbtable` est le nom de la table JDBC. Pour les magasins de données JDBC qui prennent en charge les schémas dans une base de données, spécifiez `schema.table-name`. Si aucun schéma n'est fourni, c'est le schéma « public » par défaut qui est utilisé.

Pour de plus amples informations, veuillez consulter Types et options de connexion pour ETL dans AWS Glue pour Spark.

- `format` : une spécification de format. Utilisée pour une connexion Amazon S3 ou AWS Glue prenant en charge plusieurs formats. Consultez Options de format pour les entrées et sorties dans AWS Glue pour Spark pour connaître les formats pris en charge.
- `format_options` – options de format pour le format spécifié. Consultez Options de format pour les entrées et sorties dans AWS Glue pour Spark pour connaître les formats pris en charge.
- `transformation_ctx` – Contexte de transformation à utiliser (facultatif).

write_dynamic_frame_from_catalog

`write_dynamic_frame_from_catalog(frame, database, table_name, redshift_tmp_dir, transformation_ctx = "", additional_options = {}, catalog_id = None)`

Écrit et renvoie un objet `DynamicFrame` utilisant des informations provenant d'une table et d'une base de données Data Catalog.

- `frame` – Objet `DynamicFrame` à écrire.
- `Database` – base de données Data Catalog contenant la table.
- `table_name` – nom de la table Data Catalog associée à la cible.
- `redshift_tmp_dir` – répertoire Amazon Redshift temporaire à utiliser (facultatif).
- `transformation_ctx` – Contexte de transformation à utiliser (facultatif).
- `additional_options` – Ensemble de paires nom-valeur facultatives.
- `catalog_id` – ID du catalogue (ID du compte) Data Catalog auquel vous accédez. Lorsque la valeur est Aucun, l'ID de compte par défaut de l'appelant est utilisé.

write_data_frame_from_catalog

`write_data_frame_from_catalog(frame, database, table_name, redshift_tmp_dir, transformation_ctx = "", additional_options = {}, catalog_id = None)`

Écrit et renvoie un objet `DataFrame` utilisant des informations provenant d'une table et d'une base de données Data Catalog. Cette méthode prend en charge l'écriture dans les formats de lac de données (Hudi, Iceberg et Delta Lake). Pour plus d'informations, consultez Utilisation de frameworks de lacs de données avec des tâches AWS Glue ETL.

- `frame` – Objet `DataFrame` à écrire.
- `Database` – base de données Data Catalog contenant la table.
- `table_name` – nom de la table Data Catalog qui est associée à la cible.
- `redshift_tmp_dir` – répertoire Amazon Redshift temporaire à utiliser (facultatif).
- `transformation_ctx` – Contexte de transformation à utiliser (facultatif).
- `additional_options` – Ensemble de paires nom-valeur facultatives.
  - `useSparkDataSink`— Lorsque ce paramètre est défini sur true, force AWS Glue à utiliser l'API native de Spark Data Sink pour écrire dans la table. Lorsque vous activez cette option, vous pouvez ajouter toutes les options de source de données Spark selon `additional_options` vos besoins. AWS Glue transmet ces options directement au Spark Writer.
- `catalog_id` – ID de catalogue (ID de compte) du catalogue de données auquel vous accédez. Si vous ne spécifiez aucune valeur, l'ID de compte par défaut de l'appelant est utilisé.

Limites

Lorsque vous utilisez l'option `useSparkDataSink`, tenez compte des restrictions suivantes :

- L'option `enableUpdateCatalog` n'est pas prise en charge lorsque vous utilisez l'option `useSparkDataSink`.

Exemple : Écriture dans une table Hudi à l'aide de l'enregistreur de source de données Spark

```
hudi_options = {
    'useSparkDataSink': True,
    'hoodie.table.name': <table_name>,
    'hoodie.datasource.write.storage.type': 'COPY_ON_WRITE',
    'hoodie.datasource.write.recordkey.field': 'product_id',
    'hoodie.datasource.write.table.name': <table_name>,
    'hoodie.datasource.write.operation': 'upsert',
    'hoodie.datasource.write.precombine.field': 'updated_at',
    'hoodie.datasource.write.hive_style_partitioning': 'true',
    'hoodie.upsert.shuffle.parallelism': 2,
    'hoodie.insert.shuffle.parallelism': 2,
    'hoodie.datasource.hive_sync.enable': 'true',
    'hoodie.datasource.hive_sync.database': <database_name>,
    'hoodie.datasource.hive_sync.table': <table_name>,
    'hoodie.datasource.hive_sync.use_jdbc': 'false',
```

```
    'hoodie.datasource.hive_sync.mode': 'hms'}

glueContext.write_data_frame.from_catalog(
    frame = <df_product_inserts>,
    database = <database_name>,
    table_name = <table_name>,
    additional_options = hudi_options
)
```

write_dynamic_frame_from_jdbc_conf

`write_dynamic_frame_from_jdbc_conf(frame, catalog_connection, connection_options={}, redshift_tmp_dir = "", transformation_ctx = "", catalog_id = None)`

Écrit et retourne un `DynamicFrame` à l'aide des informations de connexion JDBC spécifiées.

- `frame` – Objet `DynamicFrame` à écrire.
- `catalog_connection` – Connexion de catalogue à utiliser.
- `connection_options` – Options de connexion, telles que le chemin et la table de base de données (facultatif). Pour plus d'informations, consultez Types et options de connexion pour ETL dans AWS Glue pour Spark.
- `redshift_tmp_dir` – répertoire Amazon Redshift temporaire à utiliser (facultatif).
- `transformation_ctx` – Contexte de transformation à utiliser (facultatif).
- `catalog_id` – ID du catalogue (ID du compte) Data Catalog auquel vous accédez. Lorsque la valeur est Aucun, l'ID de compte par défaut de l'appelant est utilisé.

write_from_jdbc_conf

`write_from_jdbc_conf(frame_or_dfc, catalog_connection, connection_options={}, redshift_tmp_dir = "", transformation_ctx = "", catalog_id = None)`

Écrit et retourne un objet `DynamicFrame` ou `DynamicFrameCollection` à l'aide des informations de connexion JDBC spécifiées.

- `frame_or_dfc` – Objet `DynamicFrame` ou `DynamicFrameCollection` à écrire.
- `catalog_connection` – Connexion de catalogue à utiliser.

- `connection_options` – Options de connexion, telles que le chemin et la table de base de données (facultatif). Pour plus d'informations, consultez Types et options de connexion pour ETL dans AWS Glue pour Spark.
- `redshift_tmp_dir` – répertoire Amazon Redshift temporaire à utiliser (facultatif).
- `transformation_ctx` – Contexte de transformation à utiliser (facultatif).
- `catalog_id` – ID du catalogue (ID du compte) Data Catalog auquel vous accédez. Lorsque la valeur est Aucun, l'ID de compte par défaut de l'appelant est utilisé.

AWS Glue PySpark transforme la référence

AWS Glue fournit les transformations intégrées suivantes que vous pouvez utiliser dans les opérations PySpark ETL. Vos données passent d'une transformation à l'autre dans une structure de données appelée a DynamicFrame, qui est une extension d'un SQL Apache Spark`DataFrame`. Le `DynamicFrame` contient vos données, et vous référencez son schéma pour traiter vos données.

La plupart de ces transformations existent également en tant que méthodes de la classe `DynamicFrame`. Pour plus d'informations, consultez la section DynamicFrame transformations.

- GlueTransform classe de base
- ApplyMapping classe
- DropFields classe
- DropNullFields classe
- ErrorsAsDynamicFrame classe
- EvaluateDataQuality classe
- FillMissingValues classe
- Classe Filter
- FindIncrementalMatches classe
- FindMatches classe
- FlatMap classe
- Classe Join
- Classe Map
- MapToCollection classe
- mergeDynamicFrame

- Classe Relationalize
- RenameField classe
- ResolveChoice classe
- SelectFields classe
- SelectFromCollection classe
- Classe Simplify_ddb_json
- Classe Spigot
- SplitFields classe
- SplitRows classe
- Classe Unbox
- UnnestFrame classe

GlueTransform classe de base

Classe de base dont toutes les classes `awsglue.transforms` héritent.

Les classes définissent toutes une méthode `__call__`. Elles remplacent les méthodes de la classe `GlueTransform` répertoriées dans les sections suivantes ou sont appelées en utilisant le nom de classe par défaut.

Méthodes

- apply(cls, *args, **kwargs)
- name(cls)
- describeArgs(cls)
- describeReturn(cls)
- describeTransform(cls)
- describeErrors(cls)
- describe(cls)

apply(cls, *args, **kwargs)

Applique la transformation en appelant la classe de transformation et renvoie le résultat.

- `cls` – Objet de classe `self`.

name(cls)

Retourne le nom de la classe de transformation dérivée.

- `cls` – Objet de classe `self`.

describeArgs(cls)

- `cls` – Objet de classe `self`.

Renvoie une liste de dictionnaires, chacun correspondant à un argument nommé, dans le format suivant :

```
[
  {
    "name": "(name of argument)",
    "type": "(type of argument)",
    "description": "(description of argument)",
    "optional": "(Boolean, True if the argument is optional)",
    "defaultValue": "(Default value string, or None)(String; the default value, or
 None)"
  },
...
]
```

Lève une exception `NotImplementedError` en cas d'appel dans une transformation dérivée où elle n'est pas implémentée.

describeReturn(cls)

- `cls` – Objet de classe `self`.

Renvoie un dictionnaire avec les informations sur le type de retour, dans le format suivant :

```
{
  "type": "(return type)",
  "description": "(description of output)"
```

```
}
```

Lève une exception `NotImplementedError` en cas d'appel dans une transformation dérivée où elle n'est pas implémentée.

describeTransform(cls)

Renvoie une chaîne décrivant la transformation.

- `cls` – Objet de classe `self`.

Lève une exception `NotImplementedError` en cas d'appel dans une transformation dérivée où elle n'est pas implémentée.

describeErrors(cls)

- `cls` – Objet de classe `self`.

Renvoie une liste de dictionnaires, chacun décrivant une exception éventuelle lancée par cette transformation, dans le format suivant :

```
[
  {
    "type": "(type of error)",
    "description": "(description of error)"
  },
...
]
```

describe(cls)

- `cls` – Objet de classe `self`.

Renvoie un objet dans le format suivant :

```
{
  "transform" : {
    "name" : cls.name( ),
    "args" : cls.describeArgs( ),
    "returns" : cls.describeReturn( ),
```

```
    "raises" : cls.describeErrors( ),
    "location" : "internal"
  }
}
```

ApplyMapping classe

Applique un mappage dans un `DynamicFrame`.

Exemple

Nous vous recommandons d'utiliser le`DynamicFrame.apply_mapping()`méthode pour appliquer un mappage dans un`DynamicFrame`. Vous trouverez un exemple de code, consultez Exemple : Utilisez apply_mapping afin de renommer des champs ainsi que de modifier ses types.

Méthodes

- __call__
- s'appliquent
- name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Décrire

__call__(frame, mappings, transformation_ctx = "", info = "", stageThreshold = 0, totalThreshold = 0)

Applique un mappage déclaratif à un `DynamicFrame` spécifié.

- `frame` – le `DynamicFrame` appliquer le mappage (obligatoire).
- `mappings`— Une liste de tuples de mappage (obligatoire). Chacun étant composé comme suit : (colonne source, type source, colonne cible, type cible).

  Si la colonne source comporte un point »`.`« dans le nom, vous devez cocher une case arrière »``« autour de lui. Par exemple, pour mapper `this.old.name` (chaîne) à `thisNewName`, vous devez utiliser le tuple suivant :

```
("`this.old.name`", "string", "thisNewName", "string")
```

- `transformation_ctx` – Chaîne unique utilisée pour identifier les informations sur l'état (facultatif).
- `info` – Chaîne associée à des erreurs dans la transformation (facultatif).
- `stageThreshold` – nombre maximal d'erreurs qui peuvent avoir lieu dans la transformation avant qu'elle ne soit arrêtée.(facultatif) La valeur par défaut est zéro.
- `totalThreshold` – nombre maximal d'erreurs pouvant se produire globalement avant que le processus de traitement des erreurs ne soit arrêté.(facultatif) La valeur par défaut est zéro.

Renvoie uniquement les champs du `DynamicFrame` spécifié dans les tuples de « mappage ».

apply(cls, *args, **kwargs)

Hérité de `GlueTransform` s'appliquent.

name(cls)

Hérité de `GlueTransform` name.

describeArgs(cls)

Hérité de `GlueTransform` describeArgs.

describeReturn(cls)

Hérité de `GlueTransform` describeReturn.

describeTransform(cls)

Hérité de `GlueTransform` describeTransform.

describeErrors(cls)

Hérité de `GlueTransform` describeErrors.

describe(cls)

Hérité de `GlueTransform` describe.

DropFields classe

Déplace des champs au sein d'un `DynamicFrame`.

Exemple

Nous vous recommandons d'utiliser la`DynamicFrame.drop_fields()`méthode pour supprimer des champs d'un`DynamicFrame`. Vous trouverez un exemple de code, consultez Exemple : utilisez supprimer les chams pour supprimer les champs d'un`DynamicFrame`.

Méthodes

- __call__
- s'appliquent
- name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Décrire

__call__(frame, paths, transformation_ctx = "", info = "", stageThreshold = 0, totalThreshold = 0)

Déplace des nœuds au sein d'un `DynamicFrame`.

- `frame` – Le `DynamicFrame` dans lequel supprimer les nœuds (obligatoire).
- `paths` – Liste des chemins complets vers les nœuds à déplacer (obligatoire).
- `transformation_ctx` – Chaîne unique utilisée pour identifier les informations sur l'état (facultatif).
- `info` – Chaîne associée à des erreurs dans la transformation (facultatif).
- `stageThreshold` – nombre maximal d'erreurs qui peuvent avoir lieu dans la transformation avant qu'elle ne soit arrêtée.(facultatif) La valeur par défaut est zéro.
- `totalThreshold` – nombre maximal d'erreurs pouvant se produire globalement avant que le processus de traitement des erreurs ne soit arrêté.(facultatif) La valeur par défaut est zéro.

Renvoie un nouveau `DynamicFrame` sans les champs spécifiés.

apply(cls, *args, **kwargs)

Hérité de `GlueTransform` s'appliquent.

name(cls)

Hérité de `GlueTransform` name.

describeArgs(cls)

Hérité de `GlueTransform` describeArgs.

describeReturn(cls)

Hérité de `GlueTransform` describeReturn.

describeTransform(cls)

Hérité de `GlueTransform` describeTransform.

describeErrors(cls)

Hérité de `GlueTransform` describeErrors.

describe(cls)

Hérité de `GlueTransform` describe.

DropNullFields classe

Rejette tous les champs null dans un `DynamicFrame` dont le type est `NullType`. Il s'agit des champs comportant des valeurs manquantes ou nulles dans chaque enregistrement de l'ensemble de données `DynamicFrame`.

Exemple

Cet exemple utilise `DropNullFields` pour créer un nouveau `DynamicFrame` où les champs de type `NullType` ont été abandonnés. Afin de démontrer `DropNullFields`, nous ajoutons une nouvelle colonne nommée `empty_column` de type null pour le jeu de données `persons` déjà chargé.

> **Note**
>
> Pour accéder au jeu de données utilisé dans cet exemple, voir Exemple de code : Données de jonction et de mise en relation et suivez les instructions de la sectionÉtape 1 : analyser les données dans le compartiment Amazon S3.

```
# Example: Use DropNullFields to create a new DynamicFrame without NullType fields

from pyspark.context import SparkContext
from awsglue.context import GlueContext
from pyspark.sql.functions import lit
from pyspark.sql.types import NullType
from awsglue.dynamicframe import DynamicFrame
from awsglue.transforms import DropNullFields

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Create DynamicFrame
persons = glueContext.create_dynamic_frame.from_catalog(
    database="legislators", table_name="persons_json"
)
print("Schema for the persons DynamicFrame:")
persons.printSchema()

# Add new column "empty_column" with NullType
persons_with_nulls = persons.toDF().withColumn("empty_column",
 lit(None).cast(NullType()))
persons_with_nulls_dyf = DynamicFrame.fromDF(persons_with_nulls, glueContext,
 "persons_with_nulls")
print("Schema for the persons_with_nulls_dyf DynamicFrame:")
persons_with_nulls_dyf.printSchema()

# Remove the NullType field
persons_no_nulls = DropNullFields.apply(persons_with_nulls_dyf)
print("Schema for the persons_no_nulls DynamicFrame:")
persons_no_nulls.printSchema()
```

Output

```
Schema for the persons DynamicFrame:
root
|-- family_name: string
|-- name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- gender: string
|-- image: string
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- lang: string
|    |    |-- note: string
|    |    |-- name: string
|-- sort_name: string
|-- images: array
|    |-- element: struct
|    |    |-- url: string
|-- given_name: string
|-- birth_date: string
|-- id: string
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
|-- death_date: string

Schema for the persons_with_nulls_dyf DynamicFrame:
root
|-- family_name: string
|-- name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- gender: string
|-- image: string
```

```
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- lang: string
|    |    |-- note: string
|    |    |-- name: string
|-- sort_name: string
|-- images: array
|    |-- element: struct
|    |    |-- url: string
|-- given_name: string
|-- birth_date: string
|-- id: string
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
|-- death_date: string
|-- empty_column: null

null_fields ['empty_column']
Schema for the persons_no_nulls DynamicFrame:
root
|-- family_name: string
|-- name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- gender: string
|-- image: string
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- lang: string
|    |    |-- note: string
|    |    |-- name: string
|-- sort_name: string
```

```
|-- images: array
|    |-- element: struct
|    |    |-- url: string
|-- given_name: string
|-- birth_date: string
|-- id: string
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
|-- death_date: string
```

Méthodes

- __call__
- s'appliquent
- name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Décrire

__call__(frame, transformation_ctx = "", info = "", stageThreshold = 0, totalThreshold = 0)

Rejette tous les champs null dans un `DynamicFrame` dont le type est `NullType`. Il s'agit des champs comportant des valeurs manquantes ou nulles dans chaque enregistrement de l'ensemble de données `DynamicFrame`.

- `frame` – `DynamicFrame` dans lequel déposer les champs nuls (obligatoire).
- `transformation_ctx` – Chaîne unique utilisée pour identifier les informations sur l'état (facultatif).
- `info` – Chaîne associée à des erreurs dans la transformation (facultatif).
- `stageThreshold` – nombre maximal d'erreurs qui peuvent avoir lieu dans la transformation avant qu'elle ne soit arrêtée.(facultatif) La valeur par défaut est zéro.

- `totalThreshold` – nombre maximal d'erreurs pouvant se produire globalement avant que le processus de traitement des erreurs ne soit arrêté.(facultatif) La valeur par défaut est zéro.

Renvoie un nouveau `DynamicFrame` sans champ null.

apply(cls, *args, **kwargs)

- `cls` – cls

name(cls)

- `cls` – cls

describeArgs(cls)

- `cls` – cls

describeReturn(cls)

- `cls` – cls

describeTransform(cls)

- `cls` – cls

describeErrors(cls)

- `cls` – cls

describe(cls)

- `cls` – cls

ErrorsAsDynamicFrame classe

Renvoie un `DynamicFrame` qui contient des enregistrements imbriqués pour les erreurs survenues lors de la source `DynamicFrame` a été créé.

Exemple

Nous vous recommandons d'utiliser la `DynamicFrame.errorsAsDynamicFrame()` méthode pour récupérer et afficher les enregistrements d'erreurs. Vous trouverez un exemple de code, consultez Exemple : utilisez errorsAsDynamic Frame pour afficher les enregistrements d'erreurs.

Méthodes

- __call__
- s'appliquent
- name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Décrire

__call__(frame)

Renvoie un `DynamicFrame` qui contient des enregistrements d'erreurs liés au `DynamicFrame` source.

- `frame` – `DynamicFrame` source (obligatoire).

apply(cls, *args, **kwargs)

- `cls` – cls

name(cls)

- `cls` – cls

describeArgs(cls)

- `cls` – cls

describeReturn(cls)

- `cls` – cls

describeTransform(cls)

- `cls` – cls

describeErrors(cls)

- `cls` – cls

describe(cls)

- `cls` – cls

EvaluateDataQuality classe

Évalue un jeu de règles de qualité des données par rapport à un `DynamicFrame` et renvoie un nouveau `DynamicFrame` avec les résultats de l'évaluation.

Exemple

L'exemple de code suivant montre comment évaluer la qualité des données pour un `DynamicFrame`, puis afficher les résultats relatifs à la qualité des données.

```
from awsglue.transforms import *
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsgluedq.transforms import EvaluateDataQuality

#Create Glue context
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Define DynamicFrame
legislatorsAreas = glueContext.create_dynamic_frame.from_catalog(
    database="legislators", table_name="areas_json")

# Create data quality ruleset
```

```
ruleset = """Rules = [ColumnExists "id", IsComplete "id"]"""

# Evaluate data quality
dqResults = EvaluateDataQuality.apply(
    frame=legislatorsAreas,
    ruleset=ruleset,
    publishing_options={
        "dataQualityEvaluationContext": "legislatorsAreas",
        "enableDataQualityCloudWatchMetrics": True,
        "enableDataQualityResultsPublishing": True,
        "resultsS3Prefix": "amzn-s3-demo-bucket1",
    },
)


# Inspect data quality results
dqResults.printSchema()
dqResults.toDF().show()
```

Output

```
root
|-- Rule: string
|-- Outcome: string
|-- FailureReason: string
|-- EvaluatedMetrics: map
|    |-- keyType: string
|    |-- valueType: double


+----------------------+-------+-------------+--------------------------------------+
|Rule                  |Outcome|FailureReason|EvaluatedMetrics                      |
+----------------------+-------+-------------+--------------------------------------+
|ColumnExists "id"     |Passed |null         |{}                                    |
|IsComplete "id"       |Passed |null         |{Column.first_name.Completeness -> 1.0}|
+----------------------+-------+-------------+--------------------------------------+
```

Méthodes

- __call__
- s'appliquent
- name

- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__ (trame, jeu de règles, options de publication = {})

- `frame` – Le `DynamicFrame` dont vous souhaitez évaluer la qualité des données.
- `ruleset` – Jeu de règles DQDL (Data Quality Definition Language) au format de chaîne. Pour en savoir plus sur DQDL, consultez le guide Référence DQDL (Data Quality Definition Language).
- `publishing_options` – Dictionnaire qui spécifie les options de publication suivantes des résultats et des métriques d'une évaluation :
  - `dataQualityEvaluationContext`— Chaîne qui indique l'espace de noms sous lequel AWS Glue doit publier Amazon CloudWatch les métriques et les résultats de qualité des données. Les statistiques agrégées apparaissent dans CloudWatch, tandis que les résultats complets apparaissent dans l'interface de AWS Glue Studio.
    - Obligatoire : non
    - Valeur par défaut : `default_context`
  - `enableDataQualityCloudWatchMetrics`— Spécifie si les résultats de l'évaluation de la qualité des données doivent être publiés sur CloudWatch. Vous spécifiez un espace de noms pour les métriques à l'aide de l'option `dataQualityEvaluationContext`.
    - Obligatoire : non
    - Valeur par défaut : False
  - `enableDataQualityResultsPublishing` – Spécifie si les résultats relatifs à la qualité des données doivent être visibles dans l'onglet Data Quality (Qualité des données) de l'interface AWS Glue Studio.
    - Obligatoire : non
    - Valeur par défaut : True
  - `resultsS3Prefix`— Spécifie l'emplacement Amazon S3 où AWS Glue peut écrire les résultats de l'évaluation de la qualité des données.
    - Obligatoire : non
    - Valeur par défaut : "" (chaîne vide)

apply(cls, *args, **kwargs)

Hérité de `GlueTransform` s'appliquent.

name(cls)

Hérité de `GlueTransform` name.

describeArgs(cls)

Hérité de `GlueTransform` describeArgs.

describeReturn(cls)

Hérité de `GlueTransform` describeReturn.

describeTransform(cls)

Hérité de `GlueTransform` describeTransform.

describeErrors(cls)

Hérité de `GlueTransform` describeErrors.

describe(cls)

Hérité de `GlueTransform` describe.

FillMissingValues classe

La classe `FillMissingValues` localise les valeurs nulles et les chaînes vides dans un élément `DynamicFrame` spécifié et utilise des méthodes de machine learning, telles que la régression linéaire et la forêt aléatoire, pour prédire les valeurs manquantes. La tâche ETL utilise les valeurs de l'ensemble de données d'entrée pour entraîner le modèle de machine learning, qui prédit ensuite quelles devraient être les valeurs manquantes.

> **Tip**
>
> Si vous utilisez des jeux de données incrémentiels, chacun d'entre eux est utilisé comme données d'entraînement pour le modèle de machine learning, de sorte que les résultats peuvent ne pas être aussi précis.

Pour importer :

```
from awsglueml.transforms import FillMissingValues
```

Méthodes

- Appliquer

apply(frame, missing_values_column, output_column ="", transformation_ctx ="", info ="", stageThreshold = 0, totalThreshold = 0)

Remplit les valeurs manquantes d'un cadre dynamique dans une colonne spécifiée et renvoie un nouveau cadre avec des estimations dans une nouvelle colonne. Pour les lignes sans valeurs manquantes, la valeur de la colonne spécifiée est dupliquée dans la nouvelle colonne.

- `frame` – `DynamicFrame` dans lequel renseigner les valeurs manquantes. Obligatoire.
- `missing_values_column` – colonne contenant les valeurs manquantes (valeurs `null` et chaînes vides). Obligatoire.
- `output_column` – nom de la nouvelle colonne qui contiendra les valeurs estimées pour toutes les lignes dont la valeur était manquante. Facultatif ; la valeur par défaut est le nom de `missing_values_column` avec le suffixe `"_filled"`.
- `transformation_ctx` – Chaîne unique utilisée pour identifier les informations sur l'état (facultatif).
- `info` – Chaîne associée à des erreurs dans la transformation (facultatif).
- `stageThreshold` – Nombre maximal d'erreurs qui peuvent avoir lieu dans la transformation avant qu'elle ne soit arrêtée (facultatif ; la valeur par défaut est zéro).
- `totalThreshold` – Nombre maximal d'erreurs pouvant se produire globalement avant que le processus de traitement des erreurs ne soit arrêté (facultatif ; la valeur par défaut est zéro).

Renvoie un nouveau `DynamicFrame` avec une colonne supplémentaire qui contient des estimations pour les lignes avec des valeurs manquantes et la valeur actuelle pour les autres lignes.

Classe Filter

Construit un nouvel `DynamicFrame` contenant des enrégistrements en entrée `DynamicFrame` qui satisfont à une fonction de prédicat spécifié

Exemple

Nous vous recommandons d'utiliser la `DynamicFrame.filter()` méthode pour supprimer des champs d'un `DynamicFrame`. Vous trouverez un exemple de code, consultez Exemple : Utiliser un filtre pour obtenir une sélection de champs filtrée.

Méthodes

- __call__
- s'appliquent
- name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__(frame, f, transformation_ctx="", info="", stageThreshold=0, totalThreshold=0))

Retourne un nouvel objet `DynamicFrame` construit en sélectionnant les enregistrements en entrée `DynamicFrame` qui satisfont à une fonction de prédicat spécifiée.

- `frame` – Objet `DynamicFrame` source auquel appliquer la fonction de filtre spécifiée (requis).
- `f` – fonction de prédicat à appliquer à chaque `DynamicRecord` de l'objet `DynamicFrame`. La fonction doit accepter un `DynamicRecord` comme argument et renvoyer la valeur Vraie si le `DynamicRecord` répond aux critères du filtre, ou la valeur Faux dans le cas contraire (obligatoire).

  Un `DynamicRecord` représente un enregistrement logique dans un `DynamicFrame`. Il est comparable à une ligne dans un Spark `DataFrame`, à la différence qu'il est auto-descriptif et peut être utilisé pour les données qui ne sont pas conformes à un schéma fixe.
- `transformation_ctx` – Chaîne unique utilisée pour identifier les informations sur l'état (facultatif).
- `info` – Chaîne associée à des erreurs dans la transformation (facultatif).
- `stageThreshold` – nombre maximal d'erreurs qui peuvent avoir lieu dans la transformation avant qu'elle ne soit arrêtée.(facultatif) La valeur par défaut est zéro.

- `totalThreshold` – nombre maximal d'erreurs pouvant se produire globalement avant que le processus de traitement des erreurs ne soit arrêté.(facultatif) La valeur par défaut est zéro.

apply(cls, *args, **kwargs)

Hérité de `GlueTransform` s'appliquent.

name(cls)

Hérité de `GlueTransform` name.

describeArgs(cls)

Hérité de `GlueTransform` describeArgs.

describeReturn(cls)

Hérité de `GlueTransform` describeReturn.

describeTransform(cls)

Hérité de `GlueTransform` describeTransform.

describeErrors(cls)

Hérité de `GlueTransform` describeErrors.

describe(cls)

Hérité de `GlueTransform` describe.

FindIncrementalMatches classe

Identifie les enregistrements correspondants dans les `DynamicFrame` existants et incrémentiels, et crée un `DynamicFrame` avec un identifiant unique affecté à chaque groupe d'enregistrements correspondants.

Pour importer :

```
from awsglueml.transforms import FindIncrementalMatches
```

Méthodes

- Appliquer

appliquer (ExistingFrame, IncrementalFrame, TransformId, transformation_ctx = « », info = « », StageThreshold = 0, TotalThreshold = 0, EnforcedMatches = aucun, Scores = 0) computeMatchConfidence

Identifie les enregistrements correspondants dans l'entrée `DynamicFrame`, et crée un `DynamicFrame` avec un identifiant unique affecté à chaque groupe d'enregistrements correspondants.

- `existingFrame`— L'existant et le préapparié `DynamicFrame` pour appliquer la FindIncrementalMatches transformation. Obligatoire.
- `incrementalFrame`— L'incrémentiel `DynamicFrame` à appliquer à la FindIncrementalMatches transformation pour qu'elle corresponde `existingFrame` à. Obligatoire.
- `transformId`— L'identifiant unique associé à la FindIncrementalMatches transformation à appliquer aux enregistrements du`DynamicFrames`. Obligatoire.
- `transformation_ctx`— Chaîne unique utilisée pour identifier les stats/state informations. Facultatif.
- `info` – chaîne à associer à des erreurs dans la transformation. Facultatif.
- `stageThreshold` – nombre maximal d'erreurs qui peuvent avoir lieu dans la transformation avant qu'elle ne soit arrêtée. Facultatif. La valeur par défaut est zéro.
- `totalThreshold` – nombre maximal d'erreurs pouvant se produire globalement avant que le processus de traitement des erreurs ne soit arrêté. Facultatif. La valeur par défaut est zéro.
- `enforcedMatches` – `DynamicFrame` utilisé pour appliquer les correspondances. Facultatif. La valeur par défaut est Aucun.
- `computeMatchConfidenceScores` – Valeur booléenne indiquant s'il faut calculer un score de confiance pour chaque groupe de registres correspondant. Facultatif. La valeur par défaut est false.

Renvoie un nouveau `DynamicFrame` avec un identifiant unique affecté à chaque groupe d'enregistrements correspondants.

FindMatches classe

Identifie les enregistrements correspondants dans l'entrée `DynamicFrame`, et crée un `DynamicFrame` avec un identifiant unique affecté à chaque groupe d'enregistrements correspondants.

Pour importer :

```
from awsglueml.transforms import FindMatches
```

Méthodes

- Appliquer

appliquer (frame, TransformId, transformation_ctx = « », info = « », StageThreshold = 0, TotalThreshold = 0, EnforcedMatches = aucun, Scores = 0) computeMatchConfidence

Identifie les enregistrements correspondants dans l'entrée `DynamicFrame`, et crée un `DynamicFrame` avec un identifiant unique affecté à chaque groupe d'enregistrements correspondants.

- `frame`— Le `DynamicFrame` pour appliquer la FindMatches transformation. Obligatoire.
- `transformId`— L'identifiant unique associé à la FindMatches transformation à appliquer aux enregistrements du`DynamicFrame`. Obligatoire.
- `transformation_ctx`— Chaîne unique utilisée pour identifier les stats/state informations. Facultatif.
- `info` – chaîne à associer à des erreurs dans la transformation. Facultatif.
- `stageThreshold` – nombre maximal d'erreurs qui peuvent avoir lieu dans la transformation avant qu'elle ne soit arrêtée. Facultatif. La valeur par défaut est zéro.
- `totalThreshold` – nombre maximal d'erreurs pouvant se produire globalement avant que le processus de traitement des erreurs ne soit arrêté. Facultatif. La valeur par défaut est zéro.
- `enforcedMatches` – `DynamicFrame` utilisé pour appliquer les correspondances. Facultatif. La valeur par défaut est Aucun.
- `computeMatchConfidenceScores` – Valeur booléenne indiquant s'il faut calculer un score de confiance pour chaque groupe de registres correspondant. Facultatif. La valeur par défaut est false.

Renvoie un nouveau `DynamicFrame` avec un identifiant unique affecté à chaque groupe d'enregistrements correspondants.

FlatMap classe

Applique une transformation à chaque `DynamicFrame` d'une collection. Les résultats ne sont pas aplatis en un seul `DynamicFrame`, mais conservés sous forme de collection.

Exemples pour FlatMap

L'exemple d'extrait suivant montre comment utiliser la transformation `ResolveChoice` sur une collection de cadres dynamiques lorsqu'elle est appliquée à un `FlatMap`. Les données utilisées pour l'entrée se trouvent dans le JSON situé à l'adresse Amazon S3 de l'espace réservé `s3://bucket/path-for-data/sample.json` et contiennent les données suivantes.

Exemple de données JSON

```
[{
    "firstname": "Arnav",
    "lastname": "Desai",
    "address": {
        "street": "6 Anyroad Avenue",
        "city": "London",
        "state": "England",
        "country": "UK"
    },
    "phone": 17235550101,
    "affiliations": [
        "General Anonymous Example Products",
        "Example Independent Research",
        "Government Department of Examples"
    ]
},
{
    "firstname": "Mary",
    "lastname": "Major",
    "address": {
        "street": "7821 Spot Place",
        "city": "Centerville",
        "state": "OK",
        "country": "US"
    },
    "phone": 19185550023,
    "affiliations": [
        "Example Dot Com",
        "Example Independent Research",
        "Example.io"
    ]
},
{
    "firstname": "Paulo",
```

```
    "lastname": "Santos",
    "address": {
        "street": "123 Maple Street",
        "city": "London",
        "state": "Ontario",
        "country": "CA"
    },
    "phone": 12175550181,
    "affiliations": [
        "General Anonymous Example Products",
        "Example Dot Com"
    ]
}]
```

Example Appliquer ResolveChoice à une sortie DynamicFrameCollection et afficher.

```
#Read DynamicFrame
datasource = glueContext.create_dynamic_frame_from_options("s3", connection_options =
 {"paths":["s3://bucket/path/to/file/mysamplejson.json"]}, format="json")
datasource.printSchema()
datasource.show()

## Split to create a DynamicFrameCollection
split_frame=datasource.split_fields(["firstname","lastname","address"],"personal_info","busines
split_frame.keys()
print("---")

## Use FlatMap to run ResolveChoice
kwargs = {"choice": "cast:string"}
flat = FlatMap.apply(split_frame, ResolveChoice, frame_name="frame",
 transformation_ctx='tcx', **kwargs)
flat.keys()

##Select one of the DynamicFrames
personal_info = flat.select("personal_info")
personal_info.printSchema()
personal_info.show()
print("---")

business_info = flat.select("business_info")
business_info.printSchema()
business_info.show()
```

> **Important**
>
> Lorsque vous appelez `FlatMap.apply`, le paramètre `frame_name` doit être `"frame"`. Aucune autre valeur n'est actuellement acceptée.

Exemple de sortie

```
root
|-- firstname: string
|-- lastname: string
|-- address: struct
|    |-- street: string
|    |-- city: string
|    |-- state: string
|    |-- country: string
|-- phone: long
|-- affiliations: array
|    |-- element: string
---
{
    "firstname": "Mary",
    "lastname": "Major",
    "address": {
        "street": "7821 Spot Place",
        "city": "Centerville",
        "state": "OK",
        "country": "US"
    },
    "phone": 19185550023,
    "affiliations": [
        "Example Dot Com",
        "Example Independent Research",
        "Example.io"
    ]
}

{
    "firstname": "Paulo",
    "lastname": "Santos",
    "address": {
        "street": "123 Maple Street",
        "city": "London",
```

```
        "state": "Ontario",
        "country": "CA"
    },
    "phone": 12175550181,
    "affiliations": [
        "General Anonymous Example Products",
        "Example Dot Com"
    ]
}
---
root
|-- firstname: string
|-- lastname: string
|-- address: struct
|    |-- street: string
|    |-- city: string
|    |-- state: string
|    |-- country: string

{
    "firstname": "Mary",
    "lastname": "Major",
    "address": {
        "street": "7821 Spot Place",
        "city": "Centerville",
        "state": "OK",
        "country": "US"
    }
}

{
    "firstname": "Paulo",
    "lastname": "Santos",
    "address": {
        "street": "123 Maple Street",
        "city": "London",
        "state": "Ontario",
        "country": "CA"
    }
}
---
root
|-- phone: long
|-- affiliations: array
```

```
|    |-- element: string

{
    "phone": 19185550023,
    "affiliations": [
        "Example Dot Com",
        "Example Independent Research",
        "Example.io"
    ]
}

{
    "phone": 12175550181,
    "affiliations": [
        "General Anonymous Example Products",
        "Example Dot Com"
    ]
}
```

Méthodes

- __call__
- Appliquer
- Name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Décrire

__call__ (dfc, nom_cadre BaseTransform, transformation_ctx = « », **base_kwargs)

Applique une transformation à chaque `DynamicFrame` d'une collection et aplatit les résultats.

- `dfc` – `DynamicFrameCollection` sur lequel effectuer un flatmap (obligatoire).
- `BaseTransform` – Transformation dérivée de `GlueTransform` à appliquer à chaque membre de la collection (obligatoire).
- `frame_name` – Nom de l'argument auquel transmettre les éléments de la collection (obligatoire).

- `transformation_ctx` – Chaîne unique utilisée pour identifier les informations sur l'état (facultatif).
- `base_kwargs` – Arguments à transmettre à la transformation de base (obligatoire).

Retourne un nouvel objet `DynamicFrameCollection` en appliquant la transformation à chaque `DynamicFrame` de l'objet `DynamicFrameCollection` source.

apply(cls, *args, **kwargs)

Hérité de `GlueTransform` s'appliquent.

name(cls)

Hérité de `GlueTransform` name.

describeArgs(cls)

Hérité de `GlueTransform` describeArgs.

describeReturn(cls)

Hérité de `GlueTransform` describeReturn.

describeTransform(cls)

Hérité de `GlueTransform` describeTransform.

describeErrors(cls)

Hérité de `GlueTransform` describeErrors.

describe(cls)

Hérité de `GlueTransform` describe.

Classe Join

Effectue une jointure d'égalité sur deux `DynamicFrames`.

Exemple

Nous vous recommandons d'utiliser la `DynamicFrame.join()` méthode pour joindre `DynamicFrames`. Vous trouverez un exemple de code, consultez Exemple : Utiliser la jointure pour combiner`DynamicFrames`.

Méthodes

- __call__
- s'appliquent
- name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Décrire

__call__(frame1, frame2, keys1, keys2, transformation_ctx = "")

Effectue une jointure d'égalité sur deux `DynamicFrames`.

- `frame1` – Premier `DynamicFrame` à joindre (obligatoire).
- `frame2` – Second `DynamicFrame` à joindre (obligatoire).
- `keys1` – Clés à joindre pour la première trame (obligatoire).
- `keys2` – Clés à joindre pour la seconde trame (obligatoire).
- `transformation_ctx` – Chaîne unique utilisée pour identifier les informations sur l'état (facultatif).

Renvoie un nouveau `DynamicFrame` qui est créé en joignant les deux `DynamicFrames`.

apply(cls, *args, **kwargs)

Hérité de `GlueTransform` s'appliquent.

name(cls)

Hérité de `GlueTransform` name.

describeArgs(cls)

Hérité de `GlueTransform` describeArgs.

describeReturn(cls)

Hérité de `GlueTransform` describeReturn.

describeTransform(cls)

Hérité de `GlueTransform` describeTransform.

describeErrors(cls)

Hérité de `GlueTransform` describeErrors.

describe(cls)

Hérité de `GlueTransform` describe.

Classe Map

Crée un nouvel objet `DynamicFrame` en appliquant une fonction à tous les objets `DynamicFrame` en entrée.

Exemple

Nous vous recommandons d'utiliser le `DynamicFrame.map()` méthode pour appliquer une fonction à tous les enrégistrements dans `DynamicFrame`. Vous trouverez un exemple de code, consultez Exemple : utilisez la carte pour appliquer une fonction à chaque enregistrement dans un `DynamicFrame`.

Méthodes

- __call__
- s'appliquent
- name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Décrire

__call__(frame, f, transformation_ctx="", info="", stageThreshold=0, totalThreshold=0)

Renvoie un nouvel objet `DynamicFrame` qui résulte de l'application de la fonction spécifiée à tous les `DynamicRecords` de l'objet `DynamicFrame` d'origine.

- `frame` – Objet `DynamicFrame` d'origine auquel appliquer la fonction de mappage (obligatoire).
- `f` – fonction à tous les `DynamicRecords` de l'objet `DynamicFrame`. La fonction doit accepter un `DynamicRecord` comme argument et renvoyer un nouvel objet `DynamicRecord` produit par le mappage (obligatoire).

  Un `DynamicRecord` représente un enregistrement logique dans un `DynamicFrame`. Il est comparable à une ligne dans un `DataFrame` Apache Spark, à la différence qu'il est auto-descriptif et peut être utilisé pour les données qui ne sont pas conformes à un schéma fixe.
- `transformation_ctx` – Chaîne unique utilisée pour identifier les informations sur l'état (facultatif).
- `info` – Chaîne associée à des erreurs dans la transformation (facultatif).
- `stageThreshold` – nombre maximal d'erreurs qui peuvent avoir lieu dans la transformation avant qu'elle ne soit arrêtée.(facultatif) La valeur par défaut est zéro.
- `totalThreshold` – nombre maximal d'erreurs pouvant se produire globalement avant que le processus de traitement des erreurs ne soit arrêté.(facultatif) La valeur par défaut est zéro.

Renvoie un nouvel objet `DynamicFrame` qui résulte de l'application de la fonction spécifiée à tous les `DynamicRecords` de l'objet `DynamicFrame` d'origine.

apply(cls, *args, **kwargs)

Hérité de `GlueTransform` s'appliquent.

name(cls)

Hérité de `GlueTransform` name.

describeArgs(cls)

Hérité de `GlueTransform` describeArgs.

describeReturn(cls)

Hérité de `GlueTransform` describeReturn.

describeTransform(cls)

Hérité de `GlueTransform` describeTransform.

describeErrors(cls)

Hérité de `GlueTransform` describeErrors.

describe(cls)

Hérité de `GlueTransform` describe.

MapToCollection classe

Applique une transformation à chaque `DynamicFrame` de l'objet `DynamicFrameCollection` spécifié.

Méthodes

- __call__
- Appliquer
- Name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Décrire

__call__ (dfc, nom_cadre BaseTransform, transformation_ctx = « », **base_kwargs)

Applique une fonction de transformation à chaque `DynamicFrame` de l'objet `DynamicFrameCollection` spécifié.

- `dfc` – L'objet `DynamicFrameCollection` sur lequel appliquer la fonction de transformation (requise).
- `callable` – Une fonction de transformation appelable à appliquer à chaque membre de la collection (obligatoire).
- `transformation_ctx` – Chaîne unique utilisée pour identifier les informations sur l'état (facultatif).

Retourne un nouvel objet `DynamicFrameCollection` en appliquant la transformation à chaque `DynamicFrame` de l'objet `DynamicFrameCollection` source.

apply(cls, *args, **kwargs)

Hérité de `GlueTransform` s'appliquent

name(cls)

Hérité de `GlueTransform` name.

describeArgs(cls)

Hérité de `GlueTransform` describeArgs.

describeReturn(cls)

Hérité de `GlueTransform` describeReturn.

describeTransform(cls)

Hérité de `GlueTransform` describeTransform.

describeErrors(cls)

Hérité de `GlueTransform` describeErrors.

describe(cls)

Hérité de `GlueTransform` describe.

Classe Relationalize

Aplatit le schéma imbriqué dans une image `DynamicFrame` et fait pivoter les colonnes de tableaux de l'image aplatie.

Exemple

Nous vous recommandons d'utiliser la méthode `DynamicFrame.relationalize()` pour mettre en relation une image `DynamicFrame`. Vous trouverez un exemple de code, consultez Exemple : utiliser la méthode relationalize pour aplatir un schéma imbriqué dans une image `DynamicFrame`.

Méthodes

- __call__

- s'appliquent
- name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__(frame, staging_path=None, name='roottable', options=None, transformation_ctx = "", info = "", stageThreshold = 0, totalThreshold = 0)

Met en relation un `DynamicFrame` et produit une liste des trames qui sont générées en supprimant l'imbrication des colonnes imbriquées et en faisant pivoter des colonnes de tableaux. Vous pouvez joindre une colonne de tableau ayant été pivotée à la table racine à l'aide de la clé de jointure générée au cours de la phase de désimbrication.

- `frame` – `DynamicFrame` à mettre en relation (obligatoire).
- `staging_path` – Chemin d'accès auquel la méthode peut stocker des partitions des tables dynamiques au format CSV format (facultatif). Les tables dynamiques sont lues à partir de ce chemin.
- `name` – Nom de la table racine (facultatif).
- `options` – Dictionnaire des paramètres facultatifs. Non utilisé actuellement.
- `transformation_ctx` – Chaîne unique utilisée pour identifier les informations sur l'état (facultatif).
- `info` – Chaîne associée à des erreurs dans la transformation (facultatif).
- `stageThreshold` – nombre maximal d'erreurs qui peuvent avoir lieu dans la transformation avant qu'elle ne soit arrêtée.(facultatif) La valeur par défaut est zéro.
- `totalThreshold` – nombre maximal d'erreurs pouvant se produire globalement avant que le processus de traitement des erreurs ne soit arrêté.(facultatif) La valeur par défaut est zéro.

apply(cls, *args, **kwargs)

Hérité de `GlueTransform` s'appliquent.

name(cls)

Hérité de `GlueTransform` name.

describeArgs(cls)

Hérité de `GlueTransform` describeArgs.

describeReturn(cls)

Hérité de `GlueTransform` describeReturn.

describeTransform(cls)

Hérité de `GlueTransform` describeTransform.

describeErrors(cls)

Hérité de `GlueTransform` describeErrors.

describe(cls)

Hérité de `GlueTransform` describe.

RenameField classe

Renomme un nœud dans un `DynamicFrame`.

Exemple

Nous vous recommandons d'utiliser la méthode `DynamicFrame.rename_field()` pour renommer un champ dans une image `DynamicFrame`. Vous trouverez un exemple de code, consultez Exemple : utiliser la méthode rename_field pour renommer des champs dans une image `DynamicFrame`.

Méthodes

- __call__
- s'appliquent
- name

- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__(frame, old_name, new_name, transformation_ctx = "", info = "", stageThreshold = 0, totalThreshold = 0)

Renomme un nœud dans un `DynamicFrame`.

- `frame` – `DynamicFrame` dans lequel renommer un nœud (obligatoire).
- `old_name` – Chemin d'accès complet au nœud à renommer (obligatoire).

  Si l'ancien nom comporte des points, cela ne RenameField fonctionnera pas à moins que vous ne le contourniez (``). Par exemple, pour remplacer `this.old.name` par`thisNewName`, vous devez appeler RenameField comme suit :

  ```
  newDyF = RenameField(oldDyF, "`this.old.name`", "thisNewName")
  ```

- `new_name` – Nouveau nom, y compris le chemin d'accès complet (obligatoire).
- `transformation_ctx` – Chaîne unique utilisée pour identifier les informations sur l'état (facultatif).
- `info` – Chaîne associée à des erreurs dans la transformation (facultatif).
- `stageThreshold` – nombre maximal d'erreurs qui peuvent avoir lieu dans la transformation avant qu'elle ne soit arrêtée.(facultatif) La valeur par défaut est zéro.
- `totalThreshold` – nombre maximal d'erreurs pouvant se produire globalement avant que le processus de traitement des erreurs ne soit arrêté.(facultatif) La valeur par défaut est zéro.

apply(cls, *args, **kwargs)

Hérité de `GlueTransform` s'appliquent.

name(cls)

Hérité de `GlueTransform` name.

describeArgs(cls)

Hérité de `GlueTransform` describeArgs.

describeReturn(cls)

Hérité de `GlueTransform` describeReturn.

describeTransform(cls)

Hérité de `GlueTransform` describeTransform.

describeErrors(cls)

Hérité de `GlueTransform` describeErrors.

describe(cls)

Hérité de `GlueTransform` describe.

ResolveChoice classe

Résout un type de choix au sein d'un `DynamicFrame`.

Exemple

Nous vous recommandons d'utiliser la méthode `DynamicFrame.resolveChoice()` pour gérer les champs contenant plusieurs types dans une image `DynamicFrame`. Vous trouverez un exemple de code, consultez Exemple : utiliser la méthode resolveChoice pour gérer une colonne contenant plusieurs types.

Méthodes

- __call__
- s'appliquent
- name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__(frame, specs = None, choice = "", transformation_ctx = "", info = "", stageThreshold = 0, totalThreshold = 0)

Fournit des informations pour résoudre les types ambigus au sein d'un `DynamicFrame`. Renvoie l'image `DynamicFrame` obtenue.

- `frame` – `DynamicFrame` dans lequel résoudre le type de choix (obligatoire).
- `specs` – Liste d'ambiguïtés spécifiques à résoudre, apparaissant sous forme de tuple:`(path, action)`. La valeur `path` identifie un élément ambigu spécifique, et la valeur `action` identifie la résolution correspondante.

  Vous ne pouvez utiliser qu'un seul des paramètres `spec` et `choice`. Si le paramètre `spec` n'est pas `None`, alors le paramètre `choice` doit être une chaîne vide. Inversement, si le paramètre `choice` n'est pas une chaîne vide, alors le paramètre `spec` doit être `None`. Si aucun paramètre n'est fourni, AWS Glue essaie d'analyser le schéma et de l'utiliser pour résoudre les ambiguïtés.

  Vous pouvez spécifier l'une des stratégies de résolution suivantes dans la portion `action` d'un tuple `specs` :

  - `cast` – Vous permet de spécifier un type pour la conversion (par exemple, `cast:int`).
  - `make_cols` – Résout une ambiguïté potentielle en aplatissant les données. Par exemple, si `columnA` peut être un `int` ou un `string`, la résolution doit produire deux colonnes nommées `columnA_int` et `columnA_string` dans le `DynamicFrame` obtenu.
  - `make_struct` – Résout une ambiguïté potentielle en utilisant un champ struct pour représenter les données. Par exemple, si des données d'une colonne peuvent être un `int` ou un `string`, l'action `make_struct` produit une colonne de structures dans le `DynamicFrame` obtenu, chacune contenant à la fois un `int` et un `string`.
  - `project` – Résout une éventuelle ambiguïté en conservant uniquement les valeurs d'un type spécifié dans l'image `DynamicFrame` obtenue. Par exemple, si les données d'une colonne `ChoiceType` peuvent être de type `int` ou `string`, la spécification d'une action `project:string` supprime des valeurs de l'image `DynamicFrame` obtenue qui ne sont pas de type `string`.

  Si le `path` identifie un tableau, placez des crochets vides après le nom du tableau pour éviter toute ambiguïté. Par exemple, supposons que vous travailliez avec les données structurées comme suit :

```
"myList": [
  { "price": 100.00 },
  { "price": "$100.00" }
```

```
]
```

Vous pouvez sélectionner la version numérique plutôt que la version chaîne du prix en définissant `path` sur `"myList[].price"` et `action` sur `"cast:double"`.

- `choice` – action de résolution par défaut si le paramètre `specs` est `None`. Si le paramètre `specs` n'est pas `None`, alors la seule valeur définie doit être une chaîne vide.

  En plus des actions `specs` répertoriées précédemment, cet argument prend également en charge l'action suivante :

  - `MATCH_CATALOG` – tente de convertir chaque `ChoiceType` dans le type correspondant de table Data Catalog spécifiée.
- `database`— La base de données AWS Glue Data Catalog à utiliser avec le `MATCH_CATALOG` choix (obligatoire pour`MATCH_CATALOG`).
- `table_name`— Le nom de la table AWS Glue Data Catalog à utiliser avec l'`MATCH_CATALOG`action (obligatoire pour`MATCH_CATALOG`).
- `transformation_ctx` – Chaîne unique utilisée pour identifier les informations sur l'état (facultatif).
- `info` – Chaîne associée à des erreurs dans la transformation (facultatif).
- `stageThreshold` – nombre maximal d'erreurs qui peuvent avoir lieu dans la transformation avant qu'elle ne soit arrêtée.(facultatif) La valeur par défaut est zéro.
- `totalThreshold` – nombre maximal d'erreurs pouvant se produire globalement avant que le processus de traitement des erreurs ne soit arrêté.(facultatif) La valeur par défaut est zéro.

apply(cls, *args, **kwargs)

Hérité de `GlueTransform` s'appliquent.

name(cls)

Hérité de `GlueTransform` name.

describeArgs(cls)

Hérité de `GlueTransform` describeArgs.

describeReturn(cls)

Hérité de `GlueTransform` describeReturn.

describeTransform(cls)

Hérité de `GlueTransform` describeTransform.

describeErrors(cls)

Hérité de `GlueTransform` describeErrors.

describe(cls)

Hérité de `GlueTransform` describe.

SelectFields classe

`SelectFields`La classe crée une nouvelle`DynamicFrame`à partir d'un`DynamicFrame`, et ne conserve que les champs que vous spécifiez.`SelectFields`fournit des fonctionnalités similaires à celles d'un SQL`SELECT`instructions.

Exemple

Nous vous recommandons d'utiliser la`DynamicFrame.select_fields()`méthode pour sélectionner des champs d'un`DynamicFrame`. Vous trouverez un exemple de code, consultez Exemple : utilisez select_fields pour créer un nouveau `DynamicFrame` avec les champs sélectionnés.

Méthodes

- __call__
- s'appliquent
- name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Décrire

__call__(frame, paths, transformation_ctx = "", info = "", stageThreshold = 0, totalThreshold = 0)

Obtient les champs (nœuds) d'un `DynamicFrame`.

- `frame` – Objet `DynamicFrame` dans lequel sélectionner les champs (requis).
- `paths` – Liste des chemins complets vers les champs à sélectionner (obligatoire).
- `transformation_ctx` – Chaîne unique utilisée pour identifier les informations sur l'état (facultatif).
- `info` – Chaîne associée à des erreurs dans la transformation (facultatif).
- `stageThreshold` – nombre maximal d'erreurs qui peuvent avoir lieu dans la transformation avant qu'elle ne soit arrêtée.(facultatif) La valeur par défaut est zéro.
- `totalThreshold` – nombre maximal d'erreurs pouvant se produire globalement avant que le processus de traitement des erreurs ne soit arrêté.(facultatif) La valeur par défaut est zéro.

Renvoie un nouvel objet `DynamicFrame` contenant uniquement les champs spécifiés.

apply(cls, *args, **kwargs)

Hérité de `GlueTransform` s'appliquent.

name(cls)

Hérité de `GlueTransform` name.

describeArgs(cls)

Hérité de `GlueTransform` describeArgs.

describeReturn(cls)

Hérité de `GlueTransform` describeReturn.

describeTransform(cls)

Hérité de `GlueTransform` describeTransform.

describeErrors(cls)

Hérité de `GlueTransform` describeErrors.

describe(cls)

Hérité de `GlueTransform` describe.

SelectFromCollection classe

Sélectionne un `DynamicFrame` dans un `DynamicFrameCollection`.

Exemple

Cet exemple utilise `SelectFromCollection` pour sélectionner une image `DynamicFrame` dans une collection `DynamicFrameCollection`.

Exemple de jeu de données

L'exemple sélectionne deux images `DynamicFrames` d'une collection `DynamicFrameCollection` appelée `split_rows_collection`. Vous trouverez ci-dessous la liste des clés de la collection `split_rows_collection`.

```
dict_keys(['high', 'low'])
```

Exemple de code

```
# Example: Use SelectFromCollection to select
# DynamicFrames from a DynamicFrameCollection

from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.transforms import SelectFromCollection

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Select frames and inspect entries
frame_low = SelectFromCollection.apply(dfc=split_rows_collection, key="low")
frame_low.toDF().show()

frame_high = SelectFromCollection.apply(dfc=split_rows_collection, key="high")
frame_high.toDF().show()
```

Output

```
+---+-----+-----------------------+-------------------------+
| id|index|contact_details.val.type|contact_details.val.value|
+---+-----+-----------------------+-------------------------+
|  1|    0|                    fax|             202-225-3307|
```

```
|  1|    1|                  phone|             202-225-5731|
|  2|    0|                    fax|             202-225-3307|
|  2|    1|                  phone|             202-225-5731|
|  3|    0|                    fax|             202-225-3307|
|  3|    1|                  phone|             202-225-5731|
|  4|    0|                    fax|             202-225-3307|
|  4|    1|                  phone|             202-225-5731|
|  5|    0|                    fax|             202-225-3307|
|  5|    1|                  phone|             202-225-5731|
|  6|    0|                    fax|             202-225-3307|
|  6|    1|                  phone|             202-225-5731|
|  7|    0|                    fax|             202-225-3307|
|  7|    1|                  phone|             202-225-5731|
|  8|    0|                    fax|             202-225-3307|
|  8|    1|                  phone|             202-225-5731|
|  9|    0|                    fax|             202-225-3307|
|  9|    1|                  phone|             202-225-5731|
| 10|    0|                    fax|             202-225-6328|
| 10|    1|                  phone|             202-225-4576|
+---+-----+----------------------+-------------------------+
only showing top 20 rows

+---+-----+----------------------+-------------------------+
| id|index|contact_details.val.type|contact_details.val.value|
+---+-----+----------------------+-------------------------+
| 11|    0|                    fax|             202-225-6328|
| 11|    1|                  phone|             202-225-4576|
| 11|    2|                twitter|           RepTrentFranks|
| 12|    0|                    fax|             202-225-6328|
| 12|    1|                  phone|             202-225-4576|
| 12|    2|                twitter|           RepTrentFranks|
| 13|    0|                    fax|             202-225-6328|
| 13|    1|                  phone|             202-225-4576|
| 13|    2|                twitter|           RepTrentFranks|
| 14|    0|                    fax|             202-225-6328|
| 14|    1|                  phone|             202-225-4576|
| 14|    2|                twitter|           RepTrentFranks|
| 15|    0|                    fax|             202-225-6328|
| 15|    1|                  phone|             202-225-4576|
| 15|    2|                twitter|           RepTrentFranks|
| 16|    0|                    fax|             202-225-6328|
| 16|    1|                  phone|             202-225-4576|
| 16|    2|                twitter|           RepTrentFranks|
| 17|    0|                    fax|             202-225-6328|
```

```
| 17|    1|                   phone|             202-225-4576|
+---+-----+------------------------+-------------------------+
only showing top 20 rows
```

Méthodes

- __call__
- s'appliquent
- name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__(dfc, key, transformation_ctx = "")

Obtient un `DynamicFrame` d'un `DynamicFrameCollection`.

- `dfc` – Collection `DynamicFrameCollection` à partir de laquelle l'image `DynamicFrame` doit être sélectionnée (obligatoire).
- `key` – Clé du `DynamicFrame` à sélectionner (obligatoire).
- `transformation_ctx` – Chaîne unique utilisée pour identifier les informations sur l'état (facultatif).

apply(cls, *args, **kwargs)

Hérité de `GlueTransform` s'appliquent.

name(cls)

Hérité de `GlueTransform` name.

describeArgs(cls)

Hérité de `GlueTransform` describeArgs.

describeReturn(cls)

Hérité de `GlueTransform` describeReturn.

describeTransform(cls)

Hérité de `GlueTransform` describeTransform.

describeErrors(cls)

Hérité de `GlueTransform` describeErrors.

describe(cls)

Hérité de `GlueTransform` describe.

Classe Simplify_ddb_json

Simplifie les colonnes imbriquées dans un `DynamicFrame` qui se trouvent spécifiquement dans la structure JSON DynamoDB, et renvoie une nouvelle version simplifiée du `DynamicFrame`.

Exemple

Nous vous recommandons d'utiliser la méthode `DynamicFrame.simplify_ddb_json()` pour simplifier les colonnes imbriquées dans un `DynamicFrame` qui se trouvent spécifiquement dans la structure JSON DynamoDB. Vous trouverez un exemple de code, consultez Exemple : utilisez simplify_ddb_json pour invoquer une simplification JSON DynamoDB..

Classe Spigot

Écrit des exemples d'enregistrement sur une destination spécifiée pour vous aider à vérifier les transformations effectuées par votre tâche AWS Glue.

Exemple

Nous vous recommandons d'utiliser la méthode `DynamicFrame.spigot()` pour écrire un sous-ensemble d'enregistrements depuis une image `DynamicFrame` vers une destination spécifiée. Vous trouverez un exemple de code, consultez Exemple : utiliser la méthode spigot pour écrire des exemples de champs d'une image `DynamicFrame` vers Amazon S3.

Méthodes

- __call__

- s'appliquent
- name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__(trame, chemin, options, transformation_ctx = "")

Écrit des exemples d'enregistrements sur une destination spécifiée au cours d'une transformation.

- `frame` - `DynamicFrame` sur lequel effectuer le spigot (requis).
- `path` – Chemin d'accès de la destination sur laquelle écrire (obligatoire).
- `options` – Paires clé-valeur JSON spécifiant des options (facultatif). L'option `"topk"` indique que les premiers enregistrements k doivent être écrits. L'option `"prob"` indique la probabilité (sous forme de décimale) de sélectionner un enregistrement donné. Utilisée pour sélectionner les enregistrements à écrire.
- `transformation_ctx` – Chaîne unique utilisée pour identifier les informations sur l'état (facultatif).

apply(cls, *args, **kwargs)

Hérité de `GlueTransform` s'appliquent

name(cls)

Hérité de `GlueTransform` name

describeArgs(cls)

Hérité de `GlueTransform` describeArgs

describeReturn(cls)

Hérité de `GlueTransform` describeReturn

describeTransform(cls)

Hérité de `GlueTransform` describeTransform

describeErrors(cls)

Hérité de `GlueTransform` describeErrors

describe(cls)

Hérité de `GlueTransform` describe

SplitFields classe

Fractionne un `DynamicFrame` en deux, en fonction des champs spécifiés.

Exemple

Nous vous recommandons d'utiliser la méthode `DynamicFrame.split_fields()` pour fractionner les champs en image `DynamicFrame`. Vous trouverez un exemple de code, consultez Exemple : utiliser la méthode split_fields pour fractionner les champs sélectionnés en une image `DynamicFrame` distincte.

Méthodes

- __call__
- s'appliquent
- name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__(frame, paths, name1 = None, name2 = None, transformation_ctx = "", info = "", stageThreshold = 0, totalThreshold = 0)

Fractionne un ou plusieurs champs d'une image `DynamicFrame` en une nouvelle image `DynamicFrame` et crée une autre nouvelle image `DynamicFrame` contenant les champs restants.

- `frame` – `DynamicFrame` source à scinder en deux nouveaux objets (obligatoire).
- `paths` – Liste des chemins complets vers les champs à fractionner (obligatoire).
- `name1` – nom à attribuer au `DynamicFrame` qui contiendra les champs à fractionner (facultatif). Si aucun nom n'est fourni, le nom de la trame source est utilisé, « 1 » étant ajouté.
- `name2` – nom à attribuer au `DynamicFrame` qui contient les champs qui restent après le fractionnement des champs spécifiés (facultatif). Si aucun nom n'est fourni, le nom de la trame source est utilisé, « 2 » étant ajouté.
- `transformation_ctx` – Chaîne unique utilisée pour identifier les informations sur l'état (facultatif).
- `info` – Chaîne associée à des erreurs dans la transformation (facultatif).
- `stageThreshold` – nombre maximal d'erreurs qui peuvent avoir lieu dans la transformation avant qu'elle ne soit arrêtée.(facultatif) La valeur par défaut est zéro.
- `totalThreshold` – nombre maximal d'erreurs pouvant se produire globalement avant que le processus de traitement des erreurs ne soit arrêté.(facultatif) La valeur par défaut est zéro.

apply(cls, *args, **kwargs)

Hérité de `GlueTransform` s'appliquent.

name(cls)

Hérité de `GlueTransform` name.

describeArgs(cls)

Hérité de `GlueTransform` describeArgs.

describeReturn(cls)

Hérité de `GlueTransform` describeReturn.

describeTransform(cls)

Hérité de `GlueTransform` describeTransform.

describeErrors(cls)

Hérité de `GlueTransform` describeErrors.

describe(cls)

Hérité de `GlueTransform` describe.

SplitRows classe

Crée une collection `DynamicFrameCollection` qui contient deux images `DynamicFrames`. Une image `DynamicFrame` contient uniquement les champs spécifiés à fractionner et l'autre contient les champs restants.

Exemple

Nous vous recommandons d'utiliser la méthode `DynamicFrame.split_rows()` pour fractionner les lignes en image `DynamicFrame`. Vous trouverez un exemple de code, consultez Exemple : utiliser la méthode split_rows pour fractionner des lignes en une image `DynamicFrame`.

Méthodes

- __call__
- s'appliquent
- name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__(frame, comparison_dict, name1="frame1", name2="frame2", transformation_ctx = "", info = None, stageThreshold = 0, totalThreshold = 0)

Scinde une ou plusieurs lignes d'un objet `DynamicFrame` en un nouveau `DynamicFrame`.

- `frame` – `DynamicFrame` source à scinder en deux nouveaux objets (obligatoire).
- `comparison_dict` – Dictionnaire dans lequel la clé est le chemin d'accès complet à une colonne et la valeur est un autre dictionnaire permettant de mapper les comparateurs aux valeurs auxquelles les valeurs de la colonne sont comparées. Par exemple, `{"age": {">": 10, "<": 20}}` scinde les lignes où la valeur de « age » est comprise entre 10 et 20 (exclus), à partir des lignes où « age » est en dehors de la plage (obligatoire).

- `name1` – Nom à attribuer à l'objet `DynamicFrame` qui contient les lignes à scinder (facultatif).
- `name2` - Nom à attribuer à l'objet `DynamicFrame` qui contient les lignes qui restent après que les lignes spécifiées ont été scindées (facultatif).
- `transformation_ctx` – Chaîne unique utilisée pour identifier les informations sur l'état (facultatif).
- `info` – Chaîne associée à des erreurs dans la transformation (facultatif).
- `stageThreshold` – nombre maximal d'erreurs qui peuvent avoir lieu dans la transformation avant qu'elle ne soit arrêtée.(facultatif) La valeur par défaut est zéro.
- `totalThreshold` – nombre maximal d'erreurs pouvant se produire globalement avant que le processus de traitement des erreurs ne soit arrêté.(facultatif) La valeur par défaut est zéro.

apply(cls, *args, **kwargs)

Hérité de `GlueTransform` s'appliquent.

name(cls)

Hérité de `GlueTransform` name.

describeArgs(cls)

Hérité de `GlueTransform` describeArgs.

describeReturn(cls)

Hérité de `GlueTransform` describeReturn.

describeTransform(cls)

Hérité de `GlueTransform` describeTransform.

describeErrors(cls)

Hérité de `GlueTransform` describeErrors.

describe(cls)

Hérité de `GlueTransform` describe.

Classe Unbox

Effectue une opération unbox sur un champ de chaîne (ou le reformate) dans une image `DynamicFrame`.

Exemple

Nous vous recommandons d'utiliser la méthode `DynamicFrame.unbox()` pour effectuer une opération unbox sur un champ dans une image `DynamicFrame`. Vous trouverez un exemple de code, consultez Exemple : utiliser la méthode unbox pour effectuer une opération unbox sur un champ de chaîne en un champ struct.

Méthodes

- __call__
- s'appliquent
- name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__(frame, path, format, transformation_ctx = "", info="", stageThreshold=0, totalThreshold=0, **options)

Effectue une opération unbox sur un champ de chaîne dans un `DynamicFrame`.

- `frame` – `DynamicFrame` dans lequel effectuer une opération unbox sur un champ (obligatoire).
- `path` - Chemin d'accès complet au `StringNode` sur lequel effectuer une opération unbox (obligatoire).
- `format` – spécification de format (facultatif). Utilisée pour une connexion Amazon S3 ou AWS Glue prenant en charge plusieurs formats. Consultez Options de format pour les entrées et sorties dans AWS Glue pour Spark pour connaître les formats pris en charge.
- `transformation_ctx` – Chaîne unique utilisée pour identifier les informations sur l'état (facultatif).

- `info` – Chaîne associée à des erreurs dans la transformation (facultatif).
- `stageThreshold` – nombre maximal d'erreurs qui peuvent avoir lieu dans la transformation avant qu'elle ne soit arrêtée.(facultatif) La valeur par défaut est zéro.
- `totalThreshold` – nombre maximal d'erreurs pouvant se produire globalement avant que le processus de traitement des erreurs ne soit arrêté.(facultatif) La valeur par défaut est zéro.
- `separator` - Jeton de séparateur (facultatif).
- `escaper` – Jeton d'échappement (facultatif).
- `skipFirst` - `True` si la première ligne de données doit être ignorée, ou `False` dans le cas contraire (facultatif).
- withSchema – Chaîne contenant un schéma pour les données sur lesquelles effectuer l'opération unbox (facultatif). Doit toujours être créée à l'aide de `StructType.json`.
- `withHeader` - `True` si les données à décompresser comprennent un en-tête, ou `False` dans le cas contraire (facultatif).

apply(cls, *args, **kwargs)

Hérité de `GlueTransform` s'appliquent.

name(cls)

Hérité de `GlueTransform` name.

describeArgs(cls)

Hérité de `GlueTransform` describeArgs.

describeReturn(cls)

Hérité de `GlueTransform` describeReturn.

describeTransform(cls)

Hérité de `GlueTransform` describeTransform.

describeErrors(cls)

Hérité de `GlueTransform` describeErrors.

describe(cls)

Hérité de `GlueTransform` describe.

UnnestFrame classe

Désimbrique une image `DynamicFrame`, aplatit les objets imbriqués en éléments de niveau supérieur et génère les clés de jointure pour les objets de tableau.

Exemple

Nous vous recommandons d'utiliser la méthode `DynamicFrame.unnest()` pour aplatir les structures imbriquées dans une image `DynamicFrame`. Vous trouverez un exemple de code, consultez Exemple : utiliser la méthode unnest pour transformer des champs imbriqués en champs de niveau supérieur.

Méthodes

- __call__
- s'appliquent
- name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__(frame, transformation_ctx = "", info="", stageThreshold=0, totalThreshold=0)

Désimbrique une image `DynamicFrame`, aplatit les objets imbriqués en éléments de niveau supérieur et génère les clés de jointure pour les objets de tableau.

- `frame` – Objet `DynamicFrame` à désimbriquer (requis).
- `transformation_ctx` – Chaîne unique utilisée pour identifier les informations sur l'état (facultatif).
- `info` – Chaîne associée à des erreurs dans la transformation (facultatif).

- `stageThreshold` – nombre maximal d'erreurs qui peuvent avoir lieu dans la transformation avant qu'elle ne soit arrêtée.(facultatif) La valeur par défaut est zéro.
- `totalThreshold` – nombre maximal d'erreurs pouvant se produire globalement avant que le processus de traitement des erreurs ne soit arrêté.(facultatif) La valeur par défaut est zéro.

apply(cls, *args, **kwargs)

Hérité de `GlueTransform` s'appliquent.

name(cls)

Hérité de `GlueTransform` name.

describeArgs(cls)

Hérité de `GlueTransform` describeArgs.

describeReturn(cls)

Hérité de `GlueTransform` describeReturn.

describeTransform(cls)

Hérité de `GlueTransform` describeTransform.

describeErrors(cls)

Hérité de `GlueTransform` describeErrors.

describe(cls)

Hérité de `GlueTransform` describe.

FlagDuplicatesInColumn classe

La transformation `FlagDuplicatesInColumn` renvoie une nouvelle colonne avec une valeur spécifiée dans chaque ligne qui indique si la valeur de la colonne source de la ligne correspond à une valeur d'une ligne précédente de la colonne source. Lorsque des correspondances sont trouvées, elles sont signalées comme des doublons. L'occurrence initiale n'est pas signalée, car elle ne correspond pas à une ligne précédente.

Exemple

```
from pyspark.context import SparkContext
from pyspark.sql import SparkSession
from awsgluedi.transforms import *

sc = SparkContext()
spark = SparkSession(sc)

datasource1 = spark.read.json("s3://${BUCKET}/json/zips/raw/data")

try:
    df_output = column.FlagDuplicatesInColumn.apply(
        data_frame=datasource1,
        spark_context=sc,
        source_column="city",
        target_column="flag_col",
        true_string="True",
        false_string="False"
    )
except:
    print("Unexpected Error happened ")
    raise
```

Output

La `FlagDuplicatesInColumn` transformation ajoutera une nouvelle colonne `flag_col` à la `df_output`. DataFrame Cette colonne contiendra une valeur de chaîne indiquant si la ligne correspondante possède une valeur dupliquée dans la colonne `city` ou non. Si une ligne contient une valeur `city` dupliquée, la valeur `flag_col` contiendra la valeur `true_string` « True ». Si une ligne possède une valeur `city` unique, la valeur `flag_col` contiendra la valeur `false_string` « False ».

Le `df_output` qui en DataFrame résultera contiendra toutes les colonnes du `datasource1` original, plus la colonne `flag_col` supplémentaire indiquant les valeurs dupliquées de `city`. DataFrame

Méthodes

- __call__
- s'appliquent
- name

- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__(spark_context, data_frame, source_column, target_column, true_string=DEFAULT_TRUE_STRING, false_string=DEFAULT_FALSE_STRING)

La transformation `FlagDuplicatesInColumn` renvoie une nouvelle colonne avec une valeur spécifiée dans chaque ligne qui indique si la valeur de la colonne source de la ligne correspond à une valeur d'une ligne précédente de la colonne source. Lorsque des correspondances sont trouvées, elles sont signalées comme des doublons. L'occurrence initiale n'est pas signalée, car elle ne correspond pas à une ligne précédente.

- `source_column` : nom de la colonne source.
- `target_column` : nom de la colonne cible.
- `true_string` : chaîne à insérer dans la colonne cible lorsqu'une valeur de colonne source duplique une valeur antérieure dans cette colonne.
- `false_string` : chaîne à insérer dans la colonne cible lorsqu'une valeur de colonne source est différente des valeurs précédentes dans cette colonne.

apply(cls, *args, **kwargs)

Hérité de `GlueTransform` s'appliquent.

name(cls)

Hérité de `GlueTransform` name.

describeArgs(cls)

Hérité de `GlueTransform` describeArgs.

describeReturn(cls)

Hérité de `GlueTransform` describeReturn.

describeTransform(cls)

Hérité de `GlueTransform` describeTransform.

describeErrors(cls)

Hérité de `GlueTransform` describeErrors.

describe(cls)

Hérité de `GlueTransform` describe.

FormatPhoneNumber classe

La transformation `FormatPhoneNumber` renvoie une colonne dans laquelle une chaîne de numéro de téléphone est convertie en valeur formatée.

Exemple

```
from pyspark.context import SparkContext
from pyspark.sql import SparkSession
from awsgluedi.transforms import *

sc = SparkContext()
spark = SparkSession(sc)

input_df = spark.createDataFrame(
    [
        ("408-341-5669",),
        ("4083415669",)
    ],
    ["phone"],
)

try:
    df_output = column_formatting.FormatPhoneNumber.apply(
        data_frame=input_df,
        spark_context=sc,
        source_column="phone",
        default_region="US"
    )
    df_output.show()
except:
    print("Unexpected Error happened ")
```

```
    raise
```

Output

Le résultat sera :

```
```
+---------------+
| phone|
+---------------+
|(408) 341-5669|
|(408) 341-5669|
+---------------+
```
```

La transformation `FormatPhoneNumber` prend la valeur `source_column` `"phone"` et la valeur `default_region` `"US"`.

La transformation permet de formater correctement les deux numéros de téléphone, quel que soit leur format initial, au format étasunien standard `(408) 341-5669`.

Méthodes

- __call__
- s'appliquent
- name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__(spark_context, data_frame, source_column, phone_number_format=None, default_region=None, default_region_column=None)

La transformation `FormatPhoneNumber` renvoie une colonne dans laquelle une chaîne de numéro de téléphone est convertie en valeur formatée.

- `source_column` : nom d'une colonne existante.
- `phone_number_format` : format dans lequel le numéro de téléphone sera converti. Si aucun format n'est spécifié, le format par défaut est `E.164`, un format de numéro de téléphone standard reconnu à l'échelle internationale. Les valeurs valides sont les suivantes :
  - E164 (omettre le point après E)
- `default_region` : code de région valide composé de deux ou trois lettres majuscules qui indique la région du numéro de téléphone lorsque aucun code de pays n'est présent dans le numéro lui-même. Tout au plus, une des `defaultRegion` ou `defaultRegionColumn` peut être fournie.
- `default_region_column` : nom d'une colonne de type de données avancé `Country`. Le code de région de la colonne spécifiée est utilisé pour déterminer le code de pays pour le numéro de téléphone lorsque aucun code de pays n'est présent dans le numéro lui-même. Tout au plus, une des `defaultRegion` ou `defaultRegionColumn` peut être fournie.

apply(cls, *args, **kwargs)

Hérité de `GlueTransform` s'appliquent.

name(cls)

Hérité de `GlueTransform` name.

describeArgs(cls)

Hérité de `GlueTransform` describeArgs.

describeReturn(cls)

Hérité de `GlueTransform` describeReturn.

describeTransform(cls)

Hérité de `GlueTransform` describeTransform.

describeErrors(cls)

Hérité de `GlueTransform` describeErrors.

describe(cls)

Hérité de `GlueTransform` describe.

FormatCase classe

La transformation `FormatCase` remplace chaque chaîne d'une colonne par le type de cas spécifié.

Exemple

```
from pyspark.context import SparkContext
from pyspark.sql import SparkSession
from awsgluedi.transforms import *

sc = SparkContext()
spark = SparkSession(sc)

datasource1 = spark.read.json("s3://${BUCKET}/json/zips/raw/data")

try:
    df_output = data_cleaning.FormatCase.apply(
        data_frame=datasource1,
        spark_context=sc,
        source_column="city",
        case_type="LOWER"
    )
except:
    print("Unexpected Error happened ")
    raise
```

Output

La transformation `FormatCase` convertira les valeurs de la colonne `city` en minuscules en fonction du paramètre `case_type="LOWER"`. Le `df_output` qui en DataFrame résultera contiendra toutes les colonnes du `datasource1` original, mais avec les valeurs des colonnes `city` DataFrame en minuscules.

Méthodes

- __call__
- s'appliquent
- name
- describeArgs
- describeReturn

- describeTransform
- describeErrors
- describe

__call__(spark_context, data_frame, source_column, case_type)

La transformation `FormatCase` remplace chaque chaîne d'une colonne par le type de cas spécifié.

- `source_column` : nom d'une colonne existante.
- `case_type` : les types de casse pris en charge sont `CAPITAL`, `LOWER`, `UPPER` et `SENTENCE`.

apply(cls, *args, **kwargs)

Hérité de `GlueTransform` s'appliquent.

name(cls)

Hérité de `GlueTransform` name.

describeArgs(cls)

Hérité de `GlueTransform` describeArgs.

describeReturn(cls)

Hérité de `GlueTransform` describeReturn.

describeTransform(cls)

Hérité de `GlueTransform` describeTransform.

describeErrors(cls)

Hérité de `GlueTransform` describeErrors.

describe(cls)

Hérité de `GlueTransform` describe.

FillWithMode classe

La transformation `FillWithMode` met en forme une colonne en fonction du format de numéro de téléphone que vous spécifiez. Vous pouvez également définir une logique de disjoncteur de lien,

dans laquelle certaines valeurs sont identiques. Prenons l'exemple des valeurs suivantes : `1 2 2 3 3 4`.

Une valeur modeType de `MINIMUM` entraîne le renvoi par `FillWithMode` de 2 comme valeur de mode. Si la valeur modeType est définie sur `MAXIMUM`, le mode est 3. Pour `AVERAGE` le mode est 2,5.

Exemple

```
from awsglue.context import *
from pyspark.sql import SparkSession
from awsgluedi.transforms import *

sc = SparkContext()
spark = SparkSession(sc)

input_df = spark.createDataFrame(
    [
        (105.111, 13.12),
        (1055.123, 13.12),
        (None, 13.12),
        (13.12, 13.12),
        (None, 13.12),
    ],
    ["source_column_1", "source_column_2"],
)

try:
    df_output = data_quality.FillWithMode.apply(
        data_frame=input_df,
        spark_context=sc,
        source_column="source_column_1",
        mode_type="MAXIMUM"
    )
    df_output.show()
except:
    print("Unexpected Error happened ")
    raise
```

Output

La sortie du code donné sera :

```
```
+---------------+---------------+
|source_column_1|source_column_2|
+---------------+---------------+
105.111	13.12
1055.123	13.12
1055.123	13.12
13.12	13.12
1055.123	13.12
+---------------+---------------+
```
```

La `FillWithMode` transformation du module `awsglue.data_quality` est appliquée au `input_df`. DataFrame Elle remplace les valeurs `null` de la colonne `source_column_1` par la valeur maximale (`mode_type="MAXIMUM"`) parmi les valeurs non nulles de cette colonne.

Dans ce cas, la valeur maximale de la colonne `source_column_1` est `1 055,123`. Par conséquent, les valeurs `null` `source_column_1` sont remplacées par `1055,123` dans la sortie `df_output`. DataFrame

Méthodes

- __call__
- s'appliquent
- name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__(spark_context, data_frame, source_column, mode_type)

La transformation `FillWithMode` met en forme la casse des chaînes dans une colonne.

- `source_column` : nom d'une colonne existante.
- `mode_type` : comment résoudre les valeurs à égalité dans les données. Cette valeur doit être l'une des valeurs suivantes : `MINIMUM`, `NONE`, `AVERAGE` ou `MAXIMUM`.

apply(cls, *args, **kwargs)

Hérité de `GlueTransform` s'appliquent.

name(cls)

Hérité de `GlueTransform` name.

describeArgs(cls)

Hérité de `GlueTransform` describeArgs.

describeReturn(cls)

Hérité de `GlueTransform` describeReturn.

describeTransform(cls)

Hérité de `GlueTransform` describeTransform.

describeErrors(cls)

Hérité de `GlueTransform` describeErrors.

describe(cls)

Hérité de `GlueTransform` describe.

FlagDuplicateRows classe

La transformation `FlagDuplicateRows` renvoie une nouvelle colonne avec une valeur spécifiée dans chaque ligne qui indique si cette ligne correspond exactement à une ligne précédente dans le jeu de données. Lorsque des correspondances sont trouvées, elles sont signalées comme des doublons. L'occurrence initiale n'est pas signalée, car elle ne correspond pas à une ligne précédente.

Exemple

```
from pyspark.context import SparkContext
from pyspark.sql import SparkSession
from awsgluedi.transforms import *

sc = SparkContext()
```

```
spark = SparkSession(sc)

input_df = spark.createDataFrame(
    [
        (105.111, 13.12),
        (13.12, 13.12),
        (None, 13.12),
        (13.12, 13.12),
        (None, 13.12),
    ],
    ["source_column_1", "source_column_2"],
)

try:
    df_output = data_quality.FlagDuplicateRows.apply(
        data_frame=input_df,
        spark_context=sc,
        target_column="flag_row",
        true_string="True",
        false_string="False",
        target_index=1
    )
except:
    print("Unexpected Error happened ")
    raise
```

Output

La sortie sera une PySpark DataFrame avec une colonne supplémentaire `flag_row` qui indique si une ligne est un doublon ou non, en fonction de la `source_column_1` colonne. Le fichier `df_output` qui en résultera DataFrame contiendra les lignes suivantes :

```
```
+---------------+---------------+--------+
|source_column_1|source_column_2|flag_row|
+---------------+---------------+--------+
105.111	13.12	False
13.12	13.12	True
null	13.12	True
13.12	13.12	True
null	13.12	True
+---------------+---------------+--------+
```
```

```
```

La colonne `flag_row` indique si une ligne est un doublon ou non. La valeur `true_string` est définie sur « True » et `false_string` sur « False ». Le `target_index` est défini sur 1, ce qui signifie que la `flag_row` colonne sera insérée à la deuxième position (index 1) dans la sortie. DataFrame

Méthodes

- __call__
- s'appliquent
- name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__(spark_context, data_frame, target_column, true_string=DEFAULT_TRUE_STRING, false_string=DEFAULT_FALSE_STRING, target_index=None)

La transformation `FlagDuplicateRows` renvoie une nouvelle colonne avec une valeur spécifiée dans chaque ligne qui indique si cette ligne correspond exactement à une ligne précédente dans le jeu de données. Lorsque des correspondances sont trouvées, elles sont signalées comme des doublons. L'occurrence initiale n'est pas signalée, car elle ne correspond pas à une ligne précédente.

- `true_string` : valeur à insérer si la ligne correspond à une ligne précédente.
- `false_string` : valeur à insérer si la ligne est unique.
- `target_column` : nom de la nouvelle colonne insérée dans le jeu de données.

apply(cls, *args, **kwargs)

Hérité de `GlueTransform` s'appliquent.

name(cls)

Hérité de `GlueTransform` name.

describeArgs(cls)

Hérité de `GlueTransform` describeArgs.

describeReturn(cls)

Hérité de `GlueTransform` describeReturn.

describeTransform(cls)

Hérité de `GlueTransform` describeTransform.

describeErrors(cls)

Hérité de `GlueTransform` describeErrors.

describe(cls)

Hérité de `GlueTransform` describe.

RemoveDuplicates classe

La transformation `RemoveDuplicates` supprime une ligne entière si une valeur dupliquée est détectée dans une colonne source sélectionnée.

Exemple

```
from pyspark.context import SparkContext
from pyspark.sql import SparkSession
from awsgluedi.transforms import *

sc = SparkContext()
spark = SparkSession(sc)

input_df = spark.createDataFrame(
    [
        (105.111, 13.12),
        (13.12, 13.12),
        (None, 13.12),
        (13.12, 13.12),
        (None, 13.12),
    ],
```

```
    ["source_column_1", "source_column_2"],
)

try:
    df_output = data_quality.RemoveDuplicates.apply(
        data_frame=input_df,
        spark_context=sc,
        source_column="source_column_1"
    )
except:
    print("Unexpected Error happened ")
    raise
```

Output

Le résultat sera un PySpark DataFrame avec les doublons supprimés en fonction de la `source_column_1` colonne. Le fichier `df_output` qui en résultera DataFrame contiendra les lignes suivantes :

```
```
+---------------+---------------+
|source_column_1|source_column_2|
+---------------+---------------+
105.111	13.12
13.12	13.12
null	13.12
+---------------+---------------+
```
```

Notez que les lignes avec les `source_column_1` valeurs « 13.12 » et « null » n'apparaissent qu'une seule fois dans la sortie DataFrame, car les doublons ont été supprimés en fonction de la colonne. `source_column_1`

Méthodes

- __call__
- s'appliquent
- name
- describeArgs

- describeReturn
- describeTransform
- describeErrors
- describe

__call__(spark_context, data_frame, source_column)

La transformation `RemoveDuplicates` supprime une ligne entière si une valeur dupliquée est détectée dans une colonne source sélectionnée.

- `source_column` : nom d'une colonne existante.

apply(cls, *args, **kwargs)

Hérité de `GlueTransform` s'appliquent.

name(cls)

Hérité de `GlueTransform` name.

describeArgs(cls)

Hérité de `GlueTransform` describeArgs.

describeReturn(cls)

Hérité de `GlueTransform` describeReturn.

describeTransform(cls)

Hérité de `GlueTransform` describeTransform.

describeErrors(cls)

Hérité de `GlueTransform` describeErrors.

describe(cls)

Hérité de `GlueTransform` describe.

MonthName classe

La transformation `MonthName` crée une colonne contenant le nom du mois, à partir d'une chaîne représentant une date.

Exemple

```
from pyspark.context import SparkContext
from pyspark.sql import SparkSession
from awsgluedi.transforms import *

sc = SparkContext()
spark = SparkSession(sc)

spark.conf.set("spark.sql.legacy.timeParserPolicy", "LEGACY")

input_df = spark.createDataFrame(
    [
        ("20-2018-12",),
        ("2018-20-12",),
        ("20182012",),
        ("12202018",),
        ("20122018",),
        ("20-12-2018",),
        ("12/20/2018",),
        ("02/02/02",),
        ("02 02 2009",),
        ("02/02/2009",),
        ("August/02/2009",),
        ("02/june/2009",),
        ("02/2020/june",),
        ("2013-02-21 06:35:45.658505",),
        ("August 02 2009",),
        ("2013/02/21",),
        (None,),
    ],
    ["column_1"],
)

try:
    df_output = datetime_functions.MonthName.apply(
        data_frame=input_df,
        spark_context=sc,
        source_column="column_1",
```

```
        target_column="target_column"
    )
    df_output.show()
except:
    print("Unexpected Error happened ")
    raise
```

Output

Le résultat sera :

```
```
+------------+------------+
| column_1|target_column|
+------------+------------+
20-2018-12	December
2018-20-12	null
20182012	null
12202018	null
20122018	null
20-12-2018	December
12/20/2018	December
02/02/02	February
02 02 2009	February
02/02/2009	February
August/02/2009	August
02/june/2009	null
02/2020/june	null
2013-02-21 06:35:45.658505	February
August 02 2009	August
2013/02/21	February
null	null
+------------+------------+
```
```

La transformation `MonthName` prend la valeur `source_column` `"column_1"` et la valeur `target_column` `"target_column"`. Il tente d'extraire le nom du mois à partir des date/time chaînes de la colonne `"column_1"` et le place dans la colonne `"target_column"`. Si le format de la date/time chaîne n'est pas reconnu ou ne peut pas être analysée, la valeur « target_column » est définie sur « null`.

La transformation extrait avec succès le nom du mois à partir de différents date/time formats, tels que « 20-12-2018 », « 20/12/2018 », « 02/02/2009 », « 2013-02-21 06:35:45.658 505 » et « 02 août 2009 ».

Méthodes

- __call__
- s'appliquent
- name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__(spark_context, data_frame, target_column, source_column=None, value=None)

La transformation `MonthName` crée une colonne contenant le nom du mois, à partir d'une chaîne représentant une date.

- `source_column` : nom d'une colonne existante.
- `value` : chaîne de caractères à évaluer.
- `target_column` : nom de la colonne qui vient d'être créée.

apply(cls, *args, **kwargs)

Hérité de `GlueTransform` s'appliquent.

name(cls)

Hérité de `GlueTransform` name.

describeArgs(cls)

Hérité de `GlueTransform` describeArgs.

describeReturn(cls)

Hérité de `GlueTransform` describeReturn.

describeTransform(cls)

Hérité de `GlueTransform` describeTransform.

describeErrors(cls)

Hérité de `GlueTransform` describeErrors.

describe(cls)

Hérité de `GlueTransform` describe.

IsEven classe

La transformation `IsEven` renvoie une valeur booléenne dans une nouvelle colonne qui indique si la colonne ou la valeur source est paire. Si la colonne ou la valeur source est décimale, le résultat est false.

Exemple

```
from pyspark.context import SparkContext
from pyspark.sql import SparkSession
from awsgluedi.transforms import *

sc = SparkContext()
spark = SparkSession(sc)

input_df = spark.createDataFrame(
    [(5,), (0,), (-1,), (2,), (None,)],
    ["source_column"],
)

try:
    df_output = math_functions.IsEven.apply(
        data_frame=input_df,
        spark_context=sc,
        source_column="source_column",
        target_column="target_column",
        value=None,
        true_string="Even",
        false_string="Not even",
    )
    df_output.show()
```

```
except:
    print("Unexpected Error happened ")
    raise
```

Output

Le résultat sera :

```
```
+------------+------------+
|source_column|target_column|
+------------+------------+
5	Not even
0	Even
-1	Not even
2	Even
null	null
+------------+------------+
```
```

La transformation `IsEven` prend la valeur `source_column` "source_column" et la valeur `target_column` "target_column". Elle vérifie si la valeur de `"source_column"` est paire ou non. Si la valeur est paire, elle définit la valeur `"target_column"` sur `true_string` "Even". Si la valeur est impaire, elle définit la valeur `"target_column"` sur `false_string` "Not even". Si la valeur `"source_column"` est `null`, la valeur `"target_column"` est définie sur `null`.

La transformation identifie correctement les nombres pairs (0 et 2) et définit la valeur `"target_column"` sur "Even". Pour les nombres impairs (5 et -1), elle définit la valeur `"target_column"` sur "Not even". Pour la valeur `null` dans `"source_column"`, la valeur `"target_column"` est définie sur `null`.

Méthodes

- __call__
- s'appliquent
- name
- describeArgs
- describeReturn

- describeTransform
- describeErrors
- describe

__call__(spark_context, data_frame, target_column, source_column=None, true_string=DEFAULT_TRUE_STRING, false_string=DEFAULT_FALSE_STRING, value=None)

La transformation `IsEven` renvoie une valeur booléenne dans une nouvelle colonne qui indique si la colonne ou la valeur source est paire. Si la colonne ou la valeur source est décimale, le résultat est false.

- `source_column` : nom d'une colonne existante.
- `target_column` : le nom de la nouvelle colonne à créer.
- `true_string` : chaîne indiquant si la valeur est paire.
- `false_string` : chaîne indiquant si la valeur est impaire.

apply(cls, *args, **kwargs)

Hérité de `GlueTransform` s'appliquent.

name(cls)

Hérité de `GlueTransform` name.

describeArgs(cls)

Hérité de `GlueTransform` describeArgs.

describeReturn(cls)

Hérité de `GlueTransform` describeReturn.

describeTransform(cls)

Hérité de `GlueTransform` describeTransform.

describeErrors(cls)

Hérité de `GlueTransform` describeErrors.

describe(cls)

Hérité de `GlueTransform` describe.

CryptographicHash classe

La transformation `CryptographicHash` applique un algorithme aux valeurs de hachage de la colonne.

Exemple

```
from pyspark.context import SparkContext
from pyspark.sql import SparkSession
from awsgluedi.transforms import *

secret = "${SECRET}"
sc = SparkContext()
spark = SparkSession(sc)

input_df = spark.createDataFrame(
    [
        (1, "1234560000"),
        (2, "1234560001"),
        (3, "1234560002"),
        (4, "1234560003"),
        (5, "1234560004"),
        (6, "1234560005"),
        (7, "1234560006"),
        (8, "1234560007"),
        (9, "1234560008"),
        (10, "1234560009"),
    ],
    ["id", "phone"],
)

try:
    df_output = pii.CryptographicHash.apply(
        data_frame=input_df,
        spark_context=sc,
        source_columns=["id", "phone"],
        secret_id=secret,
        algorithm="HMAC_SHA256",
        output_format="BASE64",
    )
```

```
    df_output.show()
except:
    print("Unexpected Error happened ")
    raise
```

Output

Le résultat sera :

```
```
+---+------------+------------------+------------------+
| id| phone | id_hashed | phone_hashed |
+---+------------+------------------+------------------+
1	1234560000	QUI1zXTJiXmfIb...	juDBAmiRnnO3g...
2	1234560001	ZAUWiZ3dVTzCo...	vC8lgUqBVDMNQ...
3	1234560002	ZP4VvZWkqYifu...	Kl3QAkgswYpzB...
4	1234560003	3u8vO3wQ8EQfj...	CPBzK1P8PZZkV...
5	1234560004	eWkQJk4zAOIzx...	aLf7+mHcXqbLs...
6	1234560005	xtI9fZCJZCvsa...	dy2DFgdYWmr0p...
7	1234560006	iW9hew7jnHuOf...	wwfGMCOEv6oOv...
8	1234560007	H9V1pqvgkFhfS...	g9WKhagIXy9ht...
9	1234560008	xDhEuHaxAUbU5...	b3uQLKPY+Q5vU...
10	1234560009	GRN6nFXkxk349...	VJdsKt8VbxBbt...
+---+------------+------------------+------------------+
```
```

La transformation calcule les hachages cryptographiques des valeurs des colonnes `id` et `phone` à l'aide de l'algorithme et de la clé secrète spécifiés, et code les hachages au format Base64. Le `df_output` qui en résulte DataFrame contient toutes les colonnes du `input_df` original, plus les colonnes supplémentaires `id_hashed` et `phone_hashed` avec DataFrame les hachages calculés.

Méthodes

- __call__
- s'appliquent
- name
- describeArgs
- describeReturn
- describeTransform

- describeErrors
- describe

__call__(spark_context, data_frame, source_columns, secret_id, algorithm=None, secret_version=None, create_secret_if_missing=False, output_format=None, entity_type_filter=None)

La transformation `CryptographicHash` applique un algorithme aux valeurs de hachage de la colonne.

- `source_columns` : tableau de colonnes existantes.
- `secret_id` : ARN de la clé secrète Secrets Manager Clé utilisée dans l'algorithme de préfixe de code d'authentification de message utilisant hash (HMAC) pour hacher les colonnes source.
- `secret_version` : facultatif. Par défaut, la version du secret la plus récente.
- `entity_type_filter` : tableau facultatif de types d'entité. Peut être utilisé pour chiffrer uniquement les informations personnelles détectées dans une colonne en texte libre.
- `create_secret_if_missing` : valeur booléenne facultative. Si la valeur est true, il essaiera de créer le secret au nom de l'appelant.
- `algorithm` : l'algorithme utilisé pour hacher vos données. Valeurs d'énumération valides : MD5,,, SHA1, HMAC_ SHA256 SHA512, HMAC_MD5, HMAC_, HMAC_SHA1. SHA256 SHA512

apply(cls, *args, **kwargs)

Hérité de `GlueTransform` s'appliquent.

name(cls)

Hérité de `GlueTransform` name.

describeArgs(cls)

Hérité de `GlueTransform` describeArgs.

describeReturn(cls)

Hérité de `GlueTransform` describeReturn.

describeTransform(cls)

Hérité de `GlueTransform` describeTransform.

describeErrors(cls)

Hérité de `GlueTransform` describeErrors.

describe(cls)

Hérité de `GlueTransform` describe.

Classe de déchiffrement

La `Decrypt` transformation déchiffre l'intérieur de Glue. AWS Vos données peuvent également être déchiffrées en dehors de AWS Glue avec le AWS SDK de chiffrement. Si l'ARN de la clé KMS fourni ne correspond pas à celui utilisé pour chiffrer la colonne, l'opération de déchiffrement échoue.

Exemple

```
from pyspark.context import SparkContext
from pyspark.sql import SparkSession
from awsgluedi.transforms import *

kms = "${KMS}"
sc = SparkContext()
spark = SparkSession(sc)

input_df = spark.createDataFrame(
    [
        (1, "1234560000"),
        (2, "1234560001"),
        (3, "1234560002"),
        (4, "1234560003"),
        (5, "1234560004"),
        (6, "1234560005"),
        (7, "1234560006"),
        (8, "1234560007"),
        (9, "1234560008"),
        (10, "1234560009"),
    ],
    ["id", "phone"],
)

try:
    df_encrypt = pii.Encrypt.apply(
        data_frame=input_df,
```

```
        spark_context=sc,
        source_columns=["phone"],
        kms_key_arn=kms
    )
    df_decrypt = pii.Decrypt.apply(
        data_frame=df_encrypt,
        spark_context=sc,
        source_columns=["phone"],
        kms_key_arn=kms
    )
    df_decrypt.show()
except:
    print("Unexpected Error happened ")
    raise
```

Output

La sortie sera une PySpark DataFrame avec la colonne « id » d'origine et la colonne « phone » déchiffrée :

```
```
+---+------------+
| id| phone|
+---+------------+
1	1234560000
2	1234560001
3	1234560002
4	1234560003
5	1234560004
6	1234560005
7	1234560006
8	1234560007
9	1234560008
10	1234560009
+---+------------+
```
```

La transformation `Encrypt` prend la valeur `source_columns` comme `["phone"]` et `kms_key_arn` comme valeur de la variable d'environnement `$ {KMS}`. La transformation chiffre les valeurs de la colonne `phone` à l'aide de la clé KMS spécifiée. Le DataFrame `df_encrypt` crypté est ensuite transmis à la `Decrypt` transformation depuis le module `awsglue.pii`. Il prend la valeur

`source_columns` comme `["phone"]` et `kms_key_arn` comme valeur de la variable d'environnement `${KMS}`. La transformation déchiffre les valeurs chiffrées de la colonne `phone` à l'aide de la même clé KMS. Le fichier `df_decrypt` qui en résulte DataFrame contient la colonne `id` d'origine et la colonne `phone` décryptée.

Méthodes

- __call__
- s'appliquent
- name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__(spark_context, data_frame, source_columns, kms_key_arn)

La `Decrypt` transformation déchiffre l'intérieur de Glue. AWS Vos données peuvent également être déchiffrées en dehors de AWS Glue avec le AWS SDK de chiffrement. Si l'ARN de la clé KMS fourni ne correspond pas à celui utilisé pour chiffrer la colonne, l'opération de déchiffrement échoue.

- `source_columns` : tableau de colonnes existantes.
- `kms_key_arn`— L'ARN de la AWS clé du service de gestion des clés à utiliser pour déchiffrer les colonnes source.

apply(cls, *args, **kwargs)

Hérité de `GlueTransform` s'appliquent.

name(cls)

Hérité de `GlueTransform` name.

describeArgs(cls)

Hérité de `GlueTransform` describeArgs.

describeReturn(cls)

Hérité de `GlueTransform` describeReturn.

describeTransform(cls)

Hérité de `GlueTransform` describeTransform.

describeErrors(cls)

Hérité de `GlueTransform` describeErrors.

describe(cls)

Hérité de `GlueTransform` describe.

Classe de chiffrement

La `Encrypt` transformation chiffre les colonnes source à l'aide de la AWS clé du service de gestion des clés. La transformation `Encrypt` peut chiffrer jusqu'à 128 Mio par cellule. Elle tentera de préserver le format lors du déchiffrement. Pour préserver le type de données, les métadonnées du type de données doivent être sérialisées à moins de 1 Ko. Dans le cas contraire, vous devez définir le paramètre `preserve_data_type` sur false. Les métadonnées du type de données seront stockées en texte clair dans le contexte du chiffrement.

Exemple

```
from pyspark.context import SparkContext
from pyspark.sql import SparkSession
from awsgluedi.transforms import *

kms = "${KMS}"
sc = SparkContext()
spark = SparkSession(sc)

input_df = spark.createDataFrame(
    [
        (1, "1234560000"),
        (2, "1234560001"),
        (3, "1234560002"),
        (4, "1234560003"),
        (5, "1234560004"),
        (6, "1234560005"),
        (7, "1234560006"),
```

```
        (8, "1234560007"),
        (9, "1234560008"),
        (10, "1234560009"),
    ],
    ["id", "phone"],
)

try:
    df_encrypt = pii.Encrypt.apply(
        data_frame=input_df,
        spark_context=sc,
        source_columns=["phone"],
        kms_key_arn=kms
    )
except:
    print("Unexpected Error happened ")
    raise
```

Output

La sortie sera une PySpark DataFrame avec la colonne « id » d'origine et une colonne supplémentaire contenant les valeurs cryptées de la colonne « phone ».

```
```
+---+------------+-------------------------+
| id| phone | phone_encrypted |
+---+------------+-------------------------+
1	1234560000	EncryptedData1234...abc
2	1234560001	EncryptedData5678...def
3	1234560002	EncryptedData9012...ghi
4	1234560003	EncryptedData3456...jkl
5	1234560004	EncryptedData7890...mno
6	1234560005	EncryptedData1234...pqr
7	1234560006	EncryptedData5678...stu
8	1234560007	EncryptedData9012...vwx
9	1234560008	EncryptedData3456...yz0
10	1234560009	EncryptedData7890...123
+---+------------+-------------------------+
```
```

La transformation `Encrypt` prend la valeur `source_columns` comme `["phone"]` et `kms_key_arn` comme valeur de la variable d'environnement `$ {KMS}`. La transformation chiffre les valeurs de la

colonne `phone` à l'aide de la clé KMS spécifiée. Le `df_encrypt` qui en résulte DataFrame contient la colonne `id` d'origine, la colonne `phone` d'origine et une colonne supplémentaire nommée `phone_encrypted` contenant les valeurs cryptées de la colonne `phone`.

Méthodes

- __call__
- s'appliquent
- name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__(spark_context, data_frame, source_columns, kms_key_arn, entity_type_filter=None, preserve_data_type=None)

La `Encrypt` transformation chiffre les colonnes source à l'aide de la AWS clé du service de gestion des clés.

- `source_columns` : tableau de colonnes existantes.
- `kms_key_arn`— L'ARN AWS clé du service de gestion des clés à utiliser pour chiffrer les colonnes source.
- `entity_type_filter` : tableau facultatif de types d'entité. Peut être utilisé pour chiffrer uniquement les informations personnelles détectées dans une colonne en texte libre.
- `preserve_data_type` : valeur booléenne facultative. La valeur par défaut est true (vrai). Si la valeur est false, le type de données ne sera pas stocké.

apply(cls, *args, **kwargs)

Hérité de `GlueTransform` s'appliquent.

name(cls)

Hérité de `GlueTransform` name.

describeArgs(cls)

Hérité de `GlueTransform` describeArgs.

describeReturn(cls)

Hérité de `GlueTransform` describeReturn.

describeTransform(cls)

Hérité de `GlueTransform` describeTransform.

describeErrors(cls)

Hérité de `GlueTransform` describeErrors.

describe(cls)

Hérité de `GlueTransform` describe.

IntToIp classe

La `IntToIp` transformation convertit la valeur entière de la colonne source ou d'une autre valeur en la IPv4 valeur correspondante dans la colonne cible puis renvoie le résultat dans une nouvelle colonne.

Exemple

```
from pyspark.context import SparkContext
from pyspark.sql import SparkSession
from awsgluedi.transforms import *

sc = SparkContext()
spark = SparkSession(sc)

input_df = spark.createDataFrame(
    [
        (3221225473,),
        (0,),
        (1,),
        (100,),
        (168430090,),
```

```
        (4294967295,),
        (4294967294,),
        (4294967296,),
        (-1,),
        (None,),
    ],
    ["source_column_int"],
)

try:
    df_output = web_functions.IntToIp.apply(
        data_frame=input_df,
        spark_context=sc,
        source_column="source_column_int",
        target_column="target_column",
        value=None
    )
    df_output.show()
except:
    print("Unexpected Error happened ")
    raise
```

Output

Le résultat sera :

```
```
+---------------+---------------+
|source_column_int|target_column|
+---------------+---------------+
3221225473	192.0.0.1
0	0.0.0.0
1	0.0.0.1
100	0.0.0.100
168430090	10.0.0.10
4294967295	255.255.255.255
4294967294	255.255.255.254
4294967296	null
-1	null
null	null
+---------------+---------------+
```
```

La `IntToIp.apply` transformation prend la `source_column` comme `'"source_column_int"'` et la `target_column` comme `'"target_column' » et convertit les valeurs entières de la colonne `source_column_int` en leur représentation d'adresse correspondante et stocke le résultat dans la colonne `target_column`. IPv4

Pour les valeurs entières valides comprises dans la plage d' IPv4 adresses (0 à 4294967295), la transformation les convertit correctement en leur représentation d' IPv4 adresse (par exemple, 192.0.0.1, 0.0.0.0, 10.0.0.10, 255.255.255.255).

Pour les valeurs d'entier situées en dehors de la plage valide (par exemple, 4294967296, -1), la valeur `target_column` est définie sur `null`. Pour les valeurs `null` de la colonne `source_column_int`, la valeur `target_column` est également définie sur `null`.

Méthodes

- __call__
- s'appliquent
- name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__(spark_context, data_frame, target_column, source_column=None, value=None)

La `IntToIp` transformation convertit la valeur entière de la colonne source ou d'une autre valeur en la IPv4 valeur correspondante dans la colonne cible puis renvoie le résultat dans une nouvelle colonne.

- `sourceColumn` : nom d'une colonne existante.
- `value` : chaîne de caractères à évaluer.
- `targetColumn` : le nom de la nouvelle colonne à créer.

apply(cls, *args, **kwargs)

Hérité de `GlueTransform` s'appliquent.

name(cls)

Hérité de `GlueTransform` name.

describeArgs(cls)

Hérité de `GlueTransform` describeArgs.

describeReturn(cls)

Hérité de `GlueTransform` describeReturn.

describeTransform(cls)

Hérité de `GlueTransform` describeTransform.

describeErrors(cls)

Hérité de `GlueTransform` describeErrors.

describe(cls)

Hérité de `GlueTransform` describe.

IpToInt classe

La `IpToInt` transformation convertit la valeur du protocole Internet version 4 (IPv4) de la colonne source ou une autre valeur en valeur entière correspondante dans la colonne cible, et renvoie le résultat dans une nouvelle colonne.

Exemple

Pour AWS Glue 4.0 et versions ultérieures, créez ou mettez à jour les arguments de tâche avec `key: --enable-glue-di-transforms, value: true`

```
from pyspark.context import SparkContext
from awsgluedi.transforms import *

sc = SparkContext()

input_df = spark.createDataFrame(
    [
```

```
        ("192.0.0.1",),
        ("10.10.10.10",),
        ("1.2.3.4",),
        ("1.2.3.6",),
        ("http://12.13.14.15",),
        ("https://16.17.18.19",),
        ("1.2.3.4",),
        (None,),
        ("abc",),
        ("abc.abc.abc.abc",),
        ("321.123.123.123",),
        ("244.4.4.4",),
        ("255.255.255.255",),
    ],
    ["source_column_ip"],
)

    df_output = web_functions.IpToInt.apply(
        data_frame=input_df,
        spark_context=sc,
        source_column="source_column_ip",
        target_column="target_column",
        value=None
    )
    df_output.show()
```

Output

Le résultat sera :

```
```
+----------------+---------------+
|source_column_ip| target_column|
+----------------+---------------+
192.0.0.1	3221225473
10.10.10.10	168427722
1.2.3.4	16909060
1.2.3.6	16909062
http://12.13.14.15	null
https://16.17.18.19	null
1.2.3.4	16909060
null	null
abc	null
```
```

```
|abc.abc.abc.abc| null|
| 321.123.123.123| null|
| 244.4.4.4| 4102444804|
| 255.255.255.255| 4294967295|
+----------------+---------------+
```

La `IpToInt` transformation prend la `source_column` comme `"'source_column_ip"` et la `target_column` comme `"'target_column' » et convertit les chaînes d'adresses valides de la colonne `source_column_ip` en leur représentation entière 32 bits correspondante et stocke le résultat dans la colonne `target_column`. IPv4

Pour les chaînes d' IPv4 adresses valides (par exemple, « 192.0.0.1", « 10.10.10.10 », « 1.2.3.4 »), la transformation les convertit correctement en leur représentation entière (par exemple, 3221225473, 168427722, 16909060). Pour les chaînes qui ne sont pas des IPv4 adresses valides (par exemple URLs, les chaînes non IP comme « abc », les formats IP non valides tels que « abc.abc.abc.abc »), la valeur « target_column » est définie sur « null ». Pour les valeurs `null` de la colonne `source_column_ip`, la valeur `target_column` est également définie sur `null`.

Méthodes

- __call__
- s'appliquent
- name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__(spark_context, data_frame, target_column, source_column=None, value=None)

La `IpToInt` transformation convertit la valeur du protocole Internet version 4 (IPv4) de la colonne source ou une autre valeur en valeur entière correspondante dans la colonne cible, et renvoie le résultat dans une nouvelle colonne.

- `sourceColumn` : nom d'une colonne existante.

- `value` : chaîne de caractères à évaluer.
- `targetColumn` : le nom de la nouvelle colonne à créer.

apply(cls, *args, **kwargs)

Hérité de `GlueTransform` s'appliquent.

name(cls)

Hérité de `GlueTransform` name.

describeArgs(cls)

Hérité de `GlueTransform` describeArgs.

describeReturn(cls)

Hérité de `GlueTransform` describeReturn.

describeTransform(cls)

Hérité de `GlueTransform` describeTransform.

describeErrors(cls)

Hérité de `GlueTransform` describeErrors.

describe(cls)

Hérité de `GlueTransform` describe.

Transformations d'intégration de données

Pour AWS Glue 4.0 et versions ultérieures, créez ou mettez à jour les arguments de tâche avec`key: --enable-glue-di-transforms, value: true`.

Exemple de script de tâche :

```
from pyspark.context import SparkContext
```

```
from awsgluedi.transforms import *
sc = SparkContext()

input_df = spark.createDataFrame(
    [(5,), (0,), (-1,), (2,), (None,)],
    ["source_column"],
)

try:
    df_output = math_functions.IsEven.apply(
        data_frame=input_df,
        spark_context=sc,
        source_column="source_column",
        target_column="target_column",
        value=None,
        true_string="Even",
        false_string="Not even",
    )
    df_output.show()
except:
    print("Unexpected Error happened ")
    raise
```

Exemples de sessions utilisant des blocs-notes

```
%idle_timeout 2880
%glue_version 4.0
%worker_type G.1X
%number_of_workers 5
%region eu-west-1
```

```
%%configure
{
    "--enable-glue-di-transforms": "true"
}
```

```
from pyspark.context import SparkContext
from awsgluedi.transforms import *

sc = SparkContext()

input_df = spark.createDataFrame(
```

```
    [(5,), (0,), (-1,), (2,), (None,)],
    ["source_column"],
)

try:
    df_output = math_functions.IsEven.apply(
        data_frame=input_df,
        spark_context=sc,
        source_column="source_column",
        target_column="target_column",
        value=None,
        true_string="Even",
        false_string="Not even",
    )
    df_output.show()
except:
    print("Unexpected Error happened ")
    raise
```

Exemples de sessions utilisant AWS CLI

```
aws glue create-session --default-arguments "--enable-glue-di-transforms=true"
```

Transformations DI :

- FlagDuplicatesInColumn classe
- FormatPhoneNumber classe
- FormatCase classe
- FillWithMode classe
- FlagDuplicateRows classe
- RemoveDuplicates classe
- MonthName classe
- IsEven classe
- CryptographicHash classe
- Classe de déchiffrement
- Classe de chiffrement
- IntToIp classe

- IpToInt classe

Maven : création d'une offre groupée du plug-in pour vos applications Spark

Vous pouvez intégrer la dépendance des transformations à vos applications et distributions Spark (version 3.3) en ajoutant la dépendance du plug-in dans votre fichier Maven `pom.xml`, tout en développant vos applications Spark en local.

```
<repositories>
   ...
    <repository>
        <id>aws-glue-etl-artifacts</id>
        <url>https://aws-glue-etl-artifacts.s3.amazonaws.com/release/ </url>
    </repository>
</repositories>
...
<dependency>
    <groupId>com.amazonaws</groupId>
    <artifactId>AWSGlueTransforms</artifactId>
    <version>4.0.0</version>
</dependency>
```

Vous pouvez également télécharger les fichiers binaires directement depuis les artefacts de AWS Glue Maven et les inclure dans votre application Spark comme suit.

```
#!/bin/bash
sudo wget -v https://aws-glue-etl-artifacts.s3.amazonaws.com/release/com/amazonaws/
AWSGlueTransforms/4.0.0/AWSGlueTransforms-4.0.0.jar -P /usr/lib/spark/jars/
```

Programmation de scripts ETL AWS Glue dans Scala

Vous pouvez trouver des exemples de code et des utilitaires Scala AWS Glue dans le référentiel AWS Glue d'exemples sur le GitHub site Web.

AWS Glue prend en charge une extension du dialecte PySpark Scala pour les tâches d'extraction, de transformation et de chargement (ETL) par script. Les sections suivantes décrivent comment utiliser la bibliothèque Scala AWS Glue et l'API AWS Glue dans les scripts ETL, et fournissent la documentation de référence pour la bibliothèque.

Table des matières

- Utilisation de Scala pour programmer les scripts ETL AWS Glue
  - Test d'un programme ETL Scala dans un bloc-notes Jupyter sur un point de terminaison de développement
  - Test d'un programme ETL Scala dans un REPL Scala
- Exemple de script Scala — ETL de streaming
- APIs dans la bibliothèque AWS Glue Scala
  - com.amazonaws.services.glue
  - com.amazonaws.services.glue.ml
  - com.amazonaws.services.glue.dq
  - com.amazonaws.services.glue.types
  - com.amazonaws.services.glue.util
  - AWS GlueScala ChoiceOption APIs
    - ChoiceOption trait
    - ChoiceOption objet
      - def apply
    - Classe de cas ChoiceOptionWithResolver
    - Classe de cas MatchCatalogSchemaChoiceOption
  - DataSink Classe abstraite
    - Déf writeDynamicFrame
    - pyWriteDynamicCadre Def
    - Déf writeDataFrame
    - pyWriteDataCadre Def
    - Déf setCatalogInfo
    - def supportsFormat
    - def setFormat
    - def withFormat
    - Déf setAccumulableSize
    - Déf getOutputError RecordsAccumulable
    - errorsAsDynamicCadre Def
    - DataSink objet
      - def recordMetrics

- AWS GlueTrait Scala DataSource
- AWS GlueScala DynamicFrame APIs
  - AWS GlueClasse Scala DynamicFrame
    - val errorsCount
    - def applyMapping
    - Déf assertErrorThreshold
    - def count
    - def dropField
    - def dropFields
    - def dropNulls
    - errorsAsDynamicCadre Def
    - def filter
    - def getName
    - Déf getNumPartitions
    - Déf getSchemaIf calculé
    - Déf isSchemaComputed
    - Déf javaToPython
    - def join
    - def map
    - Déf mergeDynamicFrames
    - def printSchema
    - def recomputeSchema
    - def relationalize
    - def renameField
    - def repartition
    - def resolveChoice
    - def schema
    - def selectField
    - def selectFields
    - def show

        - Définissez par simplifier DDBJson
        - def spigot
        - def splitFields
        - def splitRows
        - Déf stageErrorsCount
        - def toDF
        - def unbox
        - def unnest
        - Def Unnest DDBJson
        - Déf withFrameSchema
        - def withName
        - Déf withTransformationContext
    - L' DynamicFrame objet
        - def apply
        - Déf emptyDynamicFrame
        - def fromPythonRDD
        - def ignoreErrors
        - def inlineErrors
        - newFrameWithErreurs de définition
- AWS GlueClasse Scala DynamicRecord
    - def addField
    - def dropField
    - def setError
    - def isError
    - def getError
    - def clearError
    - def write
    - def readFields
    - def clone
    - def schema

    - def getRoot
    - def toJson
    - Déf getFieldNode
    - def getField
    - def hashCode
    - def equals
    - DynamicRecord objet
        - Def apply
    - RecordTraverser trait
- AWS GlueScala GlueContext APIs
    - addIngestionTimeColonnes def
    - def createDataFrame FromOptions
    - forEachBatch
    - def getCatalogSink
    - def getCatalogSource
    - def get JDBCSink
    - def getSink
    - getSinkWithFormat def
    - def getSource
    - getSourceWithFormat def
    - def getSparkSession
    - def startTransaction
    - def commitTransaction
    - def cancelTransaction
    - def this
    - def this
    - def this
- MappingSpec
    - MappingSpec classe de cas
    - MappingSpec objet

    - Val orderingByTarget
    - Def apply
    - def apply
    - Def apply
- AWS GlueScala ResolveSpec APIs
    - ResolveSpec objet
        - def
        - def
    - ResolveSpec classe de cas
        - ResolveSpec méthodes def
- AWS GlueScala ArrayNode APIs
    - ArrayNode classe de cas
        - ArrayNode méthodes def
- AWS GlueScala BinaryNode APIs
    - BinaryNode classe de cas
        - BinaryNode champs de val
        - BinaryNode méthodes def
- AWS GlueScala BooleanNode APIs
    - BooleanNode classe de cas
        - BooleanNode champs de val
        - BooleanNode méthodes def
- AWS GlueScala ByteNode APIs
    - ByteNode classe de cas
        - ByteNode champs de val
        - ByteNode méthodes def
- AWS GlueScala DateNode APIs
    - DateNode classe de cas
        - DateNode champs de val
        - DateNode méthodes def
- AWS GlueScala DecimalNode APIs

    - DecimalNode classe de cas
        - DecimalNode champs de val
        - DecimalNode méthodes def
- AWS GlueScala DoubleNode APIs
    - DoubleNode classe de cas
        - DoubleNode champs de val
        - DoubleNode méthodes def
- AWS GlueScala DynamicNode APIs
    - DynamicNode classe
        - DynamicNode méthodes def
    - DynamicNode objet
        - DynamicNode méthodes def
- EvaluateDataQuality classe
    - Def apply
    - Exemple
- AWS GlueScala FloatNode APIs
    - FloatNode classe de cas
        - FloatNode champs de val
        - FloatNode méthodes def
- FillMissingValues classe
    - def apply
- FindMatches classe
    - def apply
- FindIncrementalMatches classe
    - def apply
- AWS GlueScala IntegerNode APIs
    - IntegerNode classe de cas
        - IntegerNode champs de val
        - IntegerNode méthodes def

- AWS GlueScala LongNode APIs

    - LongNode classe de cas
        - LongNode champs de val
        - LongNode méthodes def
- AWS GlueScala MapLikeNode APIs
    - MapLikeNode classe
        - MapLikeNode méthodes def
- AWS GlueScala MapNode APIs
    - MapNode classe de cas
        - MapNode méthodes def
- AWS GlueScala NullNode APIs
    - NullNode classe
    - NullNode objet du boîtier
- AWS GlueScala ObjectNode APIs
    - ObjectNode objet
        - ObjectNode méthodes def
    - ObjectNode classe de cas
        - ObjectNode méthodes def
- AWS GlueScala ScalarNode APIs
    - ScalarNode classe
        - ScalarNode méthodes def
    - ScalarNode objet
        - ScalarNode méthodes def
- AWS GlueScala ShortNode APIs
    - ShortNode classe de cas
        - ShortNode champs de val
        - ShortNode méthodes def
- AWS GlueScala StringNode APIs
    - StringNode classe de cas
        - StringNode champs de val
        - StringNode méthodes def

- AWS GlueScala TimestampNode APIs
  - TimestampNode classe de cas
    - TimestampNode champs de val
    - TimestampNode méthodes def
- AWS GlueScala GlueArgParser APIs
  - GlueArgParser objet
    - GlueArgParser méthodes def
- AWS GlueJob Scala APIs
  - Objet Job
    - Modes def Job

Utilisation de Scala pour programmer les scripts ETL AWS Glue

Vous pouvez générer automatiquement un programme ETL Scala à l'aide de la console AWS Glue et le modifier si nécessaire avant de l'affecter à une tâche. Vous pouvez également écrire votre propre programme à partir de zéro. Pour plus d'informations, consultez Configuration des propriétés des tâches pour les tâches Spark dans AWS Glue. AWS Glue compile ensuite votre programme Scala sur le serveur avant d'exécuter la tâche associée.

Pour vous assurer que votre programme compile sans erreur et s'exécute comme prévu, il est important de le charger sur un point de développement dans un REPL (Read-Eval-Print Loop) ou un bloc-notes Jupyter et de le tester là avant de l'exécuter dans une tâche. Étant donné que le processus de compilation intervient sur le serveur, vous n'aurez pas une bonne visibilité sur les éventuels problèmes qui s'y produisent.

Test d'un programme ETL Scala dans un bloc-notes Jupyter sur un point de terminaison de développement

Pour tester un programme Scala sur un point de terminaison de développement AWS Glue, définissez le point de terminaison de développement comme décrit dans Ajout d'un point de terminaison de développement.

Ensuite, connectez-le à un bloc-notes Jupyter s'exécutant localement sur votre machine ou à distance sur un serveur de bloc-notes Amazon EC2. Pour installer une version locale d'un bloc-notes Jupyter, suivez les instructions fournies dans Tutoriel : bloc-notes Jupyter dans JupyterLab.

La seule différence entre exécuter du code Scala et exécuter PySpark du code sur votre bloc-notes est que vous devez commencer chaque paragraphe du bloc-notes avec ce qui suit :

```
%spark
```

Cela empêche le serveur Notebook d'utiliser par défaut la version de l' PySparkinterpréteur Spark.

Test d'un programme ETL Scala dans un REPL Scala

Vous pouvez tester un programme Scala sur un point de terminaison de développement à l'aide d'un REPL Scala AWS Glue. Suivez les instructions indiquées dansTutoriel : Utiliser un bloc-notes SageMaker AI, sauf à la fin de la SSH-to-REPL commande, remplacez `-t gluepyspark` par`-t glue-spark-shell`. Cela appelle la fonction REPL Scala AWS Glue.

Pour fermer la fonction REPL lorsque vous avez terminé, tapez `sys.exit`.

Exemple de script Scala — ETL de streaming

Example

L'exemple de script ci-dessous se connecte à Amazon Kinesis Data Streams, utilise un schéma à partir du Data Catalog pour analyser un flux de données, joint le flux à un jeu de données statique sur Amazon S3 et génère les résultats joints dans Amazon S3 au format Parquet.

```
// This script connects to an Amazon Kinesis stream, uses a schema from the data
 catalog to parse the stream,
// joins the stream to a static dataset on Amazon S3, and outputs the joined results to
 Amazon S3 in parquet format.
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import java.util.Calendar
import org.apache.spark.SparkContext
import org.apache.spark.sql.Dataset
import org.apache.spark.sql.Row
import org.apache.spark.sql.SaveMode
import org.apache.spark.sql.SparkSession
import org.apache.spark.sql.functions.from_json
import org.apache.spark.sql.streaming.Trigger
import scala.collection.JavaConverters._

object streamJoiner {
  def main(sysArgs: Array[String]) {
```

```
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    val sparkSession: SparkSession = glueContext.getSparkSession
    import sparkSession.implicits._
    // @params: [JOB_NAME]
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)

    val staticData = sparkSession.read          // read() returns type DataFrameReader
      .format("csv")
      .option("header", "true")
      .load("s3://amzn-s3-demo-bucket/inputs/productsStatic.csv")  // load() returns a
DataFrame

    val datasource0 = sparkSession.readStream   // readstream() returns type
DataStreamReader
      .format("kinesis")
      .option("streamName", "stream-join-demo")
      .option("endpointUrl", "https://kinesis.us-east-1.amazonaws.com")
      .option("startingPosition", "TRIM_HORIZON")
      .load                                     // load() returns a DataFrame

    val selectfields1 = datasource0.select(from_json($"data".cast("string"),
glueContext.getCatalogSchemaAsSparkSchema("stream-demos", "stream-join-demo2")) as
"data").select("data.*")

    val datasink2 = selectfields1.writeStream.foreachBatch { (dataFrame: Dataset[Row],
batchId: Long) => {   //foreachBatch() returns type DataStreamWriter
      val joined = dataFrame.join(staticData, "product_id")
      val year: Int = Calendar.getInstance().get(Calendar.YEAR)
      val month :Int = Calendar.getInstance().get(Calendar.MONTH) + 1
      val day: Int = Calendar.getInstance().get(Calendar.DATE)
      val hour: Int = Calendar.getInstance().get(Calendar.HOUR_OF_DAY)

      if (dataFrame.count() > 0) {
        joined.write                            // joined.write returns type
DataFrameWriter
          .mode(SaveMode.Append)
          .format("parquet")
          .option("quote", " ")
          .save("s3://amzn-s3-demo-bucket/output/" + "/year=" + "%04d".format(year)
+ "/month=" + "%02d".format(month) + "/day=" + "%02d".format(day) + "/hour=" +
"%02d".format(hour) + "/")
      }
```

```
    }
    }  // end foreachBatch()
      .trigger(Trigger.ProcessingTime("100 seconds"))
      .option("checkpointLocation", "s3://amzn-s3-demo-bucket/checkpoint/")
      .start().awaitTermination()              // start() returns type StreamingQuery
    Job.commit()
  }
}
```

APIs dans la bibliothèque AWS Glue Scala

AWS Glueprend en charge une extension du dialecte PySpark Scala pour les tâches d'extraction, de transformation et de chargement (ETL) de scripts. Les sections suivantes décrivent le contenu APIs de la bibliothèque AWS Glue Scala.

com.amazonaws.services.glue

Le package com.amazonaws.services.glue de la bibliothèque Scala contient les AWS Glue éléments suivants : APIs

- ChoiceOption
- DataSink
- DataSource trait
- DynamicFrame
- DynamicRecord
- GlueContext
- MappingSpec
- ResolveSpec

com.amazonaws.services.glue.ml

Le package com.amazonaws.services.glue.ml de la bibliothèque Scala contient les éléments suivants : AWS Glue APIs

- FillMissingValues
- FindIncrementalMatches
- FindMatches

com.amazonaws.services.glue.dq

Le package com.amazonaws.services.glue.dq de la bibliothèque Scala contient les éléments suivants : AWS Glue APIs

- EvaluateDataQuality

com.amazonaws.services.glue.types

Le package com.amazonaws.services.glue.types de la bibliothèque Scala contient les éléments suivants : AWS Glue APIs

- ArrayNode
- BinaryNode
- BooleanNode
- ByteNode
- DateNode
- DecimalNode
- DoubleNode
- DynamicNode
- FloatNode
- IntegerNode
- LongNode
- MapLikeNode
- MapNode
- NullNode
- ObjectNode
- ScalarNode
- ShortNode
- StringNode
- TimestampNode

com.amazonaws.services.glue.util

Le package com.amazonaws.services.glue.util de la bibliothèque Scala contient les éléments suivants : AWS Glue APIs

- GlueArgParser
- Tâche

AWS GlueScala ChoiceOption APIs

Rubriques

- ChoiceOption trait
- ChoiceOption objet
- Classe de cas ChoiceOptionWithResolver
- Classe de cas MatchCatalogSchemaChoiceOption

Package : com.amazonaws.services.glue

ChoiceOption trait

```
trait ChoiceOption extends Serializable
```

ChoiceOption objet

ChoiceOption

```
object ChoiceOption
```

Une stratégie générale pour résoudre les choix applicables à tous les nœuds `ChoiceType` d'un `DynamicFrame`.

- `val CAST`
- `val MAKE_COLS`
- `val MAKE_STRUCT`
- `val MATCH_CATALOG`

- `val PROJECT`

def apply

```
def apply(choice: String): ChoiceOption
```

Classe de cas ChoiceOptionWithResolver

```
case class ChoiceOptionWithResolver(name: String, choiceResolver: ChoiceResolver)
 extends ChoiceOption {}
```

Classe de cas MatchCatalogSchemaChoiceOption

```
case class MatchCatalogSchemaChoiceOption() extends ChoiceOption {}
```

DataSink Classe abstraite

Rubriques

- Déf writeDynamicFrame
- pyWriteDynamicCadre Def
- Déf writeDataFrame
- pyWriteDataCadre Def
- Déf setCatalogInfo
- def supportsFormat
- def setFormat
- def withFormat
- Déf setAccumulableSize
- Déf getOutputError RecordsAccumulable
- errorsAsDynamicCadre Def
- DataSink objet

Package : com.amazonaws.services.glue

```
abstract class DataSink
```

L'enregistreur analogue à un encapsule un `DataSource`. `DataSink` encapsule une destination et un format dans lequel un `DynamicFrame` peut être écrit.

Déf writeDynamicFrame

```
def writeDynamicFrame( frame : DynamicFrame,
                       callSite : CallSite = CallSite("Not provided", "")
                     ) : DynamicFrame
```

pyWriteDynamicCadre Def

```
def pyWriteDynamicFrame( frame : DynamicFrame,
                         site : String = "Not provided",
                         info : String = "" )
```

Déf writeDataFrame

```
def writeDataFrame(frame: DataFrame,
                   glueContext: GlueContext,
                   callSite: CallSite = CallSite("Not provided", "")
                   ): DataFrame
```

pyWriteDataCadre Def

```
def pyWriteDataFrame(frame: DataFrame,
                     glueContext: GlueContext,
                     site: String = "Not provided",
                     info: String = ""
                     ): DataFrame
```

Déf setCatalogInfo

```
def setCatalogInfo(catalogDatabase: String,
                   catalogTableName : String,
```

```
                    catalogId : String = "")
```

def supportsFormat

```
def supportsFormat( format : String ) : Boolean
```

def setFormat

```
def setFormat( format : String,
               options : JsonOptions
             ) : Unit
```

def withFormat

```
def withFormat( format : String,
                options : JsonOptions = JsonOptions.empty
              ) : DataSink
```

Déf setAccumulableSize

```
def setAccumulableSize( size : Int ) : Unit
```

Déf getOutputError RecordsAccumulable

```
def getOutputErrorRecordsAccumulable : Accumulable[List[OutputError], OutputError]
```

errorsAsDynamicCadre Def

```
def errorsAsDynamicFrame : DynamicFrame
```

DataSink objet

```
object DataSink
```

def recordMetrics

```
def recordMetrics( frame : DynamicFrame,
                   ctxt : String
                 ) : DynamicFrame
```

AWS GlueTrait Scala DataSource

Package : com.amazonaws.services.glue

Interface de haut niveau pour produire un `DynamicFrame`.

```
trait DataSource {

  def getDynamicFrame : DynamicFrame

  def getDynamicFrame( minPartitions : Int,
                       targetPartitions : Int
                     ) : DynamicFrame
  def getDataFrame : DataFrame

  /** @param num: the number of records for sampling.
    * @param options: optional parameters to control sampling behavior. Current
 available parameter for Amazon S3 sources in options:
    *  1. maxSamplePartitions: the maximum number of partitions the sampling will
 read.
    *  2. maxSampleFilesPerPartition: the maximum number of files the sampling will
 read in one partition.
    */
  def getSampleDynamicFrame(num:Int, options: JsonOptions = JsonOptions.empty):
 DynamicFrame

  def glueContext : GlueContext

  def setFormat( format : String,
                 options : String
               ) : Unit

  def setFormat( format : String,
                 options : JsonOptions
               ) : Unit

  def supportsFormat( format : String ) : Boolean
```

```
  def withFormat( format : String,
                  options : JsonOptions = JsonOptions.empty
                ) : DataSource
}
```

AWS GlueScala DynamicFrame APIs

Package : com.amazonaws.services.glue

Table des matières

- AWS GlueClasse Scala DynamicFrame
  - val errorsCount
  - def applyMapping
  - Déf assertErrorThreshold
  - def count
  - def dropField
  - def dropFields
  - def dropNulls
  - errorsAsDynamicCadre Def
  - def filter
  - def getName
  - Déf getNumPartitions
  - Déf getSchemaIf calculé
  - Déf isSchemaComputed
  - Déf javaToPython
  - def join
  - def map
  - Déf mergeDynamicFrames
  - def printSchema
  - def recomputeSchema
  - def relationalize
  - def renameField
  - def repartition

- def resolveChoice
- def schema
- def selectField
- def selectFields
- def show
- Définissez par simplifier DDBJson
- def spigot
- def splitFields
- def splitRows
- Déf stageErrorsCount
- def toDF
- def unbox
- def unnest
- Def Unnest DDBJson
- Déf withFrameSchema
- def withName
- Déf withTransformationContext

- L' DynamicFrame objet
  - def apply
  - Déf emptyDynamicFrame
  - def fromPythonRDD
  - def ignoreErrors
  - def inlineErrors
  - newFrameWithErreurs de définition

AWS GlueClasse Scala DynamicFrame

Package : com.amazonaws.services.glue

```
class DynamicFrame extends Serializable with Logging  (
           val glueContext : GlueContext,
           _records : RDD[DynamicRecord],
           val name : String = s"",
```

```
            val transformationContext : String = DynamicFrame.UNDEFINED,
            callSite : CallSite = CallSite("Not provided", ""),
            stageThreshold : Long = 0,
            totalThreshold : Long = 0,
            prevErrors : => Long = 0,
            errorExpr : => Unit = {} )
```

Un `DynamicFrame` est une collection distribuée d'objets DynamicRecord à description automatique.

Le `DynamicFrame` est conçu pour fournir un modèle de données flexible pour les opérations ETL (extraction, transformation et chargement). Il ne nécessite pas de créer un schéma et peut être utilisé pour lire et transformer les données contenant des valeurs et des types incohérents ou complexes. Un schéma peut être calculé à la demande pour les opérations qui en nécessitent un.

Les `DynamicFrame` fournissent une plage de transformations pour le nettoyage des données et ETL. Ils prennent également en charge la conversion vers et depuis DataFrames SparkSQL afin de s'intégrer au code existant et aux nombreuses opérations d'analyse qui en découlent. DataFrames

Les paramètres suivants sont partagés entre plusieurs transformations AWS Glue qui construisent les `DynamicFrames` :

- `transformationContext` – Identificateur pour ce `DynamicFrame`. Le `transformationContext` est utilisé en tant que clé pour l'état de marque-page de tâche conservé d'une exécution à l'autre.
- `callSite` — fournit les informations de contexte pour le signalement d'erreurs. Ces valeurs sont automatiquement définies lors de l'appel à partir de Python.
- `stageThreshold` — Nombre maximal d'enregistrements d'erreurs autorisés depuis le calcul du `DynamicFrame` avant de lever une exception, à l'exclusion des enregistrements présents dans le `DynamicFrame` précédent.
- `totalThreshold` — Nombre maximal d'enregistrements d'erreur avant qu'une exception ne soit levée, y compris ceux des images précédentes.

val errorsCount

```
val errorsCount
```

Nombre d'enregistrements d'erreur dans le `DynamicFrame`. Les erreurs des opérations précédentes sont incluses dans le nombre.

def applyMapping

```
def applyMapping( mappings : Seq[Product4[String, String, String, String]],
                  caseSensitive : Boolean = true,
                  transformationContext : String = "",
                  callSite : CallSite = CallSite("Not provided", ""),
                  stageThreshold : Long = 0,
                  totalThreshold : Long = 0
                ) : DynamicFrame
```

- `mappings` – Séquence des mappages pour construire un nouveau `DynamicFrame`.
- `caseSensitive` — indique si les colonnes sources sont considérées comme sensibles à la casse. La définition de ce paramètre sur false peut s'avérer utile lors de l'intégration à des boutiques qui ne distinguent pas les majuscules des minuscules, comme le AWS Glue Data Catalog.

Sélectionne, projette et convertit les colonnes en fonction d'une séquence de mappages.

Chaque mappage se compose d'une colonne source et d'un type, ainsi que d'une colonne cible et d'un type. Les mappages peuvent être spécifiés sous forme de quatre tuples (`source_path`, `source_type`, `target_path`, `target_type`) ou d'un objet MappingSpec contenant les mêmes informations.

En plus des projections simples et de la conversion, les mappages peuvent être utilisés pour imbriquer ou désimbriquer les champs en séparant les composants du chemin d'accès par « `.` » (point).

Par exemple, supposons que vous ayez une trame `DynamicFrame` avec le schéma suivant.

```
{{{
  root
  |-- name: string
  |-- age: int
  |-- address: struct
  |    |-- state: string
  |    |-- zip: int
}}}
```

Vous pouvez effectuer l'appel suivant pour désimbriquer les champs `state` et `zip`.

```
{{{
  df.applyMapping(
   Seq(("name", "string", "name", "string"),
       ("age", "int", "age", "int"),
       ("address.state", "string", "state", "string"),
       ("address.zip", "int", "zip", "int")))
}}}
```

Le schéma obtenu est le suivant.

```
{{{
  root
  |-- name: string
  |-- age: int
  |-- state: string
  |-- zip: int
}}}
```

Vous pouvez également utiliser `applyMapping` pour réimbriquer les colonnes. Par exemple, ce qui suit inverse la transformation précédente et crée une structure nommée `address` dans la cible.

```
{{{
  df.applyMapping(
   Seq(("name", "string", "name", "string"),
       ("age", "int", "age", "int"),
       ("state", "string", "address.state", "string"),
       ("zip", "int", "address.zip", "int")))
}}}
```

Les noms de champ qui contiennent des caractères « `.` » (point) peuvent être placés entre guillemets (` `).

> **Note**
>
> La méthode `applyMapping` ne peut pas être utilisée actuellement pour mapper les colonnes imbriquées sous des tableaux.

Déf assertErrorThreshold

```
def assertErrorThreshold : Unit
```

Action qui oblige le calcul et vérifie que le nombre d'enregistrements d'erreur est inférieur à `stageThreshold` et `totalThreshold`. Lève une exception si l'une ou l'autre condition échoue.

def count

```
lazy
def count
```

Retourne le nombre d'éléments dans le `DynamicFrame`.

def dropField

```
def dropField( path : String,
               transformationContext : String = "",
               callSite : CallSite = CallSite("Not provided", ""),
               stageThreshold : Long = 0,
               totalThreshold : Long = 0
             ) : DynamicFrame
```

Renvoie un nouveau `DynamicFrame` avec la colonne spécifiée supprimée.

def dropFields

```
def dropFields( fieldNames : Seq[String],   // The column names to drop.
                transformationContext : String = "",
                callSite : CallSite = CallSite("Not provided", ""),
                stageThreshold : Long = 0,
                totalThreshold : Long = 0
              ) : DynamicFrame
```

Renvoie un nouveau `DynamicFrame` avec les colonnes spécifiées supprimées.

Vous pouvez utiliser cette méthode pour supprimer les colonnes imbriquées, y compris celles à l'intérieur de tableaux, mais pas pour supprimer des éléments de tableau spécifiques.

def dropNulls

```
def dropNulls( transformationContext : String = "",
```

```
              callSite : CallSite = CallSite("Not provided", ""),
              stageThreshold : Long = 0,
              totalThreshold : Long = 0 )
```

Renvoie un nouveau `DynamicFrame` avec toutes les colonnes null supprimées.

> **Note**
>
> Seules les colonnes de type `NullType` sont supprimées. Les valeurs null des autres colonnes ne sont pas supprimées ou modifiées.

errorsAsDynamicCadre Def

```
def errorsAsDynamicFrame
```

Renvoie un nouveau `DynamicFrame`contenant les enregistrements d'erreur de ce `DynamicFrame`.

def filter

```
def filter( f : DynamicRecord => Boolean,
            errorMsg : String = "",
            transformationContext : String = "",
            callSite : CallSite = CallSite("Not provided"),
            stageThreshold : Long = 0,
            totalThreshold : Long = 0
          ) : DynamicFrame
```

Crée un nouveau `DynamicFrame` contenant uniquement les enregistrements pour lesquels la fonction 'f' renvoie la valeur `true`. La fonction de filtre 'f' ne doit pas muter l'enregistrement d'entrée.

def getName

```
def getName : String
```

Renvoie le nom du `DynamicFrame`.

Déf getNumPartitions

```
def getNumPartitions
```

Retourne le nombre de partitions du `DynamicFrame`.

Déf getSchemaIf calculé

```
def getSchemaIfComputed : Option[Schema]
```

Renvoie le schéma s'il a déjà été calculé. N'analyse pas les données si le schéma n'a pas déjà été calculé.

Déf isSchemaComputed

```
def isSchemaComputed : Boolean
```

Renvoie la valeur `true` si le schéma a été calculé pour ce `DynamicFrame`, ou `false` dans le cas contraire. Si la méthode renvoie la valeur false, l'appel de la méthode `schema` nécessite un autre passage sur les enregistrements du `DynamicFrame`.

Déf javaToPython

```
def javaToPython : JavaRDD[Array[Byte]]
```

def join

```
def join( keys1 : Seq[String],
          keys2 : Seq[String],
          frame2 : DynamicFrame,
          transformationContext : String = "",
          callSite : CallSite = CallSite("Not provided", ""),
          stageThreshold : Long = 0,
          totalThreshold : Long = 0
        ) : DynamicFrame
```

- `keys1` — Colonnes de ce `DynamicFrame` à utiliser pour la jointure.
- `keys2` — colonnes de `frame2` à utiliser pour la jointure. Doit être de la même longueur que `keys1`.
- `frame2` — Autre `DynamicFrame` à joindre.

Renvoie le résultat de l'exécution d'une équijointure avec `frame2` à l'aide des clés spécifiées.

def map

```
def map( f : DynamicRecord => DynamicRecord,
         errorMsg : String = "",
         transformationContext : String = "",
         callSite : CallSite = CallSite("Not provided", ""),
         stageThreshold : Long = 0,
         totalThreshold : Long = 0
       ) : DynamicFrame
```

Renvoie un nouveau `DynamicFrame` construit en appliquant la fonction spécifiée 'f' à chaque enregistrement du `DynamicFrame`.

Comme cette méthode copie chaque enregistrement avant d'appliquer la fonction spécifiée, elle est sécurisée pour muter les enregistrements. Si la fonction de mappage lève une exception sur un enregistrement donné, celui-ci est marqué comme erreur, et le suivi de la pile est enregistré en tant que colonne dans l'enregistrement d'erreur.

Déf mergeDynamicFrames

```
def mergeDynamicFrames( stageDynamicFrame: DynamicFrame,  primaryKeys: Seq[String],
 transformationContext: String = "",
                         options: JsonOptions = JsonOptions.empty, callSite: CallSite =
 CallSite("Not provided"),
                         stageThreshold: Long = 0, totalThreshold: Long = 0):
 DynamicFrame
```

- `stageDynamicFrame` — trame `DynamicFrame` intermédiaire à fusionner.
- `primaryKeys` — liste des champs de clé primaire permettant de faire correspondre les enregistrements des trames `DynamicFrame` source et intermédiaire.
- `transformationContext` — chaîne unique utilisée pour récupérer les métadonnées relatives à la transformation en cours (facultatif).
- `options` — chaîne de paires nom-valeur JSON qui fournissent des informations supplémentaires pour cette transformation.
- `callSite` — permet de fournir des informations contextuelles pour le signalement d'erreurs.
- `stageThreshold` – Une `Long`. Nombre d'erreurs identifiées dans la transformation donnée et à corriger lors du traitement.

- `totalThreshold` – Une `Long`. Nombre total d'erreurs identifiées dans la transformation donnée et qui doivent être corrigées lors du traitement.

Fusionne cette trame `DynamicFrame` avec une trame `DynamicFrame` intermédiaire basée sur les clés primaires spécifiées pour identifier les enregistrements. Les registres en double (registres avec les mêmes clés primaires) ne sont pas dédupliqués. Si aucun enregistrement ne correspond dans la trame intermédiaire, tous les enregistrements (y compris les doublons) sont conservés dans la source. Si la trame intermédiaire contient des enregistrements correspondants, les enregistrements de la trame intermédiaire remplacent ceux de la source dans AWS Glue.

La trame `DynamicFrame` renvoyée contient l'enregistrement A dans les cas suivants :

1. Si `A` se trouve à la fois dans la trame source et la trame intermédiaire, c'est la valeur `A` de la trame intermédiaire qui est renvoyée.
2. Si `A` se trouve dans la table source et si `A.primaryKeys` ne se trouve pas dans la trame `stagingDynamicFrame` (en d'autres termes, `A` n'est pas mis à jour dans la table intermédiaire).

La trame source et la trame intermédiaire n'ont pas besoin d'avoir le même schéma.

Example

```
val mergedFrame: DynamicFrame = srcFrame.mergeDynamicFrames(stageFrame, Seq("id1",
 "id2"))
```

def printSchema

```
def printSchema : Unit
```

Imprime le schéma du `DynamicFrame` sur `stdout` dans un format compréhensible par les utilisateurs.

def recomputeSchema

```
def recomputeSchema : Schema
```

Force le recalcul d'un schéma. Ceci nécessite une analyse des données, mais peut « resserrer » le schéma si certains champs du schéma actuel ne sont pas présents dans les données.

Renvoie le schéma recalculé.

def relationalize

```
def relationalize( rootTableName : String,
                   stagingPath : String,
                   options : JsonOptions = JsonOptions.empty,
                   transformationContext : String = "",
                   callSite : CallSite = CallSite("Not provided"),
                   stageThreshold : Long = 0,
                   totalThreshold : Long = 0
                 ) : Seq[DynamicFrame]
```

- `rootTableName` — nom à utiliser pour le `DynamicFrame` de base dans la sortie. Les trames `DynamicFrame` qui sont créés par les tableaux pivotants commencent avec ce préfixe.
- `stagingPath` — chemin Amazon Simple Storage Service (Amazon S3) pour écrire des données intermédiaires.
- `options` — crée les relations entre les options et la configuration. Non utilisé actuellement.

Aplanit toutes les structures imbriquées et pivote les tableaux en tables distinctes.

Vous pouvez utiliser cette opération pour préparer les données profondément imbriquées en vue de leur ingestion dans une base de données relationnelle. Les structs imbriquées sont mises à plat de la même manière que la transformation unnest. De plus, les tableaux sont pivotés en tables distinctes et chaque élément du tableau devient une ligne. Par exemple, supposons que vous ayez une trame `DynamicFrame` avec les données suivantes.

```
{"name": "Nancy", "age": 47, "friends": ["Fred", "Lakshmi"]}
{"name": "Stephanie", "age": 28, "friends": ["Yao", "Phil", "Alvin"]}
{"name": "Nathan", "age": 54, "friends": ["Nicolai", "Karen"]}
```

Exécutez le code suivant.

```
{{{
  df.relationalize("people", "s3:/my_bucket/my_path", JsonOptions.empty)
}}}
```

Il génère deux tables. La première table est nommée « people » et contient les éléments suivants.

```
{{{
  {"name": "Nancy", "age": 47, "friends": 1}
  {"name": "Stephanie", "age": 28, "friends": 2}
  {"name": "Nathan", "age": 54, "friends": 3)
}}}
```

Ici, le tableau « friends » a été remplacé par une clé de jointure générée automatiquement. Une table séparée nommée `people.friends` est créée avec le contenu suivant.

```
{{{
  {"id": 1, "index": 0, "val": "Fred"}
  {"id": 1, "index": 1, "val": "Lakshmi"}
  {"id": 2, "index": 0, "val": "Yao"}
  {"id": 2, "index": 1, "val": "Phil"}
  {"id": 2, "index": 2, "val": "Alvin"}
  {"id": 3, "index": 0, "val": "Nicolai"}
  {"id": 3, "index": 1, "val": "Karen"}
}}}
```

Dans ce tableau, « `id` » est une clé de jointure qui identifie de quel enregistrement provenait l'élément de tableau, « `index` » fait référence à la position dans le tableau d'origine et « `val` » est l'entrée réelle du tableau.

La méthode `relationalize` renvoie la séquence de `DynamicFrame` créée en appliquant ce processus de façon récursive à tous les tableaux.

> **Note**
>
> La bibliothèque AWS Glue génère automatiquement les clés de jointure des nouvelles tables. Pour vous assurer que les clés de jointure sont uniques au travers des exécutions de tâche, vous devez activer les marque-pages de tâche.

def renameField

```
def renameField( oldName : String,
                 newName : String,
                 transformationContext : String = "",
                 callSite : CallSite = CallSite("Not provided", ""),
                 stageThreshold : Long = 0,
```

```
                totalThreshold : Long = 0
              ) : DynamicFrame
```

- `oldName` — Nom d'origine de la colonne.
- `newName` — Nouveau nom de la colonne.

Renvoie un nouveau `DynamicFrame` contenant le champ spécifié renommé.

Cette méthode peut être utilisée pour renommer les champs imbriqués. Par exemple, le code suivant remplace le nom `state` par `state_code` dans la structure de l'adresse.

```
{{{
  df.renameField("address.state", "address.state_code")
}}}
```

def repartition

```
def repartition( numPartitions : Int,
                 transformationContext : String = "",
                 callSite : CallSite = CallSite("Not provided", ""),
                 stageThreshold : Long = 0,
                 totalThreshold : Long = 0
               ) : DynamicFrame
```

Renvoie un nouveau `DynamicFrame` avec les partitions `numPartitions`.

def resolveChoice

```
def resolveChoice( specs : Seq[Product2[String, String]] = Seq.empty[ResolveSpec],
                   choiceOption : Option[ChoiceOption] = None,
                   database : Option[String] = None,
                   tableName : Option[String] = None,
                   transformationContext : String = "",
                   callSite : CallSite = CallSite("Not provided", ""),
                   stageThreshold : Long = 0,
                   totalThreshold : Long = 0
                 ) : DynamicFrame
```

- `choiceOption` — Action à appliquer aux colonnes `ChoiceType` non listées dans la séquence des spécifications.

- `database` — base de données Data Catalog à utiliser avec l'action `match_catalog`.
- `tableName` — table Data Catalog à utiliser avec l'action `match_catalog`.

Renvoie un nouveau `DynamicFrame` en remplaçant un ou plusieurs `ChoiceType` avec un type plus spécifique.

Il existe deux façons d'utiliser `resolveChoice`. La première consiste à spécifier une séquence de colonnes spécifiques et la façon de les résoudre. Celles-ci sont spécifiés en tant que tuples composés de paires (colonne, action).

Les actions possibles sont les suivantes :

- `cast:type` — Tente de convertir toutes les valeurs dans le type spécifié.
- `make_cols` – Convertit chaque type distinct en une colonne portant le nom `columnName_type`.
- `make_struct` — Convertit une colonne en une structure avec des clés pour chaque type distinct.
- `project:type` — Retient uniquement les valeurs du type spécifié.

L'autre mode pour `resolveChoice` consiste à spécifier une seule résolution pour tous les `ChoiceTypes`. Vous pouvez utiliser cela lorsque la liste complète des `ChoiceTypes` est inconnue avant l'exécution. En plus des actions répertoriées précédemment, ce mode prend également en charge l'action suivante :

- `match_catalogChoiceType` — Tente de convertir chaque dans le type correspondant de la table de catalogue spécifiée.

Exemples :

Résolvez la colonne `user.id` en la convertissant en int et faites que le champ `address` conserve uniquement les structs.

```
{{{
  df.resolveChoice(specs = Seq(("user.id", "cast:int"), ("address", "project:struct")))
}}}
```

Résolvez tous les objets `ChoiceType` en convertissant chaque choix en colonne séparée.

```
{{{
```

```
  df.resolveChoice(choiceOption = Some(ChoiceOption("make_cols")))
}}}
```

Résolvez tous les objets `ChoiceType` en les convertissant dans les types de la table de catalogue spécifiée.

```
{{{
  df.resolveChoice(choiceOption = Some(ChoiceOption("match_catalog")),
                   database = Some("my_database"),
                   tableName = Some("my_table"))
}}}
```

def schema

```
def schema : Schema
```

Renvoie le schéma du `DynamicFrame`.

Le schéma renvoyé est assuré de contenir chaque champ présent dans un enregistrement de ce `DynamicFrame`. Mais dans un petit nombre de cas, il peut aussi contenir des champs supplémentaires. La méthode unnest peut être utilisée pour « resserrer » le schéma basé sur les enregistrements du `DynamicFrame`.

def selectField

```
def selectField( fieldName : String,
                 transformationContext : String = "",
                 callSite : CallSite = CallSite("Not provided", ""),
                 stageThreshold : Long = 0,
                 totalThreshold : Long = 0
               ) : DynamicFrame
```

Renvoie un champ unique comme `DynamicFrame`.

def selectFields

```
def selectFields( paths : Seq[String],
                  transformationContext : String = "",
                  callSite : CallSite = CallSite("Not provided", ""),
                  stageThreshold : Long = 0,
```

```
                  totalThreshold : Long = 0
                ) : DynamicFrame
```

- `paths` — Séquence de noms de colonnes à sélectionner.

Renvoie un nouveau `DynamicFrame` contenant les colonnes spécifiées.

> **Note**
>
> La méthode `selectFields` peut uniquement être utilisée pour sélectionner les colonnes de niveau supérieur. La méthode applyMapping peut être utilisée pour sélectionner les colonnes imbriquées.

def show

```
def show( numRows : Int = 20 ) : Unit
```

- `numRows` — Nombre de lignes à imprimer.

Imprime les lignes du `DynamicFrame` au format JSON.

Définissez par simplifier DDBJson

Les exportations DynamoDB avec le connecteur d'exportation AWS Glue DynamoDB donnent lieu à des fichiers JSON de structures imbriquées spécifiques. Pour en savoir plus, consultez Data objects. `simplifyDDBJson` Simplifie les colonnes imbriquées dans ce type de données et renvoie une nouvelle colonne simplifiée DynamicFrame. DynamicFrame S'il existe plusieurs types ou un type Carte contenu dans un type Liste, les éléments de la liste ne seront pas simplifiés. Cette méthode prend uniquement en charge les données au format JSON d'exportation DynamoDB. Envisagez `unnest` pour effectuer des modifications similaires sur d'autres types de données.

```
def simplifyDDBJson() : DynamicFrame
```

Cette méthode n'utilise aucun paramètre.

Exemple d'entrée

Tenez compte du schéma suivant généré par une exportation DynamoDB :

```
root
|-- Item: struct
|    |-- parentMap: struct
|    |    |-- M: struct
|    |    |    |-- childMap: struct
|    |    |    |    |-- M: struct
|    |    |    |    |    |-- appName: struct
|    |    |    |    |    |    |-- S: string
|    |    |    |    |    |-- packageName: struct
|    |    |    |    |    |    |-- S: string
|    |    |    |    |    |-- updatedAt: struct
|    |    |    |    |    |    |-- N: string
|    |-- strings: struct
|    |    |-- SS: array
|    |    |    |-- element: string
|    |-- numbers: struct
|    |    |-- NS: array
|    |    |    |-- element: string
|    |-- binaries: struct
|    |    |-- BS: array
|    |    |    |-- element: string
|    |-- isDDBJson: struct
|    |    |-- BOOL: boolean
|    |-- nullValue: struct
|    |    |-- NULL: boolean
```

Exemple de code

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.DynamoDbDataSink
import org.apache.spark.SparkContextimport scala.collection.JavaConverters._

object GlueApp {

  def main(sysArgs: Array[String]): Unit = {
    val glueContext = new GlueContext(SparkContext.getOrCreate())
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)
```

```
    val dynamicFrame = glueContext.getSourceWithFormat(
      connectionType = "dynamodb",
      options = JsonOptions(Map(
        "dynamodb.export" -> "ddb",
        "dynamodb.tableArn" -> "ddbTableARN",
        "dynamodb.s3.bucket" -> "exportBucketLocation",
        "dynamodb.s3.prefix" -> "exportBucketPrefix",
        "dynamodb.s3.bucketOwner" -> "exportBucketAccountID",
      ))
    ).getDynamicFrame()

    val simplified = dynamicFrame.simplifyDDBJson()
    simplified.printSchema()

    Job.commit()
  }

}
```

Exemple de sortie

La transformation `simplifyDDBJson` simplifie cette exportation ainsi :

```
root
|-- parentMap: struct
|    |-- childMap: struct
|    |    |-- appName: string
|    |    |-- packageName: string
|    |    |-- updatedAt: string
|-- strings: array
|    |-- element: string
|-- numbers: array
|    |-- element: string
|-- binaries: array
|    |-- element: string
|-- isDDBJson: boolean
|-- nullValue: null
```

def spigot

```
def spigot( path : String,
            options : JsonOptions = new JsonOptions("{}"),
```

```
            transformationContext : String = "",
            callSite : CallSite = CallSite("Not provided"),
            stageThreshold : Long = 0,
            totalThreshold : Long = 0
          ) : DynamicFrame
```

Transmettez la transformation qui renvoie les mêmes enregistrements, mais écrit un sous-ensemble d'enregistrements en tant qu'effet secondaire.

- `path` — chemin d'accès dans Amazon S3 dans lequel écrire la sortie, sous la forme `s3://bucket//path`.
- `options` — Carte `JsonOptions` facultative décrivant le comportement d'échantillonnage.

Renvoie un `DynamicFrame` contenant les mêmes enregistrements que celui-ci.

Par défaut, écrit 100 enregistrements arbitraires à l'emplacement spécifié par `path`. Ce comportement peut être personnalisé à l'aide de la carte `options`. Les clés valides incluent les suivantes :

- `topk` — spécifie le nombre total d'enregistrements écrits. La valeur par défaut est 100.
- `prob` — indique la probabilité (sous forme de décimale) qu'un enregistrement individuel soit inclus. La valeur par défaut est 1.

Par exemple, l'appel suivant échantillonne l'ensemble de données en sélectionnant chaque enregistrement avec une probabilité de 20 % et en s'arrêtant après l'écriture de 200 enregistrements.

```
{{{
  df.spigot("s3://my_bucket/my_path", JsonOptions(Map("topk" -&gt; 200, "prob" -&gt;
 0.2)))
}}}
```

def splitFields

```
def splitFields( paths : Seq[String],
                 transformationContext : String = "",
                 callSite : CallSite = CallSite("Not provided", ""),
                 stageThreshold : Long = 0,
                 totalThreshold : Long = 0
               ) : Seq[DynamicFrame]
```

- `paths` — Chemins à inclure dans le premier `DynamicFrame`.

Renvoie une séquence de deux `DynamicFrames`. Le premier `DynamicFrame`contient les chemins d'accès spécifiés et le deuxième contient toutes les autres colonnes.

Exemple

Cet exemple prend une `persons` table DynamicFrame créée à partir de la `legislators` base de données du AWS Glue Data Catalog et la DynamicFrame divise en deux, les champs spécifiés étant placés dans le premier DynamicFrame et les champs restants dans un second DynamicFrame. L'exemple choisit ensuite le premier DynamicFrame parmi le résultat.

```
val InputFrame = glueContext.getCatalogSource(database="legislators",
 tableName="persons",
transformationContext="InputFrame").getDynamicFrame()

val SplitField_collection = InputFrame.splitFields(paths=Seq("family_name", "name",
 "links.note",
"links.url", "gender", "image", "identifiers.scheme", "identifiers.identifier",
 "other_names.lang",
"other_names.note", "other_names.name"), transformationContext="SplitField_collection")

val ResultFrame = SplitField_collection(0)
```

def splitRows

```
def splitRows( paths : Seq[String],
               values : Seq[Any],
               operators : Seq[String],
               transformationContext : String,
               callSite : CallSite,
               stageThreshold : Long,
               totalThreshold : Long
             ) : Seq[DynamicFrame]
```

Fractionne les lignes en fonction des prédicats qui comparent les colonnes aux constantes.

- `paths` — Colonnes à utiliser pour la comparaison.
- `values` — Valeurs constantes à utiliser pour la comparaison.
- `operators` — Opérateurs à utiliser pour la comparaison.

Renvoie une séquence de deux DynamicFrames. Le premier contient les lignes pour lesquelles le prédicat a la valeur true et le deuxième contient celles pour lesquelles la valeur est false.

Les prédicats sont spécifiés en utilisant trois séquences : « paths » contient les noms de colonne (possiblement imbriqués), « values » contient les valeurs constantes de comparaison et « operators » contient les opérateurs à utiliser pour la comparaison. Les trois séquences doivent être de la même longueur : le nème opérateur est utilisé pour comparer la nème colonne à la nème valeur.

Chaque opérateur doit être « != », « = », « <= », « < », « >= » ou « > ».

Par exemple, l'appel suivant scinde une trame DynamicFrame afin que la première trame de sortie contienne les enregistrements des personnes de plus de 65 ans des États-Unis et que la deuxième contienne tous les autres enregistrements.

```
{{{
  df.splitRows(Seq("age", "address.country"), Seq(65, "USA"), Seq("&gt;=", "="))
}}}
```

Déf stageErrorsCount

```
def stageErrorsCount
```

Renvoie le nombre d'enregistrements d'erreur créés pendant le calcul du DynamicFrame. Cela exclut les erreurs des opérations précédentes qui ont été transmises au DynamicFrame comme entrée.

def toDF

```
def toDF( specs : Seq[ResolveSpec] = Seq.empty[ResolveSpec] ) : DataFrame
```

Convertit le DynamicFrame en un DataFrame Apache Spark SQL avec le même schéma et les mêmes enregistrements.

> **Note**
>
> Étant donné que les DataFrames ne prennent pas en charge les ChoiceTypes, cette méthode convertit automatiquement les colonnes ChoiceType en StructTypes. Pour plus d'informations et pour connaître les options de résolution des choix, consultez resolveChoice.

def unbox

```
def unbox( path : String,
           format : String,
           optionString : String = "{}",
           transformationContext : String = "",
           callSite : CallSite = CallSite("Not provided"),
           stageThreshold : Long = 0,
           totalThreshold : Long = 0
         ) : DynamicFrame
```

- `path` — colonne à analyser. Doit être de type string (chaîne) ou binary (binaire).
- `format` — Le format à utiliser pour l'analyse.
- `optionString` — Options à transmettre au format, telles que le séparateur CSV.

Analyse une chaîne ou une colonne binaire intégrée selon le format spécifié. Les colonnes analysées sont imbriquées sous une structure avec le nom de colonne d'origine.

Par exemple, supposons que vous ayez un fichier CSV avec une colonne JSON imbriquée.

```
name, age, address
Sally, 36, {"state": "NE", "city": "Omaha"}
...
```

Après une analyse initiale, vous obtenez une trame `DynamicFrame` avec le schéma suivant.

```
{{{
  root
  |-- name: string
  |-- age: int
  |-- address: string
}}}
```

Vous pouvez appeler `unbox` au niveau de la colonne « address » pour analyser les composants spécifiques.

```
{{{
  df.unbox("address", "json")
}}}
```

Vous obtenez ainsi une trame `DynamicFrame` avec le schéma suivant.

```
{{{
  root
  |-- name: string
  |-- age: int
  |-- address: struct
  |    |-- state: string
  |    |-- city: string
}}}
```

def unnest

```
def unnest( transformationContext : String = "",
            callSite : CallSite = CallSite("Not Provided"),
            stageThreshold : Long = 0,
            totalThreshold : Long = 0
          ) : DynamicFrame
```

Renvoie un nouveau `DynamicFrame` avec tous les structures imbriquées mises à plat. Les noms sont construits à l'aide du caractère « `.` » (point).

Par exemple, supposons que vous ayez une trame `DynamicFrame` avec le schéma suivant.

```
{{{
  root
  |-- name: string
  |-- age: int
  |-- address: struct
  |    |-- state: string
  |    |-- city: string
}}}
```

L'appel suivant désimbrique la struct « address ».

```
{{{
  df.unnest()
}}}
```

Le schéma obtenu est le suivant.

```
{{{
  root
  |-- name: string
  |-- age: int
  |-- address.state: string
  |-- address.city: string
}}}
```

Cette méthode désimbrique également les structs imbriquées à l'intérieur des tableaux. Mais pour des raisons historiques, le nom de ces champs est précédé par le nom du tableau englobant et par « `.val` ».

Def Unnest DDBJson

```
unnestDDBJson(transformationContext : String = "",
            callSite : CallSite = CallSite("Not Provided"),
            stageThreshold : Long = 0,
            totalThreshold : Long = 0): DynamicFrame
```

Supprime l'imbrication des colonnes imbriquées dans un `DynamicFrame` qui se trouvent spécifiquement dans la structure JSON DynamoDB, et renvoie une nouvelle version non imbriquée `DynamicFrame`. Les colonnes d'un tableau de types de structure ne seront pas non-imbriquées. Notez qu'il s'agit d'un type spécifique de transformation non imbriquée qui se comporte différemment de la transformation `unnest` normale et nécessite que les données soient déjà dans la structure JSON DynamoDB. Pour plus d'informations, consultez JSON DynamoDB.

Par exemple, le schéma d'une lecture d'exportation avec la structure JSON DynamoDB pourrait ressembler à ce qui suit :

```
root
|-- Item: struct
|    |-- ColA: struct
|    |    |-- S: string
|    |-- ColB: struct
|    |    |-- S: string
|    |-- ColC: struct
|    |    |-- N: string
|    |-- ColD: struct
|    |    |-- L: array
|    |    |    |-- element: null
```

La transformation `unnestDDBJson()` convertirait ceci en :

```
root
|-- ColA: string
|-- ColB: string
|-- ColC: string
|-- ColD: array
|    |-- element: null
```

L'exemple de code suivant montre comment utiliser le connecteur d'exportation AWS Glue DynamoDB, appeler un DynamoDB JSON unnest et imprimer le nombre de partitions :

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.DynamoDbDataSink
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._


object GlueApp {

  def main(sysArgs: Array[String]): Unit = {
    val glueContext = new GlueContext(SparkContext.getOrCreate())
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)

    val dynamicFrame = glueContext.getSourceWithFormat(
      connectionType = "dynamodb",
      options = JsonOptions(Map(
        "dynamodb.export" -> "ddb",
        "dynamodb.tableArn" -> "<test_source>",
        "dynamodb.s3.bucket" -> "<bucket name>",
        "dynamodb.s3.prefix" -> "<bucket prefix>",
        "dynamodb.s3.bucketOwner" -> "<account_id of bucket>",
      ))
    ).getDynamicFrame()

    val unnested = dynamicFrame.unnestDDBJson()
    print(unnested.getNumPartitions())

    Job.commit()
```

```
  }

}
```

Déf withFrameSchema

```
def withFrameSchema( getSchema : () => Schema ) : DynamicFrame
```

- `getSchema` — fonction qui renvoie le schéma à utiliser. Spécifiée en tant que fonction de paramètre zéro pour reporter un calcul potentiellement onéreux.

Définit le schéma du `DynamicFrame` sur la valeur spécifiée. La fonction est principalement utilisée en interne pour éviter un recalcul du schéma coûteux. Le schéma transmis doit contenir toutes les colonnes présentes dans les données.

def withName

```
def withName( name : String ) : DynamicFrame
```

- `name` — Nouveau nom à utiliser.

Renvoie une copie du `DynamicFrame` avec un nouveau nom.

Déf withTransformationContext

```
def withTransformationContext( ctx : String ) : DynamicFrame
```

Renvoie une copie du `DynamicFrame` avec le contexte de transformation spécifié.

L' DynamicFrame objet

Package : com.amazonaws.services.glue

```
object DynamicFrame
```

def apply

```
def apply( df : DataFrame,
```

```
            glueContext : GlueContext
          ) : DynamicFrame
```

Déf emptyDynamicFrame

```
def emptyDynamicFrame( glueContext : GlueContext ) : DynamicFrame
```

def fromPythonRDD

```
def fromPythonRDD( rdd : JavaRDD[Array[Byte]],
                   glueContext : GlueContext
                 ) : DynamicFrame
```

def ignoreErrors

```
def ignoreErrors( fn : DynamicRecord => DynamicRecord ) : DynamicRecord
```

def inlineErrors

```
def inlineErrors( msg : String,
                  callSite : CallSite
                ) : (DynamicRecord => DynamicRecord)
```

newFrameWithErreurs de définition

```
def newFrameWithErrors( prevFrame : DynamicFrame,
                        rdd : RDD[DynamicRecord],
                        name : String = "",
                        transformationContext : String = "",
                        callSite : CallSite,
                        stageThreshold : Long,
                        totalThreshold : Long
                      ) : DynamicFrame
```

AWS GlueClasse Scala DynamicRecord

Rubriques

- def addField
- def dropField
- def setError
- def isError
- def getError
- def clearError
- def write
- def readFields
- def clone
- def schema
- def getRoot
- def toJson
- Déf getFieldNode
- def getField
- def hashCode
- def equals
- DynamicRecord objet
- RecordTraverser trait

Package : com.amazonaws.services.glue

```
class DynamicRecord extends Serializable with Writable with Cloneable
```

Un `DynamicRecord` est une structure de données à description automatique qui représente une ligne de données de l'ensemble de données en cours de traitement. Elle est auto-descriptive en ce sens que vous pouvez obtenir le schéma de la ligne représentée par le `DynamicRecord` en inspectant l'enregistrement lui-même. Un `DynamicRecord` est similaire à un `Row` dans Apache Spark.

def addField

```
def addField( path : String,
              dynamicNode : DynamicNode
            ) : Unit
```

Ajoute un DynamicNode au chemin d'accès spécifié.

- `path` — Chemin d'accès du champ à ajouter.
- `dynamicNode` — DynamicNode à ajouter au chemin d'accès spécifié.

def dropField

```
def dropField(path: String, underRename: Boolean = false): Option[DynamicNode]
```

Supprime un DynamicNode du chemin spécifié et renvoie le nœud supprimé s'il n'y a pas un tableau dans le chemin d'accès spécifié.

- `path` — Chemin d'accès du champ à supprimer.
- `underRenamedropField` : valeur True si est appelé dans le cadre d'une transformation en nouveau nom ou false dans le cas contraire (false par défaut).

Retourne un `scala.Option Option` (DynamicNode).

def setError

```
def setError( error : Error )
```

Définit cet enregistrement en tant qu'enregistrement d'erreur, comme spécifié par le paramètre `error`.

Retourne un `DynamicRecord`.

def isError

```
def isError
```

Vérifie si cet enregistrement est un enregistrement d'erreur.

def getError

```
def getError
```

Obtient `Error` si l'enregistrement est un enregistrement d'erreur. Renvoie `scala.Some` Some (Erreur) si cet enregistrement est un enregistrement d'erreur ou `scala.None` dans le cas contraire.

def clearError

```
def clearError
```

Définissez `Error` sur `scala.None.None`.

def write

```
override def write( out : DataOutput ) : Unit
```

def readFields

```
override def readFields( in : DataInput ) : Unit
```

def clone

```
override def clone : DynamicRecord
```

Clone cet enregistrement en un nouveau `DynamicRecord` et le renvoie.

def schema

```
def schema
```

Obtient `Schema` en inspectant l'enregistrement.

def getRoot

```
def getRoot : ObjectNode
```

Obtient le `ObjectNode` racine pour l'enregistrement.

def toJson

```
def toJson : String
```

Obtient la chaîne JSON pour l'enregistrement.

Déf getFieldNode

```
def getFieldNode( path : String ) : Option[DynamicNode]
```

Obtient la valeur du champ au `path` spécifié comme option de `DynamicNode`.

Renvoie `scala.Some` Some (DynamicNode) si le champ existe, ou dans le cas contraire `scala.None.None`.

def getField

```
def getField( path : String ) : Option[Any]
```

Obtient la valeur du champ au `path` spécifié comme option de `DynamicNode`.

Retourne `scala.Some` Some (valeur).

def hashCode

```
override def hashCode : Int
```

def equals

```
override def equals( other : Any )
```

DynamicRecord objet

```
object DynamicRecord
```

Def apply

```
def apply( row : Row,
           schema : SparkStructType )
```

Appliquez la méthode pour convertir un objet Row Apache Spark SQL en un DynamicRecord.

- `row` – Spark SQL Row.
- `schema` — Schema de cette ligne.

Retourne un `DynamicRecord`.

RecordTraverser trait

```
trait RecordTraverser {
  def nullValue(): Unit
  def byteValue(value: Byte): Unit
  def binaryValue(value: Array[Byte]): Unit
  def booleanValue(value: Boolean): Unit
  def shortValue(value: Short) : Unit
  def intValue(value: Int) : Unit
  def longValue(value: Long) : Unit
  def floatValue(value: Float): Unit
  def doubleValue(value: Double): Unit
  def decimalValue(value: BigDecimal): Unit
  def stringValue(value: String): Unit
  def dateValue(value: Date): Unit
  def timestampValue(value: Timestamp): Unit
  def objectStart(length: Int): Unit
  def objectKey(key: String): Unit
  def objectEnd(): Unit
  def mapStart(length: Int): Unit
  def mapKey(key: String): Unit
  def mapEnd(): Unit
  def arrayStart(length: Int): Unit
  def arrayEnd(): Unit
}
```

AWS GlueScala GlueContext APIs

Package : com.amazonaws.services.glue

```
class GlueContext extends SQLContext(sc) (
           @transient val sc : SparkContext,
           val defaultSourcePartitioner : PartitioningStrategy )
```

`GlueContext` Est le point d'entrée pour lire et écrire un DynamicFrame depuis et vers Amazon Simple Storage Service (Amazon S3), le catalogue de données AWS Glue, JDBC, etc. Cette classe fournit des fonctions d'utilitaire pour créer des objets DataSource trait et DataSink qui peuvent ensuite être utilisés pour lire et écrire les `DynamicFrame`.

Vous pouvez également utiliser `GlueContext` pour définir un nombre cible de partitions (par défaut 20) dans le `DynamicFrame` si le nombre de partitions créées à partir de la source est inférieure à un seuil minimal pour les partitions (par défaut 10).

addIngestionTimeColonnes def

```
def addIngestionTimeColumns(
         df : DataFrame,
         timeGranularity : String = "") : dataFrame
```

Ajoute des colonnes de temps d'ingestion, telles que `ingest_year`, `ingest_month`, `ingest_day`, `ingest_hour`, `ingest_minute` à l'entrée `DataFrame`. Cette fonction est automatiquement générée dans le script généré par AWS Glue lorsque vous spécifiez une table Data Catalog avec Amazon S3 comme cible. Cette fonction met automatiquement à jour la partition avec les colonnes de temps d'ingestion sur la table de sortie. Cela permet aux données de sortie d'être automatiquement partitionnées à l'heure d'ingestion sans nécessiter de colonnes d'heure d'ingestion explicites dans les données d'entrée.

- `dataFrame` – `dataFrame` auquel ajouter les colonnes de temps d'ingestion.
- `timeGranularity` — granularité des colonnes de temps. Les valeurs valides sont « `day` », « `hour` » et « `minute` ». Par exemple, si « `hour` » est transmis à la fonction, les colonnes de temps « `ingest_year` », « `ingest_month` », « `ingest_day` » et « `ingest_hour` » seront ajoutées à l'original « `dataFrame` ».

Renvoie le bloc de données après l'ajout des colonnes de granularité temporelle.

Exemple :

```
glueContext.addIngestionTimeColumns(dataFrame, "hour")
```

def createDataFrame FromOptions

```
def createDataFrameFromOptions( connectionType : String,
                          connectionOptions : JsonOptions,
                          transformationContext : String = "",
                          format : String = null,
                          formatOptions : JsonOptions = JsonOptions.empty
                        ) : DataSource
```

Renvoie un `DataFrame` créé avec la connexion et le format spécifiés. Utilisez cette fonction uniquement avec les sources AWS de streaming Glue.

- `connectionType` : type de connexion en streaming. Les valeurs valides sont `kinesis` et `kafka`.
- `connectionOptions` : options de connexion, qui sont différentes pour Kinesis et Kafka. Vous trouverez la liste de toutes les options de connexion pour chaque source de données de streaming sur la page Types et options de connexion pour ETL dans AWS Glue pour Spark. Notez les différences suivantes dans les options de connexion en streaming :
  - Les sources de streaming Kinesis nécessitent `streamARN`, `startingPosition`, `inferSchema` et `classification`.
  - Les sources de streaming Kafka nécessitent `connectionName`, `topicName`, `startingOffsets`, `inferSchema` et `classification`.
- `transformationContext` : contexte de transformation à utiliser (facultatif).
- `format` : spécification de format (facultatif). Utilisée pour une connexion Amazon S3 ou AWS Glue prenant en charge plusieurs formats. Pour plus d'informations sur les formats pris en charge, veuillez consulter Options de format pour les entrées et sorties dans AWS Glue pour Spark.
- `formatOptions` : options de format pour le format spécifié. Pour de plus amples informations sur les options de formats pris en charge, veuillez consulter Options de format de données.

Exemple pour la source de streaming Amazon Kinesis :

```
val data_frame_datasource0 =
glueContext.createDataFrameFromOptions(transformationContext = "datasource0",
 connectionType = "kinesis",
connectionOptions = JsonOptions("""{"streamName": "example_stream", "startingPosition":
 "TRIM_HORIZON", "inferSchema": "true", "classification": "json"}}""""))
```

Exemple pour la source de streaming Kafka :

```
val data_frame_datasource0 =
glueContext.createDataFrameFromOptions(transformationContext = "datasource0",
 connectionType = "kafka",
connectionOptions = JsonOptions("""{"connectionName": "example_connection",
 "topicName": "example_topic", "startingPosition": "earliest", "inferSchema": "false",
 "classification": "json", "schema":"`column1` STRING, `column2` STRING"}""" ))
```

forEachBatch

`forEachBatch(frame, batch_function, options)`

S'applique à `batch_function` transmis à chaque micro-lot lu à partir de la source de streaming.

- `frame`— Le DataFrame contenant le microlot actuel.
- `batch_function` — fonction qui sera appliquée à chaque micro-lot.
- `options` — collection de paires clé-valeur qui contient des informations sur le traitement de micro-lots. Les options suivantes sont requises :
  - `windowSize` — durée de traitement de chaque lot.
  - `checkpointLocation` — emplacement dans lequel les points de contrôle sont stockés pour la tâche ETL en streaming.
  - `batchMaxRetries` – nombre maximum de nouvelles tentatives pour ce lot en cas d'échec. La valeur par défaut est 3. Cette option n'est configurable que pour Glue version 2.0 et ultérieure.

Exemple :

```
glueContext.forEachBatch(data_frame_datasource0, (dataFrame: Dataset[Row], batchId:
 Long) =>
   {
      if (dataFrame.count() > 0)
        {
          val datasource0 = DynamicFrame(glueContext.addIngestionTimeColumns(dataFrame,
 "hour"), glueContext)
          // @type: DataSink
          // @args: [database = "tempdb", table_name = "fromoptionsoutput",
 stream_batch_time = "100 seconds",
          //      stream_checkpoint_location = "s3://from-options-testing-eu-central-1/
fromOptionsOutput/checkpoint/",
          //      transformation_ctx = "datasink1"]
          // @return: datasink1
```

```
          // @inputs: [frame = datasource0]
          val options_datasink1 = JsonOptions(
             Map("partitionKeys" -> Seq("ingest_year", "ingest_month","ingest_day",
 "ingest_hour"),
             "enableUpdateCatalog" -> true))
          val datasink1 = glueContext.getCatalogSink(
             database = "tempdb",
             tableName = "fromoptionsoutput",
             redshiftTmpDir = "",
             transformationContext = "datasink1",
             additionalOptions = options_datasink1).writeDynamicFrame(datasource0)
        }
   }, JsonOptions("""{"windowSize" : "100 seconds",
         "checkpointLocation" : "s3://from-options-testing-eu-central-1/
fromOptionsOutput/checkpoint/"}"""))
```

def getCatalogSink

```
def getCatalogSink( database : String,
        tableName : String,
        redshiftTmpDir : String = "",
        transformationContext : String = ""
        additionalOptions: JsonOptions = JsonOptions.empty,
        catalogId: String = null
) : DataSink
```

Crée un DataSink qui écrit dans un emplacement spécifié dans une table définie dans Data Catalog.

- `database` – Nom de base de données dans le catalogue de données.
- `tableName` – Nom de la table dans le catalogue de données.
- `redshiftTmpDir` – Répertoire intermédiaire temporaire à utiliser avec certains récepteurs de données. Valeur définie sur vide par défaut.
- `transformationContext` – Contexte de transformation associé au récepteur à utiliser par les signets de la tâche. Valeur définie sur vide par défaut.
- `additionalOptions` — Options supplémentaires fournies à AWS Glue.
- `catalogId` — ID du catalogue (ID du compte) Data Catalog auquel vous accédez. Lorsque la valeur est null, l'ID de compte par défaut de l'appelant est utilisé.

Renvoie le `DataSink`.

def getCatalogSource

```
def getCatalogSource( database : String,
                      tableName : String,
                      redshiftTmpDir : String = "",
                      transformationContext : String = ""
                      pushDownPredicate : String = " "
                      additionalOptions: JsonOptions = JsonOptions.empty,
                      catalogId: String = null
                    ) : DataSource
```

Crée un DataSource trait qui lit les données à partir d'une définition de table dans Data Catalog.

- `database` — nom de la base de données dans Data Catalog.
- `tableName` – Nom de la table dans le catalogue de données.
- `redshiftTmpDir` – Répertoire intermédiaire temporaire à utiliser avec certains récepteurs de données. Valeur définie sur vide par défaut.
- `transformationContext` – Contexte de transformation associé au récepteur à utiliser par les signets de la tâche. Valeur définie sur vide par défaut.
- `pushDownPredicate` – Filtre les partitions sans avoir à répertorier ni lire tous les fichiers de votre jeu de données. Pour plus d'informations, consultez Préfiltrage à l'aide des prédicats pushdown.
- `additionalOptions` — Ensemble de paires nom-valeur facultatives. Les options possibles comprennent celles répertoriées dans Types et options de connexion pour ETL dans AWS Glue pour Spark, sauf pour `endpointUrl`, `streamName`, `bootstrap.servers`, `security.protocol`, `topicName`, `classification` et `delimiter`. Est une autre option prise en charge `catalogPartitionPredicate`:

  `catalogPartitionPredicate` – Vous pouvez passer une expression de catalogue à filtrer en fonction des colonnes d'index. Cela envoie le filtrage du côté serveur. Pour en savoir plus, consultez AWS Glue Indexes de partition. Notez que `push_down_predicate` et `catalogPartitionPredicate` utilisent des syntaxes différentes. Le premier utilise la syntaxe standard SQL Spark et le dernier utilise l'analyseur JSQL.
- `catalogId` — ID du catalogue (ID du compte) Data Catalog auquel vous accédez. Lorsque la valeur est null, l'ID de compte par défaut de l'appelant est utilisé.

Renvoie le `DataSource`.

Exemple pour la source de streaming

```
val data_frame_datasource0 = glueContext.getCatalogSource(
    database = "tempdb",
    tableName = "test-stream-input",
    redshiftTmpDir = "",
    transformationContext = "datasource0",
    additionalOptions = JsonOptions("""{
        "startingPosition": "TRIM_HORIZON", "inferSchema": "false"}""")
    ).getDataFrame()
```

def get JDBCSink

```
def getJDBCSink( catalogConnection : String,
                 options : JsonOptions,
                 redshiftTmpDir : String = "",
                 transformationContext : String = "",
                 catalogId: String = null
               ) : DataSink
```

Crée un DataSink qui écrit dans une base de données JDBC spécifiée d'un objet `Connection` dans Data Catalog. L'objet `Connection` comporte des informations de connexion pour se connecter à un récepteur JDBC incluant l'URL, le nom d'utilisateur, le mot de passe, le VPC, le sous-réseau et les groupes de sécurité.

- `catalogConnection` – Nom de la connexion dans le catalogue de données qui contient l'URL JDBC sur laquelle écrire.
- `options` – Chaîne de paires nom-valeur JSON qui fournissent des informations supplémentaires nécessaires pour écrire dans un magasin de données JDBC. Cela inclut notamment les éléments suivants :
  - dbtable (obligatoire) — Nom de la table JDBC. Pour les magasins de données JDBC qui prennent en charge les schémas dans une base de données, spécifiez `schema.table-name`. Si aucun schéma n'est fourni, c'est le schéma « public » par défaut qui est utilisé. L'exemple suivant illustre un paramètre d'options qui pointe sur un schéma nommé `test` et une table nommée `test_table` dans la base de données `test_db`.

    ```
    options = JsonOptions("""{"dbtable": "test.test_table", "database": "test_db"}""")
    ```

  - database (obligatoire) — Nom de la base de données JDBC.

  - Toutes les options supplémentaires transmises directement à l'enregistreur JDBC SparkSQL. Pour plus d'informations, consultez Redshift data source for Spark.
- `redshiftTmpDir` — répertoire intermédiaire temporaire à utiliser avec certains récepteurs de données. Valeur définie sur vide par défaut.
- `transformationContext` – Contexte de transformation associé au récepteur à utiliser par les signets de la tâche. Valeur définie sur vide par défaut.
- `catalogId` — ID du catalogue (ID du compte) Data Catalog auquel vous accédez. Lorsque la valeur est null, l'ID de compte par défaut de l'appelant est utilisé.

Exemple de code :

```
getJDBCSink(catalogConnection = "my-connection-name", options =
 JsonOptions("""{"dbtable": "my-jdbc-table", "database": "my-jdbc-db"}"""),
 redshiftTmpDir = "", transformationContext = "datasink4")
```

Renvoie le `DataSink`.

def getSink

```
def getSink( connectionType : String,
             connectionOptions : JsonOptions,
             transformationContext : String = ""
           ) : DataSink
```

Crée un DataSink fichier qui écrit des données vers une destination telle qu'Amazon Simple Storage Service (Amazon S3), JDBC ou le AWS Glue Data Catalog, ou un flux de données Apache Kafka ou Amazon Kinesis.

- `connectionType` — Type de connexion. Consultez the section called “Paramètres de connexion”.
- `connectionOptions` — Chaîne JSON de paires nom-valeur JSON qui fournissent des informations supplémentaires pour établir la connexion avec le récepteur de données. Consultez the section called “Paramètres de connexion”.
- `transformationContext` – Contexte de transformation associé au récepteur à utiliser par les signets de la tâche. Valeur définie sur vide par défaut.

Renvoie le `DataSink`.

getSinkWithFormat def

```
def getSinkWithFormat( connectionType : String,
                       options : JsonOptions,
                       transformationContext : String = "",
                       format : String = null,
                       formatOptions : JsonOptions = JsonOptions.empty
                     ) : DataSink
```

Crée un DataSink qui écrit des données dans une destination comme Amazon S3, JDBC ou le catalogue de données, ou un flux de données Apache Kafka ou Amazon Kinesis. Définit également le format des données à écrire dans la destination.

- `connectionType` — Type de connexion. Consultez the section called "Paramètres de connexion".
- `options` — Chaîne JSON de paires nom-valeur JSON qui fournissent des informations supplémentaires pour établir une connexion avec le récepteur de données. Consultez the section called "Paramètres de connexion".
- `transformationContext` – Contexte de transformation associé au récepteur à utiliser par les signets de la tâche. Valeur définie sur vide par défaut.
- `format` — Format des données à écrire dans la destination.
- `formatOptions` – Chaîne JSON de paires nom-valeur qui fournissent des options supplémentaires pour le formatage des données à la destination. Consultez Options de format de données.

Renvoie le `DataSink`.

def getSource

```
def getSource( connectionType : String,
               connectionOptions : JsonOptions,
               transformationContext : String = ""
               pushDownPredicate
             ) : DataSource
```

Crée un DataSource trait fichier qui lit les données d'une source telle qu'Amazon S3, JDBC ou le AWS Glue Data Catalog. Prend également en charge les sources de données de streaming Kafka et Kinesis.

- `connectionType` – Type de données de la source de données. Consultez the section called "Paramètres de connexion".
- `connectionOptions` — Chaîne de paires nom-valeur JSON qui fournissent des informations supplémentaires pour établir la connexion avec la source de données. Pour de plus amples informations, veuillez consulter the section called "Paramètres de connexion".

  Une source de streaming Kinesis nécessite les options de connexion suivantes : `streamARN`, `startingPosition`, `inferSchema` et `classification`.

  Une source de streaming Kafka nécessite les options de connexion suivantes : `connectionName`, `topicName`, `startingOffsets`, `inferSchema` et `classification`.
- `transformationContext` – Contexte de transformation associé au récepteur à utiliser par les signets de la tâche. Valeur définie sur vide par défaut.
- `pushDownPredicate` – Prédicat sur les colonnes de partition.

Renvoie le `DataSource`.

Exemple pour la source de streaming Amazon Kinesis :

```
val kinesisOptions = jsonOptions()
data_frame_datasource0 = glueContext.getSource("kinesis",
 kinesisOptions).getDataFrame()

private def jsonOptions(): JsonOptions = {
    new JsonOptions(
      s"""{"streamARN": "arn:aws:kinesis:eu-central-1:123456789012:stream/
fromOptionsStream",
         |"startingPosition": "TRIM_HORIZON",
         |"inferSchema": "true",
         |"classification": "json"}""".stripMargin)
}
```

Exemple pour la source de streaming Kafka :

```
val kafkaOptions = jsonOptions()
```

```
val data_frame_datasource0 = glueContext.getSource("kafka",
 kafkaOptions).getDataFrame()

private def jsonOptions(): JsonOptions = {
    new JsonOptions(
      s"""{"connectionName": "ConfluentKafka",
         |"topicName": "kafka-auth-topic",
         |"startingOffsets": "earliest",
         |"inferSchema": "true",
         |"classification": "json"}""".stripMargin)
 }
```

getSourceWithFormat def

```
def getSourceWithFormat( connectionType : String,
                         options : JsonOptions,
                         transformationContext : String = "",
                         format : String = null,
                         formatOptions : JsonOptions = JsonOptions.empty
                       ) : DataSource
```

Crée un fichier DataSource trait qui lit les données d'une source telle qu'Amazon S3, JDBC ou le AWS Glue Data Catalog, et définit également le format des données stockées dans la source.

- `connectionType` – Type de données de la source de données. Consultez the section called "Paramètres de connexion".
- `options` – Chaîne de paires nom-valeur JSON qui fournissent des informations supplémentaires pour établir la connexion avec la source de données. Consultez the section called "Paramètres de connexion".
- `transformationContext` – Contexte de transformation associé au récepteur à utiliser par les signets de la tâche. Valeur définie sur vide par défaut.
- `format` – Format des données stockées à la source. Lorsque le `connectionType` est « s3 », vous pouvez également spécifier `format`. Peut être « avro », « csv », « grokLog », « ion », « json », « xml », « parquet » ou « orc ».
- `formatOptions` – Chaîne JSON de paires nom-valeur qui fournissent des options supplémentaires pour l'analyse des données à la source. Consultez Options de format de données.

Renvoie le `DataSource`.

Exemples

Créez un DynamicFrame à partir d'une source de données qui est un fichier de valeurs séparées par des virgules (CSV) sur Amazon S3 :

```
val datasource0 = glueContext.getSourceWithFormat(
    connectionType="s3",
    options =JsonOptions(s"""{"paths": [ "s3://csv/nycflights.csv"]}"""),
    transformationContext = "datasource0",
    format = "csv",
    formatOptions=JsonOptions(s"""{"withHeader":"true","separator": ","}""")
    ).getDynamicFrame()
```

Créez un DynamicFrame à partir d'une source de données PostgreSQL à l'aide d'une connexion JDBC :

```
val datasource0 = glueContext.getSourceWithFormat(
    connectionType="postgresql",
    options =JsonOptions(s"""{
      "url":"jdbc:postgresql://databasePostgres-1.rds.amazonaws.com:5432/testdb",
      "dbtable": "public.company",
      "redshiftTmpDir":"",
      "user":"username",
      "password":"password123"
    }"""),
    transformationContext = "datasource0").getDynamicFrame()
```

Créez un DynamicFrame à partir d'une source de données MySQL à l'aide d'une connexion JDBC :

```
 val datasource0 = glueContext.getSourceWithFormat(
    connectionType="mysql",
    options =JsonOptions(s"""{
      "url":"jdbc:mysql://databaseMysql-1.rds.amazonaws.com:3306/testdb",
      "dbtable": "athenatest_nycflights13_csv",
      "redshiftTmpDir":"",
      "user":"username",
      "password":"password123"
    }"""),
    transformationContext = "datasource0").getDynamicFrame()
```

def getSparkSession

```
def getSparkSession : SparkSession
```

Permet d'obtenir l'objet `SparkSession` associé à GlueContext. Utilisez cet SparkSession objet pour enregistrer des tables et UDFs pour l'utiliser avec `DataFrame` Created From DynamicFrames.

Renvoie le SparkSession.

def startTransaction

```
def startTransaction(readOnly: Boolean):String
```

Démarrer une nouvelle transaction. Appelle en interne l'API Démarrer la transaction Lake Formation.

- `readOnly` – Valeur booléenne indiquant si cette transaction doit être en lecture seule ou en lecture et en écriture. Les écritures effectuées à l'aide d'un ID de transaction en lecture seule seront rejetées. Les transactions en lecture seule n'ont pas besoin d'être validées.

Retourne l'ID de transaction.

def commitTransaction

```
def commitTransaction(transactionId: String, waitForCommit: Boolean): Boolean
```

Tentative de validation de la transaction spécifiée. `commitTransaction` peut être renvoyé avant la fin de la validation de la transaction. Appelle en interne la Lake Formation commitTransaction API.

- `transactionId` – (Chaîne) La transaction à valider.
- `waitForCommit` – (Booléen) Détermine si le `commitTransaction` retourne immédiatement. La valeur par défaut est True. Si elle est false,`commitTransaction` interroge et attend que la transaction soit validée. Le temps d'attente est limité à 1 minute en utilisant le backoff exponentiel avec un maximum de 6 tentatives de nouvelle tentative.

Renvoie une valeur de type Booléen pour indiquer si la validation a été effectuée ou non.

def cancelTransaction

```
def cancelTransaction(transactionId: String): Unit
```

Tentative d'annulation de la transaction spécifiée. Appelle en interne l'CancelTransactionAPI Lake Formation.

- `transactionId` – (Chaîne) La transaction à annuler.

Retourne une exception `TransactionCommittedException` si la transaction a déjà été validée.

def this

```
def this( sc : SparkContext,
          minPartitions : Int,
          targetPartitions : Int )
```

Crée un objet `GlueContext` utilisant le `SparkContext` spécifié, les partitions minimales et les partitions cible.

- `sc` — Le `SparkContext`.
- `minPartitions` — Nombre minimal de partitions.
- `targetPartitions` — Nombre cible de partitions.

Renvoie le `GlueContext`.

def this

```
def this( sc : SparkContext )
```

Crée un objet `GlueContext` avec le `SparkContext` fourni. Définit le nombre minimal de partitions à 10 et de partitions cibles à 20.

- `sc` — Le `SparkContext`.

Renvoie le `GlueContext`.

def this

```
def this( sparkContext : JavaSparkContext )
```

Crée un objet `GlueContext` avec le `JavaSparkContext` fourni. Définit le nombre minimal de partitions à 10 et de partitions cibles à 20.

- `sparkContext` — Le `JavaSparkContext`.

Renvoie le `GlueContext`.

MappingSpec

Package : com.amazonaws.services.glue

MappingSpec classe de cas

```
case class MappingSpec( sourcePath: SchemaPath,
                        sourceType: DataType,
                        targetPath: SchemaPath,
                        targetType: DataTyp
                       ) extends Product4[String, String, String, String] {
  override def _1: String = sourcePath.toString
  override def _2: String = ExtendedTypeName.fromDataType(sourceType)
  override def _3: String = targetPath.toString
  override def _4: String = ExtendedTypeName.fromDataType(targetType)
}
```

- `sourcePath` — `SchemaPath` du champ source.
- `sourceType` — `DataType` du champ source.
- `targetPath` — `SchemaPath` du champ cible.
- `targetType` — `DataType` du champ cible.

Un `MappingSpec` spécifie un mappage entre un chemin d'accès source et un type de données source, et un chemin cible et un type de données cible. La valeur du chemin d'accès source de l'image source s'affiche dans l'image cible du chemin cible. Le type de données source est converti dans le type de données cible.

Il s'étend depuis `Product4` afin que vous puissiez manipuler n'importe quel `Product4` dans votre interface `applyMapping`.

MappingSpec objet

```
object MappingSpec
```

L'objet `MappingSpec` coontient les membres suivants :

Val orderingByTarget

```
val orderingByTarget: Ordering[MappingSpec]
```

Def apply

```
def apply( sourcePath : String,
           sourceType : DataType,
           targetPath : String,
           targetType : DataType
         ) : MappingSpec
```

Crée un `MappingSpec`.

- `sourcePath` — Représentation sous forme de chaîne du chemin d'accès source.
- `sourceType` – source `DataType`.
- `targetPath` — Représentation sous forme de chaîne du chemin d'accès cible.
- `targetType` – cible `DataType`.

Retourne un `MappingSpec`.

def apply

```
def apply( sourcePath : String,
           sourceTypeString : String,
           targetPath : String,
           targetTypeString : String
         ) : MappingSpec
```

Crée un `MappingSpec`.

- `sourcePath` — Représentation sous forme de chaîne du chemin d'accès source.
- `sourceType` — Représentation sous forme de chaîne du type de données source.
- `targetPath` — Représentation sous forme de chaîne du chemin d'accès cible.
- `targetType` — Représentation sous forme de chaîne du type de données cible.

Renvoie un MappingSpec.

Def apply

```
def apply( product : Product4[String, String, String, String] ) : MappingSpec
```

Crée un `MappingSpec`.

- `product` — `Product4` du chemin d'accès source, type de données source, chemin d'accès cible et type de données cible.

Retourne un `MappingSpec`.

AWS GlueScala ResolveSpec APIs

Rubriques

- ResolveSpec objet
- ResolveSpec classe de cas

Package : com.amazonaws.services.glue

ResolveSpec objet

ResolveSpec

```
object ResolveSpec
```

def

```
def apply( path : String,
           action : String
         ) : ResolveSpec
```

Crée un `ResolveSpec`.

- `path` — Représentation sous forme de chaîne du champ de choix qui doit être résolu.
- `action` — action de résolution. L'action peut être `Project`, `KeepAsStruct` ou `Cast`.

Renvoie le `ResolveSpec`.

def

```
def apply( product : Product2[String, String] ) : ResolveSpec
```

Crée un `ResolveSpec`.

- `product` – `Product2` de : chemin d'accès source, action de résolution.

Renvoie le `ResolveSpec`.

ResolveSpec classe de cas

```
case class ResolveSpec extends Product2[String, String]  (
           path : SchemaPath,
           action : String )
```

Crée un `ResolveSpec`.

- `path` — Le `SchemaPath` du champ de choix qui doit être résolu.
- `action` — action de résolution. L'action peut être `Project`, `KeepAsStruct` ou `Cast`.

ResolveSpec méthodes def

```
def _1 : String
```

```
def _2 : String
```

AWS GlueScala ArrayNode APIs

Package : com.amazonaws.services.glue.types

ArrayNode classe de cas

ArrayNode

```
case class ArrayNode extends DynamicNode  (
           value : ArrayBuffer[DynamicNode] )
```

ArrayNode méthodes def

```
def add( node : DynamicNode )
```

```
def clone
```

```
def equals( other : Any )
```

```
def get( index : Int ) : Option[DynamicNode]
```

```
def getValue
```

```
def hashCode : Int
```

```
def isEmpty : Boolean
```

```
def nodeType
```

```
def remove( index : Int )
```

```
def this
```

```
def toIterator : Iterator[DynamicNode]
```

```
def toJson : String
```

```
def update( index : Int,
            node : DynamicNode )
```

AWS GlueScala BinaryNode APIs

Package : com.amazonaws.services.glue.types

BinaryNode classe de cas

BinaryNode

```
case class BinaryNode extends ScalarNode(value, TypeCode.BINARY)  (
           value : Array[Byte] )
```

BinaryNode champs de val

- `ordering`

BinaryNode méthodes def

```
def clone
```

```
def equals( other : Any )
```

```
def hashCode : Int
```

AWS GlueScala BooleanNode APIs

Package : com.amazonaws.services.glue.types

BooleanNode classe de cas

BooleanNode

```
case class BooleanNode extends ScalarNode(value, TypeCode.BOOLEAN)  (
           value : Boolean )
```

BooleanNode champs de val

- `ordering`

BooleanNode méthodes def

```
def equals( other : Any )
```

AWS GlueScala ByteNode APIs

Package : com.amazonaws.services.glue.types

ByteNode classe de cas

ByteNode

```
case class ByteNode extends ScalarNode(value, TypeCode.BYTE)  (
           value : Byte )
```

ByteNode champs de val

- `ordering`

ByteNode méthodes def

```
def equals( other : Any )
```

AWS GlueScala DateNode APIs

Package : com.amazonaws.services.glue.types

DateNode classe de cas

DateNode

```
case class DateNode extends ScalarNode(value, TypeCode.DATE)  (
           value : Date )
```

DateNode champs de val

- `ordering`

DateNode méthodes def

```
def equals( other : Any )
```

```
def this( value : Int )
```

AWS GlueScala DecimalNode APIs

Package : com.amazonaws.services.glue.types

DecimalNode classe de cas

DecimalNode

```
case class DecimalNode extends ScalarNode(value, TypeCode.DECIMAL)  (
           value : BigDecimal )
```

DecimalNode champs de val

- `ordering`

DecimalNode méthodes def

```
def equals( other : Any )
```

```
def this( value : Decimal )
```

AWS GlueScala DoubleNode APIs

Package : com.amazonaws.services.glue.types

DoubleNode classe de cas

DoubleNode

```
case class DoubleNode extends ScalarNode(value, TypeCode.DOUBLE)  (
           value : Double )
```

DoubleNode champs de val

- `ordering`

DoubleNode méthodes def

```
def equals( other : Any )
```

AWS GlueScala DynamicNode APIs

Rubriques

- DynamicNode classe
- DynamicNode objet

Package : com.amazonaws.services.glue.types

DynamicNode classe

DynamicNode

```
class DynamicNode extends Serializable with Cloneable
```

DynamicNode méthodes def

```
def getValue : Any
```

Obtenir la valeur brute et la lier à l'enregistrement actuel :

```
def nodeType : TypeCode
```

```
def toJson : String
```

Méthode pour déboguer :

```
def toRow( schema : Schema,
           options : Map[String, ResolveOption]
         ) : Row
```

```
def typeName : String
```

DynamicNode objet

DynamicNode

```
object DynamicNode
```

DynamicNode méthodes def

```
def quote( field : String,
           useQuotes : Boolean
```

```
        ) : String
```

```
def quote( node : DynamicNode,
           useQuotes : Boolean
         ) : String
```

EvaluateDataQuality classe

AWS Glue Data Quality est en version préliminaire AWS Glue et est susceptible d'être modifiée.

Package : com.amazonaws.services.glue.dq

```
object EvaluateDataQuality
```

Def apply

```
def apply(frame: DynamicFrame,
            ruleset: String,
            publishingOptions: JsonOptions = JsonOptions.empty): DynamicFrame
```

Évalue un ensemble de règles de qualité des données par rapport à un `DynamicFrame`, et renvoie un nouveau `DynamicFrame` avec les résultats de l'évaluation. Pour en savoir plus sur AWS Glue Data Quality, consultezAWS Glue Qualité des données.

- `frame` – `DynamicFrame` dont vous souhaitez évaluer la qualité des données.
- `ruleset` – Ensemble de règles DQDL (Data Quality Definition Language) au format de chaîne. Pour en savoir plus sur DQDL, consultez le guide Référence DQDL (Data Quality Definition Language).
- `publishingOptions` – Dictionnaire qui spécifie les options de publication suivantes des résultats et des métriques d'une évaluation :
  - `dataQualityEvaluationContext`— Chaîne qui indique l'espace de noms sous lequel AWS Glue doit publier Amazon CloudWatch les métriques et les résultats de qualité des données. Les statistiques agrégées apparaissent dans CloudWatch, tandis que les résultats complets apparaissent dans l'interface de AWS Glue Studio.
    - Obligatoire : non
    - Valeur par défaut : `default_context`

- `enableDataQualityCloudWatchMetrics`— Spécifie si les résultats de l'évaluation de la qualité des données doivent être publiés sur CloudWatch. Vous spécifiez un espace de noms pour les métriques à l'aide de l'option `dataQualityEvaluationContext`.
  - Obligatoire : non
  - Valeur par défaut : False
- `enableDataQualityResultsPublishing` – Spécifie si les résultats relatifs à la qualité des données doivent être visibles dans l'onglet Data Quality (Qualité des données) de l'interface AWS Glue Studio.
  - Obligatoire : non
  - Valeur par défaut : True
- `resultsS3Prefix`— Spécifie l'emplacement Amazon S3 où AWS Glue peut écrire les résultats de l'évaluation de la qualité des données.
  - Obligatoire : non
  - Valeur par défaut : "" (chaîne vide)

Exemple

L'exemple de code suivant montre comment évaluer la qualité des données pour `DynamicFrame` avant d'effectuer une transformation `SelectFields`. Le script vérifie que l'ensemble des règles de qualité des données sont conformes avant de tenter la transformation.

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.MappingSpec
import com.amazonaws.services.glue.errors.CallSite
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._
import com.amazonaws.services.glue.dq.EvaluateDataQuality

object GlueApp {
  def main(sysArgs: Array[String]) {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    // @params: [JOB_NAME]
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)
```

```
    // Create DynamicFrame with data
    val Legislators_Area = glueContext.getCatalogSource(database="legislators",
 tableName="areas_json", transformationContext="S3bucket_node1").getDynamicFrame()

    // Define data quality ruleset
    val DQ_Ruleset = """
      Rules = [ColumnExists "id"]
    """

    // Evaluate data quality
    val DQ_Results = EvaluateDataQuality.apply(frame=Legislators_Area,
 ruleset=DQ_Ruleset, publishingOptions=JsonOptions("""{"dataQualityEvaluationContext":
 "Legislators_Area", "enableDataQualityMetrics": "true",
 "enableDataQualityResultsPublishing": "true"}"""))
    assert(DQ_Results.filter(_.getField("Outcome").contains("Failed")).count == 0,
 "Failing DQ rules for Legislators_Area caused the job to fail.")

    // Script generated for node Select Fields
    val SelectFields_Results = Legislators_Area.selectFields(paths=Seq("id", "name"),
 transformationContext="Legislators_Area")

    Job.commit()
  }
}
```

AWS GlueScala FloatNode APIs

Package : com.amazonaws.services.glue.types

FloatNode classe de cas

FloatNode

```
case class FloatNode extends ScalarNode(value, TypeCode.FLOAT)  (
           value : Float )
```

FloatNode champs de val

- `ordering`

FloatNode méthodes def

```
def equals( other : Any )
```

FillMissingValues classe

Package : com.amazonaws.services.glue.ml

```
object FillMissingValues
```

def apply

```
def apply(frame: DynamicFrame,
          missingValuesColumn: String,
          outputColumn: String = "",
          transformationContext: String = "",
          callSite: CallSite = CallSite("Not provided", ""),
          stageThreshold: Long = 0,
          totalThreshold: Long = 0): DynamicFrame
```

Remplit les valeurs manquantes d'un cadre dynamique dans une colonne spécifiée et renvoie un nouveau cadre avec des estimations dans une nouvelle colonne. Pour les lignes sans valeurs manquantes, la valeur de la colonne spécifiée est dupliquée dans la nouvelle colonne.

- `frame`— Le champ DynamicFrame dans lequel les valeurs manquantes doivent être remplies. Obligatoire.
- `missingValuesColumn` — colonne contenant les valeurs manquantes (valeurs `null` et chaînes vides). Obligatoire.
- `outputColumn` — nom de la nouvelle colonne qui contiendra les valeurs estimées pour toutes les lignes dont la valeur était manquante. Facultatif ; la valeur par défaut est la valeur de `missingValuesColumn` avec le suffixe `"_filled"`.
- `transformationContext` — chaîne unique utilisée pour identifier les informations sur l'état (facultatif).
- `callSite` — utilisé pour fournir les informations de contexte pour le signalement d'erreurs.
- `stageThreshold` — nombre maximal d'erreurs qui peuvent avoir lieu dans la transformation avant qu'elle ne soit arrêtée (facultatif ; la valeur par défaut est zéro).
- `totalThreshold` — nombre maximal d'erreurs pouvant se produire globalement avant que le processus de traitement des erreurs ne soit arrêté (facultatif ; la valeur par défaut est zéro).

Renvoie une nouvelle image dynamique avec une colonne supplémentaire qui contient des estimations pour les lignes avec des valeurs manquantes et la valeur actuelle pour les autres lignes.

FindMatches classe

Package : com.amazonaws.services.glue.ml

```
object FindMatches
```

def apply

```
def apply(frame: DynamicFrame,
            transformId: String,
            transformationContext: String = "",
            callSite: CallSite = CallSite("Not provided", ""),
            stageThreshold: Long = 0,
            totalThreshold: Long = 0,
            enforcedMatches: DynamicFrame = null): DynamicFrame,
   computeMatchConfidenceScores: Boolean
```

Recherchez des correspondances dans une image d'entrée et renvoyez une nouvelle image avec une nouvelle colonne contenant un ID unique par groupe de correspondance.

- `frame`— L' DynamicFrame endroit où trouver des correspondances. Obligatoire.
- `transformId`— Un identifiant unique associé à la FindMatches transformation à appliquer au cadre d'entrée. Obligatoire.
- `transformationContext` — Identifiant de ce `DynamicFrame`. Le `transformationContext` est utilisé en tant que clé pour l'état de signet de tâche conservé d'une exécution à l'autre. Facultatif.
- `callSite` – Permet de fournir des informations contextuelles pour le signalement d'erreurs. Ces valeurs sont automatiquement définies lors de l'appel à partir de Python. Facultatif.
- `stageThreshold` – Nombre maximal d'enregistrements d'erreurs autorisés depuis le calcul du `DynamicFrame` avant de lever une exception, à l'exclusion des enregistrements présents dans le `DynamicFrame` précédent. Facultatif. La valeur par défaut est zéro.
- `totalThreshold` – Nombre maximal d'enregistrements d'erreur avant qu'une exception ne soit levée, y compris ceux des images précédentes. Facultatif. La valeur par défaut est zéro.
- `enforcedMatches` – Image pour les correspondances forcées. Facultatif. La valeur par défaut est `null`.

- `computeMatchConfidenceScores` — Valeur booléenne indiquant s'il faut calculer un score de confiance pour chaque groupe d'enregistrements correspondants. Facultatif. La valeur par défaut est false.

Renvoie une nouvelle image dynamique avec un identifiant unique affecté à chaque groupe d'enregistrements correspondants.

FindIncrementalMatches classe

Package : com.amazonaws.services.glue.ml

```
object FindIncrementalMatches
```

def apply

```
apply(existingFrame: DynamicFrame,
            incrementalFrame: DynamicFrame,
            transformId: String,
            transformationContext: String = "",
            callSite: CallSite = CallSite("Not provided", ""),
            stageThreshold: Long = 0,
            totalThreshold: Long = 0,
            enforcedMatches: DynamicFrame = null): DynamicFrame,
   computeMatchConfidenceScores: Boolean
```

Recherchez des correspondances dans les images existantes et incrémentielles, et renvoyez une nouvelle image avec une colonne contenant un ID unique par groupe de correspondance.

- `existingframe` – Image existante à laquelle un ID correspondant a été attribué pour chaque groupe. Obligatoire.
- `incrementalframe` – Image incrémentielle utilisée pour rechercher des correspondances avec l'image existante. Obligatoire.
- `transformId`— Un identifiant unique associé à la FindIncrementalMatches transformation à appliquer aux trames d'entrée. Obligatoire.
- `transformationContext` — Identifiant de ce `DynamicFrame`. Le `transformationContext` est utilisé en tant que clé pour l'état de signet de tâche conservé d'une exécution à l'autre. Facultatif.

- `callSite` – Permet de fournir des informations contextuelles pour le signalement d'erreurs. Ces valeurs sont automatiquement définies lors de l'appel à partir de Python. Facultatif.
- `stageThreshold` – Nombre maximal d'enregistrements d'erreurs autorisés depuis le calcul du `DynamicFrame` avant de lever une exception, à l'exclusion des enregistrements présents dans le `DynamicFrame` précédent. Facultatif. La valeur par défaut est zéro.
- `totalThreshold` – Nombre maximal d'enregistrements d'erreur avant qu'une exception ne soit levée, y compris ceux des images précédentes. Facultatif. La valeur par défaut est zéro.
- `enforcedMatches` – Image pour les correspondances forcées. Facultatif. La valeur par défaut est `null`.
- `computeMatchConfidenceScores` — Valeur booléenne indiquant s'il faut calculer un score de confiance pour chaque groupe d'enregistrements correspondants. Facultatif. La valeur par défaut est false.

Renvoie une nouvelle image dynamique avec un identifiant unique affecté à chaque groupe d'enregistrements correspondants.

AWS GlueScala IntegerNode APIs

Package : com.amazonaws.services.glue.types

IntegerNode classe de cas

IntegerNode

```
case class IntegerNode extends ScalarNode(value, TypeCode.INT)  (
           value : Int )
```

IntegerNode champs de val

- `ordering`

IntegerNode méthodes def

```
def equals( other : Any )
```

AWS GlueScala LongNode APIs

Package : com.amazonaws.services.glue.types

LongNode classe de cas

LongNode

```
case class LongNode extends ScalarNode(value, TypeCode.LONG)  (
           value : Long )
```

LongNode champs de val

- `ordering`

LongNode méthodes def

```
def equals( other : Any )
```

AWS GlueScala MapLikeNode APIs

Package : com.amazonaws.services.glue.types

MapLikeNode classe

MapLikeNode

```
class MapLikeNode extends DynamicNode  (
           value : mutable.Map[String, DynamicNode] )
```

MapLikeNode méthodes def

```
def clear : Unit
```

```
def get( name : String ) : Option[DynamicNode]
```

```
def getValue
```

```
def has( name : String ) : Boolean
```

```
def isEmpty : Boolean
```

```
def put( name : String,
         node : DynamicNode
       ) : Option[DynamicNode]
```

```
def remove( name : String ) : Option[DynamicNode]
```

```
def toIterator : Iterator[(String, DynamicNode)]
```

```
def toJson : String
```

```
def toJson( useQuotes : Boolean ) : String
```

Exemple : Soit le fichier JSON suivant :

```
{"foo": "bar"}
```

Si `useQuotes == true`, `toJson` génère `{"foo": "bar"}`. Si `useQuotes == false`, `toJson` génère `{foo: bar}` @return.

AWS GlueScala MapNode APIs

Package : com.amazonaws.services.glue.types

MapNode classe de cas

MapNode

```
case class MapNode extends MapLikeNode(value)  (
           value : mutable.Map[String, DynamicNode] )
```

MapNode méthodes def

```
def clone
```

```
def equals( other : Any )
```

```
def hashCode : Int
```

```
def nodeType
```

```
def this
```

AWS GlueScala NullNode APIs

Rubriques

- NullNode classe
- NullNode objet du boîtier

Package : com.amazonaws.services.glue.types

NullNode classe

NullNode

```
class NullNode
```

NullNode objet du boîtier

NullNode

```
case object NullNode extends NullNode
```

AWS GlueScala ObjectNode APIs

Rubriques

- ObjectNode objet
- ObjectNode classe de cas

Package : com.amazonaws.services.glue.types

ObjectNode objet

ObjectNode

```
object ObjectNode
```

ObjectNode méthodes def

```
def apply( frameKeys : Set[String],
           v1 : mutable.Map[String, DynamicNode],
           v2 : mutable.Map[String, DynamicNode],
           resolveWith : String
         ) : ObjectNode
```

ObjectNode classe de cas

ObjectNode

```
case class ObjectNode extends MapLikeNode(value)  (
           val value : mutable.Map[String, DynamicNode] )
```

ObjectNode méthodes def

```
def clone
```

```
def equals( other : Any )
```

```
def hashCode : Int
```

```
def nodeType
```

```
def this
```

AWS GlueScala ScalarNode APIs

Rubriques

- ScalarNode classe
- ScalarNode objet

Package : com.amazonaws.services.glue.types

ScalarNode classe

ScalarNode

```
class ScalarNode extends DynamicNode  (
           value : Any,
           scalarType : TypeCode )
```

ScalarNode méthodes def

```
def compare( other : Any,
             operator : String
           ) : Boolean
```

```
def getValue
```

```
def hashCode : Int
```

```
def nodeType
```

```
def toJson
```

ScalarNode objet

ScalarNode

```
object ScalarNode
```

ScalarNode méthodes def

```
def apply( v : Any ) : DynamicNode
```

```
def compare( tv : Ordered[T],
             other : T,
             operator : String
           ) : Boolean
```

```
def compareAny( v : Any,
                y : Any,
                o : String )
```

```
def withEscapedSpecialCharacters( jsonToEscape : String ) : String
```

AWS GlueScala ShortNode APIs

Package : com.amazonaws.services.glue.types

ShortNode classe de cas

ShortNode

```
case class ShortNode extends ScalarNode(value, TypeCode.SHORT)  (
           value : Short )
```

ShortNode champs de val

- `ordering`

ShortNode méthodes def

```
def equals( other : Any )
```

AWS GlueScala StringNode APIs

Package : com.amazonaws.services.glue.types

StringNode classe de cas

StringNode

```
case class StringNode extends ScalarNode(value, TypeCode.STRING)  (
           value : String )
```

StringNode champs de val

- `ordering`

StringNode méthodes def

```
def equals( other : Any )
```

```
def this( value : UTF8String )
```

AWS GlueScala TimestampNode APIs

Package : com.amazonaws.services.glue.types

TimestampNode classe de cas

TimestampNode

```
case class TimestampNode extends ScalarNode(value, TypeCode.TIMESTAMP)  (
           value : Timestamp )
```

TimestampNode champs de val

- `ordering`

TimestampNode méthodes def

```
def equals( other : Any )
```

```
def this( value : Long )
```

AWS GlueScala GlueArgParser APIs

Package : com.amazonaws.services.glue.util

GlueArgParser objet

GlueArgParser

```
object GlueArgParser
```

Strictement compatible avec la version Python de `utils.getResolvedOptions` dans le package `AWSGlueDataplanePython`.

GlueArgParser méthodes def

```
def getResolvedOptions( args : Array[String],
                        options : Array[String]
                      ) : Map[String, String]
```

```
def initParser( userOptionsSet : mutable.Set[String] ) : ArgumentParser
```

Example Extraction des arguments transmis à une tâche

Pour extraire les arguments d'une tâche, vous pouvez utiliser la méthode `getResolvedOptions`. Prenons l'exemple suivant, qui récupère un argument de tâche nommé `aws_region`.

```
val args = GlueArgParser.getResolvedOptions(sysArgs,
 Seq("JOB_NAME","aws_region").toArray)
Job.init(args("JOB_NAME"), glueContext, args.asJava)
val region = args("aws_region")
println(region)
```

AWS GlueJob Scala APIs

Package : com.amazonaws.services.glue.util

Objet Job

Job

```
object Job
```

Modes def Job

```
def commit
```

```
def init( jobName : String,
          glueContext : GlueContext,
          args : java.util.Map[String, String] = Map[String, String]().asJava
        ) : this.type
```

```
def init( jobName : String,
          glueContext : GlueContext,
          endpoint : String,
          args : java.util.Map[String, String]
        ) : this.type
```

```
def isInitialized
```

```
def reset
```

```
def runId
```

Fonctionnalités et optimisations de la programmation des scripts ETL AWS Glue pour Spark

Les sections suivantes décrivent les techniques et les valeurs qui s'appliquent généralement à la programmation ETL (extraction, transformation et chargement) AWS Glue pour Spark quel que soit le langage.

Rubriques

- Types et options de connexion pour ETL dans AWS Glue pour Spark
- Options de format pour les entrées et sorties dans AWS Glue pour Spark
- AWS Support de Glue Data Catalog pour les tâches Spark SQL
- Utilisation des marque-pages de tâche
- Utilisation de la détection des données sensibles en dehors de AWS Glue Studio
- API Visual Job AWS Glue

Types et options de connexion pour ETL dans AWS Glue pour Spark

Dans AWS Glue Spark, diverses méthodes PySpark et transformations Scala spécifient le type de connexion à l'aide d'un `connectionType` paramètre. Ils spécifient des options de connexion à l'aide d'un paramètre `connectionOptions` ou `options`.

Le paramètre `connectionType` peut prendre les valeurs indiquées dans le tableau suivant. Les valeurs de paramètre associées `connectionOptions` (ou `options`) pour chaque type sont

documentées dans les sections suivantes. Sauf indication contraire, les paramètres s'appliquent lorsque la connexion est utilisée comme source ou comme collecteur.

Pour obtenir un exemple de code qui illustre le paramétrage et l'utilisation des options de connexion, consultez la page d'accueil de chaque type de connexion.

connectionType	Se connecte à
dynamodb	Base de données Amazon DynamoDB
kinesis	Amazon Kinesis Data Streams
s3	Amazon S3
documentdb	Base de données Amazon DocumentDB (compatible avec MongoDB)
openSearch	Amazon OpenSearch Service.
redshift	Base de données Amazon Redshift
kafka	Kafka ou Amazon Managed Streaming for Apache Kafka
azurecosmos	Azure Cosmos pour NoSQL.
azuresql	Azure SQL.
bigquery	Google BigQuery.
mongodb	Base de données MongoDB, y compris MongoDB Atlas.
sqlserver	Base de données Microsoft SQL Server (consultez Connexions JDBC)
mysql	Base de données MySQL (consultez Connexions JDBC)
oracle	Base de données Oracle (consultez Connexions JDBC)
postgresql	Base de données PostgreSQL (consultez Connexions JDBC)
saphana	SAP HANA.
snowflake	Lac de données Snowflake

`connectionType`	Se connecte à
teradata	Teradata Vantage.
vertica	Vertica.
personnalisé.*	Magasins de données Spark, Athena ou JDBC (voir Valeurs personnalisées et AWS Marketplace ConnectionType
marketplace.*	Magasins de données Spark, Athena ou JDBC (voir Valeurs personnalisées et AWS Marketplace ConnectionType)

Connexions DynamoDB

Vous pouvez utiliser AWS Glue for Spark pour lire et écrire dans des tables de DynamoDB dans Glue. AWS Vous vous connectez à DynamoDB à l'aide des autorisations IAM associées à votre tâche Glue. AWS AWS Glueprend en charge l'écriture de données dans la table DynamoDB d'un autre AWS compte. Pour de plus amples informations, veuillez consulter the section called “Accès entre régions et entre comptes aux tables DynamoDB”.

Le connecteur DynamoDB d'origine utilise des objets DynamicFrame Glue pour travailler avec les données extraites de DynamoDB. AWS Glue 5.0+ introduit une nouveauté Connecteur DynamoDB avec prise en charge de Spark DataFrame qui fournit le support natif de Spark DataFrame .

Outre le connecteur AWS Glue DynamoDB ETL, vous pouvez lire depuis DynamoDB à l'aide du connecteur d'exportation DynamoDB, qui appelle une demande DynamoDB et la stocke dans un emplacement Amazon `ExportTableToPointInTime` S3 que vous fournissez, au format DynamoDB JSON. AWS Gluecrée ensuite un DynamicFrame objet en lisant les données depuis le site d'exportation Amazon S3.

Le dispositif d’écriture DynamoDB est disponible dans la version 1.0 ou les versions ultérieures de AWS Glue. Le dispositif de connecteur d’exportation DynamoDB AWS Glue est disponible dans la version 2.0 ou les versions ultérieures d’AWS Glue. Le nouveau connecteur DataFrame DynamoDB est disponible dans la version 5.0 ou les versions ultérieures de AWS Glue.

Pour plus d’informations sur DynamoDB, consultez la documentation Amazon DynamoDB.

> **Note**
>
> Le lecteur ETL DynamoDB ne prend pas en charge les filtres ou les prédicats pushdown.

Configuration de connexions DynamoDB

Pour vous connecter à DynamoDB AWS depuis Glue, accordez au rôle IAM associé à AWS votre tâche Glue l'autorisation d'interagir avec DynamoDB. Pour plus d'informations sur les autorisations requises pour lire ou écrire à partir de DynamoDB, consultez Actions, ressources et clés de condition pour DynamoDB dans la documentation IAM.

Dans les situations suivantes, vous autre peut-être besoin d'une configuration supplémentaire :

- Lorsque vous utilisez le connecteur d'exportation DynamoDB, vous devez configurer IAM pour que votre tâche puisse demander des exportations de tables DynamoDB. En outre, vous devrez identifier un compartiment Amazon S3 pour l'exportation et fournir les autorisations appropriées dans IAM pour que DynamoDB puisse y écrire, et pour que votre tâche AWS Glue puisse y lire. Pour plus d'informations, consultez Demande d'exportation de table dans DynamoDB.
- Si votre tâche AWS Glue comporte des exigences de connectivité Amazon VPC spécifiques, utilisez le type de connexion `NETWORK` AWS Glue pour fournir des options réseau. L'accès à DynamoDB étant autorisé par IAM, il n'est pas nécessaire de recourir à un type de connexion AWS Glue DynamoDB.

Lecture et écriture dans DynamoDB

Les exemples de code suivants montrent comment lire et écrire des tables DynamoDB. Ils montrent la lecture d'une table et l'écriture dans une autre table.

Python

```
import sys
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job
from awsglue.utils import getResolvedOptions

args = getResolvedOptions(sys.argv, ["JOB_NAME"])
glue_context= GlueContext(SparkContext.getOrCreate())
```

```
job = Job(glue_context)
job.init(args["JOB_NAME"], args)

dyf = glue_context.create_dynamic_frame.from_options(
    connection_type="dynamodb",
    connection_options={"dynamodb.input.tableName": test_source,
        "dynamodb.throughput.read.percent": "1.0",
        "dynamodb.splits": "100"
    }
)
print(dyf.getNumPartitions())

glue_context.write_dynamic_frame_from_options(
    frame=dyf,
    connection_type="dynamodb",
    connection_options={"dynamodb.output.tableName": test_sink,
        "dynamodb.throughput.write.percent": "1.0"
    }
)

job.commit()
```

Scala

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.DynamoDbDataSink
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._


object GlueApp {

  def main(sysArgs: Array[String]): Unit = {
    val glueContext = new GlueContext(SparkContext.getOrCreate())
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)

    val dynamicFrame = glueContext.getSourceWithFormat(
      connectionType = "dynamodb",
      options = JsonOptions(Map(
```

```
        "dynamodb.input.tableName" -> test_source,
        "dynamodb.throughput.read.percent" -> "1.0",
        "dynamodb.splits" -> "100"
      ))
    ).getDynamicFrame()

    print(dynamicFrame.getNumPartitions())

    val dynamoDbSink: DynamoDbDataSink =  glueContext.getSinkWithFormat(
      connectionType = "dynamodb",
      options = JsonOptions(Map(
        "dynamodb.output.tableName" -> test_sink,
        "dynamodb.throughput.write.percent" -> "1.0"
      ))
    ).asInstanceOf[DynamoDbDataSink]

    dynamoDbSink.writeDynamicFrame(dynamicFrame)

    Job.commit()
  }

}
```

Utilisation du connecteur d'exportation DynamoDB

Le connecteur d'exportation fonctionne mieux que le connecteur ETL lorsque la taille de la table DynamoDB est supérieure à 80 Go. En outre, étant donné que la demande d'exportation est effectuée en dehors des processus Spark dans une tâche AWS Glue, vous pouvez activer l'auto scaling des tâches AWS Glue pour enregistrer l'utilisation du DPU pendant la demande d'exportation. Avec le connecteur d'exportation, vous n'avez pas non plus besoin de configurer le nombre de fractionnements pour le parallélisme de l'exécuteur Spark ou le pourcentage de lecture du débit DynamoDB.

> **Note**
>
> DynamoDB a des exigences spécifiques pour recourir aux demandes `ExportTableToPointInTime`. Pour plus d'informations, consultez Demande d'exportation de table dans DynamoDB. Par exemple, Point-in-Time-Restore (PITR) doit être activé sur la table pour utiliser ce connecteur. Le connecteur DynamoDB prend également en charge le AWS KMS chiffrement pour les exportations DynamoDB vers Amazon S3. La saisie de

votre configuration de sécurité dans la configuration de la tâche AWS Glue active AWS KMS le chiffrement pour une exportation DynamoDB. La clé KMS doit être située dans la même région que le compartiment Simple Storage Service (Amazon S3).
Notez que des frais supplémentaires pour l'exportation DynamoDB et les coûts de stockage Simple Storage Service (Amazon S3) s'appliquent. Les données exportées dans Simple Storage Service (Amazon S3) sont conservées une fois l'exécution d'une tâche terminée, ce qui vous permet de les réutiliser sans exportations DynamoDB supplémentaires. L'une des conditions requises pour utiliser ce connecteur est que point-in-time la restauration (PITR) soit activée pour la table.
Le connecteur ETL DynamoDB ou le connecteur d'exportation ne prennent pas en charge les filtres ou les prédicats pushdown à appliquer à la source DynamoDB.

Les exemples de code suivants montrent comment lire (via le connecteur d'exportation) et comment imprimer le nombre de partitions.

Python

```
import sys
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job
from awsglue.utils import getResolvedOptions

args = getResolvedOptions(sys.argv, ["JOB_NAME"])
glue_context= GlueContext(SparkContext.getOrCreate())
job = Job(glue_context)
job.init(args["JOB_NAME"], args)

dyf = glue_context.create_dynamic_frame.from_options(
    connection_type="dynamodb",
    connection_options={
        "dynamodb.export": "ddb",
        "dynamodb.tableArn": test_source,
        "dynamodb.s3.bucket": bucket_name,
        "dynamodb.s3.prefix": bucket_prefix,
        "dynamodb.s3.bucketOwner": account_id_of_bucket,
    }
)
print(dyf.getNumPartitions())
```

```
job.commit()
```

Scala

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.DynamoDbDataSink
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._


object GlueApp {

  def main(sysArgs: Array[String]): Unit = {
    val glueContext = new GlueContext(SparkContext.getOrCreate())
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)

    val dynamicFrame = glueContext.getSourceWithFormat(
      connectionType = "dynamodb",
      options = JsonOptions(Map(
        "dynamodb.export" -> "ddb",
        "dynamodb.tableArn" -> test_source,
        "dynamodb.s3.bucket" -> bucket_name,
        "dynamodb.s3.prefix" -> bucket_prefix,
        "dynamodb.s3.bucketOwner" -> account_id_of_bucket,
      ))
    ).getDynamicFrame()

    print(dynamicFrame.getNumPartitions())

    Job.commit()
  }

}
```

Ces exemples montrent comment effectuer la lecture (via le connecteur d'exportation) et imprimer le nombre de partitions à partir d'une table AWS Glue Data Catalog comportant une dynamodb classification :

Python

```
import sys
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job
from awsglue.utils import getResolvedOptions

args = getResolvedOptions(sys.argv, ["JOB_NAME"])
glue_context= GlueContext(SparkContext.getOrCreate())
job = Job(glue_context)
job.init(args["JOB_NAME"], args)

dynamicFrame = glue_context.create_dynamic_frame.from_catalog(
        database=catalog_database,
        table_name=catalog_table_name,
        additional_options={
            "dynamodb.export": "ddb",
            "dynamodb.s3.bucket": s3_bucket,
            "dynamodb.s3.prefix": s3_bucket_prefix
        }
    )
print(dynamicFrame.getNumPartitions())

job.commit()
```

Scala

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.DynamoDbDataSink
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._


object GlueApp {

  def main(sysArgs: Array[String]): Unit = {
    val glueContext = new GlueContext(SparkContext.getOrCreate())
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)
```

```
    val dynamicFrame = glueContext.getCatalogSource(
        database = catalog_database,
        tableName = catalog_table_name,
        additionalOptions = JsonOptions(Map(
            "dynamodb.export" -> "ddb",
            "dynamodb.s3.bucket" -> s3_bucket,
            "dynamodb.s3.prefix" -> s3_bucket_prefix
        ))
    ).getDynamicFrame()
    print(dynamicFrame.getNumPartitions())
)
```

Simplification de l'utilisation de l'exportation DynamoDB au format JSON

Les exportations DynamoDB avec le connecteur d'exportation AWS Glue DynamoDB donnent lieu à des fichiers JSON de structures imbriquées spécifiques. Pour plus d'informations, consultez la section Objets de données. AWS Gluefournit une DynamicFrame transformation qui peut transformer ces structures en une easier-to-use forme destinée à des applications en aval.

La transformation peut être appelée de deux manières. Vous pouvez définir l'option de connexion `"dynamodb.simplifyDDBJson"` avec la valeur `"true"` lorsque vous appelez une méthode pour lire à partir de DynamoDB. Vous pouvez également appeler la transformation en tant que méthode disponible indépendamment dans la bibliothèque AWS Glue.

Tenez compte du schéma suivant généré par une exportation DynamoDB :

```
root
|-- Item: struct
|    |-- parentMap: struct
|    |    |-- M: struct
|    |    |    |-- childMap: struct
|    |    |    |    |-- M: struct
|    |    |    |    |    |-- appName: struct
|    |    |    |    |    |    |-- S: string
|    |    |    |    |    |-- packageName: struct
|    |    |    |    |    |    |-- S: string
|    |    |    |    |    |-- updatedAt: struct
|    |    |    |    |    |    |-- N: string
|    |-- strings: struct
|    |    |-- SS: array
```

```
|    |    |    |-- element: string
|    |-- numbers: struct
|    |    |-- NS: array
|    |    |    |-- element: string
|    |-- binaries: struct
|    |    |-- BS: array
|    |    |    |-- element: string
|    |-- isDDBJson: struct
|    |    |-- BOOL: boolean
|    |-- nullValue: struct
|    |    |-- NULL: boolean
```

La transformation `simplifyDDBJson` simplifie cette exportation ainsi :

```
root
|-- parentMap: struct
|    |-- childMap: struct
|    |    |-- appName: string
|    |    |-- packageName: string
|    |    |-- updatedAt: string
|-- strings: array
|    |-- element: string
|-- numbers: array
|    |-- element: string
|-- binaries: array
|    |-- element: string
|-- isDDBJson: boolean
|-- nullValue: null
```

> **Note**
>
> `simplifyDDBJson`est disponible dans AWS Glue 3.0 et versions ultérieures. La transformation `unnestDDBJson` est également disponible pour simplifier l'exportation DynamoDB au format JSON. Nous encourageons les utilisateurs à passer de `simplifyDDBJson` à `unnestDDBJson`.

Configuration du parallélisme des opérations de DynamoDB

Pour améliorer les performances, vous pouvez régler certains paramètres disponibles pour le connecteur DynamoDB. Lorsque vous ajustez les paramètres de parallélisme, votre objectif est de

maximiser l'utilisation des AWS Glue workers provisionnés. Ensuite, si vous avez besoin de plus de performances, nous vous recommandons d'étendre votre travail en augmentant le nombre de DPUs.

Vous pouvez modifier le parallélisme dans une opération de lecture de DynamoDB à l'aide du paramètre `dynamodb.splits` lorsque vous utilisez le connecteur ETL. Lors d'une tâche de lecture avec le connecteur d'exportation, vous n'avez pas besoin de configurer le nombre de fractionnements pour le parallélisme de l'exécuteur Spark. Vous pouvez modifier le parallélisme dans une opération d'écriture de DynamoDB avec `dynamodb.output.numParallelTasks`.

Lecture avec le connecteur ETL DynamoDB

Nous vous recommandons de calculer `dynamodb.splits` en fonction du nombre maximum d'employés défini dans la configuration de votre tâche et du calcul `numSlots` suivant. En cas de mise à l'échelle automatique, le nombre réel d'employés disponibles peut changer et passer en dessous de ce plafond. Pour plus d'informations sur la définition du nombre maximum d'employés, consultez Nombre d'employés (`NumberOfWorkers`) dans the section called "Configuration des propriétés de tâches Spark".

- `numExecutors = NumberOfWorkers - 1`

  À des fins de contexte, un exécuteur est réservé au pilote Spark ; d'autres exécuteurs sont utilisés pour traiter les données.
- `numSlotsPerExecutor =`

  AWS Glue 3.0 and later versions
  - 4 si `WorkerType` a la valeur `G.1X`
  - 8 si `WorkerType` a la valeur `G.2X`
  - 16 si `WorkerType` a la valeur `G.4X`
  - 32 si `WorkerType` a la valeur `G.8X`

  AWS Glue 2.0 and legacy versions
  - 8 si `WorkerType` a la valeur `G.1X`
  - 16 si `WorkerType` a la valeur `G.2X`
- `numSlots = numSlotsPerExecutor * numExecutors`

Nous vous recommandons de définir `dynamodb.splits` sur le nombre d'emplacements disponibles, `numSlots`.

Écriture dans DynamoDB

Le paramètre `dynamodb.output.numParallelTasks` est utilisé pour déterminer le WCU par tâche Spark, à l'aide du calcul suivant :

```
permittedWcuPerTask = ( TableWCU * dynamodb.throughput.write.percent ) / dynamodb.output.numParallelTasks
```

Le dispositif d'écriture DynamoDB fonctionne de manière optimale si la configuration représente avec précision le nombre de tâches Spark écrites dans DynamoDB. Dans certains cas, il se peut que vous ayez besoin de remplacer le calcul par défaut afin d'améliorer les performances d'écriture. Si vous ne spécifiez pas ce paramètre, le WCU autorisé pour chaque tâche Spark est automatiquement calculé par la formule suivante :

-
  - `numPartitions = dynamicframe.getNumPartitions()`
  - `numSlots` (tel que défini précédemment dans cette section)
  - `numParallelTasks = min(numPartitions, numSlots)`
- Exemple 1. DPU = 10, = Norme. WorkerType L'entrée DynamicFrame comporte 100 partitions RDD.
  - `numPartitions = 100`
  - `numExecutors = (10 - 1) * 2 - 1 = 17`
  - `numSlots = 4 * 17 = 68`
  - `numParallelTasks = min(100, 68) = 68`
- Exemple 2. DPU = 10, = Norme. WorkerType L'entrée DynamicFrame comporte 20 partitions RDD.
  - `numPartitions = 20`
  - `numExecutors = (10 - 1) * 2 - 1 = 17`
  - `numSlots = 4 * 17 = 68`
  - `numParallelTasks = min(20, 68) = 20`

> **Note**
>
> Les jobs sur les anciennes versions de AWS Glue et ceux utilisant des outils Standard Workers nécessitent des méthodes différentes pour calculer le nombre d'emplacements. Si vous avez besoin d'optimiser les performances de ces tâches, nous vous recommandons de passer aux versions compatibles de AWS Glue.

Référence des options de connexion DynamoDB

Désigne une connexion à Amazon DynamoDB.

Les options de connexion sont différentes pour une connexion source et une connexion collecteur.

« connectionType » : « dynamodb » avec le connecteur ETL comme source

Utilisez les options de connexion suivantes en `"connectionType": "dynamodb"` tant que source, lorsque vous utilisez le connecteur ETL AWS Glue DynamoDB :

- `"dynamodb.input.tableName"` : (obligatoire) table DynamoDB à lire.
- `"dynamodb.throughput.read.percent"` : (Facultatif) Le pourcentage d'unités de capacité de lecture à utiliser. La valeur définie par défaut est « 0,5 ». Valeurs acceptées : de 0,1 à 1,5 inclus.
  - `0.5` représente la vitesse de lecture par défaut, ce qui signifie que AWS Glue tente de consommer la moitié de la capacité de lecture de la table. Si vous augmentez la valeur au-delà de `0.5`, AWS Glue augmente le débit de demandes. Si vous réduisez la valeur en dessous de `0.5`, le débit de demandes de lecture baisse. (La vitesse de lecture réelle dépend de facteurs tels que l'existence d'une distribution uniforme des clés dans la table DynamoDB.)
  - Lorsque la table DynamoDB est en mode à la demande, AWS Glue gère la capacité de lecture de la table à 40 000. Pour exporter une table volumineuse, nous vous recommandons de basculer votre table DynamoDB en mode à la demande.
- `"dynamodb.splits"` : (facultatif) définit en combien de fractionnements nous partitionnons cette table DynamoDB lors de la lecture. La valeur définie par défaut est « 1 ». Valeurs acceptées : de 1 à 1 000 000 inclus.

  1 montre qu'il n'y a pas de parallélisme. Nous vous recommandons fortement de spécifier une valeur plus élevée pour de meilleures performances en utilisant la formule ci-dessous. Pour plus d'informations sur la définition appropriée d'une valeur, consultez the section called “Parallélisme de DynamoDB”.
- `"dynamodb.sts.roleArn"` : (facultatif) l'ARN du rôle IAM à endosser pour l'accès entre comptes. Ce paramètre est disponible dans AWS Glue 1.0 ou une version ultérieure.
- `"dynamodb.sts.roleSessionName"` : (facultatif) nom de séance STS. La valeur par défaut est définie sur « glue-dynamodb-read-sts -session ». Ce paramètre est disponible dans AWS Glue 1.0 ou une version ultérieure.

« connectionType » : « dynamodb » avec le connecteur d'exportation DynamoDB AWS Glue comme source

Utilisez les options de connexion suivantes avec « connectionType » : « dynamodb » en tant que source, lorsque vous utilisez le connecteur d'exportation DynamoDB AWS Glue, disponible uniquement pour AWS Glue version 2.0 et au-delà :

- `"dynamodb.export"` : (Obligatoire) Une valeur de chaîne :
  - Si la valeur est définie sur `ddb`, le connecteur d'exportation DynamoDB AWS Glue est activé où un nouveau paramètre `ExportTableToPointInTimeRequest` sera invoqué pendant la tâche AWS Glue. Une nouvelle exportation sera générée avec l'emplacement transmis depuis `dynamodb.s3.bucket` et `dynamodb.s3.prefix`.
  - Si la valeur est définie sur `s3`, le connecteur d'exportation DynamoDB AWS Glue est activé, mais ignore la création d'une nouvelle exportation DynamoDB et utilise plutôt les paramètres `dynamodb.s3.bucket` et `dynamodb.s3.prefix` en tant qu'emplacement Amazon S3 d'une exportation antérieure de cette table.
- `"dynamodb.tableArn"` : (obligatoire) table DynamoDB à lire.
- `"dynamodb.unnestDDBJson"` : (Facultatif) Par défaut : false. Valeurs valides : booléen. S'il est paramétré sur true, il effectue une transformation irréprochable de la structure JSON DynamoDB présente dans les exportations. Définir `"dynamodb.unnestDDBJson"` et `"dynamodb.simplifyDDBJson"` sur true en même temps constitue une erreur. Dans les AWS versions 3.0 et ultérieures, nous vous recommandons de l'utiliser `"dynamodb.simplifyDDBJson"` pour un meilleur comportement lors de la simplification des types de cartes DynamoDB. Pour de plus amples informations, veuillez consulter the section called “Simplification de l’utilisation de l’exportation DynamoDB au format JSON”.
- `"dynamodb.simplifyDDBJson"` : (Facultatif) Par défaut : false. Valeurs valides : booléen. S’il est paramétré sur true, il effectue une transformation pour simplifier le schéma de la structure JSON DynamoDB présente dans les exportations. Cela a le même objectif que l’option `"dynamodb.unnestDDBJson"`, mais fournit une meilleure prise en charge des types de cartes DynamoDB ou même des types de cartes imbriqués dans votre table DynamoDB. Cette option est disponible dans AWS Glue 3.0 et versions ultérieures. Définir `"dynamodb.unnestDDBJson"` et `"dynamodb.simplifyDDBJson"` sur true en même temps constitue une erreur. Pour de plus amples informations, veuillez consulter the section called “Simplification de l’utilisation de l’exportation DynamoDB au format JSON”.

- `"dynamodb.s3.bucket"` : (facultatif) Indique l'emplacement du compartiment Amazon S3 dans lequel le processus DynamoDB `ExportTableToPointInTime` doit être mené. Le format de fichier de l'exportation est DynamoDB JSON.
  - `"dynamodb.s3.prefix"` : (Facultatif) Indique l'emplacement du préfixe Amazon S3 dans le compartiment Amazon S3 dans lequel les charges `ExportTableToPointInTime` de DynamoDB doivent être stockées. Si `dynamodb.s3.prefix` ni `dynamodb.s3.bucket` ne sont indiqués, ces valeurs seront par défaut à l'emplacement du répertoire temporaire indiqué dans la configuration de la tâche AWS Glue. Pour plus d'informations, consultez la section Paramètres spéciaux utilisés par AWS Glue.
  - `"dynamodb.s3.bucketOwner"` : indique le propriétaire du compartiment nécessaire à l'accès Amazon S3 entre comptes.
- `"dynamodb.sts.roleArn"`: (Facultatif) L'ARN du rôle IAM à utiliser pour l'accès entre comptes L'accès and/or interrégional à la table DynamoDB. Remarque : Le même ARN de rôle IAM sera utilisé pour accéder à l'emplacement Amazon S3 spécifié pour la demande de `ExportTableToPointInTime`.
- `"dynamodb.sts.roleSessionName"` : (facultatif) nom de séance STS. La valeur par défaut est définie sur « glue-dynamodb-read-sts -session ».
- `"dynamodb.exportTime"` (Facultatif) Valeurs valides : chaînes représentant des instants ISO-8601. point-in-timeA où l'exportation doit être effectuée.
- `"dynamodb.sts.region"` : (Obligatoire si vous passez un appel entre régions à l'aide d'un point de terminaison régional) Région hébergeant la table DynamoDB que vous souhaitez lire.

« connectionType » : « dynamodb » avec le connecteur ETL comme récepteur

Utilisez les options de connexion suivantes avec `"connectionType": "dynamodb"` comme collecteur :

- `"dynamodb.output.tableName"` : (obligatoire) table DynamoDB où écrire.
- `"dynamodb.throughput.write.percent"` : (facultatif) pourcentage d'unités de capacité de lecture (WCU) à utiliser. La valeur définie par défaut est « 0,5 ». Valeurs acceptées : de 0,1 à 1,5 inclus.
  - `0.5` représente la vitesse d'écriture par défaut, ce qui signifie que AWS Glue tente de consommer la moitié de la capacité d'écriture de la table. Si vous augmentez la valeur au-delà de 0,5, AWS Glue augmente le débit de demandes. Si vous réduisez la valeur en dessous de

0,5, le débit de demandes d'écriture baisse. (La vitesse d'écriture réelle dépend de facteurs tels que l'existence d'une distribution uniforme des clés dans la table DynamoDB.)

- Lorsque la table DynamoDB est en mode à la demande, AWS Glue gère la capacité d'écriture de la table à `40000`. Pour importer une table volumineuse, nous vous recommandons de basculer votre table DynamoDB en mode à la demande.

- `"dynamodb.output.numParallelTasks"` : (facultatif) définit le nombre de tâches parallèles qui écrivent dans DynamoDB simultanément. Utilisé pour calculer le WCU permissif par tâche Spark. Dans la plupart des cas, AWS Glue calcule une valeur par défaut raisonnable pour cette valeur. Pour de plus amples informations, veuillez consulter the section called "Parallélisme de DynamoDB".
- `"dynamodb.output.retry"` : (facultatif) définit le nombre de tentatives que nous effectuons lorsqu'il y a un `ProvisionedThroughputExceededException` de DynamoDB. La valeur définie par défaut est « 10 ».
- `"dynamodb.sts.roleArn"` : (facultatif) l'ARN du rôle IAM à endosser pour l'accès entre comptes.
- `"dynamodb.sts.roleSessionName"` : (facultatif) nom de séance STS. La valeur par défaut est définie sur « glue-dynamodb-write-sts -session ».

Accès entre régions et entre comptes aux tables DynamoDB

AWS Glue Les tâches ETL prennent en charge l'accès entre régions et entre comptes aux tables DynamoDB. AWS Glue Les tâches ETL permettent à la fois de lire les données de la table DynamoDB d'un autre AWS compte et d'écrire des données dans la table DynamoDB d'un autre AWS compte. AWS Gluepermet également de lire à partir d'une table DynamoDB d'une autre région et d'écrire dans une table DynamoDB d'une autre région. Cette section donne des instructions sur la configuration de l'accès et fournit un exemple de script.

Les procédures de cette section font référence à un didacticiel IAM pour créer un rôle IAM et accorder l'accès au rôle. Le didacticiel traite également de la prise en charge d'un rôle, mais ici, vous utiliserez plutôt un script de tâche pour endosser le rôle dans AWS Glue. Ce didacticiel contient également des informations sur les pratiques générales entre comptes. Pour plus d'informations, voir Tutoriel : Déléguer l'accès entre AWS comptes à l'aide de rôles IAM dans le Guide de l'utilisateur IAM.

Créer un rôle

Suivez l'étape 1 du didacticiel pour créer un rôle IAM dans le compte A. Lorsque vous définissez les autorisations du rôle, vous pouvez choisir d'associer des politiques existantes telles que `AmazonDynamoDBReadOnlyAccess` ou d'autoriser le rôle `AmazonDynamoDBFullAccess` à read/write DynamoDB. L'exemple suivant illustre la création d'un rôle nommé `DynamoDBCrossAccessRole`, avec la politique d'autorisation `AmazonDynamoDBFullAccess`.

Accorder l'accès au rôle

Suivez l'étape 2 du didacticiel du Guide de l'utilisateur IAM pour permettre au compte B de basculer vers le rôle qui vient d'être créé. L'exemple suivant crée une politique avec l'instruction suivante :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::111122223333:role/DynamoDBCrossAccessRole"
  }
}
```

Vous pouvez ensuite associer cette politique à celle group/role/user que vous souhaitez utiliser pour accéder à DynamoDB.

Endosser le rôle dans le script de tâche AWS Glue

Maintenant, vous pouvez vous connecter au compte B et créer une tâche AWS Glue. Pour créer une tâche, reportez-vous aux instructions de la page Configuration des propriétés des tâches pour les tâches Spark dans AWS Glue.

Dans le script de tâche, vous devez utiliser le paramètre `dynamodb.sts.roleArn` pour endosser le rôle `DynamoDBCrossAccessRole`. En supposant que ce rôle vous permet d'obtenir les informations d'identification temporaires, qui doivent être utilisées pour accéder à DynamoDB dans le compte B. Consultez ces exemples de scripts.

Pour une lecture entre comptes dans toutes les régions (connecteur ETL) :

```
import sys
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job
from awsglue.utils import getResolvedOptions

args = getResolvedOptions(sys.argv, ["JOB_NAME"])
glue_context= GlueContext(SparkContext.getOrCreate())
job = Job(glue_context)
job.init(args["JOB_NAME"], args)

dyf = glue_context.create_dynamic_frame_from_options(
    connection_type="dynamodb",
    connection_options={
    "dynamodb.region": "us-east-1",
    "dynamodb.input.tableName": "test_source",
    "dynamodb.sts.roleArn": "<DynamoDBCrossAccessRole's ARN>"
    }
)
dyf.show()
job.commit()
```

Pour une lecture entre comptes dans toutes les régions (connecteur ETL) :

```
import sys
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job
from awsglue.utils import getResolvedOptions

args = getResolvedOptions(sys.argv, ["JOB_NAME"])
glue_context= GlueContext(SparkContext.getOrCreate())
job = Job(glue_context)
job.init(args["JOB_NAME"], args)

dyf = glue_context.create_dynamic_frame_from_options(
    connection_type="dynamodb",
    connection_options={
        "dynamodb.export": "ddb",
        "dynamodb.tableArn": "<test_source ARN>",
        "dynamodb.sts.roleArn": "<DynamoDBCrossAccessRole's ARN>"
    }
)
```

```
dyf.show()
job.commit()
```

Pour une lecture et une écriture entre comptes dans toutes les régions :

```
import sys
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job
from awsglue.utils import getResolvedOptions

args = getResolvedOptions(sys.argv, ["JOB_NAME"])
glue_context= GlueContext(SparkContext.getOrCreate())
job = Job(glue_context)
job.init(args["JOB_NAME"], args)

dyf = glue_context.create_dynamic_frame_from_options(
    connection_type="dynamodb",
    connection_options={
        "dynamodb.region": "us-east-1",
        "dynamodb.input.tableName": "test_source"
    }
)
dyf.show()

glue_context.write_dynamic_frame_from_options(
    frame=dyf,
    connection_type="dynamodb",
    connection_options={
        "dynamodb.region": "us-west-2",
        "dynamodb.output.tableName": "test_sink",
        "dynamodb.sts.roleArn": "<DynamoDBCrossAccessRole's ARN>"
    }
)

job.commit()
```

Connecteur DynamoDB avec prise en charge de Spark DataFrame

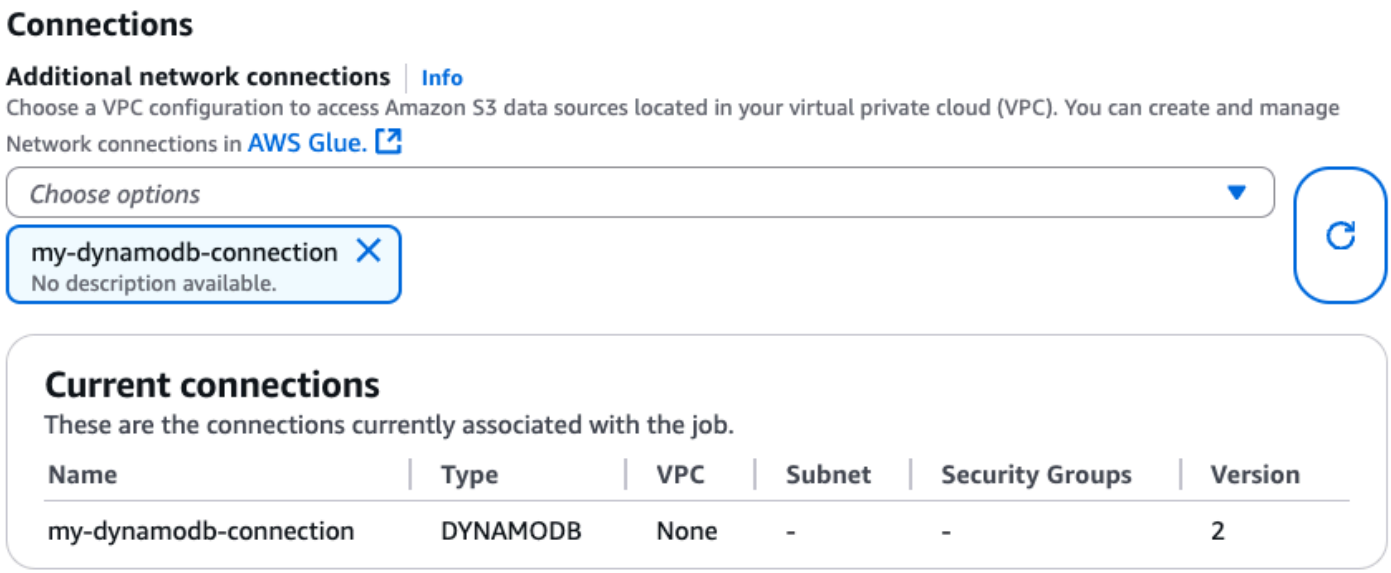
Le connecteur DynamoDB compatible avec DataFrame Spark vous permet de lire et d'écrire dans des tables de DynamoDB à l'aide de Spark. DataFrame APIs Les étapes de configuration du connecteur sont les mêmes que pour le connecteur DynamicFrame basé et peuvent être consultées ici.

Pour charger dans la bibliothèque de connecteurs DataFrame basée, veillez à associer une connexion DynamoDB à la tâche Glue.

> **Note**
>
> L'interface utilisateur de la console Glue ne prend actuellement pas en charge la création d'une connexion DynamoDB. Vous pouvez utiliser Glue CLI (CreateConnection) pour créer une connexion DynamoDB :

```
aws glue create-connection \
    --connection-input '{ \
        "Name": "my-dynamodb-connection", \
        "ConnectionType": "DYNAMODB", \
        "ConnectionProperties": {}, \
        "ValidateCredentials": false, \
        "ValidateForComputeEnvironments": ["SPARK"] \
    }'
```

Lors de la création de la connexion DynamoDB, vous pouvez l'associer à votre tâche Glue via CLI CreateJob(UpdateJob,) ou directement sur la page « Détails de la tâche » :



Après avoir vérifié qu'une connexion avec DYNAMODB Type est attachée à votre tâche Glue, vous pouvez utiliser les opérations de lecture, d'écriture et d'exportation suivantes à partir du connecteur DataFrame basé.

Lecture et écriture dans DynamoDB à l'aide du connecteur basé DataFrame

Les exemples de code suivants montrent comment lire et écrire dans des tables DynamoDB via DataFrame le connecteur basé. Ils montrent la lecture d'une table et l'écriture dans une autre table.

Python

```
import sys
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job
from awsglue.utils import getResolvedOptions

args = getResolvedOptions(sys.argv, ["JOB_NAME"])
glue_context= GlueContext(SparkContext.getOrCreate())
spark = glueContext.spark_session
job = Job(glue_context)
job.init(args["JOB_NAME"], args)

# Read from DynamoDB
df = spark.read.format("dynamodb") \
    .option("dynamodb.input.tableName", "test-source") \
    .option("dynamodb.throughput.read.ratio", "0.5") \
    .option("dynamodb.consistentRead", "false") \
    .load()

print(df.rdd.getNumPartitions())

# Write to DynamoDB
df.write \
  .format("dynamodb") \
  .option("dynamodb.output.tableName", "test-sink") \
  .option("dynamodb.throughput.write.ratio", "0.5") \
  .option("dynamodb.item.size.check.enabled", "true") \
  .save()

job.commit()
```

Scala

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {

    val glueContext = new GlueContext(SparkContext.getOrCreate())
    val spark = glueContext.getSparkSession
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)

    val df = spark.read
      .format("dynamodb")
      .option("dynamodb.input.tableName", "test-source")
      .option("dynamodb.throughput.read.ratio", "0.5")
      .option("dynamodb.consistentRead", "false")
      .load()

    print(df.rdd.getNumPartitions)

    df.write
      .format("dynamodb")
      .option("dynamodb.output.tableName", "test-sink")
      .option("dynamodb.throughput.write.ratio", "0.5")
      .option("dynamodb.item.size.check.enabled", "true")
      .save()

    job.commit()
  }
}
```

Utilisation de l'exportation DynamoDB via le connecteur basé DataFrame

L'opération d'exportation est préférable à l'opération de lecture pour les tables DynamoDB de plus de 80 Go. Les exemples de code suivants montrent comment lire depuis une table, exporter vers S3 et imprimer le nombre de partitions via le connecteur DataFrame basé.

> **Note**
>
> La fonctionnalité d'exportation DynamoDB est disponible via l'objet Scala. `DynamoDBExport` Les utilisateurs de Python peuvent y accéder via l'interopérabilité JVM de Spark ou utiliser le SDK AWS pour Python (boto3) avec l'API DynamoDB. `ExportTableToPointInTime`

Scala

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.util.{GlueArgParser, Job}
import org.apache.spark.SparkContext
import glue.spark.dynamodb.DynamoDBExport
import scala.collection.JavaConverters._

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val glueContext = new GlueContext(SparkContext.getOrCreate())
    val spark = glueContext.getSparkSession
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)

    val options = Map(
      "dynamodb.export" -> "ddb",
      "dynamodb.tableArn" -> "arn:aws:dynamodb:us-east-1:123456789012:table/my-
table",
      "dynamodb.s3.bucket" -> "my-s3-bucket",
      "dynamodb.s3.prefix" -> "my-s3-prefix",
      "dynamodb.simplifyDDBJson" -> "true"
    )
    val df = DynamoDBExport.fullExport(spark, options)

    print(df.rdd.getNumPartitions)
    df.count()

    Job.commit()
  }
}
```

Options de configuration

Options de lecture

Option	Description	Par défaut
`dynamodb.input.tableName`	Nom de la table DynamoDB (obligatoire)	-
`dynamodb.throughput.read`	Les unités de capacité de lecture (RCU) à utiliser. S'il n'est pas spécifié, `dynamodb.throughput.read.ratio` il est utilisé pour le calcul.	-
`dynamodb.throughput.read.ratio`	Le ratio d'unités de capacité de lecture (RCU) à utiliser	0.5
`dynamodb.table.read.capacity`	Capacité de lecture de la table à la demande utilisée pour calculer le débit. Ce paramètre n'est efficace que dans les tables de capacité à la demande. Par défaut, ce sont les unités de lecture à débit chaud.	-
`dynamodb.splits`	Définit le nombre de segments utilisés dans les opérations de scan en parallèle. S'il n'est pas fourni, le connecteur calculera une valeur par défaut raisonnable.	-
`dynamodb.consistentRead`	S'il faut utiliser des lectures très cohérentes	FALSE
`dynamodb.input.retry`	Définit le nombre de tentatives que nous effectuons en cas d'exception réessayable.	10

Options d'écriture

Option	Description	Par défaut
`dynamodb.output.tableName`	Nom de la table DynamoDB (obligatoire)	-
`dynamodb.throughput.write`	Les unités de capacité d'écriture (WCU) à utiliser. S'il n'est pas spécifié, `dynamodb.throughput.write.ratio` il est utilisé pour le calcul.	-
`dynamodb.throughput.write.ratio`	Le ratio d'unités de capacité d'écriture (WCU) à utiliser	0.5
`dynamodb.table.write.capacity`	Capacité d'écriture de la table à la demande utilisée pour calculer le débit. Ce paramètre n'est efficace que dans les tables de capacité à la demande. Par défaut, ce sont les unités d'écriture à débit chaud.	-
`dynamodb.item.size.check.enabled`	Si c'est vrai, le connecteur calcule la taille de l'élément et abandonne si la taille dépasse la taille maximale, avant d'écrire dans la table DynamoDB.	TRUE
`dynamodb.output.retry`	Définit le nombre de tentatives que nous effectuons en cas d'exception réessayable.	10

Options d'exportation

Option	Description	Par défaut
`dynamodb.export`	S'il est défini sur, `ddb` active le connecteur d'exportation AWS Glue DynamoDB dans lequel un `ExportTableToPointInTimeRequet` nouveau connecteur sera invoqué pendant la AWS tâche Glue.	`ddb`

Option	Description	Par défaut
	Une nouvelle exportation sera générée avec l'emplacement transmis depuis `dynamodb.s3.bucket` et `dynamodb.s3.prefix`. S'il est défini sur, `s3` active le connecteur d'exportation AWS Glue DynamoDB mais ignore la création d'une nouvelle exportation DynamoDB et utilise à la place le et `dynamodb.s3.bucket` comme emplacement Amazon S3 de `dynamodb.s3.prefix` l'ancienne exportation de cette table.	
`dynamodb.tableArn`	La table DynamoDB à partir de laquelle lire. Obligatoire si `dynamodb.export` est défini sur `ddb`.	
`dynamodb.simplifyDDBJson`	S'il est défini sur`true`, exécute une transformation pour simplifier le schéma de la structure JSON DynamoDB présente dans les exportations.	FALSE
`dynamodb.s3.bucket`	Le compartiment S3 pour stocker les données temporaires lors de l'exportation DynamoDB (obligatoire)	
`dynamodb.s3.prefix`	Le préfixe S3 pour stocker les données temporaires lors de l'exportation DynamoDB	
`dynamodb.s3.bucketOwner`	Indiquez le propriétaire du compartiment requis pour l'accès entre comptes Amazon S3	
`dynamodb.s3.sse.algorithm`	Type de chiffrement utilisé sur le compartiment dans lequel les données temporaires seront stockées. Les valeurs valides sont `AES256` et `KMS`.	

Option	Description	Par défaut
`dynamodb.s3.sse.kmsKeyId`	ID de la clé AWS KMS gérée utilisée pour chiffrer le compartiment S3 dans lequel les données temporaires seront stockées (le cas échéant).	
`dynamodb.exportTime`	point-in-timeA où l'exportation doit être effectuée. Valeurs valides : chaînes représentant des instants ISO-8601.	

Options générales

Option	Description	Par défaut
`dynamodb.sts.roleArn`	L'ARN du rôle IAM à assumer pour l'accès entre comptes	-
`dynamodb.sts.roleSessionName`	Nom de session STS	`glue-dynamodb-sts-session`
`dynamodb.sts.region`	Région pour le client STS (pour l'hypothèse d'un rôle interrégional)	Identique à l'`region`option

Connexions Kinesis

Vous pouvez utiliser une connexion Kinesis pour lire et écrire dans Amazon Kinesis Data Streams à l'aide d'informations stockées dans une table de catalogue de données ou en fournissant des informations permettant d'accéder directement au flux de données. Vous pouvez lire les informations de Kinesis dans un Spark DataFrame, puis les convertir en Glue AWS . DynamicFrame Vous pouvez DynamicFrames écrire dans Kinesis au format JSON. Si vous accédez directement au flux de données, utilisez ces options pour fournir des informations sur la façon d'accéder au flux de données.

Si vous utilisez `getCatalogSource` ou `create_data_frame_from_catalog` pour consommer des enregistrements à partir d'une source de streaming Kinesis, la tâche dispose des informations relatives à la base de données Data Catalog et au nom de la table, et peut les utiliser pour

obtenir certains paramètres de base pour la lecture à partir de la source de streaming Kinesis. Si vous utilisez `getSource`, `getSourceWithFormat`, `createDataFrameFromOptions` ou `create_data_frame_from_options`, vous devez spécifier ces paramètres de base à l'aide des options de connexion décrites ici.

Vous pouvez spécifier les options de connexion pour Kinesis à l'aide des arguments suivants pour les méthodes spécifiées dans la classe `GlueContext`.

- Scala
  - `connectionOptions` : utiliser avec `getSource`, `createDataFrameFromOptions`, `getSink`
  - `additionalOptions` : utiliser avec `getCatalogSource`, `getCatalogSink`
  - `options` : utiliser avec `getSourceWithFormat`, `getSinkWithFormat`
- Python
  - `connection_options` : utiliser avec `create_data_frame_from_options`, `write_dynamic_frame_from_options`
  - `additional_options` : utiliser avec `create_data_frame_from_catalog`, `write_dynamic_frame_from_catalog`
  - `options` : utiliser avec `getSource`, `getSink`

Pour les remarques et les restrictions concernant les tâches ETL de streaming, consultez the section called “Restrictions et notes sur ETL en streaming”.

Configurer Kinesis

Pour vous connecter à un flux de données Kinesis dans le cadre d'une tâche AWS Glue Spark, vous aurez besoin de certaines conditions préalables :

- En cas de lecture, la tâche AWS Glue doit disposer d'autorisations IAM de niveau d'accès en lecture pour le flux de données Kinesis.
- En cas d'écriture, la tâche AWS Glue doit disposer d'autorisations IAM de niveau d'accès Write pour le flux de données Kinesis.

Dans certains cas, vous devrez configurer des prérequis supplémentaires :

- Si votre tâche AWS Glue est configurée avec des connexions réseau supplémentaires (généralement pour se connecter à d'autres ensembles de données) et que l'une de ces

connexions fournit des options de réseau Amazon VPC, cela indiquera à votre tâche de communiquer via Amazon VPC. Dans ce cas, vous devrez également configurer votre flux de données Kinesis pour qu'il communique via Amazon VPC. Vous pouvez le faire en créant un point de terminaison d'un VPC d'interface entre votre Amazon VPC et votre flux de données Kinesis. Pour plus d'informations, consultez Using Kinesis Data Streams with Interface VPC Endpoints.

- Lorsque vous spécifiez Amazon Kinesis Data Streams dans un autre compte, vous devez configurer les rôles et les stratégies pour autoriser l'accès intercompte. Pour de plus amples informations, veuillez consulter la rubrique Exemple : Lire à partir d'un flux Kinesis dans un autre compte.

Pour plus d'informations sur les prérequis de la tâche ETL de streaming, consultez the section called "Tâches ETL en streaming".

Exemple : lecture à partir de flux Kinesis

Exemple : lecture à partir de flux Kinesis

Utilisez conjointement avec the section called "forEachBatch".

Exemple pour la source de streaming Amazon Kinesis :

```
kinesis_options =
   { "streamARN": "arn:aws:kinesis:us-east-2:777788889999:stream/fromOptionsStream",
     "startingPosition": "TRIM_HORIZON",
     "inferSchema": "true",
     "classification": "json"
   }
data_frame_datasource0 =
 glueContext.create_data_frame.from_options(connection_type="kinesis",
 connection_options=kinesis_options)
```

Exemple : écriture dans Kinesis Streams

Exemple : lecture à partir de flux Kinesis

Utilisez conjointement avec the section called "forEachBatch".

Exemple pour la source de streaming Amazon Kinesis :

```
kinesis_options =
   { "streamARN": "arn:aws:kinesis:us-east-2:777788889999:stream/fromOptionsStream",
```

```
    "startingPosition": "TRIM_HORIZON",
    "inferSchema": "true",
    "classification": "json"
  }
data_frame_datasource0 =
 glueContext.create_data_frame.from_options(connection_type="kinesis",
 connection_options=kinesis_options)
```

Référence des options de connexion Kinesis

Désigne des options de connexion à Amazon Kinesis Data Streams.

Utilisez les options de connexion suivantes pour les sources de données en streaming Kinesis :

- `"streamARN"` (Obligatoire) Utilisé pour la lecture/l'écriture. ARN du flux de données Kinesis.
- `"classification"` (Obligatoire pour la lecture) Utilisé pour la lecture. Format de fichier utilisé par les données de l'enregistrement. Obligatoire, sauf s'il est fourni par le catalogue de données.
- `"streamName"` : (facultatif) utilisé pour la lecture. Nom du flux de données Kinesis à partir duquel lire. Utilisé avec `endpointUrl`.
- `"endpointUrl"` : (facultatif) utilisé pour la lecture. Par défaut : « https://kinesis.us-east-1.amazonaws.com ». AWS Point de terminaison du flux Kinesis. Vous n'avez pas besoin de modifier ce paramètre, sauf si vous vous connectez à une région spéciale.
- `"partitionKey"` : (facultatif) utilisé pour l'écriture. Clé de partition Kinesis utilisée lors de la production d'enregistrements.
- `"delimiter"` : (facultatif) utilisé pour la lecture. Séparateur de valeurs utilisé lorsque `classification` est CSV. La valeur par défaut est « `,` ».
- `"startingPosition"` : (facultatif) utilisé pour la lecture. La position de départ dans le flux de données Kinesis à partir duquel lire les données. Les valeurs possibles sont `"latest"`, `"trim_horizon"`, `"earliest"`, ou une chaîne d'horodatage au format UTC dans le modèle `yyyy-mm-ddTHH:MM:SSZ` (où `Z` représente un décalage de fuseau horaire UTC avec un +/-. Par exemple : « 2023-04-04T08:00:00-04:00 »). La valeur par défaut est `"latest"`. Remarque : la chaîne d'horodatage au format UTC pour n'`"startingPosition"`est prise en charge que pour la version 4.0 ou ultérieure de AWS Glue.
- `"failOnDataLoss"` : (facultatif) échec de la tâche si une partition active est manquante ou a expiré. La valeur par défaut est `"false"`.
- `"awsSTSRoleARN"` : (facultatif) utilisé pour la lecture/l'écriture. Le nom de ressource Amazon (ARN) du rôle à assumer en utilisant AWS Security Token Service (AWS STS). Ce rôle doit

disposer des autorisations nécessaires pour décrire ou lire des registres pour le flux de données Kinesis. Vous devez utiliser ce paramètre lorsque vous accédez à un flux de données dans un autre compte. Utilisez conjointement avec `"awsSTSSessionName"`.

- `"awsSTSSessionName"` : (facultatif) utilisé pour la lecture/l'écriture. Un identifiant de la séance assumant le rôle à l'aide d' AWS STS. Vous devez utiliser ce paramètre lorsque vous accédez à un flux de données dans un autre compte. Utilisez conjointement avec `"awsSTSRoleARN"`.
- `"awsSTSEndpoint"`: (Facultatif) Le AWS STS point de terminaison à utiliser lors de la connexion à Kinesis avec un rôle assumé. Cela permet d'utiliser le point de AWS STS terminaison régional dans un VPC, ce qui n'est pas possible avec le point de terminaison global par défaut.
- `"maxFetchTimeInMs"` : (facultatif) utilisé pour la lecture. Le temps maximal passé dans l'exécuteur de tâches pour lire des enregistrements du lot actuel à partir du flux de données Kinesis, exprimé en millisecondes (ms). Plusieurs appels d'API `GetRecords` peuvent être effectués pendant cette période. La valeur par défaut est `1000`.
- `"maxFetchRecordsPerShard"` : (facultatif) utilisé pour la lecture. Nombre maximal d'enregistrements à récupérer par partition dans le flux de données Kinesis par microlot. Remarque : Le client peut dépasser cette limite si la tâche de streaming a déjà lu des enregistrements supplémentaires provenant de Kinesis (dans le même appel get-records). Si `maxFetchRecordsPerShard` doit être strict, il doit être un multiple de `maxRecordPerRead`. La valeur par défaut est `100000`.
- `"maxRecordPerRead"` : (facultatif) utilisé pour la lecture. Nombre maximal d'enregistrements à extraire du flux de données Kinesis dans chaque opération `getRecords`. La valeur par défaut est `10000`.
- `"addIdleTimeBetweenReads"` : (facultatif) utilisé pour la lecture. Ajoute un délai entre deux opérations `getRecords` consécutives. La valeur par défaut est `"False"`. Cette option n'est configurable que pour Glue version 2.0 et ultérieure.
- `"idleTimeBetweenReadsInMs"` : (facultatif) utilisé pour la lecture. Délai minimum entre deux opérations `getRecords`, en ms. La valeur par défaut est `1000`. Cette option n'est configurable que pour Glue version 2.0 et ultérieure.
- `"describeShardInterval"` : (facultatif) utilisé pour la lecture. Intervalle de temps minimum entre deux appels d'API `ListShards` pour que votre script envisage le repartitionnement. Pour plus d'informations, consultez Politiques de repartitionnement dans le Guide du développeur Amazon Kinesis Data Streams. La valeur par défaut est `1s`.
- `"numRetries"` : (facultatif) utilisé pour la lecture. Le nombre maximal de nouvelles tentatives pour les demandes d'API Kinesis Data Streams. La valeur par défaut est `3`.

- `"retryIntervalMs"` : (facultatif) utilisé pour la lecture. Le délai de réflexion (spécifié en ms) avant de réessayer l'appel d'API Kinesis Data Streams. La valeur par défaut est `1000`.
- `"maxRetryIntervalMs"` : (facultatif) utilisé pour la lecture. Le délai d'attente maximal (spécifié en ms) entre deux tentatives d'appel d'API Kinesis Data Streams. La valeur par défaut est `10000`.
- `"avoidEmptyBatches"` : (facultatif) utilisé pour la lecture. Évite de créer une tâche de micro-lot vide en vérifiant les données non lues dans le flux de données Kinesis avant le démarrage du lot. La valeur par défaut est `"False"`.
- `"schema"` : (obligatoire lorsque inferSchema est défini sur false) utilisé pour la lecture. Schéma à utiliser pour traiter la charge utile. Si la classification est `avro`, le schéma fourni doit être au format de schéma Avro. Si la classification n'est pas `avro`, le schéma fourni doit être au format de schéma DDL.

  Voici quelques exemples de schémas.

  Example in DDL schema format

  ```
  `column1` INT, `column2` STRING , `column3` FLOAT
  ```

  Example in Avro schema format

  ```
  {
    "type":"array",
    "items":
    {
      "type":"record",
      "name":"test",
      "fields":
      [
        {
          "name":"_id",
          "type":"string"
        },
        {
          "name":"index",
          "type":
          [
            "int",
            "string",
            "float"
          ]
        }
  ```

```
    ]
  }
}
```

- `"inferSchema"` : (facultatif) utilisé pour la lecture. La valeur par défaut est « false ». S'il est défini sur « true », le schéma sera détecté lors de l'exécution à partir de la charge utile dans `foreachbatch`.
- `"avroSchema"` : (obsolète) utilisé pour la lecture. Paramètre utilisé pour spécifier un schéma de données Avro lorsque le format Avro est utilisé. Ce paramètre est désormais obsolète. Utilisez le paramètre `schema`.
- `"addRecordTimestamp"` : (facultatif) utilisé pour la lecture. Lorsque cette option est définie sur « true », la sortie de données contient une colonne supplémentaire nommée « __src_timestamp » qui indique l'heure à laquelle l'enregistrement correspondant est reçu par le flux. La valeur par défaut est « false ». Cette option est prise en charge dans AWS Glue version 4.0 ou ultérieure.
- `"emitConsumerLagMetrics"` : (facultatif) utilisé pour la lecture. Lorsque l'option est définie sur « vrai », pour chaque lot, elle émet les métriques correspondant à la durée comprise entre le plus ancien enregistrement reçu par le flux et l'heure AWS Glue à laquelle il arrive CloudWatch. Le nom de la métrique est « glue.driver.streaming ». maxConsumerLagInMs». La valeur par défaut est « false ». Cette option est prise en charge dans AWS Glue version 4.0 ou ultérieure.
- `"fanoutConsumerARN"` : (facultatif) utilisé pour la lecture. ARN d'un consommateur de flux Kinesis pour le flux spécifié dans `streamARN`. Utilisé pour activer le mode diffusion améliorée pour votre connexion Kinesis. Pour plus d'informations sur la consommation d'un flux Kinesis avec diffusion améliorée, consultez the section called “Utilisation de la diffusion améliorée dans les tâches de streaming Kinesis”.
- `"recordMaxBufferedTime"` : (facultatif) utilisé pour l'écriture. Par défaut : 1 000 (ms). Durée maximale pendant laquelle un enregistrement est mis en mémoire tampon en attendant d'être écrit.
- `"aggregationEnabled"` : (facultatif) utilisé pour l'écriture. Valeur par défaut : vraie. Spécifie si les enregistrements doivent être agrégés avant de les envoyer à Kinesis.
- `"aggregationMaxSize"` : (facultatif) utilisé pour l'écriture. Par défaut : 51 200 (octets). Si un enregistrement est supérieur à cette limite, il contourne l'agrégateur. Remarque Kinesis impose une limite de 50 Ko à la taille des enregistrements. Si vous définissez ce paramètre au-delà de 50 Ko, les enregistrements surdimensionnés seront rejetés par Kinesis.
- `"aggregationMaxCount"` : (facultatif) utilisé pour l'écriture. Par défaut : 4294967295. Nombre maximum de résultats à regrouper dans un enregistrement agrégé.

- `"producerRateLimit"` : (facultatif) utilisé pour l'écriture. Par défaut : 150 (%). Limite le débit par partition envoyée par un seul producteur (votre tâche, par exemple), sous forme de pourcentage de la limite du backend.
- `"collectionMaxCount"` : (facultatif) utilisé pour l'écriture. Par défaut : 500. Nombre maximum d'articles à inclure dans une PutRecords demande.
- `"collectionMaxSize"` : (facultatif) utilisé pour l'écriture. Par défaut : 5242880 (octets). Quantité maximale de données à envoyer avec une PutRecords demande.

Utilisation de la diffusion améliorée dans les tâches de streaming Kinesis

Un client bénéficiant de la diffusion améliorée est capable de recevoir des enregistrements provenant d'un flux Kinesis avec un débit dédié qui peut être supérieur à celui des consommateurs classiques. Cela se fait en optimisant le protocole de transfert utilisé pour fournir des données à un client Kinesis, tel que votre tâche. Pour plus d'informations sur la diffusion améliorée de Kinesis, consultez la documentation Kinesis.

En mode de diffusion améliorée, les options de connexion `maxRecordPerRead` et `idleTimeBetweenReadsInMs` ne s'appliquent plus, car ces paramètres ne sont pas configurables lors de la diffusion améliorée. Les options de configuration pour les nouvelles tentatives fonctionnent comme décrit.

Utilisez les procédures suivantes pour activer et désactiver la diffusion améliorée pour votre tâche de streaming. Vous devez enregistrer un consommateur de flux pour chaque tâche qui consommera des données de votre flux.

Pour activer la consommation de diffusion améliorée sur votre tâche :

1. Enregistrez un consommateur de flux pour votre tâche à l'aide de l'API Kinesis. Suivez les instructions de la documentation Kinesis pour enregistrer un consommateur avec une diffusion améliorée en utilisant à l'aide de l'API Kinesis Data Streams. Il vous suffira de suivre la première étape : appeler RegisterStreamConsumer. Votre demande doit renvoyer un ARN,*`consumerARN`*.
2. Définissez l'option de connexion `fanoutConsumerARN` sur *`consumerARN`* dans les arguments de votre méthode de connexion.
3. Redémarrez votre tâche.

Pour désactiver la consommation de diffusion améliorée sur votre tâche :

1. Supprimez l'option de connexion `fanoutConsumerARN` de votre appel de méthode.
2. Redémarrez votre tâche.
3. Suivez les instructions de la documentation Kinesis pour annuler l'enregistrement d'un consommateur. Ces instructions s'appliquent à la console, mais peuvent également être obtenues via l'API Kinesis. Pour plus d'informations sur la désinscription des utilisateurs de streaming via l'API Kinesis, consultez la documentation Kinesis. DeregisterStreamConsumer

Connexions Amazon S3

Vous pouvez utiliser AWS Glue for Spark pour lire et écrire des fichiers dans Amazon S3. AWS Glue for Spark prend en charge de nombreux formats de données courants stockés dans Amazon S3 prêts à l'emploi, notamment CSV, Avro, JSON, Orc et Parquet. Pour de plus amples informations sur les formats de données pris en charge, consultez the section called “Options de format de données”. Chaque format de données peut prendre en charge un ensemble différent de fonctionnalités de AWS Glue. Consultez la page de votre format de données pour connaître les spécificités de la prise en charge des fonctionnalités. En outre, vous pouvez lire et écrire des fichiers versionnés stockés dans les cadres de lacs de données Hudi, Iceberg et Delta Lake. Pour plus d'informations sur les cadres de lac de données, consultez the section called “Cadres de lac de données”.

Avec AWS Glue, vous pouvez partitionner vos objets Amazon S3 dans une structure de dossiers pendant l'écriture, puis les récupérer par partition pour améliorer les performances grâce à une configuration simple. Vous pouvez également définir la configuration pour regrouper les petits fichiers lors de la transformation de vos données afin d'améliorer les performances. Vous pouvez lire et écrire des archives `bzip2` et `gzip` dans Amazon S3.

Rubriques

- Configuration de connexions S3
- Référence des options de connexion Amazon S3
- Syntaxes de connexion obsolètes pour les formats de données
- Exclusion des classes de stockage Amazon S3
- Gestion des partitions pour la sortie ETL dans AWS Glue
- Lecture des fichiers en entrée dans des groupes de plus grande taille
- Types de points de terminaison d'un VPC pour Amazon S3

Configuration de connexions S3

Pour vous connecter à Amazon S3 dans le cadre d'une tâche AWS Glue with Spark, vous aurez besoin de certaines conditions préalables :

- La tâche AWS Glue doit disposer d'autorisations IAM pour les buckets Amazon S3 concernés.

Dans certains cas, vous devrez configurer des prérequis supplémentaires :

- Lorsque vous configurez un accès intercompte, il convient de mettre en place des contrôles d'accès appropriés sur le compartiment Amazon S3.
- Pour des raisons de sécurité, vous pouvez choisir d'acheminer vos requêtes Amazon S3 via un Amazon VPC. Cette approche peut engendrer des problèmes en matière de bande passante et de disponibilité. Pour de plus amples informations, veuillez consulter the section called “Types de points de terminaison d'un VPC pour Amazon S3”.

Référence des options de connexion Amazon S3

Désigne une connexion à Amazon S3.

Comme Amazon S3 gère des fichiers plutôt que des tables, en plus de spécifier les propriétés de connexion fournies dans ce document, vous devrez indiquer une configuration supplémentaire concernant votre type de fichier. Vous indiquez ces informations par le biais des options de format de données. Pour plus d'informations sur ces options de format, consultez the section called “Options de format de données”. Vous pouvez également spécifier ces informations en les intégrant au catalogue de données AWS Glue.

Pour illustrer la distinction entre les options de connexion et les options de format, considérez comment la méthode the section called “create_dynamic_frame_from_options” utilise `connection_type`, `connection_options`, `format` et `format_options`. Cette section traite spécifiquement des paramètres fournis à `connection_options`.

Utilisez les options de connexion suivantes avec `"connectionType": "s3"` :

- `"paths"` : (obligatoire) une liste de chemins Amazon S3 à lire.
- `"exclusions"` : (Facultatif) Chaîne contenant une liste JSON des modèles glob de style Unix à exclure. Par exemple, `"[\"**.pdf\"]"` permet d'exclure tous les fichiers PDF. Pour plus d'informations sur la syntaxe glob qui prend en charge AWS Glue, consultez Inclure et Exclure les modèles.

- `"compressionType"` : ou « `compression` » : (facultatif) spécifie la manière dont les données sont comprimées. Utilisez `"compressionType"` pour les sources Amazon S3 et `"compression"` pour les cibles Amazon S3. Ce n'est généralement pas nécessaire si le fichier de données a une extension standard. Les valeurs possibles sont `"gzip"` et `"bzip2"`). Des formats de compression supplémentaires peuvent être pris en charge pour des formats spécifiques. Pour connaître les spécificités de la prise en charge des fonctionnalités, consultez la page sur le format des données.
- `"groupFiles"` : (facultatif) le groupement de fichiers est activé par défaut lorsque l'entrée contient plus de 50 000 fichiers. Pour activer le groupement lorsqu'il y a moins de 50 000 fichiers, définissez ce paramètre sur `"inPartition"`. Pour désactiver le groupement lorsqu'il y a plus de 50 000 fichiers, définissez ce paramètre sur `"none"`.
- `"groupSize"` : (Facultatif) Taille du groupe cible, en octets. La valeur par défaut est calculée en fonction de la dimension des données en entrée et de la dimension de votre cluster. Lorsqu'il y a moins de 50 000 fichiers en entrée, `"groupFiles"` doit être défini sur `"inPartition"` pour que cela prenne effet.
- `"recurse"` : (Facultatif) Si ce paramètre est défini sur « true », les fichiers sont lus de manière récursive dans tous les sous-répertoires des chemins spécifiés.
- `"maxBand"` : (facultatif, avancé) cette option permet de contrôler la durée, en millisecondes, au delà de laquelle la liste `s3` est susceptible d'être cohérente. Les fichiers dont l'horodatage de modification se situe dans les dernières `maxBand` millisecondes sont suivis lorsque des `JobBookmarks` sont utilisés pour prendre en compte une cohérence éventuelle Amazon S3. La plupart des utilisateurs n'ont pas besoin de définir cette option. La valeur par défaut est 900 000 millisecondes, soit 15 minutes.
- `"maxFilesInBand"` : (Facultatif, avancé) Cette option spécifie le nombre maximal de fichiers à enregistrer à partir des dernières `maxBand` secondes. Si ce nombre est dépassé, les fichiers supplémentaires sont ignorés et traités dans l'exécution de tâche suivante. La plupart des utilisateurs n'ont pas besoin de définir cette option.
- `"isFailFast"` : (facultatif) cette option détermine si une tâche ETL AWS Glue Glue lance des exceptions d'analyse de lecteur. Si elle est définie sur `true`, les tâches échouent rapidement si quatre tentatives de la tâche Spark échouent à analyser correctement les données.
- `"catalogPartitionPredicate"` : (facultatif) utilisé pour la lecture. Le contenu d'une clause `WHERE` SQL. Utilisé lors de la lecture de tables du catalogue de données comportant un très grand nombre de partitions. Récupère les partitions correspondantes à partir des index du catalogue de données. Utilisé avec `push_down_predicate`, une option sur la méthode the section called

"create_dynamic_frame_from_catalog" (et d'autres méthodes similaires). Pour de plus amples informations, veuillez consulter the section called "Prédicats de partition de catalogue".

- `"partitionKeys"` : (facultatif) utilisé pour l'écriture. Un tableau de chaînes d'étiquettes de colonnes. AWS Glue partitionnera vos données conformément à cette configuration. Pour de plus amples informations, veuillez consulter the section called "Écriture des partitions".
- `"excludeStorageClasses"` : (facultatif) utilisé pour la lecture. Tableau de chaînes spécifiant les classes de stockage Amazon S3. AWS Glue exclura les objets Amazon S3 en fonction de cette configuration. Pour de plus amples informations, veuillez consulter the section called "Exclusion des classes de stockage Amazon S3".

Syntaxes de connexion obsolètes pour les formats de données

Certains formats de données sont accessibles à l'aide d'une syntaxe de type de connexion spécifique. Cette syntaxe est obsolète. Nous vous recommandons de spécifier vos formats en utilisant le type de connexion `s3` et les options de format fournies dans the section called "Options de format de données" à la place.

« connectionType » : « orc »

Désigne une connexion à des fichiers stockés dans Amazon S3 au format de fichier Apache Hive Optimized Row Columnar (ORC).

Utilisez les options de connexion suivantes avec `"connectionType": "orc"` :

- `paths` : (obligatoire) une liste de chemins Amazon S3 à lire.
- (Autres name/value paires d'options) : Toutes les options supplémentaires, y compris les options de formatage, sont transmises directement au SparkSQL`DataSource`.

« connectionType » : « parquet »

Désigne une connexion aux fichiers stockés dans Amazon S3 au format de fichier Apache Parquet.

Utilisez les options de connexion suivantes avec `"connectionType": "parquet"` :

- `paths` : (obligatoire) une liste de chemins Amazon S3 à lire.
- (Autres name/value paires d'options) : Toutes les options supplémentaires, y compris les options de formatage, sont transmises directement au SparkSQL`DataSource`.

Exclusion des classes de stockage Amazon S3

Si vous exécutez des tâches ETL AWS Glue qui lisent des fichiers ou des partitions à partir d'Amazon Simple Storage Service (Amazon S3), vous pouvez exclure certains types de classe de stockage Amazon S3.

Les classes de stockage suivantes sont disponibles dans Amazon S3 :

- `STANDARD` — pour le stockage à usage général des données fréquemment consultées.
- `INTELLIGENT_TIERING` — pour les données ayant des modèles d'accès inconnus ou variables.
- `STANDARD_IA` et `ONEZONE_IA` — pour les données à longue durée de vie, mais consultées moins fréquemment.
- `GLACIER`, `DEEP_ARCHIVE` et `REDUCED_REDUNDANCY` — pour l'archivage à long terme et la conservation numérique.

Pour de plus amples informations, veuillez consulter Classes de stockage Amazon S3 dans le Guide du développeur Amazon S3.

Les exemples de cette section montrent comment exclure les classes de stockage `GLACIER` et `DEEP_ARCHIVE`. Ces classes vous permettent de répertorier les fichiers, mais ne vous permettent pas de les lire, sauf s'ils sont restaurés. (Pour de plus amples informations, veuillez consulter Restauration d'objets archivés dans le Guide du développeur Amazon S3.)

En utilisant les exclusions de classe de stockage, vous pouvez garantir que vos tâches AWS Glue fonctionnent sur les tables ayant des partitions sur ces niveaux de classe de stockage. Sans exclusion, les tâches qui lisent les données de ces niveaux échouent avec l'erreur suivante : AmazonS3Exception: The operation is not valid for the object's storage class (AmazonS3Exception : l'opération n'est pas valide pour la classe de stockage de l'objet).

Vous pouvez filtrer les classes de stockage Amazon S3 de différentes façons dans AWS Glue.

Rubriques

- Exclusion des classes de stockage Amazon S3 lors de la création d'une image dynamique
- Exclusion de classes de stockage Amazon S3 d'une table Data Catalog

Exclusion des classes de stockage Amazon S3 lors de la création d'une image dynamique

Pour exclure les classes de stockage Amazon S3 lors de la création d'une image dynamique, utilisez `excludeStorageClasses` dans `additionalOptions`. AWS Glue utilise automatiquement sa propre implémentation de `Lister` Amazon S3 pour répertorier et exclure les fichiers correspondant aux classes de stockage spécifiées.

Les exemples Python et Scala suivants montrent comment exclure les classes de stockage `GLACIER` et `DEEP_ARCHIVE` lors de la création d'une image dynamique.

Exemple Python :

```
glueContext.create_dynamic_frame.from_catalog(
    database = "my_database",
    tableName = "my_table_name",
    redshift_tmp_dir = "",
    transformation_ctx = "my_transformation_context",
    additional_options = {
        "excludeStorageClasses" : ["GLACIER", "DEEP_ARCHIVE"]
    }
)
```

Exemple Scala :

```
val* *df = glueContext.getCatalogSource(
    nameSpace, tableName, "", "my_transformation_context",
    additionalOptions = JsonOptions(
        Map("excludeStorageClasses" -> List("GLACIER", "DEEP_ARCHIVE"))
    )
).getDynamicFrame()
```

Exclusion de classes de stockage Amazon S3 d'une table Data Catalog

Vous pouvez spécifier les exclusions de classe de stockage à utiliser par une tâche AWS Glue ETL en tant que paramètre de table dans le catalogue de données AWS Glue. Vous pouvez inclure ce paramètre dans l'`CreateTable`opération à l'aide du AWS Command Line Interface (AWS CLI) ou par programmation à l'aide de l'API. Pour plus d'informations, voir Structure du tableau et CreateTable.

Vous pouvez également spécifier des classes de stockage exclues sur la console AWS Glue.

Pour exclure des classes de stockage Amazon S3 (console)

1. Connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/.
2. Dans le panneau de navigation de gauche, sélectionnez Tables.
3. Choisissez le nom de la table dans la liste, puis choisissez Modifier la table.
4. Dans Propriétés de la table, ajoutez **`excludeStorageClasses`** en tant que clé et **`[\"GLACIER\",\"DEEP_ARCHIVE\"]`** en tant que valeur.
5. Cliquez sur Appliquer.

Gestion des partitions pour la sortie ETL dans AWS Glue

Le partitionnement est une technique importante technique pour organiser les ensembles de données afin qu'ils puissent être interrogés de manière efficace. Il organise les données en une structure de répertoires hiérarchique fondée sur les valeurs distinctes d'une ou de plusieurs colonnes.

Par exemple, vous pouvez décider de partitionner vos journaux d'application dans Amazon Simple Storage Service (Amazon S3) par date, réparties par année, par mois et par jour. Les fichiers qui correspondent à une seule journée de données sont ensuite placés sous un préfixe, tel que `s3://my_bucket/logs/year=2018/month=01/day=23/`. Les systèmes comme Amazon Athena, Amazon Redshift Spectrum, et maintenant AWS Glue peuvent utiliser ces partitions pour filtrer les données en fonction de la valeur de partition, sans avoir à lire toutes les données sous-jacentes depuis Amazon S3.

Les robots d'exploration ne se contentent pas de déduire les types de fichiers et les schémas, ils identifient également automatiquement la structure de partition de votre ensemble de données lorsqu'ils alimentent le catalogue de données AWS Glue. Les colonnes de partition résultantes sont disponibles pour l'interrogation dans les tâches ETL AWS Glue ou les moteurs de requête tels qu'Amazon Athena.

Après avoir analysé une table, vous pouvez afficher les partitions créées par l'crawler. Dans la console AWS Glue, dans le panneau de navigation de gauche, sélectionnez Tables. Sélectionnez la table créée par l'crawler, puis View partitions (Afficher les partitions).

Pour les chemins partitionnés de style Apache Hive de type `key=val`, les crawlers remplissent automatiquement le nom de la colonne à l'aide du nom de clé. Sinon, ils utilisent des noms par défaut comme `partition_0`, `partition_1`, etc. Vous pouvez modifier les noms par défaut sur la console. Pour ce faire, accédez à la table. Vérifiez si des index existent sous l'onglet Index. Si tel

est le cas, vous devez les supprimer pour continuer (vous pourrez ensuite les recréer en utilisant les nouveaux noms de colonnes). Choisissez ensuite Modifier le schéma et modifiez les noms des colonnes de partition qui s'y trouvent.

Dans vos scripts ETL, vous pouvez ensuite filtrer sur les colonnes de partition. Étant donné que les informations de partition sont stockées dans le catalogue de données, utilisez les appels d'API `from_catalog` pour inclure les colonnes de partition dans le fichier `DynamicFrame`. Par exemple, utilisez `create_dynamic_frame.from_catalog` plutôt que `create_dynamic_frame.from_options`.

Le partitionnement est une technique d'optimisation qui réduit l'analyse des données. Pour plus d'informations sur le processus permettant d'identifier le moment où cette technique est appropriée, consultez la section Réduire la quantité de données analysées dans le guide des meilleures pratiques pour le réglage des performances de AWS Glue pour les tâches Apache Spark sur le guide AWS prescriptif.

Préfiltrage à l'aide des prédicats pushdown

Dans de nombreux cas, vous pouvez utiliser un prédicat pushdown pour filtrer les partitions sans avoir à répertorier et à lire tous les fichiers de votre ensemble de données. Au lieu de lire l'ensemble de données et de filtrer ensuite un DynamicFrame fichier, vous pouvez appliquer le filtre directement sur les métadonnées de partition dans le catalogue de données. Ensuite, vous ne listez et ne lisez que ce dont vous avez réellement besoin dans un DynamicFrame.

Par exemple, en Python, vous pouvez écrire ce qui suit.

```
glue_context.create_dynamic_frame.from_catalog(
    database = "my_S3_data_set",
    table_name = "catalog_data_table",
    push_down_predicate = my_partition_predicate)
```

Cela crée un DynamicFrame qui charge uniquement les partitions du catalogue de données qui répondent à l'expression de prédicat. En fonction de la taille d'un sous-ensemble de vos données de chargement, vous pouvez économiser beaucoup de temps de traitement.

L'expression de prédicat peut être n'importe quelle expression booléenne prise en charge par Spark SQL. Tout ce que vous pouvez placer dans une clause `WHERE` d'une requête SQL Spark fonctionne. Par exemple, l'expression de prédicat `pushDownPredicate = "(year=='2017' and month=='04')"` charge uniquement les partitions dans Data Catalog qui disposent à la fois d'une

`year` égale à 2017 et un `month` égal à 04. Pour plus d'informations, consultez la documentation Apache Spark SQL, en particulier le document Référence des fonctions Scala SQL.

Filtrage côté serveur à l'aide de prédicats de partition de catalogue

L'option `push_down_predicate` est appliquée après avoir répertorié toutes les partitions du catalogue et avant de répertorier les fichiers d'Amazon S3 pour ces partitions. Si vous disposez d'un grand nombre de partitions pour une table, la liste des partitions du catalogue peut encore entraîner une surcharge de temps supplémentaire. Pour remédier à cette surcharge, vous pouvez utiliser l'élagage des partitions côté serveur avec l'`catalogPartitionPredicate`option qui utilise les index de partition dans le catalogue de données Glue AWS . Cela accélère considérablement le filtrage des partitions lorsque vous avez des millions de partitions dans une table. Vous pouvez utiliser à la fois `push_down_predicate` et `catalogPartitionPredicate` dans `additional_options` si votre `catalogPartitionPredicate` a besoin d'une syntaxe de prédicat qui n'est pas encore prise en charge avec les index de partition de catalogue.

Python :

```
dynamic_frame = glueContext.create_dynamic_frame.from_catalog(
    database=dbname,
    table_name=tablename,
    transformation_ctx="datasource0",
    push_down_predicate="day>=10 and customer_id like '10%'",
    additional_options={"catalogPartitionPredicate":"year='2021' and month='06'"}
)
```

Scala :

```
val dynamicFrame = glueContext.getCatalogSource(
    database = dbname,
    tableName = tablename,
    transformationContext = "datasource0",
    pushDownPredicate="day>=10 and customer_id like '10%'",
    additionalOptions = JsonOptions("""{
        "catalogPartitionPredicate": "year='2021' and month='06'"}""")
    ).getDynamicFrame()
```

> **Note**
>
> `push_down_predicate` et `catalogPartitionPredicate` utilisent des syntaxes différentes. Le premier utilise la syntaxe standard SQL Spark et le dernier utilise l'analyseur JSQL.

Écriture des partitions

Par défaut, a n' DynamicFrame est pas partitionné lors de son écriture. Tous les fichiers de sortie sont écrits au niveau supérieur du chemin de sortie spécifié. Jusqu'à récemment, le seul moyen d'écrire un DynamicFrame dans des partitions était de le convertir en SQL Spark DataFrame avant de l'écrire.

Cependant, vous pouvez DynamicFrames désormais prendre en charge le partitionnement natif à l'aide d'une séquence de clés, en utilisant l'`partitionKeys`option lorsque vous créez un récepteur. Par exemple, le code Python suivant écrit un ensemble de données sur Amazon S3 au format Parquet, dans les répertoires partitionnés par le type de champ. À partir de là, vous pouvez traiter ces partitions à l'aide d'autres systèmes, comme Amazon Athena.

```
glue_context.write_dynamic_frame.from_options(
    frame = projectedEvents,
    connection_type = "s3",
    connection_options = {"path": "$outpath", "partitionKeys": ["type"]},
    format = "parquet")
```

Lecture des fichiers en entrée dans des groupes de plus grande taille

Vous pouvez définir les propriétés de vos tables pour activer une tâche ETL AWS Glue regroupant les fichiers lorsqu'ils sont lus à partir d'un magasin de données Amazon S3. Ces propriétés permettent à chaque tâche ETL de lire un groupe de fichiers d'entrée en une seule partition en mémoire, ce qui est particulièrement utile lorsqu'il y a un grand nombre de petits fichiers dans votre magasin de données Amazon S3. Lorsque vous définissez certaines propriétés, vous demandez à AWS Glue de regrouper les fichiers au sein d'une partition de données Amazon S3 et de définir la taille des groupes à lire. Vous pouvez également définir ces options lors de la lecture à partir d'un magasin de données Amazon S3 avec la méthode `create_dynamic_frame.from_options`.

Pour activer le regroupement de fichiers pour une table, vous définissez les paires clé-valeur dans le champ des paramètres de la structure de votre table. Utilisez la notation JSON pour définir une

valeur pour le champ des paramètres de votre table. Pour plus d'informations sur la modification des propriétés d'une table, consultez Affichage et gestion des détails d'une table.

Vous pouvez utiliser cette méthode pour activer le regroupement des tables dans Data Calalog avec les magasins de données Amazon S3.

groupFiles

Définissez groupFiles sur `inPartition` afin d'activer le regroupement de fichiers au sein d'une partition de données Amazon S3. AWS Glue active automatiquement le regroupement s'il y a plus de 50 000 fichiers d'entrée, comme dans l'exemple suivant.

```
'groupFiles': 'inPartition'
```

groupSize

Définissez groupSize sur la taille cible des groupes en octets. La propriété groupSize est facultative. Si aucune valeur n'est définie, AWS Glue calcule une taille pour utiliser tous les cœurs d'UC dans le cluster tout en réduisant le nombre total de tâches ETL et les partitions en mémoire.

Par exemple, ce qui suit définit une taille de groupe de 1 Mo.

```
'groupSize': '1048576'
```

Notez que le `groupsize` doit être défini avec le résultat d'un calcul. Par exemple, 1024 * 1024 = 1048576.

recurse

Définissez recurse (récursif) à `True` pour lire les fichiers de manière récursive dans tous les sous-répertoires lorsque vous spécifiez `paths` en tant que liste de chemins. Vous n'avez pas besoin de définir recurse si `paths` est un tableau de clés d'objet dans Amazon S3, ou si le format d'entrée est parquet/orc, comme dans l'exemple suivant.

```
'recurse':True
```

Si vous lisez à partir d'Amazon S3 directement à l'aide de la méthode `create_dynamic_frame.from_options`, ajoutez ces options de connexion. Par exemple, ce qui suit tente de placer les fichiers par groupes de 1 Mo.

```
df = glueContext.create_dynamic_frame.from_options("s3", {'paths': ["s3://s3path/"],
 'recurse':True, 'groupFiles': 'inPartition', 'groupSize': '1048576'}, format="json")
```

> **Note**
>
> `groupFiles`est pris en charge pour les formats de données DynamicFrames créés à partir des formats de données suivants : csv, ion, GrokLog, json et xml. Cette option n'est pas prise en charge pour avro, parquet et orc.

Types de points de terminaison d'un VPC pour Amazon S3

Pour des raisons de sécurité, de nombreux AWS clients exécutent leurs applications dans un environnement Amazon Virtual Private Cloud (Amazon VPC). Avec Amazon VPC, vous pouvez lancer des EC2 instances Amazon dans un cloud privé virtuel, qui est logiquement isolé des autres réseaux, y compris de l'Internet public. Avec un VPC Amazon, vous contrôlez sa plage d'adresses IP, ses sous-réseaux, ses tables de routage, ses passerelles réseau et ses paramètres de sécurité.

> **Note**
>
> Si vous avez créé votre AWS compte après le 04/12/2013, vous disposez déjà d'un VPC par défaut dans chaque région. AWS Vous pouvez commencer immédiatement à utiliser votre VPC par défaut sans configuration supplémentaire.
> Pour plus d'informations, veuillez consulter Vos VPC et sous-réseaux par défaut dans le Guide de l'utilisateur Amazon VPC.

De nombreux clients ont des préoccupations légitimes liées à la confidentialité et la sécurité lors de l'envoi et de la réception de données via le réseau Internet public. Les clients peuvent faire face à ces préoccupations en utilisant un réseau VPN pour acheminer l'ensemble du trafic réseau d'Amazon

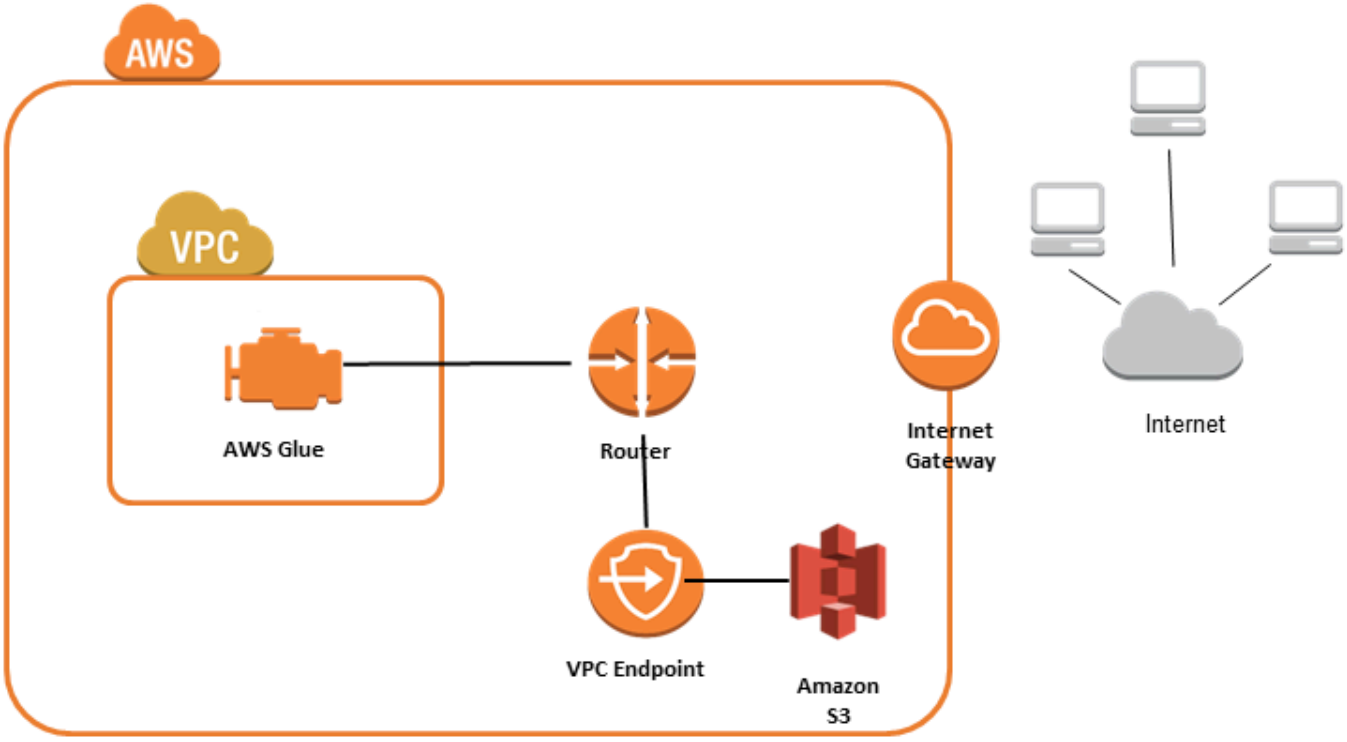
S3 via leur propre infrastructure réseau d'entreprise. Toutefois, cette approche peut engendrer des problèmes en matière de bande passante et de disponibilité.

Les points de terminaison d'un VPC pour Amazon S3 peuvent atténuer ces problèmes. Un point de terminaison d'un VPC pour Amazon S3 permet à AWS Glue d'utiliser des adresses IP privées pour accéder à Amazon S3 sans exposition au réseau Internet public. AWS Glue ne nécessite pas d'adresses IP publiques et vous n'avez pas besoin de passerelle Internet, de périphérique NAT ou de passerelle réseau privé virtuel dans votre VPC. Vous utilisez des politiques de point de terminaison pour contrôler l'accès à Amazon S3. Le trafic entre votre VPC et le AWS service ne quitte pas le réseau Amazon.

Lorsque vous créez un point de terminaison d'un VPC pour Amazon S3, toutes les demandes envoyées à un point de terminaison Amazon S3 au sein de la région (par exemple, s3.us-west-2.amazonaws.com) sont acheminées vers un point de terminaison Amazon S3 privé dans le réseau Amazon. Vous n'avez pas besoin de modifier vos applications exécutées sur les EC2 instances Amazon de votre VPC : le nom du point de terminaison reste le même, mais le chemin vers Amazon S3 reste entièrement au sein du réseau Amazon et n'accède pas à l'Internet public.

Pour plus d'informations sur les points de terminaison d'un VPC, veuillez consulter Points de terminaison d'un VPC dans le Guide de l'utilisateur Amazon VPC.

Le diagramme suivant montre comment AWS Glue peut utiliser un point de terminaison d'un VPC pour accéder à Amazon S3.



Pour configurer l'accès à Amazon S3

1. Connectez-vous à la console Amazon VPC AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/vpc/
2. Dans le panneau de navigation de gauche, sélectionnez Points de terminaison.
3. Sélectionnez Create Endpoint (Créer un point de terminaison) et suivez les étapes pour créer un point de terminaison d'un VPC Amazon S3 de type Passerelle.

Connexions Amazon DocumentDB

Vous pouvez utiliser AWS Glue for Spark pour lire et écrire dans des tables d'Amazon DocumentDB. Vous pouvez vous connecter à Amazon DocumentDB à l'aide des informations d'identification enregistrées AWS Secrets Manager via une connexion AWS Glue.

Pour plus d'informations sur Amazon DocumentDB, consultez la documentation Amazon DocumentDB.

> **Note**
>
> Les clusters élastiques Amazon DocumentDB ne sont actuellement pas pris en charge lors de l'utilisation du connecteur AWS Glue. Pour plus d'informations sur les clusters élastiques, consultez Using Amazon DocumentDB elastic clusters.

Lecture et écriture dans les collections Amazon DocumentDB

> **Note**
>
> Lorsque vous créez une tâche ETL qui se connecte à Amazon DocumentDB pour la propriété de tâche `Connections`, vous devez désigner un objet de connexion qui spécifie le virtual private cloud (VPC) dans lequel Amazon DocumentDB est exécuté. Pour l'objet de connexion, le type de connexion doit être `JDBC` et le `JDBC URL` doit être `mongo://<DocumentDB_host>:27017`.

> **Note**
>
> Ces exemples de code ont été développés pour AWS Glue 3.0. Pour migrer vers AWS Glue 4.0, consultezthe section called "MongoDB". Le paramètre `uri` a été modifié.

> **Note**
>
> Lorsque vous utilisez Amazon DocumentDB, le paramètre `retryWrites` doit être défini sur false dans certaines situations, par exemple lorsque le document écrit spécifie `_id`. Pour plus d'informations, consultez Différences fonctionnelles avec MongoDB dans la documentation Amazon DocumentDB.

Le script Python suivant illustre l'utilisation des types et des options de connexion pour la lecture et l'écriture dans Amazon DocumentDB.

```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext, SparkConf
from awsglue.context import GlueContext
from awsglue.job import Job
import time

## @params: [JOB_NAME]
args = getResolvedOptions(sys.argv, ['JOB_NAME'])

sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session

job = Job(glueContext)
job.init(args['JOB_NAME'], args)

output_path = "s3://some_bucket/output/" + str(time.time()) + "/"
documentdb_uri = "mongodb://<mongo-instanced-ip-address>:27017"
documentdb_write_uri = "mongodb://<mongo-instanced-ip-address>:27017"

read_docdb_options = {
    "uri": documentdb_uri,
    "database": "test",
    "collection": "coll",
    "username": "username",
    "password": "1234567890",
    "ssl": "true",
    "ssl.domain_match": "false",
    "partitioner": "MongoSamplePartitioner",
    "partitionerOptions.partitionSizeMB": "10",
    "partitionerOptions.partitionKey": "_id"
}

write_documentdb_options = {
    "retryWrites": "false",
    "uri": documentdb_write_uri,
    "database": "test",
    "collection": "coll",
    "username": "username",
    "password": "pwd"
}
```

```
# Get DynamicFrame from  DocumentDB
dynamic_frame2 =
 glueContext.create_dynamic_frame.from_options(connection_type="documentdb",

 connection_options=read_docdb_options)

# Write DynamicFrame to MongoDB and DocumentDB
glueContext.write_dynamic_frame.from_options(dynamic_frame2,
 connection_type="documentdb",

 connection_options=write_documentdb_options)

job.commit()
```

Le script Scala suivant illustre l'utilisation des types et des options de connexion pour la lecture et l'écriture dans Amazon DocumentDB.

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.MappingSpec
import com.amazonaws.services.glue.errors.CallSite
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.DynamicFrame
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._

object GlueApp {
  val DOC_URI: String = "mongodb://<mongo-instanced-ip-address>:27017"
  val DOC_WRITE_URI: String = "mongodb://<mongo-instanced-ip-address>:27017"
  lazy val documentDBJsonOption = jsonOptions(DOC_URI)
  lazy val writeDocumentDBJsonOption = jsonOptions(DOC_WRITE_URI)
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)

    // Get DynamicFrame from DocumentDB
    val resultFrame2: DynamicFrame = glueContext.getSource("documentdb",
 documentDBJsonOption).getDynamicFrame()
```

```
    // Write DynamicFrame to DocumentDB
    glueContext.getSink("documentdb", writeJsonOption).writeDynamicFrame(resultFrame2)

    Job.commit()
  }

  private def jsonOptions(uri: String): JsonOptions = {
    new JsonOptions(
      s"""{"uri": "${uri}",
         |"database":"test",
         |"collection":"coll",
         |"username": "username",
         |"password": "pwd",
         |"ssl":"true",
         |"ssl.domain_match":"false",
         |"partitioner": "MongoSamplePartitioner",
         |"partitionerOptions.partitionSizeMB": "10",
         |"partitionerOptions.partitionKey": "_id"}""".stripMargin)
  }
}
```

Référence des options de connexion Amazon DocumentDB

Désigne une connexion à Amazon DocumentDB (compatible avec MongoDB).

Les options de connexion sont différentes pour une connexion source et une connexion collecteur.

« connectionType »: « documentdb » comme cource

Utilisez les options de connexion suivantes avec `"connectionType": "documentdb"` comme source :

- `"uri"` : (obligatoire) hôte Amazon DocumentDB à lire, dans le format `mongodb://<host>:<port>`.
- `"database"` : (obligatoire) base de données Amazon DocumentDB à lire.
- `"collection"` : (obligatoire) collection Amazon DocumentDB à lire.
- `"username"` : (obligatoire) nom d'utilisateur Amazon DocumentDB.
- `"password"` : (obligatoire) mot de passe Amazon DocumentDB.
- `"ssl"` : (obligatoire si vous utilisez SSL) si votre connexion utilise SSL, vous devez inclure cette option avec la valeur `"true"`.

- `"ssl.domain_match"` : (obligatoire si vous utilisez SSL) si votre connexion utilise SSL, vous devez inclure cette option avec la valeur `"false"`.
- `"batchSize"` : (Facultatif) : Nombre de documents à renvoyer par lot, utilisé dans le curseur des lots internes.
- `"partitioner"` : (facultatif) : nom de classe de l'outil du partitionneur pour lire les données d'entrée à partir d'Amazon DocumentDB. Le connecteur fournit les partitionneurs suivants :
  - `MongoDefaultPartitioner`(par défaut) (Non pris en charge dans AWS Glue 4.0)
  - `MongoSamplePartitioner`(Non pris en charge dans AWS Glue 4.0)
  - `MongoShardedPartitioner`
  - `MongoSplitVectorPartitioner`
  - `MongoPaginateByCountPartitioner`
  - `MongoPaginateBySizePartitioner`(Non pris en charge dans AWS Glue 4.0)
- `"partitionerOptions"` (Facultatif) : Options pour le partitionneur désigné. Les options suivantes sont prises en charge pour chaque partitionneur :
  - `MongoSamplePartitioner`: `partitionKey`, `partitionSizeMB`, `samplesPerPartition`
  - `MongoShardedPartitioner`: `shardkey`
  - `MongoSplitVectorPartitioner` : `partitionKey`, partitionSizeMB
  - `MongoPaginateByCountPartitioner`: `partitionKey`, `numberOfPartitions`
  - `MongoPaginateBySizePartitioner` : `partitionKey`, partitionSizeMB

  Pour plus d'informations sur ces options, consultez Configuration du partitionneur dans la documentation MongoDB.

« connectionType »: « documentdb » comme collecteur

Utilisez les options de connexion suivantes avec `"connectionType": "documentdb"` comme collecteur :

- `"uri"` : (obligatoire) hôte Amazon DocumentDB où écrire, dans le format `mongodb://<host>:<port>`.
- `"database"` : (obligatoire) base de données Amazon DocumentDB où écrire.
- `"collection"` : (obligatoire) collection Amazon DocumentDB où écrire.
- `"username"` : (obligatoire) nom d'utilisateur Amazon DocumentDB.

- `"password"` : (obligatoire) mot de passe Amazon DocumentDB.
- `"extendedBsonTypes"` : (facultatif) si la valeur est `true`, cela autorise les types BSON étendus lors de l'écriture de données dans Amazon DocumentDB. La valeur par défaut est `true`.
- `"replaceDocument"` : (Facultatif) Si `true`, remplace l'ensemble du document lors de l'enregistrement de jeux de données contenant un champ `_id`. Si `false`, seuls les champs du document qui correspondent aux champs du jeu de données sont mis à jour. La valeur par défaut est `true`.
- `"maxBatchSize"` : (Facultatif) : Taille maximale du lot pour les opérations en bloc lors de l'enregistrement des données. La valeur par défaut est 512.
- `"retryWrites"`: (Facultatif) : Réessayez automatiquement certaines opérations d'écriture une seule fois si AWS Glue rencontre une erreur réseau.

OpenSearch Connexions de service

Vous pouvez utiliser AWS Glue for Spark pour lire et écrire dans des tables dans OpenSearch Service in AWS Glue 4.0 et versions ultérieures. Vous pouvez définir ce qu'il faut lire dans OpenSearch Service à l'aide d'une OpenSearch requête. Vous vous connectez au OpenSearch Service à l'aide des informations d'authentification HTTP de base stockées AWS Secrets Manager via une connexion AWS Glue. Cette fonctionnalité n'est pas compatible avec le OpenSearch Service serverless.

Pour plus d'informations sur Amazon OpenSearch Service, consultez la documentation Amazon OpenSearch Service.

Configuration des connexions OpenSearch de service

Pour vous connecter à OpenSearch Service from AWS Glue, vous devez créer et stocker vos identifiants de OpenSearch service dans un AWS Secrets Manager secret, puis associer ce secret à une connexion OpenSearch Service AWS Glue.

Prérequis :

- Identifiez le point de terminaison *`aosEndpoint`* et le port du domaine que *`aosPort`* vous souhaitez consulter, ou créez la ressource en suivant les instructions de la documentation Amazon OpenSearch Service. Pour plus d'informations sur la création d'un domaine, consultez la section Création et gestion OpenSearch de domaines Amazon Service dans la documentation Amazon OpenSearch Service.

Un point de terminaison de domaine Amazon OpenSearch Service aura le formulaire par défaut suivant, https://search - *domainName* -*unstructuredIdContent*. *region*.es.amazonaws.com. Pour plus d'informations sur l'identification du point de terminaison de votre domaine, consultez la section Création et gestion des domaines Amazon OpenSearch Service dans la documentation Amazon OpenSearch Service.

Identifiez ou générez des informations d'authentification HTTP de base, *aosUser* et *aosPassword* pour votre domaine.

Pour configurer une connexion au OpenSearch service :

1. Dans AWS Secrets Manager, créez un secret à l'aide de vos informations d'identification OpenSearch de service. Pour créer un secret dans Secrets Manager, suivez le didacticiel disponible dans la section Créer un AWS Secrets Manager secret dans la AWS Secrets Manager documentation. Après avoir créé le secret, conservez le nom du secret *secretName* pour l'étape suivante.

   - Lorsque vous sélectionnez des paires clé/valeur, créez une paire pour la clé `USERNAME` contenant la valeur. *aosUser*
   - Lorsque vous sélectionnez des paires clé/valeur, créez une paire pour la clé `PASSWORD` contenant la valeur. *aosPassword*

2. Dans la console AWS Glue, créez une connexion en suivant les étapes décrites dansthe section called “Ajouter une AWS Glue connexion”. Après avoir créé la connexion, conservez le nom de la connexion pour une utilisation future dans AWS Glue. *connectionName*

   - Lorsque vous sélectionnez un type de connexion, sélectionnez OpenSearch Service.
   - Lorsque vous sélectionnez un point de terminaison de domaine, fournissez*aosEndpoint*.
   - Lorsque vous sélectionnez un port, indiquez*aosPort*.
   - Lorsque vous sélectionnez un AWS secret, fournissez*secretName*.

Après avoir créé une connexion AWS Glue OpenSearch Service, vous devez suivre les étapes suivantes avant d'exécuter votre tâche AWS Glue :

- Accordez au rôle IAM associé à votre tâche AWS Glue l'autorisation de lire*secretName*.

- Dans la configuration de votre tâche AWS Glue*connectionName*, fournissez une connexion réseau supplémentaire.

Lecture à partir des index OpenSearch de service

Prérequis :

- Un index des OpenSearch services que vous aimeriez lire,*aosIndex*.
- Une connexion AWS Glue OpenSearch Service configurée pour fournir des informations d'authentification et de localisation réseau. Pour cela, suivez les étapes de la procédure précédente, Pour configurer une connexion au OpenSearch service. Vous aurez besoin du nom de la connexion AWS Glue,*connectionName*.

Cet exemple lit un index provenant d'Amazon OpenSearch Service. Vous devrez fournir le paramètre `pushdown`.

Par exemple :

```
opensearch_read = glueContext.create_dynamic_frame.from_options(
    connection_type="opensearch",
    connection_options={
        "connectionName": "connectionName",
        "opensearch.resource": "aosIndex",
        "pushdown": "true",
    }
)
```

Vous pouvez également fournir une chaîne de requête pour filtrer les résultats renvoyés dans votre DynamicFrame. Vous devrez configurer `opensearch.query`.

`opensearch.query`peut prendre une chaîne de paramètres de requête URL *queryString* ou un objet *queryObject* JSON DSL de requête. Pour plus d'informations sur la requête DSL, consultez la section Requête DSL dans la OpenSearch documentation. Pour fournir une chaîne de paramètres de requête URL, ajoutez `?q=` à votre requête, comme vous le feriez pour une URL complète. Pour fournir un objet de requête DSL, échapper une chaîne objet JSON avant de le fournir.

Par exemple :

```
            queryObject = "{ "query": { "multi_match": { "query": "Sample", "fields":
 [ "sample" ] } } }"
            queryString = "?q=queryString"

            opensearch_read_query = glueContext.create_dynamic_frame.from_options(
    connection_type="opensearch",
    connection_options={
        "connectionName": "connectionName",
        "opensearch.resource": "aosIndex",
        "opensearch.query": queryString,
        "pushdown": "true",
    }
)
```

Pour plus d'informations sur la façon de créer une requête en dehors de sa syntaxe spécifique, consultez la section Syntaxe des chaînes de requête dans la OpenSearch documentation.

Lorsque vous lisez des OpenSearch collections contenant des données de type tableau, vous devez spécifier quels champs sont de type tableau dans votre appel de méthode à l'aide du `opensearch.read.field.as.array.include` paramètre.

Par exemple, en lisant le document suivant, vous rencontrerez les champs `genre` et `actor` du tableau :

```
{
    "_index": "movies",
    "_id": "2",
    "_version": 1,
    "_seq_no": 0,
    "_primary_term": 1,
    "found": true,
    "_source": {
        "director": "Frankenheimer, John",
        "genre": [
            "Drama",
            "Mystery",
            "Thriller",
            "Crime"
        ],
        "year": 1962,
        "actor": [
            "Lansbury, Angela",
            "Sinatra, Frank",
```

```
            "Leigh, Janet",
            "Harvey, Laurence",
            "Silva, Henry",
            "Frees, Paul",
            "Gregory, James",
            "Bissell, Whit",
            "McGiver, John",
            "Parrish, Leslie",
            "Edwards, James",
            "Flowers, Bess",
            "Dhiegh, Khigh",
            "Payne, Julie",
            "Kleeb, Helen",
            "Gray, Joe",
            "Nalder, Reggie",
            "Stevens, Bert",
            "Masters, Michael",
            "Lowell, Tom"
        ],
        "title": "The Manchurian Candidate"
    }
}
```

Dans ce cas, vous devez inclure les noms de ces champs dans votre appel de méthode. Par exemple :

```
"opensearch.read.field.as.array.include": "genre,actor"
```

Si votre champ de tableau est imbriqué dans la structure de votre document, faites-y référence en utilisant la notation par points : `"genre,actor,foo.bar.baz"`. Cela permet de spécifier un tableau `baz` inclus dans votre document source par le biais du document intégré `foo` contenant le document intégré `bar`.

Écrire dans les tables OpenSearch de service

Cet exemple écrit les informations d'un fichier existant DynamicFrame *`dynamicFrame`* dans le OpenSearch Service. Si l'index contient déjà des informations, AWS Glue ajoutera les données de votre DynamicFrame. Vous devrez fournir le paramètre `pushdown`.

Prérequis :

- Une table de OpenSearch service à laquelle vous souhaitez écrire. Vous aurez besoin des informations d'identification de la table. Appelons ça*tableName*.
- Une connexion AWS Glue OpenSearch Service configurée pour fournir des informations d'authentification et de localisation réseau. Pour cela, suivez les étapes de la procédure précédente, Pour configurer une connexion au OpenSearch service. Vous aurez besoin du nom de la connexion AWS Glue,*connectionName*.

Par exemple :

```
glueContext.write_dynamic_frame.from_options(
    frame=dynamicFrame,
    connection_type="opensearch",
    connection_options={
      "connectionName": "connectionName",
      "opensearch.resource": "aosIndex",
    },
)
```

OpenSearch Référence des options de connexion au service

- `connectionName` — Obligatoire. Utilisé pour la lecture/l'écriture. Nom d'une connexion AWS Glue OpenSearch Service configurée pour fournir des informations d'authentification et de localisation réseau à votre méthode de connexion.
- `opensearch.resource` — Obligatoire. Utilisé pour la lecture/l'écriture. Valeurs valides : noms d' OpenSearch index. Le nom de l'index avec lequel votre méthode de connexion va interagir.
- `opensearch.query` : utilisé pour la lecture. Valeurs valides : chaîne JSON échappée ou, lorsque cette chaîne commence par `?`, la partie de recherche d'une URL. Une OpenSearch requête qui filtre ce qui doit être récupéré lors de la lecture. Pour plus d'informations sur l'utilisation de ce paramètre, consultez la section précédente the section called “Lire depuis le OpenSearch service”.
- `pushdown` — Obligatoire si. Utilisé pour la lecture. Valeurs valides : booléen. Demande à Spark de transmettre les requêtes de lecture OpenSearch afin que la base de données ne renvoie que les documents pertinents.
- `opensearch.read.field.as.array.include` : obligatoire en cas de lecture de données de type tableau. Utilisé pour la lecture. Valeurs valides : listes de noms de champs séparés par des virgules. Spécifie les champs à lire sous forme de tableaux à partir de OpenSearch documents. Pour plus d'informations sur l'utilisation de ce paramètre, consultez la section précédente the section called “Lire depuis le OpenSearch service”.

Connexions Redshift

Vous pouvez utiliser AWS Glue for Spark pour lire et écrire dans les tables des bases de données Amazon Redshift. Lors de la connexion aux bases de données Amazon Redshift, AWS Glue déplace les données via Amazon S3 pour atteindre un débit maximal, en utilisant le code SQL et les commandes Amazon Redshift. `COPY` `UNLOAD` Dans AWS Glue 4.0 et versions ultérieures, vous pouvez utiliser l'intégration Amazon Redshift pour Apache Spark pour lire et écrire avec des optimisations et des fonctionnalités spécifiques à Amazon Redshift, au-delà de celles disponibles lors de la connexion via les versions précédentes.

Découvrez comment AWS Glue facilite plus que jamais la migration des utilisateurs d'Amazon Redshift vers AWS Glue pour l'intégration des données sans serveur et l'ETL.

Configuration des connexions Redshift

Pour utiliser les clusters Amazon Redshift dans AWS Glue, vous devez remplir certaines conditions préalables :

- Un répertoire Amazon S3 à utiliser pour le stockage temporaire lors de la lecture et de l'écriture dans la base de données.
- Un Amazon VPC permettant la communication entre votre cluster Amazon Redshift, votre tâche AWS Glue et votre répertoire Amazon S3.
- Autorisations IAM appropriées sur le job AWS Glue et le cluster Amazon Redshift.

Configuration des rôles IAM

Configurer le rôle pour le cluster Amazon Redshift

Votre cluster Amazon Redshift doit être capable de lire et d'écrire sur Amazon S3 pour pouvoir s'intégrer aux tâches AWS Glue. Pour autoriser cela, vous pouvez associer des rôles IAM au cluster Amazon Redshift auquel vous souhaitez vous connecter. Votre rôle doit disposer d'une politique autorisant la lecture et l'écriture dans votre répertoire temporaire Amazon S3. Votre rôle doit avoir une relation de confiance permettant au service `redshift.amazonaws.com` de `AssumeRole`.

Pour associer un rôle IAM à Amazon Redshift

1. Conditions préalables : un compartiment ou un répertoire Amazon S3 utilisé pour le stockage temporaire de fichiers.

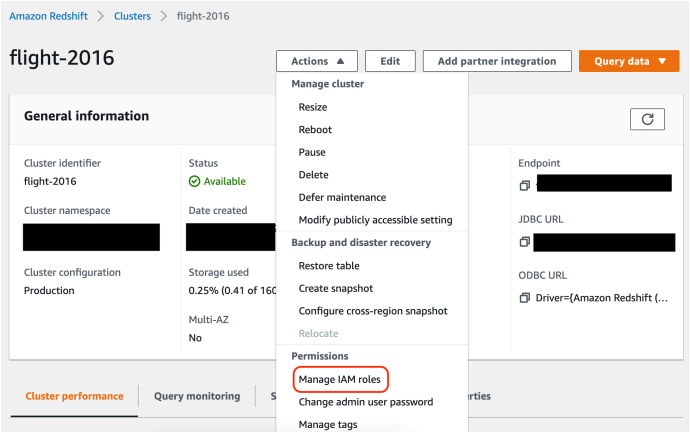
2. Identifiez les autorisations Amazon S3 dont votre cluster Amazon Redshift aura besoin. Lors du déplacement de données vers et depuis un cluster Amazon Redshift, les tâches AWS Glue émettent des instructions COPY et UNLOAD à l'encontre d'Amazon Redshift. Si votre tâche modifie une table dans Amazon Redshift, AWS Glue émettra également des instructions CREATE LIBRARY. Pour plus d'informations sur les autorisations Amazon S3 spécifiques requises pour qu'Amazon Redshift exécute ces instructions, consultez la documentation Amazon Redshift : Amazon Redshift : Permissions to access other Resources. AWS
3. Dans la console IAM, créez une politique IAM avec les autorisations nécessaires. Pour plus d'informations sur la création de politiques, consultez Création de politiques IAM.
4. Dans la console IAM, créez un rôle et une relation de confiance permettant à Amazon Redshift d'assumer ce rôle. Suivez les instructions de la documentation IAM pour créer un rôle pour un AWS service (console)

   - Lorsqu'on vous demande de choisir un cas d'utilisation du AWS service, choisissez « Redshift - Personnalisable ».
   - Lorsque vous êtes invité à joindre une politique, choisissez celle que vous avez définie précédemment.

   > **Note**
   >
   > Pour plus d'informations sur la configuration des rôles pour Amazon Redshift, consultez la section Autoriser Amazon Redshift à accéder à d'autres AWS services en votre nom dans la documentation Amazon Redshift.

5. Dans la console Amazon Redshift, associez le rôle à votre cluster Amazon Redshift. Suivez les instructions de la documentation Amazon Redshift.

   Sélectionnez l'option en surbrillance dans la console Amazon Redshift pour configurer ce paramètre :



> **Note**
>
> Par défaut, les tâches AWS Glue transmettent les informations d'identification temporaires Amazon Redshift créées à l'aide du rôle que vous avez spécifié pour exécuter la tâche. Nous vous déconseillons d'utiliser ces informations d'identification. Pour des raisons de sécurité, ces informations d'identification expirent au bout d'une heure.

Configuration du rôle pour la tâche AWS Glue

La tâche AWS Glue a besoin d'un rôle pour accéder au compartiment Amazon S3. Vous n'avez pas besoin d'autorisations IAM pour le cluster Amazon Redshift. Votre accès est contrôlé par la connectivité dans Amazon VPC et par les informations d'identification de votre base de données.

Configurer Amazon VPC

Pour configurer l'accès pour les magasins de données Amazon Redshift

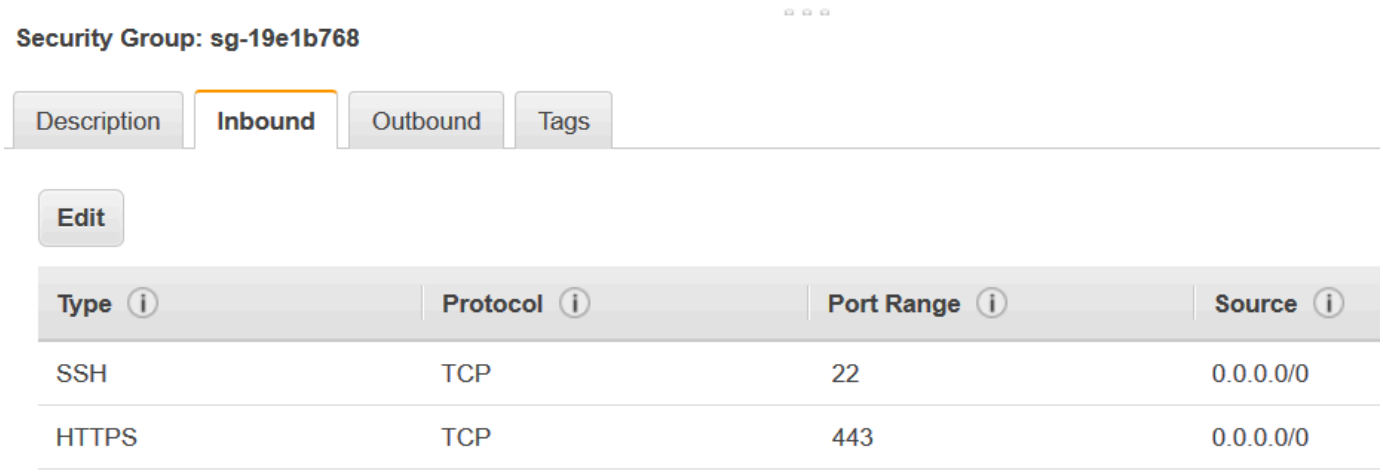
1. Connectez-vous à la console Amazon Redshift AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/redshiftv2/
2. Dans le volet de navigation de gauche, choisissez Clusters.
3. Choisissez le nom du cluster auquel vous souhaitez accéder à partir d'AWS Glue.
4. Dans la section Cluster Properties (Propriétés du cluster), choisissez un groupe de sécurité dans VPC security groups (Groupes de sécurité VPC) afin de permettre à AWS Glue de l'utiliser. Enregistrez le nom du groupe de sécurité que vous avez choisi pour référence ultérieure. Lorsque vous choisissez le groupe de sécurité, la liste des groupes de sécurité de la console Amazon EC2 s'ouvre.

5. Choisissez le groupe de sécurité à modifier et accédez à l'onglet Inbound (Entrant).
6. Ajoutez une règle avec référence circulaire afin de permettre aux composants AWS Glue de communiquer. Plus spécifiquement, ajoutez ou confirmez qu'il existe une règle Type `All TCP`, que Protocol (Protocole) est `TCP`, que Port Range (Plage de ports) comprend tous les ports et que la valeur de Source est le même nom de groupe de sécurité que la valeur de Group ID (ID du groupe).

   La règle entrante ressemble à ce qui suit :

Type	Protocole	Plage de ports	Source
Tous les TCP	TCP	0-65535	database-security-group

   Par exemple :



7. Ajoutez également une règle pour le trafic sortant. Ouvrez le trafic sortant pour tous les ports, par exemple :

Type	Protocole	Plage de ports	Destination
Tout le trafic	ALL	ALL	0.0.0.0/0

   Ou créez une règle avec référence circulaire où Type est `All TCP`, Protocol (Protocole) est `TCP`, Port Range (Plage de ports) comprend tous les ports et dont la valeur de Destination est

le même nom de groupe de sécurité que la valeur de Group ID (ID du groupe). Si vous utilisez un point de terminaison d'un VPC Amazon S3, ajoutez également une règle HTTPS pour l'accès à Amazon S3. Le *s3-prefix-list-id* est requis dans la règle du groupe de sécurité pour autoriser le trafic entre le VPC et le point de terminaison Amazon S3 VPC.

Par exemple :

Type	Protocole	Plage de ports	Destination
Tous les TCP	TCP	0-65535	*security-group*
HTTPS	TCP	443	*s3-prefix-list-id*

Configurer AWS Glue

Vous devez créer une connexion AWS Glue Data Catalog qui fournit les informations de connexion Amazon VPC.

Pour configurer la connectivité Amazon Redshift entre Amazon VPC et Glue dans la console AWS

1. Créez une connexion au catalogue de données en suivant les étapes décrites dans :the section called "Ajouter une AWS Glue connexion". Après avoir créé la connexion, conservez le nom de la connexion pour l'étape suivante. *connectionName*

   - Lorsque vous sélectionnez un type de connexion, sélectionnez Amazon Redshift.
   - Lorsque vous sélectionnez un cluster Redshift, sélectionnez le nom de votre cluster.
   - Fournissez les informations de connexion par défaut pour un utilisateur Amazon Redshift sur votre cluster.
   - Vos paramètres Amazon VPC seront automatiquement configurés.

   > **Note**
   >
   > Vous devrez fournir `PhysicalConnectionRequirements` manuellement pour votre Amazon VPC lorsque vous créez une connexion Amazon Redshift via le kit AWS SDK.

2. Dans la configuration de votre tâche AWS Glue*connectionName*, fournissez une connexion réseau supplémentaire.

Exemple : lecture à partir de tables Amazon Redshift

Vous pouvez lire à partir de clusters Amazon Redshift et des environnements Amazon Redshift sans serveur.

Conditions préalables : une table Amazon Redshift à partir de laquelle vous souhaitez lire. Suivez les étapes décrites dans la section précédente, the section called "Configurer Redshift" après quoi vous devriez avoir l'URI Amazon S3 pour un répertoire temporaire *temp-s3-dir* et un rôle IAM (dans le compte*role-account-id*). *rs-role-name*

Using the Data Catalog

Conditions préalables supplémentaires : une base de données et une table de catalogue de données pour la table Amazon Redshift à partir de laquelle vous souhaitez lire. Pour plus d'informations sur le catalogue de données, consultez Découverte et catalogage des données. Après avoir créé une entrée pour votre table Amazon Redshift, vous identifierez votre connexion à l'aide d'un *redshift-dc-database-name* et. *redshift-table-name*

Configuration : dans les options de votre fonction, vous identifierez votre table de catalogue de données avec les paramètres `database` et `table_name`. Vous identifierez votre répertoire temporaire Amazon S3 avec `redshift_tmp_dir`. Vous fournirez également *rs-role-name* en utilisant la `aws_iam_role` clé dans le `additional_options` paramètre.

```
 glueContext.create_dynamic_frame.from_catalog(
    database = "redshift-dc-database-name",
    table_name = "redshift-table-name",
    redshift_tmp_dir = args["temp-s3-dir"],
    additional_options = {"aws_iam_role": "arn:aws:iam::role-account-id:role/rs-
role-name"})
```

Connecting directly

Conditions préalables supplémentaires : vous aurez besoin du nom de votre table Amazon Redshift (. *redshift-table-name* Vous aurez besoin des informations de connexion JDBC

pour le cluster Amazon Redshift qui stocke cette table. Vous fournirez vos informations de connexion avec *hostport*,*redshift-database-name*, *username* et*password*.

Vous pouvez récupérer vos informations de connexion depuis la console Amazon Redshift lorsque vous travaillez avec des clusters Amazon Redshift. Pour l'utilisation d'Amazon Redshift sans serveur, consultez Connexion à Amazon Redshift sans serveur dans la documentation Amazon Redshift.

Configuration : dans les options de votre fonction, vous identifierez vos paramètres de connexions avec `url`, `dbtable`, `user` et `password`. Vous identifierez votre répertoire temporaire Amazon S3 avec `redshift_tmp_dir`. Vous pouvez spécifier votre rôle IAM en utilisant `aws_iam_role` lorsque vous utilisez `from_options`. La syntaxe est similaire à celle de la connexion via le catalogue de données, mais vous placez les paramètres sur la carte `connection_options`.

C'est une mauvaise pratique de coder en dur les mots de passe dans les scripts AWS Glue. Envisagez de stocker vos mots de passe AWS Secrets Manager et de les récupérer dans votre script avec le SDK for Python (Boto3).

```
my_conn_options = {
    "url": "jdbc:redshift://host:port/redshift-database-name",
    "dbtable": "redshift-table-name",
    "user": "username",
    "password": "password",
    "redshiftTmpDir": args["temp-s3-dir"],
    "aws_iam_role": "arn:aws:iam::account id:role/rs-role-name"
}

df = glueContext.create_dynamic_frame.from_options("redshift", my_conn_options)
```

Exemple : écrire sur des tables Amazon Redshift

Vous pouvez sur des clusters Amazon Redshift et des environnements Amazon Redshift sans serveur.

Conditions préalables : un cluster Amazon Redshift et suivez les étapes de la the section called "Configurer Redshift" section précédente, après quoi vous devriez avoir l'URI Amazon S3 pour un répertoire temporaire et un rôle IAM*temp-s3-dir*,*rs-role-name*, (dans le compte). *role-*

account-id Vous aurez également besoin d'un `DynamicFrame` dont vous souhaitez écrire le contenu dans la base de données.

Using the Data Catalog

Conditions préalables supplémentaires : une base de données de catalogue de données pour le cluster Amazon Redshift et la table sur laquelle vous souhaitez écrire. Pour plus d'informations sur le catalogue de données, consultez Découverte et catalogage des données. Vous allez identifier votre connexion avec *redshift-dc-database-name* et la table cible avec*redshift-table-name*.

Configuration : dans les options de votre fonction, vous identifierez votre base de données de catalogue de données avec les paramètres `database`, puis vous fournirez votre table avec `table_name`. Vous identifierez votre répertoire temporaire Amazon S3 avec `redshift_tmp_dir`. Vous fournirez également *rs-role-name* en utilisant la `aws_iam_role` clé dans le `additional_options` paramètre.

```
glueContext.write_dynamic_frame.from_catalog(
   frame = input dynamic frame,
   database = "redshift-dc-database-name",
   table_name = "redshift-table-name",
   redshift_tmp_dir = args["temp-s3-dir"],
   additional_options = {"aws_iam_role": "arn:aws:iam::account-id:role/rs-role-
name"})
```

Connecting through a AWS Glue connection

Vous pouvez vous connecter directement à Amazon Redshift à l'aide de la méthode `write_dynamic_frame.from_options`. Toutefois, plutôt que d'insérer vos informations de connexion directement dans votre script, vous pouvez référencer les informations de connexion stockés dans une connexion au catalogue de données à l'aide de la méthode `from_jdbc_conf`. Vous pouvez le faire sans indexation de site web ou sans créer de tables de catalogue de données pour votre base de données. Pour plus d'informations sur les connexions au catalogue de données, consultez Connexion aux données.

Conditions préalables supplémentaires : une base au catalogue de données pour votre base de données, une table Amazon Redshift à partir de laquelle vous souhaitez lire.

Configuration : vous allez identifier votre connexion au catalogue de données avec*dc-connection-name*. Vous identifierez votre base de données et votre table Amazon Redshift avec *redshift-table-name* et. *redshift-database-name* Vous fournirez les informations de connexion à votre catalogue de données avec `catalog_connection` et vos informations Amazon Redshift avec `dbtable` et `database`. La syntaxe est similaire à celle de la connexion via le catalogue de données, mais vous placez les paramètres sur la carte `connection_options`.

```
my_conn_options = {
    "dbtable": "redshift-table-name",
    "database": "redshift-database-name",
    "aws_iam_role": "arn:aws:iam::role-account-id:role/rs-role-name"
}

glueContext.write_dynamic_frame.from_jdbc_conf(
    frame = input dynamic frame,
    catalog_connection = "dc-connection-name",
    connection_options = my_conn_options,
    redshift_tmp_dir = args["temp-s3-dir"])
```

Référence des options de connexion Amazon Redshift

Les options de connexion de base utilisées pour toutes les connexions JDBC AWS Glue afin de configurer des informations telles que`url`, `user` et `password` sont cohérentes entre tous les types de JDBC. Pour plus d'informations sur les paramètres JDBC standards, consultez the section called "Paramètres de connexion JDBC".

Le type de connexion Amazon Redshift propose des options de connexion supplémentaires :

- `"redshiftTmpDir"` : (obligatoire) chemin Amazon S3 où les données temporaires peuvent être stockées lors de la copie à partir de la base de données.
- `"aws_iam_role"` : (facultatif) ARN pour un rôle IAM. La tâche AWS Glue transmettra ce rôle au cluster Amazon Redshift pour accorder au cluster les autorisations nécessaires pour exécuter les instructions de la tâche.

Options de connexion supplémentaires disponibles dans AWS Glue 4.0+

Vous pouvez également transmettre les options du nouveau connecteur Amazon Redshift via les options de connexion AWS Glue. Pour obtenir la liste complète des options de connecteur prises en charge, consultez la section relative aux paramètres SQL de Spark dans Intégration Amazon Redshift pour Apache Spark.

Pour vous faciliter la tâche, nous vous rappelons ici certaines nouvelles options :

Nom	Obligatoire	Par défaut	Description
autopushdown	Non	TRUE	Applique le pushdown des prédicats et des requêtes en capturant et en analysant les plans logiques de Spark pour les opérations SQL. Les opération s sont traduites en une requête SQL, puis exécutées dans Amazon Redshift pour améliorer les performances.
autopushdown.s3_result_cache	Non	FALSE	Met en cache la requête SQL pour décharger les données du mappage des chemins Amazon S3 en mémoire, afin que la même requête n'ait pas besoin de s'exécuter à nouveau dans la même session Spark. Pris

Nom	Obligatoire	Par défaut	Description
			en charge uniquement lorsqu'`autopushdown` est activé.
unload_s3_format	Non	PARQUET	PARQUET : décharge les résultats de la requête au format Parquet. TEXTE : décharge les résultats de la requête séparés par une barre verticale au format texte.
sse_kms_key	Non	N/A	La clé AWS SSE-KMS à utiliser pour le chiffrement pendant les `UNLOAD` opérations au lieu du chiffrement par défaut pour. AWS

Nom	Obligatoire	Par défaut	Description
extracopyoptions	Non	N/A	Une liste d'options supplémentaires à ajouter à la commande `COPY` d'Amazon Redshift lors du chargement de données, telles que `TRUNCATECOLUMNS` ou `MAXERROR n` (pour en savoir plus sur les autres options, voir COPY : paramètres facultatifs). Notez que ces options étant ajoutées à la fin de la commande `COPY`, seules des options qui ont du sens peuvent être utilisées à la fin de la commande. Cela devrait couvrir la plupart des cas d'utilisation possibles.
csvnullstring (expérimental)	Non	NULL	La valeur de chaîne à écrire pour les valeurs nulles lors de l'utilisation du `tempformat` CSV. Il doit s'agir d'une valeur qui n'apparaît pas dans vos données réelles.

Ces nouveaux paramètres peuvent être utilisés comme suit.

Nouvelles options destinées à améliorer les performances

Le nouveau connecteur intègre de nouvelles options d'amélioration des performances :

- `autopushdown` : activée par défaut.
- `autopushdown.s3_result_cache` : désactivée par défaut.
- `unload_s3_format` : `PARQUET` par défaut.

Pour plus d'informations sur l'utilisation de ces options, consultez Intégration Amazon Redshift pour Apache Spark. Nous vous recommandons de ne pas activer `autopushdown.s3_result_cache` lorsque vous effectuez des opérations mixtes (lecture et écriture), car les résultats mis en cache peuvent contenir des informations périmées. L'option `unload_s3_format` est définie sur `PARQUET` par défaut pour la commande `UNLOAD`, afin d'améliorer les performances et de réduire les coûts de stockage. Pour utiliser le comportement par défaut de la commande `UNLOAD`, réinitialisez l'option sur `TEXT`.

Nouvelle option de chiffrement pour la lecture

Par défaut, les données du dossier temporaire que AWS Glue utilise lors de la lecture des données de la table Amazon Redshift sont chiffrées à l'aide du chiffrement `SSE-S3`. Pour utiliser les clés gérées par le client à partir de AWS Key Management Service (AWS KMS) pour chiffrer vos données, vous pouvez définir d'`("sse_kms_key" # kmsKey)`où KSMKey est l'ID de clé AWS KMS, au lieu de l'ancienne option `("extraunloadoptions" # s"ENCRYPTED KMS_KEY_ID '$kmsKey'")` de configuration de la version 3.0. AWS Glue

```
datasource0 = glueContext.create_dynamic_frame.from_catalog(
  database = "database-name",
  table_name = "table-name",
  redshift_tmp_dir = args["TempDir"],
  additional_options = {"sse_kms_key":"<KMS_KEY_ID>"},
  transformation_ctx = "datasource0"
)
```

Prise en charge de l'URL JDBC basée sur IAM

Le nouveau connecteur prend en charge une URL JDBC basée sur IAM, vous n'avez donc pas besoin de transmettre un code secret. user/password Avec une URL JDBC basée sur IAM, le

connecteur utilise le rôle d'exécution des tâches pour accéder à la source de données Amazon Redshift.

Étape 1 : associez la politique minimale requise suivante à votre rôle d'exécution de tâches AWS Glue.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": "redshift:GetClusterCredentials",
            "Resource": [
                "arn:aws:redshift:us-east-1:111122223333:dbgroup:<cluster name>/
*",
                "arn:aws:redshift:*:111122223333:dbuser:*/*",
                "arn:aws:redshift:us-east-1:111122223333:dbname:<cluster name>/
<database name>"
            ]
        },
        {
            "Sid": "VisualEditor1",
            "Effect": "Allow",
            "Action": "redshift:DescribeClusters",
            "Resource": "*"
        }
    ]
}
```

Étape 2 : utilisez l'URL JDBC basée sur IAM comme suit. Spécifiez une nouvelle option `DbUser` avec le nom d'utilisateur Amazon Redshift qui permet de vous connecter.

```
conn_options = {
     // IAM-based JDBC URL
    "url": "jdbc:redshift:iam://<cluster name>:<region>/<database name>",
    "dbtable": dbtable,
    "redshiftTmpDir": redshiftTmpDir,
    "aws_iam_role": aws_iam_role,
```

```
    "DbUser": "<Redshift User name>" // required for IAM-based JDBC URL
    }

redshift_write = glueContext.write_dynamic_frame.from_options(
    frame=dyf,
    connection_type="redshift",
    connection_options=conn_options
)

redshift_read = glueContext.create_dynamic_frame.from_options(
    connection_type="redshift",
    connection_options=conn_options
)
```

> **Note**
>
> `DynamicFrame` ne prend actuellement en charge qu'une URL JDBC basée sur IAM avec un `DbUser` dans le flux de travail `GlueContext.create_dynamic_frame.from_options`.

Migration de AWS Glue version 3.0 vers la version 4.0

Dans AWS Glue 4.0, les tâches ETL ont accès à un nouveau connecteur Amazon Redshift Spark et à un nouveau pilote JDBC avec différentes options et configurations. Le nouveau connecteur et le nouveau pilote Amazon Redshift ont été conçus dans un souci de performance et garantissent la cohérence transactionnelle de vos données. Ces produits sont décrits dans la documentation Amazon Redshift. Pour en savoir plus, consultez :

- Intégration Amazon Redshift pour Apache Spark
- Pilote Amazon Redshift JDBC, version 2.1

Restriction portant sur les noms et les identifiants des tables et des colonnes

Concernant le nom de la table Redshift, les exigences du nouveau connecteur et du nouveau pilote Amazon Redshift Spark sont plus restreintes. Pour plus d'informations, consultez la section Noms et identifiants qui explique comment définir le nom de votre table Amazon Redshift. Le flux de travail des signets de tâches peut ne pas fonctionner avec un nom de table non conforme aux règles et avec certains caractères (l'espace, par exemple).

Si vous utilisez d'anciennes tables dont les noms ne sont pas conformes aux règles relatives aux noms et identifiants et que vous rencontrez des problèmes liés aux signets (tâches dédiées au retraitement des données d'anciennes tables Amazon Redshift), nous vous recommandons de renommer vos tables. Pour plus d'informations, consultez les exemples d'utilisation de la commande ALTER TABLE.

Modification de tempformat par défaut dans le Dataframe

Le connecteur Spark de AWS Glue version 3.0 définit par défaut le `tempformat` sur CSV lors de l'écriture dans Amazon Redshift. Par souci de cohérence, dans AWS Glue version 3.0, `DynamicFrame` définit toujours par défaut le `tempformat` de manière à utiliser le format `CSV`. Si vous avez déjà utilisé Spark Dataframe APIs directement avec le connecteur Amazon Redshift Spark, vous pouvez définir explicitement le format CSV dans `tempformat` `DataframeReader` les options/. `Writer` Sinon, `tempformat` est défini par défaut sur `AVRO` dans le nouveau connecteur Spark.

Changement de comportement : mapper le type de données Amazon Redshift REAL sur le type de données FLOAT de Spark au lieu de DOUBLE

Dans AWS Glue version 3.0, Amazon Redshift `REAL` est converti en type `DOUBLE` de Spark. Le nouveau connecteur Amazon Redshift Spark a mis à jour le comportement afin que le type Amazon Redshift `REAL` soit converti en type `FLOAT` de Spark et inversement. Si, conformément à un ancien cas d'utilisation, vous souhaitez toujours que le type `REAL` Amazon Redshift soit mappé sur un type Spark `DOUBLE`, vous pouvez utiliser la solution alternative suivante :

- Pour un `DynamicFrame`, mappez le type `Float` sur un type `Double` avec `DynamicFrame.ApplyMapping`. Pour un `Dataframe`, vous devez utiliser `cast`.

Exemple de code :

```
dyf_cast = dyf.apply_mapping([('a', 'long', 'a', 'long'), ('b', 'float', 'b',
 'double')])
```

Gestion du type de données VARBYTE

Lorsque vous travaillez avec les types de données AWS Glue 3.0 et Amazon Redshift, AWS Glue 3.0 convertit Amazon `VARBYTE` Redshift en type Spark. `STRING` Cependant, le dernier connecteur Amazon Redshift Spark ne prend pas en charge le type de données `VARBYTE`. Pour contourner cette restriction, vous pouvez créer une vue Redshift qui transforme les colonnes `VARBYTE` en type de données compatible. Utilisez ensuite le nouveau connecteur pour charger les données à partir de

cette vue au lieu de la table d'origine, ce qui garantit la compatibilité tout en maintenant l'accès à vos données VARBYTE.

Exemple de requête Redshift :

```
CREATE VIEW view_name AS SELECT FROM_VARBYTE(varbyte_column, 'hex') FROM table_name
```

Connexions Kafka

Vous pouvez utiliser une connexion Kafka pour lire et écrire dans des flux de données Kafka à l'aide d'informations stockées dans une table de catalogue de données ou en fournissant des informations permettant d'accéder directement au flux de données. La connexion prend en charge un cluster Kafka ou un cluster Amazon Managed Streaming for Apache Kafka. Vous pouvez lire les informations de Kafka dans un Spark DataFrame, puis les convertir en AWS Glue DynamicFrame. Vous pouvez DynamicFrames écrire dans Kafka au format JSON. Si vous accédez directement au flux de données, utilisez ces options pour fournir des informations sur la façon d'accéder au flux de données.

Si vous utilisez `getCatalogSource` ou `create_data_frame_from_catalog` pour consommer des enregistrements à partir d'une source de streaming Kafka, ou `getCatalogSink` ou `write_dynamic_frame_from_catalog` pour écrire des enregistrements dans Kafka, et que la tâche dispose des informations relatives à la base de données Data Catalog et au nom de la table, et peut les utiliser pour obtenir certains paramètres de base pour la lecture à partir de la source de streaming Kafka. Si vous utilisez `getSource`, `getCatalogSink`, `getSourceWithFormat`, `getSinkWithFormat`, `createDataFrameFromOptions`, `create_data_frame_from_options` ou `write_dynamic_frame_from_catalog`, vous devez spécifier ces paramètres de base à l'aide des options de connexion décrites ici.

Vous pouvez spécifier les options de connexion pour Kafka à l'aide des arguments suivants pour les méthodes spécifiées dans la classe `GlueContext`.

- Scala
  - `connectionOptions` : utiliser avec `getSource`, `createDataFrameFromOptions`, `getSink`
  - `additionalOptions` : utiliser avec `getCatalogSource`, `getCatalogSink`
  - `options` : utiliser avec `getSourceWithFormat`, `getSinkWithFormat`
- Python
  - `connection_options` : utiliser avec `create_data_frame_from_options`, `write_dynamic_frame_from_options`

- `additional_options` : utiliser avec `create_data_frame_from_catalog`, `write_dynamic_frame_from_catalog`
- `options` : utiliser avec `getSource`, `getSink`

Pour les remarques et les restrictions concernant les tâches ETL de streaming, consultez the section called "Restrictions et notes sur ETL en streaming".

Rubriques

- Configurer Kafka
- Exemple : lecture à partir de flux Kafka
- Exemple : écriture dans les flux Kafka
- Référence des options de connexion de Kafka

Configurer Kafka

Il n'y a aucune AWS condition préalable pour se connecter aux flux Kafka disponibles sur Internet.

Vous pouvez créer une connexion AWS Glue Kafka pour gérer vos identifiants de connexion. Pour de plus amples informations, veuillez consulter the section called "Création d'une connexion pour un flux de données Kafka". Dans la configuration de votre tâche AWS Glue, fournissez *`connectionName`* une connexion réseau supplémentaire, puis, dans votre appel de méthode, fournissez *`connectionName`* le `connectionName` paramètre.

Dans certains cas, vous devrez configurer des prérequis supplémentaires :

- Si vous utilisez Amazon Managed Streaming pour Apache Kafka avec l'authentification IAM, vous aurez besoin d'une configuration IAM appropriée.
- Si vous utilisez Amazon Managed Streaming pour Apache Kafka avec un Amazon VPC, vous aurez besoin d'une configuration Amazon VPC appropriée. Vous devez créer une connexion AWS Glue fournissant les informations de connexion Amazon VPC. Vous aurez besoin de la configuration de votre tâche pour inclure la connexion AWS Glue en tant que connexion réseau supplémentaire.

Pour plus d'informations sur les prérequis de la tâche ETL de streaming, consultez the section called "Tâches ETL en streaming".

Exemple : lecture à partir de flux Kafka

Utilisez conjointement avec the section called "forEachBatch".

Exemple pour la source de streaming Kafka :

```
kafka_options =
    { "connectionName": "ConfluentKafka",
      "topicName": "kafka-auth-topic",
      "startingOffsets": "earliest",
      "inferSchema": "true",
      "classification": "json"
    }
data_frame_datasource0 =
 glueContext.create_data_frame.from_options(connection_type="kafka",
 connection_options=kafka_options)
```

Exemple : écriture dans les flux Kafka

Exemples pour écrire dans Kafka :

Exemple avec la méthode `getSink` :

```
data_frame_datasource0 =
glueContext.getSink(
 connectionType="kafka",
 connectionOptions={
  JsonOptions("""{
   "connectionName": "ConfluentKafka",
   "classification": "json",
   "topic": "kafka-auth-topic",
   "typeOfData": "kafka"}
  """)},
 transformationContext="dataframe_ApacheKafka_node1711729173428")
 .getDataFrame()
```

Exemple avec la méthode `write_dynamic_frame.from_options` :

```
kafka_options =
    { "connectionName": "ConfluentKafka",
      "topicName": "kafka-auth-topic",
      "classification": "json"
    }
```

```
data_frame_datasource0 =
 glueContext.write_dynamic_frame.from_options(connection_type="kafka",
 connection_options=kafka_options)
```

Référence des options de connexion de Kafka

Lors de la lecture, utilisez les options de connexion suivantes avec `"connectionType": "kafka"` :

- `"bootstrap.servers"`(Obligatoire) Une liste de serveurs bootstrap URLs, par exemple, commeb-1.vpc-test-2.o4q88o.c6.kafka.us-east-1.amazonaws.com:9094. Cette option doit être spécifiée dans l'appel d'API ou définie dans les métadonnées de la table dans le catalogue de données.
- `"security.protocol"`(Obligatoire) Le protocole utilisé pour communiquer avec les agents. Les valeurs possibles sont `"SSL"` ou `"PLAINTEXT"`.
- `"topicName"` : (requis) liste de rubriques séparées par des virgules auxquelles s'abonner. Vous devez spécifier un seul et unique élément parmi `"topicName"`, `"assign"` ou `"subscribePattern"`.
- `"assign"` : (requis) chaîne JSON indiquant le `TopicPartitions` spécifique à utiliser. Vous devez spécifier un seul et unique élément parmi `"topicName"`, `"assign"` ou `"subscribePattern"`.

  Exemple : '{"topicA":[0,1],"topicB":[2,4]}'
- `"subscribePattern"` : (obligatoire) chaîne d'expression rationnelle Java qui identifie la liste de rubriques à laquelle vous souhaitez vous abonner. Vous devez spécifier un seul et unique élément parmi `"topicName"`, `"assign"` ou `"subscribePattern"`.

  Exemple : 'topic.*'
- `"classification"` : (obligatoire) le format de fichier utilisé par les données de l'enregistrement. Obligatoire, sauf s'il est fourni par le catalogue de données.
- `"delimiter"` : (facultatif) le séparateur de valeurs utilisé lorsque `classification` est CSV. La valeur par défaut est « `,` ».
- `"startingOffsets"` : (facultatif) position de départ dans la rubrique Kafka à partir de laquelle lire les données. Les valeurs possibles sont `"earliest"` ou `"latest"`. La valeur par défaut est `"latest"`.
- `"startingTimestamp"`: (Facultatif, disponible uniquement pour AWS la version 4.0 ou ultérieure de Glue) Horodatage de l'enregistrement dans la rubrique Kafka à partir duquel les données

doivent être lues. La valeur possible est une chaîne d'horodatage au format UTC dans le modèle `yyyy-mm-ddTHH:MM:SSZ` (où `Z` représente un décalage de fuseau horaire UTC avec un +/-. Par exemple : « 2023-04-04T08:00:00-04:00 »).

Remarque : seule l'une des propriétés « StartingOffsets » ou « StartingTimestamp » peut figurer dans la liste des options de connexion du script de streaming AWS Glue. L'inclusion de ces deux propriétés entraînera l'échec de la tâche.

- `"endingOffsets"` : (facultatif) point de fin lorsqu'une requête par lots est terminée. Les valeurs possibles sont `"latest"` ou une chaîne JSON qui spécifie un décalage de fin pour chaque `TopicPartition`.

  Pour la chaîne JSON, le format est `{"topicA":{"0":23,"1":-1},"topicB":{"0":-1}}`. La valeur `-1` en tant que décalage représente `"latest"`.
- `"pollTimeoutMs"` : (facultatif) délai d'attente en millisecondes pour interroger les données de Kafka dans les exécuteurs de tâches Spark. La valeur par défaut est `600000`.
- `"numRetries"` : (facultatif) nombre de nouvelles tentatives avant de ne pas récupérer les décalages Kafka. La valeur par défaut est `3`.
- `"retryIntervalMs"` : (facultatif) temps d'attente en millisecondes avant d'essayer de récupérer les décalages Kafka. La valeur par défaut est `10`.
- `"maxOffsetsPerTrigger"` : (facultatif) limite de taux sur le nombre maximal de décalages qui sont traités par intervalle de déclenchement. Le nombre total spécifié de décalages est réparti proportionnellement entre les `topicPartitions` des différents volumes. La valeur par défaut est null, ce qui signifie que le consommateur lit tous les décalages jusqu'au dernier décalage connu.
- `"minPartitions"` : (facultatif) nombre minimum de partitions à lire à partir de Kafka. La valeur par défaut est nulle, ce qui signifie que le nombre de partitions Spark est égal au nombre de partitions Kafka.
- `"includeHeaders"`: (facultatif) indique s'il faut inclure les en-têtes Kafka. Lorsque l'option est définie sur « true » (vrai), la sortie de données contiendra une colonne supplémentaire nommée « glue_streaming_kafka_headers » avec le type `Array[Struct(key: String, value: String)]`. La valeur définie par défaut est « false ». Cette option est disponible dans AWS Glue version 3.0 ou ultérieure.
- `"schema"` : (requis lorsque inferSchema est défini sur false) schéma à utiliser pour traiter la charge utile. Si la classification est `avro`, le schéma fourni doit être au format de schéma Avro. Si la classification n'est pas `avro`, le schéma fourni doit être au format de schéma DDL.

  Voici quelques exemples de schémas.

Example in DDL schema format

```
'column1' INT, 'column2' STRING , 'column3' FLOAT
```

Example in Avro schema format

```
{
"type":"array",
"items":
{
"type":"record",
"name":"test",
"fields":
[
  {
    "name":"_id",
    "type":"string"
  },
  {
    "name":"index",
    "type":
    [
      "int",
      "string",
      "float"
    ]
  }
]
}
}
```

- `"inferSchema"` : (facultatif) la valeur par défaut est « false ». S'il est défini sur « true », le schéma sera détecté lors de l'exécution à partir de la charge utile dans `foreachbatch`.
- `"avroSchema"` : (obsolète) paramètre utilisé pour spécifier un schéma de données Avro lorsque le format Avro est utilisé. Ce paramètre est désormais obsolète. Utilisez le paramètre `schema`.
- `"addRecordTimestamp"` : (facultatif) lorsque cette option est définie sur « true », la sortie de données contient une colonne supplémentaire nommée « __src_timestamp » qui indique l'heure à laquelle l'enregistrement correspondant est reçu par la rubrique. La valeur par défaut est « false ». Cette option est prise en charge dans AWS Glue version 4.0 ou ultérieure.

- `"emitConsumerLagMetrics"`: (Facultatif) Lorsque l'option est définie sur « vrai », pour chaque lot, elle émet les métriques correspondant à la durée comprise entre le plus ancien enregistrement reçu par le sujet et le moment où il arrive CloudWatch. AWS Glue Le nom de la métrique est « glue.driver.streaming ». maxConsumerLagInMs». La valeur par défaut est « false ». Cette option est prise en charge dans AWS Glue version 4.0 ou ultérieure.

Lors de l'écriture, utilisez les options de connexion suivantes avec `"connectionType": "kafka"` :

- `"connectionName"`(Obligatoire) Nom de la connexion AWS Glue utilisée pour se connecter au cluster Kafka (similaire à la source Kafka).
- `"topic"` (Obligatoire) Si une colonne de rubrique existe, sa valeur est utilisée comme rubrique lors de l'écriture de la ligne donnée dans Kafka, sauf si l'option de configuration de la rubrique est définie. C'est-à-dire que l'option de configuration `topic` remplace la colonne de rubrique.
- `"partition"` (Facultatif) Si un numéro de partition valide est spécifié, cette `partition` sera utilisée lors de l'envoi de l'enregistrement.

  Si aucune partition n'est spécifiée, mais qu'une `key` est présente, une partition sera choisie en utilisant le hachage de la clé.

  Si ni une `key`, ni une `partition` ne sont présentes, une partition sera choisie en partitionnant de manière permanente ces modifications lorsqu'au moins des octets batch.size seront produits sur la partition.
- `"key"` (Facultatif) Utilisé pour le partitionnement si `partition` est nul.
- `"classification"` (Facultatif) Format de fichier utilisé par les données de l'enregistrement. Nous prenons uniquement en charge les formats JSON, CSV et Avro.

  Avec le format Avro, nous pouvons fournir une valeur avroSchema personnalisée avec laquelle il est possible de sérialiser, mais notez que cela doit également être fourni sur la source pour la désérialisation. Sinon, par défaut, il utilise Apache AvroSchema pour la sérialisation.

En outre, vous pouvez optimiser le récepteur Kafka selon les besoins en mettant à jour les paramètres de configuration du producteur Kafka. Notez qu'il n'existe aucune liste d'autorisation pour les options de connexion, toutes les paires clé-valeur sont conservées telles quelles sur le récepteur.

Cependant, il existe une petite liste de refus d'options qui ne prendra pas effet. Pour plus d'informations, consultez Kafka specific configurations.

Connexions Azure Cosmos DB

Vous pouvez utiliser AWS Glue for Spark pour lire et écrire dans des conteneurs existants dans Azure Cosmos DB à l'aide de l'API NoSQL de AWS Glue 4.0 et des versions ultérieures. Vous pouvez définir ce qu'il faut lire dans Azure Cosmos DB à l'aide d'une requête SQL. Vous vous connectez à Azure Cosmos DB à l'aide d'une clé Azure Cosmos DB stockée AWS Secrets Manager via une connexion AWS Glue.

Pour plus d'informations sur Azure Cosmos DB pour NoSQL, consultez la documentation Azure.

Configuration des connexions Azure Cosmos DB

Pour vous connecter à Azure Cosmos DB depuis AWS Glue, vous devez créer et stocker votre clé de base de données Azure Cosmos dans un AWS Secrets Manager secret, puis associer ce secret à une connexion Azure Cosmos DB AWS Glue.

Prérequis :

- Dans Azure, vous devrez identifier ou générer une clé de base de données Azure Cosmos à utiliser par AWS Glue,`cosmosKey`. Pour plus d'informations, consultez la section Accès sécurisé aux données dans Azure Cosmos DB dans la documentation Azure.

Pour configurer une connexion à Azure Cosmos DB :

1. Dans AWS Secrets Manager, créez un secret à l'aide de votre clé de base de données Azure Cosmos. Pour créer un secret dans Secrets Manager, suivez le didacticiel disponible dans la section Créer un AWS Secrets Manager secret dans la AWS Secrets Manager documentation. Après avoir créé le secret, conservez le nom du secret *`secretName`* pour l'étape suivante.

   - Lorsque vous sélectionnez des paires clé/valeur, créez une paire pour la clé `spark.cosmos.accountKey` contenant la valeur. *`cosmosKey`*

2. Dans la console AWS Glue, créez une connexion en suivant les étapes décrites dansthe section called "Ajouter une AWS Glue connexion". Après avoir créé la connexion, conservez le nom de la connexion pour une utilisation future dans AWS Glue. *`connectionName`*

   - Lorsque vous sélectionnez un type de connexion, sélectionnez Azure Cosmos DB.
   - Lorsque vous sélectionnez un AWS secret, fournissez*`secretName`*.

Après avoir créé une connexion AWS Glue Azure Cosmos DB, vous devez effectuer les étapes suivantes avant d'exécuter votre tâche AWS Glue :

- Accordez au rôle IAM associé à votre tâche AWS Glue l'autorisation de lire*secretName*.
- Dans la configuration de votre tâche AWS Glue*connectionName*, fournissez une connexion réseau supplémentaire.

Lire à partir de la Azure Cosmos DB pour les conteneurs NoSQL

Prérequis :

- Un conteneur Azure Cosmos DB pour NoSQL à partir duquel vous souhaitez lire. Vous aurez besoin des informations d'identification du conteneur.

  Un conteneur Azure Cosmos pour NoSQL est identifié par sa base de données et son conteneur. Vous devez fournir les noms de base de données et de conteneur lorsque vous vous connectez à l'API Azure Cosmos pour NoSQL. *cosmosDBName* *cosmosContainerName*
- Une connexion AWS Glue Azure Cosmos DB configurée pour fournir des informations d'authentification et de localisation réseau. Pour cela, suivez les étapes de la procédure précédente, Pour configurer une connexion à Azure Cosmos DB. Vous aurez besoin du nom de la connexion AWS Glue,*connectionName*.

Par exemple :

```
azurecosmos_read = glueContext.create_dynamic_frame.from_options(
    connection_type="azurecosmos",
    connection_options={
    "connectionName": connectionName,
    "spark.cosmos.database": cosmosDBName,
    "spark.cosmos.container": cosmosContainerName,
    }
)
```

Vous pouvez également fournir une requête SQL SELECT pour filtrer les résultats renvoyés à votre DynamicFrame. Vous devrez configurer `query`.

Par exemple :

```
azurecosmos_read_query = glueContext.create_dynamic_frame.from_options(
```

```
    connection_type="azurecosmos",
    connection_options={
        "connectionName": "connectionName",
        "spark.cosmos.database": cosmosDBName,
        "spark.cosmos.container": cosmosContainerName,
        "spark.cosmos.read.customQuery": "query"
    }
)
```

Écrire dans Azure Cosmos DB pour les conteneurs NoSQL

Cet exemple écrit des informations à partir d'une base de données existante DynamicFrame *dynamicFrame* dans Azure Cosmos DB. Si le conteneur contient déjà des informations, AWS Glue ajoutera les données de votre DynamicFrame. Si les informations contenues dans le conteneur ont un schéma différent de celui des informations que vous écrivez, vous rencontrerez des erreurs.

Prérequis :

- Une table Azure Cosmos DB dans laquelle vous souhaitez écrire. Vous aurez besoin des informations d'identification du conteneur. Vous devez créer le conteneur avant d'appeler la méthode de connexion.

  Un conteneur Azure Cosmos pour NoSQL est identifié par sa base de données et son conteneur. Vous devez fournir les noms de base de données et de conteneur lorsque vous vous connectez à l'API Azure Cosmos pour NoSQL. *cosmosDBName cosmosContainerName*
- Une connexion AWS Glue Azure Cosmos DB configurée pour fournir des informations d'authentification et de localisation réseau. Pour cela, suivez les étapes de la procédure précédente, Pour configurer une connexion à Azure Cosmos DB. Vous aurez besoin du nom de la connexion AWS Glue,*connectionName*.

Par exemple :

```
azurecosmos_write = glueContext.write_dynamic_frame.from_options(
    frame=dynamicFrame,
    connection_type="azurecosmos",
    connection_options={
    "connectionName": connectionName,
    "spark.cosmos.database": cosmosDBName,
    "spark.cosmos.container": cosmosContainerName
)
```

Référence des options de connexion Azure Cosmos DB

- `connectionName` — Obligatoire. Utilisé pour la lecture/l'écriture. Nom d'une connexion AWS Glue Azure Cosmos DB configurée pour fournir des informations d'authentification et de localisation réseau à votre méthode de connexion.
- `spark.cosmos.database` — Obligatoire. Utilisé pour la lecture/l'écriture. Valeurs valides : noms des bases de données. Nom de la base de données Azure Cosmos DB pour NoSQL.
- `spark.cosmos.container` — Obligatoire. Utilisé pour la lecture/l'écriture. Valeurs valides : noms des conteneurs. Nom du conteneur Azure Cosmos DB pour NoSQL.
- `spark.cosmos.read.customQuery` : utilisé pour la lecture. Valeurs valides : requêtes SELECT SQL. Requête personnalisée pour sélectionner les documents à lire.

Connexions Azure SQL

Vous pouvez utiliser AWS Glue for Spark pour lire et écrire dans des tables sur des instances gérées Azure SQL dans AWS Glue 4.0 et versions ultérieures. Vous pouvez définir ce qu'il faut lire dans Azure SQL à l'aide d'une requête SQL. Vous vous connectez à Azure SQL à l'aide des informations d'identification d'utilisateur et de mot de passe stockées AWS Secrets Manager via une connexion AWS Glue.

Pour plus d'informations sur Azure SQL, consultez la documentation Azure SQL.

Configuration des connexions Azure SQL

Pour vous connecter à Azure SQL depuis AWS Glue, vous devez créer et stocker vos informations d'identification Azure SQL dans un AWS Secrets Manager secret, puis associer ce secret à une connexion Azure SQL AWS Glue.

Pour configurer une connexion à Azure SQL :

1. Dans AWS Secrets Manager, créez un secret à l'aide de vos informations d'identification Azure SQL. Pour créer un secret dans Secrets Manager, suivez le didacticiel disponible dans la section Créer un AWS Secrets Manager secret dans la AWS Secrets Manager documentation. Après avoir créé le secret, conservez le nom du secret *`secretName`* pour l'étape suivante.

   - Lorsque vous sélectionnez des paires clé/valeur, créez une paire pour la clé `user` contenant la valeur. *`azuresqlUsername`*

- Lorsque vous sélectionnez des paires clé/valeur, créez une paire pour la clé `password` contenant la valeur. *`azuresqlPassword`*

2. Dans la console AWS Glue, créez une connexion en suivant les étapes décrites dansthe section called "Ajouter une AWS Glue connexion". Après avoir créé la connexion, conservez le nom de la connexion pour une utilisation future dans AWS Glue. *`connectionName`*

   - Lorsque vous sélectionnez un type de connexion, sélectionnez Azure SQL.
   - Lorsque vous fournissez une URL Azure SQL, fournissez une URL de point de terminaison JDBC.

     La URL doit avoir le format suivant :
     `jdbc:sqlserver://`*`databaseServerName`*`:`*`databasePort`*`;databaseName=`*`azuresqlDBnam`*

     AWS Glue nécessite les propriétés d'URL suivantes :

     - `databaseName` – une base de données par défaut dans Azure SQL à laquelle se connecter.

     Pour plus d'informations sur JDBC URLs pour Azure SQL Managed Instances, consultez la documentation Microsoft.
   - Lorsque vous sélectionnez un AWS secret, indiquez-le*`secretName`*.

Après avoir créé une connexion AWS Glue Azure SQL, vous devez effectuer les étapes suivantes avant d'exécuter votre tâche AWS Glue :

- Accordez au rôle IAM associé à votre tâche AWS Glue l'autorisation de lire*`secretName`*.
- Dans la configuration de votre tâche AWS Glue*`connectionName`*, fournissez une connexion réseau supplémentaire.

Lire à partir de tables SQL Azure

Prérequis :

- Une table Azure SQL à partir de laquelle vous souhaitez lire. Vous aurez besoin d'informations d'identification pour la table, *`databaseName`* et*`tableIdentifier`*.

  Une table Azure SQL est identifiée par sa base de données, son schéma et son nom de table. Vous devez fournir le nom de la base de données et le nom de la table lorsque vous vous

connectez à Azure SQL. Vous devez également fournir le schéma s'il n'est pas « public » par défaut. La base de données est fournie via une propriété d'URL dans*connectionName*, un schéma et un nom de table via ledbtable.

- Une connexion SQL AWS Glue Azure configurée pour fournir des informations d'authentification. Suivez les étapes de la procédure précédente Pour configurer une connexion à Azure SQL afin de configurer vos informations d'authentification. Vous aurez besoin du nom de la connexion AWS Glue,*connectionName*.

Par exemple :

```
azuresql_read_table = glueContext.create_dynamic_frame.from_options(
    connection_type="azuresql",
    connection_options={
        "connectionName": "connectionName",
        "dbtable": "tableIdentifier"
    }
)
```

Vous pouvez également fournir une requête SQL SELECT pour filtrer les résultats renvoyés à votre DynamicFrame. Vous devrez configurer `query`.

Par exemple :

```
azuresql_read_query = glueContext.create_dynamic_frame.from_options(
    connection_type="azuresql",
    connection_options={
        "connectionName": "connectionName",
        "query": "query"
    }
)
```

Écrire dans des tables Azure SQL

Cet exemple écrit des informations à partir d'un DynamicFrame fichier existant *dynamicFrame* dans Azure SQL. Si le tableau contient déjà des informations, AWS Glue ajoutera les données de votre DynamicFrame.

Prérequis :

- Une table Azure SQL dans laquelle vous souhaitez écrire. Vous aurez besoin d'informations d'identification pour la table, *databaseName* et*tableIdentifier*.

  Une table Azure SQL est identifiée par sa base de données, son schéma et son nom de table. Vous devez fournir le nom de la base de données et le nom de la table lorsque vous vous connectez à Azure SQL. Vous devez également fournir le schéma s'il n'est pas « public » par défaut. La base de données est fournie via une propriété d'URL dans*connectionName*, un schéma et un nom de table via ledbtable.
- Informations d'authentification Azure SQL. Suivez les étapes de la procédure précédente Pour configurer une connexion à Azure SQL afin de configurer vos informations d'authentification. Vous aurez besoin du nom de la connexion AWS Glue,*connectionName*.

Par exemple :

```
azuresql_write = glueContext.write_dynamic_frame.from_options(
    connection_type="azuresql",
    connection_options={
        "connectionName": "connectionName",
        "dbtable": "tableIdentifier"
    }
)
```

Référence des options de connexion Azure SQL

- connectionName — Obligatoire. Utilisé pour la lecture/l'écriture. Le nom d'une connexion SQL AWS Glue Azure configurée pour fournir des informations d'authentification à votre méthode de connexion.
- databaseName : utilisé pour la lecture/l'écriture. Valeurs valides : noms des bases de données Azure SQL. Le nom de la base de données dans Azure SQL à laquelle se connecter.
- dbtable — obligatoire pour l'écriture, obligatoire pour la lecture à moins qu'une query ne soit fournie. Utilisé pour les combinaisons de Read/Write. Valid Values: Names of Azure SQL tables, or period separated schema/table noms. Utilisées pour spécifier la table et le schéma qui identifient la table à laquelle se connecter. Le schéma par défaut est « public ». Si votre table n'est pas dans un schéma par défaut, fournissez ces informations dans le formulaire *schemaName*.*tableName*.
- query : utilisé pour la lecture. Une requête Transact-SQL SELECT définissant ce qui doit être récupéré lors de la lecture à partir d'Azure SQL. Pour plus d'informations, consultez la documentation Microsoft.

BigQuery connexions

Vous pouvez utiliser AWS Glue for Spark pour lire et écrire dans des tableaux de Google BigQuery dans AWS Glue 4.0 et versions ultérieures. Vous pouvez le lire à l' BigQuery aide d'une requête Google SQL. Vous vous connectez à BigQuery l'aide des informations d'identification stockées AWS Secrets Manager via une connexion AWS Glue.

Pour plus d'informations sur Google BigQuery, consultez le BigQuery site Web de Google Cloud.

Configuration des BigQuery connexions

Pour vous connecter à Google BigQuery depuis AWS Glue, vous devez créer et stocker vos informations d'identification Google Cloud Platform dans un AWS Secrets Manager secret, puis associer ce secret à une connexion Google BigQuery AWS Glue.

Pour configurer une connexion à BigQuery :

1. Dans Google Cloud Platform, créez et identifiez les ressources pertinentes :
   - Créez ou identifiez un projet GCP contenant des BigQuery tables auxquelles vous souhaitez vous connecter.
   - Activez l' BigQuery API. Pour plus d'informations, voir Utiliser l'API BigQuery Storage Read pour lire les données des tables.
2. Dans Google Cloud Platform, créez et exportez les informations d'identification du compte de service :

   Vous pouvez utiliser l'assistant BigQuery d'identification pour accélérer cette étape : créer des informations d'identification.

   Pour créer un compte de service dans GCP, suivez le didacticiel disponible dans la section Créer des comptes de service.
   - Lorsque vous sélectionnez un projet, sélectionnez le projet contenant votre BigQuery tableau.
   - Lorsque vous sélectionnez des rôles GCP IAM pour votre compte de service, ajoutez ou créez un rôle qui accordera les autorisations appropriées pour exécuter des BigQuery tâches de lecture, d'écriture ou de création BigQuery de tables.

   Pour créer des informations d'identification pour votre compte de service, suivez le didacticiel disponible dans la section Créer une clé de compte de service.

- Lorsque vous sélectionnez le type de clé, sélectionnez JSON.

Vous devriez maintenant avoir téléchargé un fichier JSON contenant les informations d'identification de votre compte de service. Il doit ressembler à l'exemple ci-dessous.

```
{
  "type": "service_account",
  "project_id": "*****",
  "private_key_id": "*****",
  "private_key": "*****",
  "client_email": "*****",
  "client_id": "*****",
  "auth_uri": "https://accounts.google.com/o/oauth2/auth",
  "token_uri": "https://oauth2.googleapis.com/token",
  "auth_provider_x509_cert_url": "https://www.googleapis.com/oauth2/v1/certs",
  "client_x509_cert_url": "*****",
  "universe_domain": "googleapis.com"
}
```

3. Téléchargez le fichier JSON de vos informations d'identification sur un emplacement Amazon S3 correctement sécurisé. Conservez le chemin d'accès au fichier *s3secretpath* pour les étapes futures.
4. Dans AWS Secrets Manager, créez un secret à l'aide de vos informations d'identification Google Cloud Platform. Pour créer un secret dans Secrets Manager, suivez le didacticiel disponible dans la section Créer un AWS Secrets Manager secret dans la AWS Secrets Manager documentation. Après avoir créé le secret, conservez le nom du secret *secretName* pour l'étape suivante.

   Lorsque vous créez des Key/value paires, spécifiez les clés et les valeurs comme suit :

   - Pour les `client_id` cléstoken_uri,,client_x509_cert_url,,private_key_id,,project_id,,universe_domain, spécifiez les valeurs correspondantes dans le fichier JSON téléchargé.
   - Pour la `spark.hadoop.google.cloud.auth.service.account.json.keyfile` clé, spécifiez le*s3secretpath*.
5. Dans le catalogue de données AWS Glue, créez une connexion en suivant les étapes décrites dansthe section called "Ajouter une AWS Glue connexion". Après avoir créé la connexion, conservez le nom de la connexion pour l'étape suivante. *connectionName*

- Lorsque vous sélectionnez un type de connexion, sélectionnez Google BigQuery.
- Lorsque vous sélectionnez un AWS secret, indiquez-le*secretName*.

6. Accordez au rôle IAM associé à votre tâche AWS Glue l'autorisation de lire*secretName*.
7. Dans la configuration de votre tâche AWS Glue*connectionName*, fournissez une connexion réseau supplémentaire.

Lecture à partir de BigQuery tables

Prérequis :

- Un BigQuery tableau que vous aimeriez lire. Vous aurez besoin des noms de BigQuery table et de jeu de données dans le formulaire`[dataset].[table]`. Appelons ça*tableName*.
- Le projet de facturation de la BigQuery table. Vous aurez besoin du nom du projet,*parentProject*. S'il n'existe aucun projet parent de facturation, utilisez le projet contenant la table.
- BigQuery informations d'authentification. Suivez les étapes pour gérer vos informations de connexion avec AWS Glue afin de configurer vos informations d'authentification. Vous aurez besoin du nom de la connexion AWS Glue,*connectionName*.

Par exemple :

```
bigquery_read = glueContext.create_dynamic_frame.from_options(
    connection_type="bigquery",
    connection_options={
        "connectionName": "connectionName",
        "parentProject": "parentProject",
        "sourceType": "table",
        "table": "tableName",
    }
```

Vous pouvez également fournir une requête pour filtrer les résultats renvoyés à votre DynamicFrame. Vous devrez configurer `query`, `sourceType`, `viewsEnabled` et `materializationDataset`.

Par exemple :

Prérequis supplémentaires :

Vous devrez créer ou identifier un BigQuery jeu de données dans lequel vous BigQuery pourrez écrire des vues matérialisées pour vos requêtes. *materializationDataset*

Vous devrez accorder les autorisations GCP IAM appropriées à votre compte de service pour créer des tables dans. *materializationDataset*

```
glueContext.create_dynamic_frame.from_options(
            connection_type="bigquery",
            connection_options={
                "connectionName": "connectionName",
                "materializationDataset": materializationDataset,
                "parentProject": "parentProject",
                "viewsEnabled": "true",
                "sourceType": "query",
                "query": "select * from bqtest.test"
            }
        )
```

Écrire sur des BigQuery tables

Cet exemple écrit directement dans le BigQuery service. BigQuery supporte également la méthode d'écriture « indirecte ». Pour plus d'informations sur les écritures indirectes, consultez the section called “Utiliser l'écriture indirecte avec Google BigQuery”.

Prérequis :

- Une BigQuery table dans laquelle vous souhaitez écrire. Vous aurez besoin des noms de BigQuery table et de jeu de données dans le formulaire`[dataset].[table]`. Vous pouvez également fournir un nouveau nom de table qui sera automatiquement créé. Appelons ça*tableName*.
- Le projet de facturation de la BigQuery table. Vous aurez besoin du nom du projet,*parentProject*. S'il n'existe aucun projet parent de facturation, utilisez le projet contenant la table.
- BigQuery informations d'authentification. Suivez les étapes pour gérer vos informations de connexion avec AWS Glue afin de configurer vos informations d'authentification. Vous aurez besoin du nom de la connexion AWS Glue,*connectionName*.

Par exemple :

```
bigquery_write = glueContext.write_dynamic_frame.from_options(
```

```
    frame=frameToWrite,
    connection_type="bigquery",
    connection_options={
        "connectionName": "connectionName",
        "parentProject": "parentProject",
        "writeMethod": "direct",
        "table": "tableName",
    }
)
```

BigQuery référence des options de connexion

- `project` : par défaut, le compte de service Google Cloud par défaut. Utilisé pour la lecture/l'écriture. Le nom d'un projet Google Cloud associé à votre table.
- `table` : (obligatoire) utilisé pour la lecture/l'écriture. Le nom de votre BigQuery table dans le format`[[project:]dataset.]`.
- `dataset` : obligatoire lorsqu'il n'est pas défini par l'option `table`. Utilisé pour la lecture/l'écriture. Le nom du jeu de données contenant votre BigQuery table.
- `parentProject` : par défaut, le compte de service Google Cloud par défaut. Utilisé pour la lecture/l'écriture. Nom d'un projet Google Cloud associé à `project` utilisé pour la facturation.
- `sourceType` : utilisé pour la lecture. Obligatoire lors de la lecture. Valeurs valides : `table` `query` indique à AWS Glue si vous allez lire par table ou par requête.
- `materializationDataset` : utilisé pour la lecture. Valeurs valides : chaînes. Nom d'un BigQuery ensemble de données utilisé pour stocker les matérialisations des vues.
- `viewsEnabled` : utilisé pour la lecture. Par défaut : false. Valeurs valides : true, false. Détermine si BigQuery vous souhaitez utiliser les vues.
- `query` : utilisé pour la lecture. Utilisé quand `viewsEnabled` est true. Une requête GoogleSQL DQL.
- `temporaryGcsBucket` : utilisé pour écrire. Obligatoire lorsque `writeMethod` est défini sur la valeur par défaut (`indirect`). Nom d'un bucket Google Cloud Storage utilisé pour stocker une forme intermédiaire de vos données lorsque vous y écrivez BigQuery.
- `writeMethod` : `indirect` par défaut. Valeurs valides : `direct`, `indirect`. Utilisé pour écrire. Spécifie la méthode utilisée pour écrire les données.
  - S'il est défini sur`direct`, votre connecteur écrira à l'aide de l'API BigQuery Storage Write.

- S'il est défini sur`indirect`, votre connecteur écrira dans Google Cloud Storage, puis le transférera à l' BigQuery aide d'une opération de chargement. Votre compte de service Google Cloud aura besoin des autorisations GCS appropriées.

Utiliser l'écriture indirecte avec Google BigQuery

Cet exemple utilise l'écriture indirecte, qui écrit des données dans Google Cloud Storage et les copie dans Google BigQuery.

Prérequis :

Vous aurez besoin d'un bucket Google Cloud Storage temporaire*temporaryBucket*.

Le rôle GCP IAM pour le compte de service GCP de AWS Glue nécessitera les autorisations GCS appropriées pour y accéder. *temporaryBucket*

Configuration supplémentaire :

Pour configurer l'écriture indirecte avec BigQuery :

1. Évaluez the section called “Configuration BigQuery” et localisez ou retéléchargez le fichier JSON de vos informations d'identification GCP. Identifiez*secretName*, le AWS Secrets Manager secret de la connexion Google BigQuery AWS Glue utilisée dans le cadre de votre travail.
2. Téléchargez le fichier JSON de vos informations d'identification sur un emplacement Amazon S3 correctement sécurisé. Conservez le chemin d'accès au fichier *s3secretpath* pour les étapes futures.
3. Modifier*secretName*, ajouter la `spark.hadoop.google.cloud.auth.service.account.json.keyfile` clé. Définissez la valeur sur *s3secretpath*.
4. Accordez à votre job AWS Glue Amazon S3 IAM des autorisations d'accès*s3secretpath*.

Vous pouvez désormais fournir l'emplacement temporaire de votre compartiment GCS à votre méthode d'écriture. Vous n'avez pas besoin de fournir `writeMethod`, car `indirect` est historiquement la valeur par défaut.

```
bigquery_write = glueContext.write_dynamic_frame.from_options(
    frame=frameToWrite,
    connection_type="bigquery",
```

```
    connection_options={
        "connectionName": "connectionName",
        "parentProject": "parentProject",
        "temporaryGcsBucket": "temporaryBucket",
        "table": "tableName",
    }
)
```

Connexions JDBC

Certains types de bases de données, généralement relationnelles, prennent en charge la connexion via le standard JDBC. Pour plus d'informations sur JDBC, consultez la documentation de l'API Java JDBC. AWS Glue prend en charge nativement la connexion à certaines bases de données via ses connecteurs JDBC. Les bibliothèques JDBC sont fournies dans les jobs Glue Spark. AWS Lorsque vous vous connectez à ces types de bases de données à l'aide des bibliothèques AWS Glue, vous avez accès à un ensemble standard d'options.

Les valeurs connectionType JDBC sont les suivantes :

- `"connectionType": "sqlserver"` : Désigne une connexion à une base de données Microsoft SQL Server.
- `"connectionType": "mysql"` : Désigne une connexion à une base de données MySQL.
- `"connectionType": "oracle"` : Désigne une connexion à une base de données Oracle.
- `"connectionType": "postgresql"` : Désigne une connexion à une base de données PostgreSQL.
- `"connectionType": "redshift"` : désigne une connexion à une base de données Amazon Redshift. Pour de plus amples informations, veuillez consulter the section called "Connexions Redshift".

Le tableau suivant répertorie les versions du pilote JDBC prises en charge par AWS Glue.

Produit (langue française non garantie)	Versions du pilote JDBC pour Glue 5.1	Versions du pilote JDBC pour Glue 5.0	Versions de pilotes JDBC pour Glue 4.0	Versions de pilotes JDBC pour Glue 3.0	Versions de pilotes JDBC pour Glue 0.9, 1.0, 2.0
Microsoft SQL S	10.2.0	10.2.0	9.4.0	7.x	6.x

Produit (langue française non garantie)	Versions du pilote JDBC pour Glue 5.1	Versions du pilote JDBC pour Glue 5.0	Versions de pilotes JDBC pour Glue 4.0	Versions de pilotes JDBC pour Glue 3.0	Versions de pilotes JDBC pour Glue 0.9, 1.0, 2.0
MySQL	8,0,33	8,0,33	8.0.23	8.0.23	5.1
Oracle Databas	23,3.0,23,09	23,3.0,23,09	21,7	21,1	11.2
PostgreSQL	42.7.3	42.7.3	42,3.6	42,2,18	42.1.x
Amazon Redshift *	redshift-jdbc42-2.1.0.29	redshift-jdbc42-2.1.0.29	redshift-jdbc42-2.1.0.16	redshift-jdbc41-1.2.12.1017	redshift-jdbc41-1.2.12.1017

* Pour le type de connexion Amazon Redshift, toutes les autres name/value paires d'options incluses dans les options de connexion pour une connexion JDBC, y compris les options de formatage, sont transmises directement au SparkSQL sous-jacent. DataSource Dans les jobs AWS Glue with Spark de AWS Glue 4.0 et des versions ultérieures, le connecteur natif AWS Glue pour Amazon Redshift utilise l'intégration Amazon Redshift pour Apache Spark. Pour plus d'informations, consultez Intégration d'Amazon Redshift à Apache Spark. Pour les versions précédentes, consultez Amazon Redshift data source for Spark.

Pour configurer votre Amazon VPC afin de vous connecter aux magasins de données Amazon RDS à l'aide de JDBC, reportez-vous à the section called "Configuration d'Amazon VPC pour se connecter aux entrepôts de données Amazon RDS".

> **Note**
>
> AWS Les tâches Glue ne sont associées qu'à un seul sous-réseau lors d'une exécution. Cela peut avoir une incidence sur votre capacité à vous connecter à plusieurs sources de données par le biais de la même tâche. Ce comportement n'est pas limité aux sources JDBC.

Rubriques

- Référence des options de connexion JDBC
- Utiliser sampleQuery

- Utiliser un pilote JDBC personnalisé
- Lecture en parallèle à partir de tables JDBC
- Configuration d'Amazon VPC pour les connexions JDBC aux entrepôts de données Amazon RDS depuis AWS Glue

Référence des options de connexion JDBC

Si une connexion JDBC AWS Glue est déjà définie, vous pouvez réutiliser les propriétés de configuration qui y sont définies, telles que : url, utilisateur et mot de passe ; vous n'avez donc pas à les spécifier dans le code en tant qu'options de connexion. Cette fonctionnalité est disponible dans AWS Glue 3.0 et versions ultérieures. Pour ce faire, utilisez les propriétés de connexion suivantes :

- `"useConnectionProperties"` : à définir sur « true » pour indiquer que vous souhaitez utiliser la configuration à partir d'une connexion.
- `"connectionName"` : saisissez le nom de connexion à partir duquel récupérer la configuration. La connexion doit être définie dans la même région que la tâche.

Utilisez ces options de connexion avec les connexions JDBC :

- `"url"` : (Obligatoire) URL JDBC de la base de données.
- `"dbtable"` : (obligatoire) table de la base de données à partir de laquelle la lecture doit s'effectuer. Pour les magasins de données JDBC qui prennent en charge les schémas dans une base de données, spécifiez `schema.table-name`. Si aucun schéma n'est fourni, c'est le schéma « public » par défaut qui est utilisé.
- `"user"` : (Obligatoire) Nom d'utilisateur à utiliser lors de la connexion.
- `"password"` : (Obligatoire) Mot de passe à utiliser lors de la connexion.
- (Facultatif) les options suivantes vous permettent de fournir un pilote JDBC personnalisé. Utilisez ces options si vous devez utiliser un pilote que AWS Glue ne prend pas en charge de manière native.

  Les tâches ETL peuvent utiliser différentes versions de pilote JDBC pour la source de données et la cible, même si la source et la cible sont le même produit de base de données. Cela vous permet de migrer des données entre les bases de données source et cible avec différentes versions. Pour utiliser ces options, vous devez d'abord charger le fichier JAR du pilote JDBC vers Amazon S3.

  - `"customJdbcDriverS3Path"` : chemin Amazon S3 du pilote JDBC personnalisé.

  - `"customJdbcDriverClassName"` : nom de la classe du pilote JDBC.
- `"bulkSize"` : (Facultatif) Utilisé pour configurer des insertions parallèles pour accélérer les charges groupées dans les cibles JDBC. Spécifiez une valeur entière pour le degré de parallélisme à utiliser lors de l'écriture ou de l'insertion de données. Cette option permet d'améliorer les performances d'écriture dans les bases de données telles que le référentiel d'utilisateurs Arch (AUR).
- `"hashfield"` : (facultatif) chaîne utilisée pour indiquer le nom d'une colonne de la table JDBC à utiliser pour diviser les données en partitions lors de la lecture de tables JDBC en parallèle. Indiquez « hashfield » OU « hashexpression ». Pour de plus amples informations, veuillez consulter the section called "Lecture en parallèle à partir de JDBC".
- `"hashexpression"` : (facultatif) une clause de sélection SQL renvoyant un nombre entier. Utilisée pour diviser les données d'une table JDBC en partitions lors de la lecture de tables JDBC en parallèle. Indiquez « hashfield » OU « hashexpression ». Pour de plus amples informations, veuillez consulter the section called "Lecture en parallèle à partir de JDBC".
- `"hashpartitions"` : (facultatif) un entier positif. Utilisé pour indiquer le nombre de lectures parallèles de la table JDBC lors de la lecture de tables JDBC en parallèle. Par défaut : 7. Pour de plus amples informations, veuillez consulter the section called "Lecture en parallèle à partir de JDBC".
- `"sampleQuery"` : (facultatif) une instruction de requête SQL personnalisée. Utilisé pour spécifier un sous-ensemble d'informations dans une table afin de récupérer un échantillon du contenu de la table. Lorsqu'elle est configurée sans tenir compte de vos données, elle peut être moins efficace que DynamicFrame les méthodes, entraînant des délais d'attente ou des erreurs de mémoire insuffisante. Pour de plus amples informations, veuillez consulter the section called "Utiliser sampleQuery".
- `"enablePartitioningForSampleQuery"` : (facultatif) une valeur booléenne. Par défaut : faux. Utilisé pour permettre la lecture des tables JDBC en parallèle lorsque vous indiquez `sampleQuery`. S'il est défini sur true, **`sampleQuery`** il doit se terminer par « where » ou « and » pour que AWS Glue ajoute des conditions de partitionnement. Pour de plus amples informations, veuillez consulter the section called "Utiliser sampleQuery".
- `"sampleSize"` : (facultatif) un entier positif. Limite le nombre de lignes renvoyées par l'exemple de requête. Fonctionne uniquement lorsque `enablePartitioningForSampleQuery` a la valeur « true ». Si le partitionnement n'est pas activé, vous devez plutôt ajouter directement `"limit x"` dans `sampleQuery` pour limiter la taille. Pour de plus amples informations, veuillez consulter the section called "Utiliser sampleQuery".

Utiliser sampleQuery

Cette section explique comment utiliser `sampleQuery`, `sampleSize` et `enablePartitioningForSampleQuery`.

`sampleQuery` peut être un moyen efficace d'échantillonner quelques lignes de votre jeu de données. Par défaut, la requête est exécutée par un exécuteur unique. Lorsqu'elle est configurée sans tenir compte de vos données, elle peut être moins efficace que DynamicFrame les méthodes, entraînant des délais d'attente ou des erreurs de mémoire insuffisante. L'exécution de SQL sur la base de données sous-jacente dans le cadre de votre pipeline ETL n'est généralement nécessaire que pour des raisons de performances. Si vous essayez de prévisualiser quelques lignes de votre jeu de données, pensez à utiliser the section called "show". Si vous essayez de transformer votre ensemble de données à l'aide de SQL, pensez the section called "toDF" à l'utiliser pour définir une transformation SparkSQL par rapport à vos données dans un formulaire. DataFrame

Bien que votre requête puisse manipuler diverses tables, `dbtable` reste obligatoire.

Utilisation de sampleQuery pour récupérer un échantillon de votre table

Lorsque vous utilisez le comportement SampleQuery par défaut pour récupérer un échantillon de vos données, AWS Glue ne s'attend pas à un débit important. Il exécute donc votre requête sur un seul exécuteur. Afin de limiter les données que vous fournissez et de ne pas entraîner de problèmes de performances, nous vous suggérons de fournir une clause `LIMIT` à SQL.

Example Utiliser SampleQuery sans partitionnement

L'exemple de code suivant illustre comment utiliser `sampleQuery` sans partitionnement.

```
//A full sql query statement.
val query = "select name from $tableName where age > 0 limit 1"
val connectionOptions = JsonOptions(Map(
    "url" -> url,
    "dbtable" -> tableName,
    "user" -> user,
    "password" -> password,
    "sampleQuery" -> query ))
val dyf = glueContext.getSource("mysql", connectionOptions)
          .getDynamicFrame()
```

Utilisation de sampleQuery sur des jeux de données plus volumineux

Si vous lisez un jeu de données volumineux, vous devrez peut-être activer le partitionnement JDBC pour interroger une table en parallèle. Pour de plus amples informations, veuillez consulter the section called "Lecture en parallèle à partir de JDBC". Pour utiliser `sampleQuery` avec le partitionnement JDBC, paramétrez `enablePartitioningForSampleQuery` sur la valeur « true ». L'activation de cette fonctionnalité nécessite que vous apportiez quelques modifications à votre `sampleQuery`.

Lorsque vous utilisez le partitionnement JDBC avec`sampleQuery`, votre requête doit se terminer par « where » ou « and » pour que AWS Glue ajoute des conditions de partitionnement.

Si vous souhaitez limiter les résultats de votre sampleQuery lorsque vous lisez des tables JDBC en parallèle, définissez le paramètre `"sampleSize"` plutôt que d'indiquer une clause `LIMIT`.

Example Utiliser SampleQuery avec le partitionnement JDBC

L'exemple de code suivant illustre comment utiliser `sampleQuery` avec partitionnement JDBC.

```
//note that the query should end with "where" or "and" if use with JDBC partitioning.
val query = "select name from $tableName where age > 0 and"

//Enable JDBC partitioning by setting hashfield.
//to use sampleQuery with partitioning, set enablePartitioningForSampleQuery.
//use sampleSize to limit the size of returned data.
val connectionOptions = JsonOptions(Map(
    "url" -> url,
    "dbtable" -> tableName,
    "user" -> user,
    "password" -> password,
    "hashfield" -> primaryKey,
    "sampleQuery" -> query,
    "enablePartitioningForSampleQuery" -> true,
    "sampleSize" -> "1" ))
val dyf = glueContext.getSource("mysql", connectionOptions)
          .getDynamicFrame()
```

Notes et restrictions :

Les exemples de requêtes ne peuvent pas être utilisés avec des signets de tâche. L'état du signet sera ignoré lorsque la configuration sera fournie pour les deux.

Utiliser un pilote JDBC personnalisé

Les exemples de code suivants montrent comment lire et écrire des bases de données JDBC avec des pilotes JDBC personnalisés. Ils présentent la lecture d'une version d'un produit de base de données et l'écriture dans une version ultérieure du même produit.

Python

```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext, SparkConf
from awsglue.context import GlueContext
from awsglue.job import Job
import time
from pyspark.sql.types import StructType, StructField, IntegerType, StringType

sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session

# Construct JDBC connection options
connection_mysql5_options = {
    "url": "jdbc:mysql://<jdbc-host-name>:3306/db",
    "dbtable": "test",
    "user": "admin",
    "password": "pwd"}

connection_mysql8_options = {
    "url": "jdbc:mysql://<jdbc-host-name>:3306/db",
    "dbtable": "test",
    "user": "admin",
    "password": "pwd",
    "customJdbcDriverS3Path": "s3://amzn-s3-demo-bucket/mysql-connector-
java-8.0.17.jar",
    "customJdbcDriverClassName": "com.mysql.cj.jdbc.Driver"}

connection_oracle11_options = {
    "url": "jdbc:oracle:thin:@//<jdbc-host-name>:1521/ORCL",
    "dbtable": "test",
    "user": "admin",
    "password": "pwd"}
```

```
connection_oracle18_options = {
    "url": "jdbc:oracle:thin:@//<jdbc-host-name>:1521/ORCL",
    "dbtable": "test",
    "user": "admin",
    "password": "pwd",
    "customJdbcDriverS3Path": "s3://amzn-s3-demo-bucket/ojdbc10.jar",
    "customJdbcDriverClassName": "oracle.jdbc.OracleDriver"}

# Read from JDBC databases with custom driver
df_mysql8 = glueContext.create_dynamic_frame.from_options(connection_type="mysql",

 connection_options=connection_mysql8_options)

# Read DynamicFrame from MySQL 5 and write to MySQL 8
df_mysql5 = glueContext.create_dynamic_frame.from_options(connection_type="mysql",

 connection_options=connection_mysql5_options)
glueContext.write_from_options(frame_or_dfc=df_mysql5, connection_type="mysql",
                               connection_options=connection_mysql8_options)

# Read DynamicFrame from Oracle 11 and write to Oracle 18
df_oracle11 =
 glueContext.create_dynamic_frame.from_options(connection_type="oracle",

 connection_options=connection_oracle11_options)
glueContext.write_from_options(frame_or_dfc=df_oracle11, connection_type="oracle",
                               connection_options=connection_oracle18_options)
```

Scala

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.MappingSpec
import com.amazonaws.services.glue.errors.CallSite
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.DynamicFrame
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._


object GlueApp {
```

```
  val MYSQL_5_URI: String = "jdbc:mysql://<jdbc-host-name>:3306/db"
  val MYSQL_8_URI: String = "jdbc:mysql://<jdbc-host-name>:3306/db"
  val ORACLE_11_URI: String = "jdbc:oracle:thin:@//<jdbc-host-name>:1521/ORCL"
  val ORACLE_18_URI: String = "jdbc:oracle:thin:@//<jdbc-host-name>:1521/ORCL"

  // Construct JDBC connection options
  lazy val mysql5JsonOption = jsonOptions(MYSQL_5_URI)
  lazy val mysql8JsonOption = customJDBCDriverJsonOptions(MYSQL_8_URI, "s3://amzn-
s3-demo-bucket/mysql-connector-java-8.0.17.jar", "com.mysql.cj.jdbc.Driver")
  lazy val oracle11JsonOption = jsonOptions(ORACLE_11_URI)
  lazy val oracle18JsonOption = customJDBCDriverJsonOptions(ORACLE_18_URI, "s3://
amzn-s3-demo-bucket/ojdbc10.jar", "oracle.jdbc.OracleDriver")

  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)

    // Read from JDBC database with custom driver
    val df_mysql8: DynamicFrame = glueContext.getSource("mysql",
 mysql8JsonOption).getDynamicFrame()

    // Read DynamicFrame from MySQL 5 and write to MySQL 8
    val df_mysql5: DynamicFrame = glueContext.getSource("mysql",
 mysql5JsonOption).getDynamicFrame()
    glueContext.getSink("mysql", mysql8JsonOption).writeDynamicFrame(df_mysql5)

    // Read DynamicFrame from Oracle 11 and write to Oracle 18
    val df_oracle11: DynamicFrame = glueContext.getSource("oracle",
 oracle11JsonOption).getDynamicFrame()
    glueContext.getSink("oracle", oracle18JsonOption).writeDynamicFrame(df_oracle11)

    Job.commit()
  }

  private def jsonOptions(url: String): JsonOptions = {
    new JsonOptions(
      s"""{"url": "${url}",
         |"dbtable":"test",
         |"user": "admin",
         |"password": "pwd"}""".stripMargin)
  }
```

```
  private def customJDBCDriverJsonOptions(url: String, customJdbcDriverS3Path:
 String, customJdbcDriverClassName: String): JsonOptions = {
    new JsonOptions(
      s"""{"url": "${url}",
         |"dbtable":"test",
         |"user": "admin",
         |"password": "pwd",
         |"customJdbcDriverS3Path": "${customJdbcDriverS3Path}",
         |"customJdbcDriverClassName" :
 "${customJdbcDriverClassName}"}""".stripMargin)
  }
}
```

Lecture en parallèle à partir de tables JDBC

Vous pouvez définir les propriétés de votre table JDBC afin d'autoriser AWS Glue à lire les données en parallèle. Lorsque vous définissez certaines propriétés, vous demandez à AWS Glue d'exécuter des requêtes SQL parallèles sur des partitions logiques de vos données. Vous pouvez contrôler le partitionnement en définissant un champ de hachage ou une expression de hachage. Vous pouvez également contrôler le nombre de lectures parallèles utilisées pour accéder à vos données.

La lecture en parallèle à partir de tables JDBC est une technique d'optimisation susceptible d'améliorer les performances. Pour plus d'informations sur le processus permettant d'identifier le moment où cette technique est appropriée, consultez la section Réduire la quantité de données analysées dans le guide des meilleures pratiques pour le réglage des performances AWS Glue pour les tâches Apache Spark sur les directives AWS prescriptives.

Pour activer les lectures parallèles, vous pouvez définir des paires clé-valeur dans le champ des paramètres de la structure de votre table. Utilisez la notation JSON pour définir une valeur pour le champ des paramètres de votre table. Pour plus d'informations sur la modification des propriétés d'une table, consultez Affichage et gestion des détails d'une table. Vous pouvez également activer les lectures parallèles lorsque vous appelez les méthodes d'extraction, de transformation et de chargement (ETL) `create_dynamic_frame_from_options` et `create_dynamic_frame_from_catalog`. Pour plus d'informations sur la spécification d'options dans ces méthodes, consultez from_options et from_catalog.

Vous pouvez utiliser cette méthode pour les tables JDBC, c'est-à-dire la plupart des tables dont la base de données est un magasin de données JDBC. Ces propriétés sont ignorées lors de la lecture des tables Amazon Redshift et Amazon S3.

hashfield

Définissez `hashfield` sur le nom d'une colonne de la table JDBC à utiliser pour diviser les données en partitions. Pour de meilleurs résultats, cette colonne doit avoir une distribution uniforme des valeurs pour répartir les données entre les partitions. Cette colonne peut avoir n'importe quel type de données. AWS Glue génère des requêtes qui ne se chevauchent pas et qui s'exécutent en parallèle pour lire les données partitionnées à partir de cette colonne. Par exemple, si vos données sont réparties de façon uniforme par mois, vous pouvez utiliser la colonne `month` pour lire chaque mois de données en parallèle.

```
'hashfield': 'month'
```

AWS Glue crée une requête pour hacher la valeur du champ vers un numéro de partition et exécute la requête pour toutes les partitions en parallèle. Pour utiliser votre propre requête de partition d'une lecture de table, fournissez un élément `hashexpression` au lieu d'un élément `hashfield`.

hashexpression

Définissez `hashexpression` sur une expression SQL (conforme à la syntaxe du moteur de base de données JDBC) qui renvoie un nombre entier. Une expression simple représente le nom d'une colonne numérique dans la table. AWS Glue génère des requêtes SQL pour lire les données JDBC en parallèle à l'aide de l'élément `hashexpression` dans la clause `WHERE` pour partitionner les données.

Par exemple, utilisez la colonne numérique `customerID` pour lire les données partitionnées en fonction d'un numéro de client.

```
'hashexpression': 'customerID'
```

Pour qu'AWS Glue contrôle le partitionnement, fournissez un élément `hashfield` au lieu de `hashexpression`.

hashpartitions

Définissez l'élément `hashpartitions` sur le nombre de lectures parallèles de la table JDBC. Si cette propriété n'est pas définie, la valeur par défaut est 7.

Par exemple, définissez le nombre de lectures parallèles sur 5 afin qu'AWS Glue lise vos données avec cinq requêtes (ou moins).

```
'hashpartitions': '5'
```

Configuration d'Amazon VPC pour les connexions JDBC aux entrepôts de données Amazon RDS depuis AWS Glue

Lorsque vous utilisez JDBC pour vous connecter à des bases de données dans Amazon RDS, vous devez effectuer une configuration supplémentaire. Pour permettre aux composants AWS Glue de communiquer avec Amazon RDS, vous devez configurer l'accès à vos entrepôts de données de données Amazon RDS dans Amazon VPC. Pour permettre à AWS Glue de communiquer entre ses composants, spécifiez un groupe de sécurité avec une règle entrante avec référence circulaire pour tous les ports TCP. En créant une règle avec référence circulaire, vous pouvez restreindre la source au même groupe de sécurité dans le VPC. Une règle d'autoréférencement n'ouvrira pas le VPC à tous les réseaux. Le groupe de sécurité par défaut pour votre VPC peut déjà avoir une règle entrante avec référence circulaire pour ALL Traffic.

Pour configurer l'accès entre les magasins de données AWS Glue et Amazon RDS

1. Connectez-vous à la console Amazon RDS AWS Management Console et ouvrez-la à https://console.aws.amazon.com/rds/l'adresse.
2. Dans la console Amazon RDS, identifiez le ou les groupes de sécurité utilisés pour contrôler l'accès à votre base de données Amazon RDS.

   Dans le volet de navigation de gauche, choisissez Bases de données, puis sélectionnez l'instance à laquelle vous souhaitez vous connecter dans la liste du volet principal.

   Sur la page détaillée de la base de données, recherchez Groupes de sécurité VPC dans l'onglet Connectivité et sécurité.
3. En fonction de l'architecture de votre réseau, identifiez le groupe de sécurité associé qu'il est préférable de modifier pour autoriser l'accès au service AWS Glue. Enregistrez son nom *`database-security-group`* pour référence future. S'il n'existe aucun groupe de sécurité approprié, suivez les instructions pour fournir l'accès à votre instance de base de données dans votre VPC en créant un groupe de sécurité dans la documentation Amazon RDS.

4. Connectez-vous à la console Amazon VPC AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/vpc/
5. Dans la console Amazon VPC, déterminez comment effectuer la mise à jour. *database-security-group*

   Dans le volet de navigation de gauche, choisissez Groupes ***database-security-group*** de sécurité, puis sélectionnez-les dans la liste du volet principal.
6. Identifiez l'ID du groupe de sécurité pour*database-security-group*,*database-sg-id*. Conservez-le en vue d'une référence future.

   Sur la page détaillée du groupe de sécurité, recherchez ID du groupe de sécurité.
7. Modifiez les règles entrantes pour*database-security-group*, ajoutez une règle d'autoréférencement pour permettre aux AWS Glue composants de communiquer. Plus précisément, ajoutez ou confirmez qu'il existe une règle dans laquelle le type est`All TCP`, le protocole est`TCP`, la plage de ports inclut tous les ports et la source est*database-sg-id*. Vérifiez que le groupe de sécurité que vous avez saisi pour Source est le même que le groupe de sécurité que vous modifiez.

   Sur la page détaillée du groupe de sécurité, sélectionnez Modifier les règles entrantes.

   La règle entrante ressemble à ce qui suit :

Type	Protocole	Plage de ports	Source
Tous les TCP	TCP	0-65535	*database-sg-id*

8. Ajoutez des règles pour le trafic sortant.

   Sur la page détaillée du groupe de sécurité, sélectionnez Modifier les règles sortantes.

   Si votre groupe de sécurité autorise tout le trafic sortant, vous n'avez pas besoin de règles distinctes. Par exemple :

Type	Protocole	Plage de ports	Destination
Tout le trafic	ALL	ALL	0.0.0.0/0

Si votre architecture réseau est conçue pour vous permettre de restreindre le trafic sortant, créez les règles de trafic sortant suivantes :

Créez une règle d'autoréférencement où le type est`All TCP`, le protocole est`TCP`, la plage de ports inclut tous les ports et la destination est. *`database-sg-id`* Vérifiez que le groupe de sécurité que vous avez saisi pour Destination est le même que le groupe de sécurité que vous modifiez.

Si vous utilisez un point de terminaison d'un VPC Amazon S3, ajoutez une règle HTTPS pour autoriser le trafic du VPC vers Amazon S3. Créez une règle dans laquelle le type est`HTTPS`, le protocole est`TCP`, la plage de ports est `443` et la destination est l'ID de la liste de préfixes gérée pour le point de terminaison de la passerelle Amazon S3,*`s3-prefix-list-id`*. Pour plus d'informations sur les listes de préfixes et les points de terminaison de passerelle Amazon S3, consultez Gateway endpoints for Amazon S3 dans la documentation Amazon VPC.

Par exemple :

Type	Protocole	Plage de ports	Destination
Tous les TCP	TCP	0-65535	*`database-sg-id`*
HTTPS	TCP	443	*`s3-prefix-list-id`*

Connexions MongoDB

Vous pouvez utiliser AWS Glue for Spark pour lire et écrire dans des tables dans MongoDB et MongoDB Atlas dans AWS Glue 4.0 et versions ultérieures. Vous pouvez vous connecter à MongoDB en utilisant les informations d'identification du nom d'utilisateur et du mot de passe stockées AWS Secrets Manager via une connexion AWS Glue.

Pour plus d'informations sur MongoDB, consultez la documentation MongoDB.

Configuration de connexions MongoDB

Pour vous connecter à MongoDB depuis AWS Glue, vous aurez besoin de vos informations d'identification MongoDB, et. *`mongodbUser`* *`mongodbPass`*

Pour vous connecter à MongoDB depuis AWS Glue, vous aurez peut-être besoin de certaines conditions préalables :

- Si votre instance MongoDB se trouve dans un Amazon VPC, configurez Amazon VPC pour permettre à votre tâche AWS Glue de communiquer avec l'instance MongoDB sans que le trafic ne transite par l'Internet public.

  Dans Amazon VPC, identifiez ou créez un VPC, un sous-réseau et un groupe de sécurité que AWS Glue utilisera lors de l'exécution de la tâche. En outre, vous devez vous assurer qu'Amazon VPC est configuré pour autoriser le trafic réseau entre votre instance MongoDB et cet emplacement. Selon la configuration de votre réseau, cela peut nécessiter des modifications des règles du groupe de sécurité, du réseau ACLs, des passerelles NAT et des connexions d'appairage.

Vous pouvez ensuite configurer AWS Glue pour l'utiliser avec MongoDB.

Pour configurer une connexion à MongoDB :

1. Vous pouvez éventuellement créer un secret à AWS Secrets Manager l'aide de vos informations d'identification MongoDB. Pour créer un secret dans Secrets Manager, suivez le didacticiel disponible dans la section Créer un AWS Secrets Manager secret dans la AWS Secrets Manager documentation. Après avoir créé le secret, conservez le nom du secret *secretName* pour l'étape suivante.

   - Lorsque vous sélectionnez des paires clé/valeur, créez une paire pour la clé `username` contenant la valeur. *mongodbUser*

     Lorsque vous sélectionnez des paires clé/valeur, créez une paire pour la clé `password` contenant la valeur. *mongodbPass*

2. Dans la console AWS Glue, créez une connexion en suivant les étapes décrites dansthe section called “Ajouter une AWS Glue connexion”. Après avoir créé la connexion, conservez le nom de la connexion pour une utilisation future dans AWS Glue. *connectionName*

   - Lorsque vous sélectionnez un type de connexion, sélectionnez MongoDB ou MongoDB Atlas.
   - Lorsque vous sélectionnez une URL MongoDB ou une URL MongoDB Atlas, indiquez le nom d'hôte de votre instance MongoDB.

     Une URL MongoDB est fournie au format `mongodb://`*mongoHost*`:`*mongoPort*`/`*mongoDBname*.

Une URL MongoDB Atlas est fournie au format `mongodb +srv://`*`mongoHost`*`/`*`mongoDBname`*.

- Si vous avez choisi de créer un secret Secrets Manager, choisissez le type AWS Secrets Manager d'identifiant.

  Ensuite, dans AWS Secret, fournissez*`secretName`*.

- Si vous choisissez de fournir un nom d'utilisateur et un mot de passe, fournissez *`mongodbUser`* et*`mongodbPass`*.

3. Dans les situations suivantes, vous pouvez avoir besoin d'une configuration supplémentaire :

   - Pour les instances MongoDB hébergées AWS dans un Amazon VPC
     - Vous devrez fournir les informations de connexion Amazon VPC à la connexion AWS Glue qui définit vos informations de sécurité MongoDB. Lorsque vous créez ou mettez à jour votre connexion, définissez le VPC, le sous-réseau et les groupes de sécurité dans les options réseau.

Après avoir créé une connexion AWS Glue MongoDB, vous devez effectuer les actions suivantes avant d'appeler votre méthode de connexion :

- Si vous avez choisi de créer un secret Secrets Manager, accordez au rôle IAM associé à votre tâche AWS Glue l'autorisation de lecture*`secretName`*.
- Dans la configuration de votre tâche AWS Glue*`connectionName`*, fournissez une connexion réseau supplémentaire.

Pour utiliser votre connexion AWS Glue MongoDB dans AWS Glue for Spark, indiquez l'`connectionName`option dans votre appel de méthode de connexion. Vous pouvez également suivre les étapes décrites the section called “Intégration à MongoDB” pour utiliser la connexion en conjonction avec le AWS Glue Data Catalog.

Lecture depuis MongoDB à l'aide d'une connexion Glue AWS

Prérequis :

- Une collection MongoDB à partir de laquelle vous souhaitez lire. Vous aurez besoin des informations d'identification pour la collection.

Une collection MongoDB est identifiée par un nom de base de données et un nom de collection,,*mongodbName*. *mongodbCollection*

- Une connexion AWS Glue MongoDB configurée pour fournir des informations d'authentification. Suivez les étapes de la procédure précédente Pour configurer une connexion à MongoDB afin de configurer vos informations d'authentification. Vous aurez besoin du nom de la connexion AWS Glue,*connectionName*.

Par exemple :

```
mongodb_read = glueContext.create_dynamic_frame.from_options(
    connection_type="mongodb",
    connection_options={
        "connectionName": "connectionName",
        "database": "mongodbName",
        "collection": "mongodbCollection",
        "partitioner":
 "com.mongodb.spark.sql.connector.read.partitioner.SinglePartitionPartitioner",
        "partitionerOptions.partitionSizeMB": "10",
        "partitionerOptions.partitionKey": "_id",
        "disableUpdateUri": "false",
    }
)
```

Écrire dans des tables MongoDB

Cet exemple écrit les informations d'un fichier existant DynamicFrame *dynamicFrame* dans MongoDB.

Prérequis :

- Une collection MongoDB sur laquelle vous souhaitez écrire. Vous aurez besoin des informations d'identification pour la collection.

  Une collection MongoDB est identifiée par un nom de base de données et un nom de collection,,*mongodbName*. *mongodbCollection*
- Une connexion AWS Glue MongoDB configurée pour fournir des informations d'authentification. Suivez les étapes de la procédure précédente Pour configurer une connexion à MongoDB afin de configurer vos informations d'authentification. Vous aurez besoin du nom de la connexion AWS Glue,*connectionName*.

Par exemple :

```
glueContext.write_dynamic_frame.from_options(
    frame=dynamicFrame,
    connection_type="mongodb",
    connection_options={
        "connectionName": "connectionName",
        "database": "mongodbName",
        "collection": "mongodbCollection",
        "disableUpdateUri": "false",
        "retryWrites": "false",
    },
)
```

Lire et écrire dans des tables MongoDB

Cet exemple écrit les informations d'un fichier existant DynamicFrame *dynamicFrame* dans MongoDB.

Prérequis :

- Une collection MongoDB à partir de laquelle vous souhaitez lire. Vous aurez besoin des informations d'identification pour la collection.

  Une collection MongoDB sur laquelle vous souhaitez écrire. Vous aurez besoin des informations d'identification pour la collection.

  Une collection MongoDB est identifiée par un nom de base de données et un nom de collection,,*mongodbName*. *mongodbCollection*
- informations d'authentification MongoDB, et. *mongodbUser* *mongodbPassword*

Par exemple :

Python

```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext, SparkConf
from awsglue.context import GlueContext
from awsglue.job import Job
```

```
import time

## @params: [JOB_NAME]
args = getResolvedOptions(sys.argv, ['JOB_NAME'])

sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session

job = Job(glueContext)
job.init(args['JOB_NAME'], args)

output_path = "s3://some_bucket/output/" + str(time.time()) + "/"
mongo_uri = "mongodb://<mongo-instanced-ip-address>:27017"
mongo_ssl_uri = "mongodb://<mongo-instanced-ip-address>:27017"
write_uri = "mongodb://<mongo-instanced-ip-address>:27017"

read_mongo_options = {
    "uri": mongo_uri,
    "database": "mongodbName",
    "collection": "mongodbCollection",
    "username": "mongodbUsername",
    "password": "mongodbPassword",
    "partitioner": "MongoSamplePartitioner",
    "partitionerOptions.partitionSizeMB": "10",
    "partitionerOptions.partitionKey": "_id"}

ssl_mongo_options = {
    "uri": mongo_ssl_uri,
    "database": "mongodbName",
    "collection": "mongodbCollection",
    "ssl": "true",
    "ssl.domain_match": "false"
}

write_mongo_options = {
    "uri": write_uri,
    "database": "mongodbName",
    "collection": "mongodbCollection",
    "username": "mongodbUsername",
    "password": "mongodbPassword",
}

# Get DynamicFrame from MongoDB
```

```
dynamic_frame =
 glueContext.create_dynamic_frame.from_options(connection_type="mongodb",

 connection_options=read_mongo_options)

# Write DynamicFrame to MongoDB
glueContext.write_dynamic_frame.from_options(dynamicFrame,
 connection_type="mongodb", connection_options=write_mongo_options)

job.commit()
```

Scala

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.MappingSpec
import com.amazonaws.services.glue.errors.CallSite
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.DynamicFrame
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._

object GlueApp {
  val DEFAULT_URI: String = "mongodb://<mongo-instanced-ip-address>:27017"
  val WRITE_URI: String = "mongodb://<mongo-instanced-ip-address>:27017"
  lazy val defaultJsonOption = jsonOptions(DEFAULT_URI)
  lazy val writeJsonOption = jsonOptions(WRITE_URI)
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)

    // Get DynamicFrame from MongoDB
    val dynamicFrame: DynamicFrame = glueContext.getSource("mongodb",
 defaultJsonOption).getDynamicFrame()

    // Write DynamicFrame to MongoDB
    glueContext.getSink("mongodb", writeJsonOption).writeDynamicFrame(dynamicFrame)

    Job.commit()
  }
```

```
  private def jsonOptions(uri: String): JsonOptions = {
    new JsonOptions(
      s"""{"uri": "${uri}",
         |"database":"mongodbName",
         |"collection":"mongodbCollection",
         |"username": "mongodbUsername",
         |"password": "mongodbPassword",
         |"ssl":"true",
         |"ssl.domain_match":"false",
         |"partitioner": "MongoSamplePartitioner",
         |"partitionerOptions.partitionSizeMB": "10",
         |"partitionerOptions.partitionKey": "_id"}""".stripMargin)
  }
}
```

Référence des options de connexion MongoDB

Désigne une connexion à MongoDB. Les options de connexion sont différentes pour une connexion source et une connexion collecteur.

Les propriétés de connexion suivantes sont partagées entre les connexions source et récepteur :

- `connectionName` : utilisé pour la lecture/l'écriture. Le nom d'une connexion AWS Glue MongoDB configurée pour fournir des informations d'authentification et de mise en réseau à votre méthode de connexion. Lorsqu'une connexion AWS Glue est configurée comme décrit dans la section précédentethe section called "Configuration de MongoDB", la fourniture `connectionName` remplacera le besoin de fournir `"uri"` les options `"username"` et de `"password"` connexion.
- `"uri"`: (Obligatoire) L'hôte MongoDB à lire, formaté comme `mongodb://<host>:<port>`. Utilisé dans les versions de AWS Glue antérieures à AWS Glue 4.0.
- `"connection.uri"`: (Obligatoire) L'hôte MongoDB à lire, formaté comme `mongodb://<host>:<port>`. Utilisé dans AWS Glue 4.0 et versions ultérieures.
- `"username"` : (Obligatoire) Le nom d'utilisateur MongoDB.
- `"password"` : (Obligatoire) Le mot de passe MongoDB.
- `"database"`: (Obligatoire) La base de données MongoDB à lire. Cette option peut également être transmise dans `additional_options` lors de l'appel à `glue_context.create_dynamic_frame_from_catalog` dans votre script de tâche.

- `"collection"` : (Obligatoire) La collection MongoDB à lire. Cette option peut également être transmise dans `additional_options` lors de l'appel à `glue_context.create_dynamic_frame_from_catalog` dans votre script de tâche.

« connectionType »: « mongodb » comme source

Utilisez les options de connexion suivantes avec `"connectionType": "mongodb"` comme source :

- `"ssl"` : (Facultatif) Si `true`, lance une connexion SSL. La valeur par défaut est `false`.
- `"ssl.domain_match"` : (Facultatif) Si `true` et `ssl` est `true`, la vérification de la correspondance de domaine est effectuée. La valeur par défaut est `true`.
- `"batchSize"` : (Facultatif) : Nombre de documents à renvoyer par lot, utilisé dans le curseur des lots internes.
- `"partitioner"` : (Facultatif) : Le nom de classe du partitionneur pour lire les données d'entrée de MongoDB. Le connecteur fournit les partitionneurs suivants :
  - `MongoDefaultPartitioner`(par défaut) (Non pris en charge dans AWS Glue 4.0)
  - `MongoSamplePartitioner`(Nécessite MongoDB 3.2 ou version ultérieure) (Non pris en charge dans AWS Glue 4.0)
  - `MongoShardedPartitioner`(Non pris en charge dans AWS Glue 4.0)
  - `MongoSplitVectorPartitioner`(Non pris en charge dans AWS Glue 4.0)
  - `MongoPaginateByCountPartitioner`(Non pris en charge dans AWS Glue 4.0)
  - `MongoPaginateBySizePartitioner`(Non pris en charge dans AWS Glue 4.0)
  - `com.mongodb.spark.sql.connector.read.partitioner.SinglePartitionPartitioner`
  - `com.mongodb.spark.sql.connector.read.partitioner.ShardedPartitioner`
  - `com.mongodb.spark.sql.connector.read.partitioner.PaginateIntoPartitionsPartit`
- `"partitionerOptions"` (Facultatif) : Options pour le partitionneur désigné. Les options suivantes sont prises en charge pour chaque partitionneur :
  - `MongoSamplePartitioner`: `partitionKey`, `partitionSizeMB`, `samplesPerPartition`
  - `MongoShardedPartitioner`: `shardkey`
  - `MongoSplitVectorPartitioner`: `partitionKey`, `partitionSizeMB`
  - `MongoPaginateByCountPartitioner`: `partitionKey`, `numberOfPartitions`
  - `MongoPaginateBySizePartitioner`: `partitionKey`, `partitionSizeMB`

Pour plus d'informations sur ces options, consultez Configuration du partitionneur dans la documentation MongoDB.

« connectionType »: « mongodb » comme collecteur

Utilisez les options de connexion suivantes avec `"connectionType": "mongodb"` comme collecteur :

- `"ssl"` : (Facultatif) Si `true`, lance une connexion SSL. La valeur par défaut est `false`.
- `"ssl.domain_match"` : (Facultatif) Si `true` et `ssl` est `true`, la vérification de la correspondance de domaine est effectuée. La valeur par défaut est `true`.
- `"extendedBsonTypes"` : (facultatif) si la valeur est `true`, cela active les types BSON étendus lors de l'écriture de données dans MongoDB. La valeur par défaut est `true`.
- `"replaceDocument"` : (Facultatif) Si `true`, remplace l'ensemble du document lors de l'enregistrement de jeux de données contenant un champ `_id`. Si `false`, seuls les champs du document qui correspondent aux champs du jeu de données sont mis à jour. La valeur par défaut est `true`.
- `"maxBatchSize"` : (Facultatif) : Taille maximale du lot pour les opérations en bloc lors de l'enregistrement des données. La valeur par défaut est 512.
- `"retryWrites"`: (Facultatif) : Réessayez automatiquement certaines opérations d'écriture une seule fois si AWS Glue rencontre une erreur réseau.

Connexions SAP HANA

Vous pouvez utiliser AWS Glue for Spark pour lire et écrire dans des tables dans SAP HANA dans AWS Glue 4.0 et versions ultérieures. Vous pouvez définir ce qu'il faut lire dans SAP HANA à l'aide d'une requête SQL. Vous vous connectez à SAP HANA à l'aide des informations d'identification JDBC stockées via AWS Secrets Manager une connexion AWS Glue SAP HANA.

Pour plus d'informations sur SAP HANA JDBC, consultez la documentation SAP HANA.

Configuration des connexions SAP HANA

Pour vous connecter à SAP HANA depuis AWS Glue, vous devez créer et stocker vos informations d'identification SAP HANA dans un AWS Secrets Manager secret, puis associer ce secret à une connexion SAP HANA AWS Glue. Vous devrez configurer la connectivité réseau entre votre service SAP HANA et AWS Glue.

Pour vous connecter à SAP HANA, vous aurez peut-être besoin de certaines conditions préalables :

- Si votre service SAP HANA se trouve dans un Amazon VPC, configurez Amazon VPC pour permettre à votre tâche AWS Glue de communiquer avec le service SAP HANA sans que le trafic ne transite par l'Internet public.

  Dans Amazon VPC, identifiez ou créez un VPC, un sous-réseau et un groupe de sécurité que AWS Glue utilisera lors de l'exécution de la tâche. En outre, vous devez vous assurer qu'Amazon VPC est configuré pour autoriser le trafic réseau entre votre point de terminaison SAP HANA et cet emplacement. Votre tâche devra établir une connexion TCP avec votre port JDBC SAP HANA. Pour plus d'informations sur les ports SAP HANA, consultez la documentation SAP HANA. Selon la configuration de votre réseau, cela peut nécessiter des modifications des règles du groupe de sécurité, du réseau ACLs, des passerelles NAT et des connexions d'appairage.
- Il n'y a aucune condition préalable supplémentaire si votre point de terminaison SAP HANA est accessible sur Internet.

Pour configurer une connexion à SAP HANA :

1. Dans AWS Secrets Manager, créez un secret à l'aide de vos informations d'identification SAP HANA. Pour créer un secret dans Secrets Manager, suivez le didacticiel disponible dans la section Créer un AWS Secrets Manager secret dans la AWS Secrets Manager documentation. Après avoir créé le secret, conservez le nom du secret *secretName* pour l'étape suivante.

   - Lorsque vous sélectionnez des paires clé/valeur, créez une paire pour la clé `username/USERNAME` contenant la valeur. *saphanaUsername*
   - Lorsque vous sélectionnez des paires clé/valeur, créez une paire pour la clé `password/PASSWORD` contenant la valeur. *saphanaPassword*
2. Dans la console AWS Glue, créez une connexion en suivant les étapes décrites dansthe section called “Ajouter une AWS Glue connexion”. Après avoir créé la connexion, conservez le nom de la connexion pour une utilisation future dans AWS Glue. *connectionName*

   - Lorsque vous sélectionnez un type de connexion, sélectionnez SAP HANA.
   - Lorsque vous fournissez l'URL SAP HANA, indiquez l'URL de votre instance.

     SAP HANA JDBC se présente sous la URLs forme `jdbc:sap://`*saphanaHostname*`:`*saphanaPort*`/?`*databaseName*`=`*saphanaDBname*`,`*Paramete*

     AWS Glue nécessite les paramètres d'URL JDBC suivants :

    - `databaseName` – une base de données par défaut dans SAP HANA à laquelle se connecter.
  - Lorsque vous sélectionnez un AWS secret, fournissez*secretName*.

Après avoir créé une connexion AWS Glue SAP HANA, vous devez suivre les étapes suivantes avant d'exécuter votre tâche AWS Glue :

- Accordez au rôle IAM associé à votre tâche AWS Glue l'autorisation de lire*secretName*.
- Dans la configuration de votre tâche AWS Glue*connectionName*, fournissez une connexion réseau supplémentaire.

Lecture à partir de tables SAP HANA

Prérequis :

- Une table SAP HANA à partir de laquelle vous souhaitez lire. Vous aurez besoin des informations d'identification de la table.

  Une table peut être spécifiée avec un nom de table SAP HANA et un nom de schéma, sous forme de *schemaName*.*tableName*. Le nom du schéma et le séparateur « . » ne sont pas obligatoires si la table se trouve dans le schéma par défaut, « public ». Appelle ça*tableIdentifier*. Notez que la base de données est fournie sous forme de paramètre d'URL JDBC dans `connectionName`.
- Une connexion AWS Glue SAP HANA configurée pour fournir des informations d'authentification. Suivez les étapes de la procédure précédente Pour configurer une connexion à SAP HANA afin de configurer vos informations d'authentification. Vous aurez besoin du nom de la connexion AWS Glue,*connectionName*.

Par exemple :

```
saphana_read_table = glueContext.create_dynamic_frame.from_options(
    connection_type="saphana",
    connection_options={
        "connectionName": "connectionName",
        "dbtable": "tableIdentifier",
    }
)
```

Vous pouvez également fournir une requête SQL SELECT pour filtrer les résultats renvoyés à votre DynamicFrame. Vous devrez configurer `query`.

Par exemple :

```
saphana_read_query = glueContext.create_dynamic_frame.from_options(
    connection_type="saphana",
    connection_options={
        "connectionName": "connectionName",
        "query": "query"
    }
)
```

Écrire dans des tables SAP HANA

Cet exemple écrit les informations d'un fichier existant DynamicFrame *dynamicFrame* dans SAP HANA. Si le tableau contient déjà des informations, AWS Glue génère une erreur.

Prérequis :

- Une table SAP HANA dans laquelle vous souhaitez écrire.

  Une table peut être spécifiée avec un nom de table SAP HANA et un nom de schéma, sous forme de *schemaName*.*tableName*. Le nom du schéma et le séparateur « . » ne sont pas obligatoires si la table se trouve dans le schéma par défaut, « public ». Appelle ça*tableIdentifier*. Notez que la base de données est fournie sous forme de paramètre d'URL JDBC dans `connectionName`.
- Informations d'authentification SAP HANA. Suivez les étapes de la procédure précédente Pour configurer une connexion à SAP HANA afin de configurer vos informations d'authentification. Vous aurez besoin du nom de la connexion AWS Glue,*connectionName*.

Par exemple :

```
options = {
    "connectionName": "connectionName",
    "dbtable": 'tableIdentifier'
}

    saphana_write = glueContext.write_dynamic_frame.from_options(
        frame=dynamicFrame,
        connection_type="saphana",
```

```
        connection_options=options
)
```

Référence des options de connexion de SAP HANA

- `connectionName` — Obligatoire. Utilisé pour la lecture/l'écriture. Nom d'une connexion AWS Glue SAP HANA configurée pour fournir des informations d'authentification et de mise en réseau à votre méthode de connexion.
- `databaseName` : utilisé pour la lecture/l'écriture. Valeurs valides : noms des bases de données dans SAP HANA. Nom de base de données à laquelle se connecter.
- `dbtable` — obligatoire pour l'écriture, obligatoire pour la lecture à moins qu'une `query` ne soit fournie. Utilisé pour la lecture/l'écriture. Valeurs valides : contenu d'une clause SAP HANA SQL FROM. Identifie une table dans SAP HANA à laquelle se connecter. Vous pouvez également fournir un SQL autre qu'un nom de table, comme une sous-requête. Pour plus d'informations, consultez la clause From dans la documentation SAP HANA.
- `query` : utilisé pour la lecture. Une requête SAP HANA SQL SELECT définissant ce qui doit être récupéré lors de la lecture à partir de SAP HANA.

Connexions Snowflake

Vous pouvez utiliser AWS Glue for Spark pour lire et écrire dans des tables dans Snowflake dans AWS Glue 4.0 et versions ultérieures. Vous pouvez lire depuis Snowflake à l'aide d'une requête SQL. Vous pouvez vous connecter à Snowflake à l'aide d'un nom d'utilisateur et d'un mot de passe. Vous pouvez vous référer aux informations d'identification Snowflake stockées dans le catalogue AWS Secrets Manager de données AWS Glue. Les informations d'identification Snowflake du catalogue de données pour AWS Glue for Spark sont stockées séparément des informations d'identification du catalogue de données Snowflake pour les robots d'exploration. Vous devez choisir un type de connexion `SNOWFLAKE` et non un type de connexion `JDBC` configuré pour se connecter à Snowflake.

Pour en savoir plus sur Snowflake, consultez le site web Snowflake. Pour plus d'informations sur Snowflake on AWS, consultez Snowflake Data Warehouse sur Amazon Web Services.

Configuration des connexions Snowflake

Il n'y a aucune AWS condition préalable pour se connecter aux bases de données Snowflake disponibles sur Internet.

Vous pouvez éventuellement effectuer la configuration suivante pour gérer vos informations de connexion avec AWS Glue.

Pour gérer vos informations de connexion avec AWS Glue

1. Dans Snowflake, générez un utilisateur *snowflakeUser* et un mot de passe. *snowflakePassword*
2. Dans AWS Secrets Manager, créez un secret à l'aide de vos informations d'identification Snowflake. Pour créer un secret dans Secrets Manager, suivez le didacticiel disponible dans la section Créer un AWS Secrets Manager secret dans la AWS Secrets Manager documentation. Après avoir créé le secret, conservez le nom du secret *secretName* pour l'étape suivante.
   - Lorsque vous sélectionnez des paires clé/valeur, créez une paire pour *snowflakeUser* avec la clé. `USERNAME`
   - Lorsque vous sélectionnez des paires clé/valeur, créez une paire pour *snowflakePassword* avec la clé. `PASSWORD`
   - Lorsque vous sélectionnez des paires clé/valeur, vous pouvez fournir à votre entrepôt Snowflake la clé `sfWarehouse`.
   - Lorsque vous sélectionnez des paires clé/valeur, vous pouvez fournir des propriétés de connexion Snowflake supplémentaires en utilisant les noms de propriété Spark correspondants comme clés. Les propriétés prises en charge sont les suivantes :
     - `sfDatabase` : nom de la base de données Snowflake
     - `sfSchema` : nom du schéma Snowflake
     - `sfRole` : nom du rôle Snowflake
     - `pem_private_key` : clé privée pour l'authentification par paire de clés
3. Dans le catalogue de données AWS Glue, créez une connexion en choisissant Connections, puis Create connection. Suivez les étapes décrites dans l'assistant de connexion pour finaliser le processus :
   - Lorsque vous sélectionnez une source de données, sélectionnez Snowflake, puis Suivant.
   - Saisissez les détails de connexion tels que l'hôte et le port. Lorsque vous saisissez l'URL Snowflake hôte, indiquez l'URL de votre instance Snowflake. L'URL utilisera généralement un nom d'hôte sous la forme *account_identifier*`.snowflakecomputing.com`. Cependant, le format de l'URL peut varier en fonction du type de compte Snowflake (par exemple AWS, Azure ou hébergé par Snowflake).

- Lorsque vous sélectionnez le rôle de service IAM, choisissez dans le menu déroulant. Il s'agit du rôle IAM de votre compte qui sera utilisé pour accéder à l'adresse IP AWS Secrets Manager et l'attribuer si un VPC est spécifié.
- Lorsque vous sélectionnez un AWS secret, indiquez-le*secretName*.

4. À l'étape suivante de l'assistant, définissez les propriétés de votre connexion Snowflake.
5. Dans la dernière étape de l'assistant, examinez vos paramètres, puis terminez le processus de création de votre connexion.

Dans les situations suivantes, vous pouvez avoir besoin des éléments suivants :

- Pour Snowflake hébergé sur un Amazon AWS VPC
  - Vous aurez besoin d'une configuration Amazon VPC appropriée pour Snowflake. Pour plus d'informations sur la configuration de votre Amazon VPC, consultez AWS PrivateLink et Snowflake dans la documentation Snowflake.
  - Vous aurez besoin d'une configuration Amazon VPC appropriée pour Glue AWS . the section called "Configuration de points de terminaison d'un VPC d'interface (AWS PrivateLink) pour AWS Glue ".
  - Vous devrez créer une connexion AWS Glue Data Catalog qui fournit les informations de connexion Amazon VPC (en plus de l'identifiant d'un AWS Secrets Manager secret qui définit vos informations de sécurité Snowflake). Votre URL changera lors de l'utilisation AWS PrivateLink, comme décrit dans la documentation Snowflake dont le lien figure dans un article précédent.
  - Vous aurez besoin de la configuration de votre tâche pour inclure la connexion au catalogue de données en tant que connexion réseau supplémentaire.

Lecture à partir de tables Snowflake

Conditions préalables : une table Snowflake à partir de laquelle vous souhaitez lire. Vous aurez besoin du nom de la table Snowflake,. *tableName* Vous aurez besoin de votre URL*snowflakeUrl*, nom d'utilisateur *snowflakeUser* et mot de passe Snowflake. *snowflakePassword* Si aucun espace de noms par défaut n'est défini pour votre utilisateur Snowflake, vous aurez besoin du nom de la base de données Snowflake et du nom du schéma. *databaseName schemaName* De plus, si votre utilisateur Snowflake n'a pas défini d'entrepôt par défaut, vous aurez besoin d'un nom d'entrepôt. *warehouseName*

Par exemple :

Conditions préalables supplémentaires : suivez les étapes pour gérer vos informations d'identification de connexion avec AWS Glue pour les configurer*snowflakeUrl*, *snowflakeUsername* et*snowflakePassword*. Pour passer en revue ces étapes, reportez-vous à la section précédente, the section called "Configuration de Snowflake". Pour sélectionner la connexion réseau supplémentaire à laquelle se connecter, nous utiliserons le paramètre `connectionName`.

```
snowflake_read = glueContext.create_dynamic_frame.from_options(
  connection_type="snowflake",
  connection_options={
        "connectionName": "connectionName",
        "dbtable": "tableName",
        "sfDatabase": "databaseName",
        "sfSchema": "schemaName",
        "sfWarehouse": "warehouseName",
    }
)
```

De plus, vous pouvez utiliser les paramètres `autopushdown` et `query` pour lire une partie d'une table Snowflake. Cela peut être nettement plus efficace que de filtrer vos résultats une fois qu'ils ont été chargés dans Spark. Prenons un exemple où toutes les ventes sont stockées dans la même table, mais où vous n'avez besoin d'analyser que les ventes d'un certain magasin pendant les jours fériés. Si ces informations sont stockées dans la table, vous pouvez utiliser le pushdown de prédicat pour récupérer les résultats comme suit :

```
snowflake_node = glueContext.create_dynamic_frame.from_options(
    connection_type="snowflake",
    connection_options={
        "autopushdown": "on",
        "query": "select * from sales where store='1' and IsHoliday='TRUE'",
        "connectionName": "snowflake-glue-conn",
        "sfDatabase": "databaseName",
        "sfSchema": "schemaName",
        "sfWarehouse": "warehouseName",
    }
)
```

Écrire sur les tables Snowflake

Conditions préalables : une base de données Snowflake sur laquelle vous souhaitez écrire. Vous aurez besoin d'un nom de table actuel ou souhaité,*tableName*. Vous aurez besoin de

votre URL*snowflakeUrl*, nom d'utilisateur *snowflakeUser* et mot de passe Snowflake. *snowflakePassword* Si aucun espace de noms par défaut n'est défini pour votre utilisateur Snowflake, vous aurez besoin du nom de la base de données Snowflake et du nom du schéma. *databaseName schemaName* De plus, si votre utilisateur Snowflake n'a pas défini d'entrepôt par défaut, vous aurez besoin d'un nom d'entrepôt. *warehouseName*

Par exemple :

Conditions préalables supplémentaires : suivez les étapes pour gérer vos informations d'identification de connexion avec AWS Glue pour les configurer*snowflakeUrl*, *snowflakeUsername* et*snowflakePassword*. Pour passer en revue ces étapes, reportez-vous à la section précédente, the section called "Configuration de Snowflake". Pour sélectionner la connexion réseau supplémentaire à laquelle se connecter, nous utiliserons le paramètre `connectionName`.

```
glueContext.write_dynamic_frame.from_options(
    connection_type="snowflake",
    connection_options={
        "connectionName": "connectionName",
        "dbtable": "tableName",
        "sfDatabase": "databaseName",
        "sfSchema": "schemaName",
        "sfWarehouse": "warehouseName",
    },
)
```

Référence des options de connexion Snowflake

Le type de connexion Snowflake accepte les options de connexion suivantes :

Vous pouvez récupérer certains paramètres de cette section à partir d'une connexion au catalogue de données (`sfUrl`, `sfUser`, `sfPassword`), auquel cas vous n'êtes pas obligé de les fournir. Pour ce faire, fournissez le paramètre `connectionName`.

Vous pouvez récupérer les paramètres de connexion à partir des AWS Secrets Manager secrets à l'aide du `secretId` paramètre. Lorsque vous utilisez Secrets Manager, les propriétés Spark suivantes peuvent être automatiquement récupérées si elles sont présentes dans le secret :

- `sfUser` (à l'aide d'une clé `USERNAME` ou `sfUser`)
- `sfPassword` (à l'aide d'une clé `PASSWORD` ou `sfPassword`)
- `sfWarehouse` (à l'aide d'une clé `sfWarehouse`)

- `sfDatabase` (à l'aide d'une clé `sfDatabase`)
- `sfSchema` (à l'aide d'une clé `sfSchema`)
- `sfRole` (à l'aide d'une clé `sfRole`)
- `pem_private_key` (à l'aide d'une clé `pem_private_key`)

Ordre de priorité des propriétés : lorsque la même propriété est spécifiée à plusieurs emplacements, AWS Glue utilise l'ordre de priorité suivant (du plus haut au plus bas) :

1. Options de connexion explicitement fournies dans votre code de tâche
2. Propriétés de la connexion au catalogue de données
3. AWS Secrets Manager valeurs secrètes (lorsqu'elles `secretId` sont spécifiées)
4. Paramètres utilisateur par défaut de Snowflake

Les paramètres suivants sont généralement utilisés lors de la connexion à Snowflake.

- `sfDatabase` : obligatoire si aucune valeur par défaut de l'utilisateur n'est définie dans Snowflake. Utilisé pour la lecture/l'écriture. La base de données à utiliser pour la session après la connexion.
- `sfSchema` : obligatoire si aucune valeur par défaut de l'utilisateur n'est définie dans Snowflake. Utilisé pour la lecture/l'écriture. Le schéma à utiliser pour la session après la connexion.
- `sfWarehouse` : obligatoire si aucune valeur par défaut de l'utilisateur n'est définie dans Snowflake. Utilisé pour la lecture/l'écriture. L'entrepôt virtuel par défaut à utiliser pour la session après la connexion.
- `sfRole` : obligatoire si aucune valeur par défaut de l'utilisateur n'est définie dans Snowflake. Utilisé pour la lecture/l'écriture. Le rôle de sécurité par défaut à utiliser pour la session après la connexion.
- `sfUrl` : (obligatoire) utilisé pour la lecture/l'écriture. Indique le nom d'hôte de votre compte au format suivant : *`account_identifier`*`.snowflakecomputing.com`. Pour plus d'informations sur les identifiants de compte, consultez Identifiants de compte dans la documentation de Snowflake.
- `sfUser` : (obligatoire) utilisé pour la lecture/l'écriture. Nom de connexion de l'utilisateur Snowflake.
- `sfPassword` : (Obligatoire sauf si la clé `pem_private_key` est fournie) utilisé pour la lecture/l'écriture. Mot de passe de l'utilisateur Snowflake.
- `dbtable` : obligatoire lorsque vous travaillez avec des tables complètes. Utilisé pour la lecture/l'écriture. Le nom de la table à lire ou de la table dans laquelle les données sont écrites. Lors de la lecture, toutes les colonnes et tous les enregistrements sont récupérés.

- `pem_private_key` : utilisé pour la lecture/l'écriture. Chaîne de clé privée codée en b64 non chiffrée. Clé privée de l'utilisateur Snowflake. Il est courant de la copier à partir d'un fichier PEM. Pour plus d'informations, consultez Authentification par paire de clés et rotation des paires de clés dans la documentation Snowflake.
- `query` : obligatoire lors de la lecture avec une requête. Utilisé pour la lecture. La requête exacte (instruction `SELECT`) à exécuter

Les options suivantes sont utilisées pour configurer des comportements spécifiques lors du processus de connexion à Snowflake.

- `preactions` : utilisé pour la lecture/l'écriture. Valeurs valides : liste d'instructions SQL séparées par des points-virgules sous forme de chaîne. Les instructions SQL sont exécutées avant le transfert des données entre AWS Glue et Snowflake. Si une instruction contient `%s`, `%s` est remplacé par le nom de table référencé pour l'opération.
- `postactions` : utilisé pour la lecture/l'écriture. Les instructions SQL sont exécutées après le transfert des données entre AWS Glue et Snowflake. Si une instruction contient `%s`, `%s` est remplacé par le nom de table référencé pour l'opération.
- `autopushdown` : `"on"` par défaut. Valeurs valides : `"on"`, `"off"`. Ce paramètre contrôle si le pushdown automatique des requêtes est activé. Lorsque l'option pushdown est activée, si une partie de la requête peut être « poussée vers le bas » sur le serveur Snowflake, elle est poussée vers le bas au moment de l'exécution de la requête sur Spark. Cela améliore les performances de certaines requêtes. Pour savoir si votre requête peut être poussée vers le bas, consultez Pushdown dans la documentation de Snowflake.

En outre, certaines des options disponibles sur le connecteur Snowflake Spark peuvent être prises en charge dans Glue AWS . Pour plus d'informations sur les options disponibles sur le connecteur Snowflake Spark, consultez la section Réglage des options de configuration du connecteur dans la documentation Snowflake.

Méthodes d'authentification Snowflake

AWS Glue prend en charge les méthodes d'authentification suivantes pour se connecter à Snowflake :

- Authentification par nom d'utilisateur et mot de passe : indiquez les paramètres `sfUser` et `sfPassword`.

- Authentification par paire de clés : indiquez les paramètres `sfUser` et `pem_private_key`. Lorsque vous utilisez l'authentification par paire de clés, le paramètre `sfPassword` n'est pas obligatoire.

Les deux méthodes d'authentification sont entièrement prises en charge et peuvent être configurées à l'aide de n'importe quelle combinaison d'options de connexion, de connexions au catalogue de données ou de AWS Secrets Manager secrets.

Limites du connecteur Snowflake

La connexion à Snowflake avec AWS Glue for Spark est soumise aux restrictions suivantes.

- Ce connecteur ne prend pas en charge les signets de tâches. Pour plus d'informations sur les signets de tâche, consultez the section called "Suivi des données traitées à l'aide de signets de tâche".
- Ce connecteur ne prend pas en charge les lectures et écritures Snowflake via les tables du catalogue de données AWS Glue à l'aide des méthodes `create_dynamic_frame.from_catalog` et`write_dynamic_frame.from_catalog`.
- Ce connecteur prend en charge username/password l'authentification et l'authentification par paire de clés. Les autres méthodes d'authentification (telles que OAuth SAML) ne sont actuellement pas prises en charge.
- Ce connecteur n'est pas pris en charge dans les tâches de streaming.
- Ce connecteur prend en charge les requêtes basées sur des instructions `SELECT` lors de la récupération d'informations (par exemple avec le paramètre `query`). Les autres types de requêtes (telles que `SHOW`, `DESC` ou les instructions DML) ne sont pas pris en charge.
- Snowflake limite la taille du texte de la requête (c'est-à-dire les instructions SQL) soumis par les clients Snowflake à 1 Mo par instruction. Pour plus d'informations, consultez Limites de la taille du texte de requête.

Connexions Teradata Vantage

Vous pouvez utiliser AWS Glue for Spark pour lire et écrire dans des tables existantes dans Teradata Vantage dans AWS Glue 4.0 et versions ultérieures. Vous pouvez définir ce qu'il faut lire dans Teradata à l'aide d'une requête SQL. Vous pouvez vous connecter à Teradata à l'aide du nom d'utilisateur et du mot de passe enregistrés AWS Secrets Manager via une connexion AWS Glue.

Pour de plus amples informations sur Teradata, veuillez consulter la documentation Teradata

Configuration de connexions Teradata

Pour vous connecter à Teradata depuis AWS Glue, vous devez créer et stocker vos informations d'identification Teradata dans un AWS Secrets Manager secret, puis associer ce secret à une connexion AWS Glue Teradata. Si votre instance Teradata se trouve dans un Amazon VPC, vous devrez également fournir des options réseau à votre connexion AWS Glue Teradata.

Pour vous connecter à Teradata depuis AWS Glue, vous aurez peut-être besoin de certaines conditions préalables :

- Si vous accédez à votre environnement Teradata via Amazon VPC, configurez Amazon VPC pour permettre à votre tâche AWS Glue de communiquer avec l'environnement Teradata. Nous vous déconseillons d'accéder à l'environnement Teradata via l'Internet public.

  Dans Amazon VPC, identifiez ou créez un VPC, un sous-réseau et un groupe de sécurité que AWS Glue utilisera lors de l'exécution de la tâche. En outre, vous devez vous assurer qu'Amazon VPC est configuré pour autoriser le trafic réseau entre votre instance Teradata et cet emplacement. Votre tâche devra établir une connexion TCP avec votre port client Teradata. Pour de plus amples informations sur les ports Teradata, consultez la documentation Teradata.

  Selon la configuration de votre réseau, la connectivité VPC sécurisée peut nécessiter des modifications dans Amazon VPC et dans d'autres services réseau. Pour plus d'informations sur AWS la connectivité, consultez les options de AWS connectivité dans la documentation Teradata.

Pour configurer une connexion AWS Glue Teradata :

1. Dans votre configuration Teradata, identifiez ou créez un utilisateur et un mot de passe auxquels AWS Glue se connectera, *`teradataUser`* et*`teradataPassword`*. Pour plus d'informations, consultez la présentation de Vantage Security dans la documentation Teradata.
2. Dans AWS Secrets Manager, créez un secret à l'aide de vos informations d'identification Teradata. Pour créer un secret dans Secrets Manager, suivez le didacticiel disponible dans la section Créer un AWS Secrets Manager secret dans la AWS Secrets Manager documentation. Après avoir créé le secret, conservez le nom du secret *`secretName`* pour l'étape suivante.

   - Lorsque vous sélectionnez des paires clé/valeur, créez une paire pour la clé `user` contenant la valeur. *`teradataUsername`*
   - Lorsque vous sélectionnez des paires clé/valeur, créez une paire pour la clé `password` contenant la valeur. *`teradataPassword`*

3. Dans la console AWS Glue, créez une connexion en suivant les étapes décrites dansthe section called "Ajouter une AWS Glue connexion". Après avoir créé la connexion, conservez le nom de la connexion pour l'étape suivante. *connectionName*

   - Lorsque vous sélectionnez un type de connexion, sélectionnez Teradata.
   - Lorsque vous fournissez l'URL JDBC, indiquez l'URL de votre instance. Vous pouvez également coder en dur certains paramètres de connexion séparés par des virgules dans votre URL JDBC. L'URL doit être conforme au format suivant : `jdbc:teradata://`*teradataHostname*`/`*ParameterName*`=`*ParameterValue*`,`*ParameterName*

     Les paramètres d'URL pris en charge sont les suivants :

     - `DATABASE` – nom de la base de données sur l'hôte à laquelle accéder par défaut.
     - `DBS_PORT` – le port de base de données utilisé lors de l'exécution sur un port non standard.
   - Lorsque vous sélectionnez un type d'identifiant, sélectionnez AWS Secrets Manager, puis définissez AWS Secret sur*secretName*.
4. Dans les situations suivantes, vous pouvez avoir besoin d'une configuration supplémentaire :

   - Pour les instances Teradata hébergées sur AWS un Amazon VPC
     - Vous devrez fournir les informations de connexion Amazon VPC à la connexion AWS Glue qui définit vos informations de sécurité Teradata. Lorsque vous créez ou mettez à jour votre connexion, définissez le VPC, le sous-réseau et les groupes de sécurité dans les options réseau.

Après avoir créé une connexion AWS Glue Teradata, vous devez effectuer les étapes suivantes avant d'appeler votre méthode de connexion.

- Accordez au rôle IAM associé à votre tâche AWS Glue l'autorisation de lire*secretName*.
- Dans la configuration de votre tâche AWS Glue*connectionName*, fournissez une connexion réseau supplémentaire.

Lecture à partir de Teradata

Prérequis :

- Une table Teradata à partir de laquelle vous souhaitez lire. Vous aurez besoin du nom de la table,*tableName*.

- Une connexion AWS Glue Teradata configurée pour fournir des informations d'authentification. Suivez les étapes pour configurer une connexion à Teradata afin de configurer vos informations d'authentification. Vous aurez besoin du nom de la connexion AWS Glue,*connectionName*.

Par exemple :

```
teradata_read_table = glueContext.create_dynamic_frame.from_options(
    connection_type="teradata",
    connection_options={
        "connectionName": "connectionName",
        "dbtable": "tableName"
    }
)
```

Vous pouvez également fournir une requête SQL SELECT pour filtrer les résultats renvoyés à votre DynamicFrame. Vous devrez configurer `query`.

Par exemple :

```
teradata_read_query = glueContext.create_dynamic_frame.from_options(
    connection_type="teradata",
    connection_options={
        "connectionName": "connectionName",
        "query": "query"
    }
)
```

Écrire dans les tables Teradata

Prérequis : une table Teradata dans laquelle vous souhaitez écrire,. *tableName* Vous devez créer la table avant d'appeler la méthode de connexion.

Par exemple :

```
teradata_write = glueContext.write_dynamic_frame.from_options(
    connection_type="teradata",
    connection_options={
        "connectionName": "connectionName",
        "dbtable": "tableName"
    }
```

```
)
```

Référence des options de connexion Teradata

- `connectionName` — Obligatoire. Utilisé pour la lecture/l'écriture. Nom d'une connexion AWS Glue Teradata configurée pour fournir des informations d'authentification et de mise en réseau à votre méthode de connexion.
- `dbtable` — obligatoire pour l'écriture, obligatoire pour la lecture à moins qu'une `query` ne soit fournie. Utilisé pour la lecture/l'écriture. Le nom de la table avec lequelle votre méthode de connexion va interagir.
- `query` : utilisé pour la lecture. Une requête SELECT SQL définissant ce qui doit être récupéré lors de la lecture à partir de Teradata.

Connexions Teradata Vantage NOS

La connexion Teradata NOS (Native Object Store) est une nouvelle connexion pour Teradata Vantage qui utilise la requête Teradata WRITE_NOS pour lire des tables existantes et la requête READ_NOS pour écrire dans des tables. Ces requêtes utilisent Amazon S3 comme répertoire intermédiaire. Le connecteur Teradata NOS est donc plus rapide que le connecteur Teradata existant (basé sur JDBC), surtout lorsqu'il s'agit de gérer de grandes quantités de données.

Vous pouvez utiliser la connexion Teradata NOS dans Spark AWS Glue pour lire et écrire dans des tables existantes dans Teradata Vantage dans les AWS Glue versions 5.0 et ultérieures. Vous pouvez définir ce qu'il faut lire dans Teradata à l'aide d'une requête SQL. Vous pouvez vous connecter à Teradata à l'aide des identifiants de nom d'utilisateur et de mot de passe enregistrés AWS Secrets Manager via une AWS Glue connexion.

Pour plus d'informations sur Teradata, veuillez consulter la documentation Teradata

Rubriques

- Création d'une connexion Teradata NOS
- Lecture à partir de tables Teradata
- Écrire dans les tables Teradata
- Référence des options de connexion Teradata
- Fournir des options dans l'interface utilisateur AWS Glue Visual ETL

Création d'une connexion Teradata NOS

Pour vous connecter à Teradata NOS depuis AWS Glue, vous devez créer et stocker vos informations d'identification Teradata dans un AWS Secrets Manager secret, puis associer ce secret à une connexion AWS Glue Teradata NOS. Si votre instance Teradata se trouve dans un Amazon VPC, vous devrez également fournir des options réseau à AWS Glue votre connexion Teradata NOS.

Prérequis :

- Si vous accédez à votre environnement Teradata via Amazon VPC, configurez Amazon VPC pour permettre à AWS Glue votre tâche de communiquer avec l'environnement Teradata. Nous vous déconseillons d'accéder à l'environnement Teradata via l'Internet public.
- Dans Amazon VPC, identifiez ou créez un VPC, un sous-réseau et un groupe de sécurité qui AWS Glue seront utilisés lors de l'exécution de la tâche. En outre, vous devez vous assurer qu'Amazon VPC est configuré pour autoriser le trafic réseau entre votre instance Teradata et cet emplacement. Votre tâche devra établir une connexion TCP avec votre port client Teradata. Pour plus d'informations sur les ports Teradata, consultez Security Groups for Teradata Vantage.
- Selon la configuration de votre réseau, la connectivité VPC sécurisée peut nécessiter des modifications dans Amazon VPC et dans d'autres services réseau. Pour plus d'informations sur AWS la connectivité, consultez la section Options de AWS connectivité dans la documentation Teradata.

Pour configurer une connexion AWS Glue Teradata NOS :

1. Dans votre configuration Teradata, identifiez ou créez un *`teradataUsername`* et auquel *`teradataPassword`* AWS Glue vous vous connecterez. Pour plus d'informations, consultez Vantage Security Overview dans la documentation Teradata.
2. Dans AWS Secrets Manager, créez un secret à l'aide de vos informations d'identification Teradata. Pour créer un secret dans AWS Secrets Manager, suivez le didacticiel disponible dans la section Créer un AWS Secrets Manager secret dans la AWS Secrets Manager documentation. Après avoir créé le secret, conservez le nom du secret *`secretName`* pour l'étape suivante.
   - Lorsque vous sélectionnez des Key/value paires, créez une paire pour la clé USERNAME avec la valeur*`teradataUsername`*.
   - Lorsque vous sélectionnez des Key/value paires, créez une paire pour la clé PASSWORD avec la valeur*`teradataPassword`*.

3. Dans la AWS Glue console, créez une connexion en suivant les étapes décrites dans Ajouter une AWS Glue connexion. Après avoir créé la connexion, conservez le nom de la connexion pour l'étape suivante. *connectionName*

   - Lorsque vous sélectionnez un type de connexion, sélectionnez Teradata Vantage NOS.
   - Lorsque vous fournissez l'URL JDBC, indiquez l'URL de votre instance. Vous pouvez également coder en dur certains paramètres de connexion séparés par des virgules dans votre URL JDBC. L'URL doit être conforme au format suivant : `jdbc:teradata://teradataHostname/ParameterName=ParameterValue,ParameterName=ParameterValue` .
   - Les paramètres d'URL pris en charge sont les suivants :
     - `DATABASE` – nom de la base de données sur l'hôte à laquelle accéder par défaut.
     - `DBS_PORT` – le port de base de données utilisé lors de l'exécution sur un port non standard.
   - Lorsque vous sélectionnez un type d'identifiant, sélectionnez AWS Secrets Manager, puis définissez AWS Secret sur*secretName*.
4. Dans les situations suivantes, vous pouvez avoir besoin d'une configuration supplémentaire :

   - Pour les instances Teradata AWS hébergées sur un Amazon VPC, vous devrez fournir les informations de connexion Amazon VPC à AWS Glue la connexion qui définit vos informations de sécurité Teradata. Lorsque vous créez ou mettez à jour votre connexion, définissez le VPC, le sous-réseau et les groupes de sécurité dans les options réseau.

Après avoir créé une connexion AWS Glue Teradata Vantage NOS, vous devez effectuer les étapes suivantes avant d'appeler votre méthode de connexion.

1. Accordez au rôle IAM associé à votre AWS Glue travail l'autorisation de lire*secretName*.
2. Dans la configuration de votre AWS Glue tâche, *connectionName* indiquez une connexion réseau supplémentaire sous Connexions.

Lecture à partir de tables Teradata

Prérequis :

- Une table Teradata à partir de laquelle vous souhaitez lire. Vous aurez besoin du nom de la table,*tableName*.

- L'environnement Teradata dispose d'un accès en écriture au chemin Amazon S3 spécifié par l'`staging_fs_url`option,*`stagingFsUrl`*.
- Le rôle IAM associé à la AWS Glue tâche dispose d'un accès en écriture à l'emplacement Amazon S3 spécifié par `staging_fs_url` option.
- Une connexion AWS Glue Teradata NOS configurée pour fournir des informations d'authentification. Suivez la procédure Pour configurer une connexion AWS Glue Teradata NOS : pour configurer vos informations d'authentification. Vous aurez besoin du nom de la AWS Glue connexion,*`connectionName`*.

Exemple :

```
teradata_read_table = glueContext.create_dynamic_frame.from_options(
    connection_type= "teradatanos",
    connection_options={
        "connectionName": "connectionName",
        "dbtable": "tableName",
        "staging_fs_url": "stagingFsUrl"
    }
)
```

Vous pouvez également fournir une requête SQL SELECT pour filtrer les résultats renvoyés à votre DynamicFrame. Vous devrez configurer query. Si vous configurez à la fois dbTable et query, le connecteur ne parvient pas à lire les données. Par exemple :

```
teradata_read_query = glueContext.create_dynamic_frame.from_options(
    connection_type="teradatanos",
    connection_options={
        "connectionName": "connectionName",
        "query": "query",
        "staging_fs_url": "stagingFsUrl"
    }
)
```

En outre, vous pouvez utiliser l' DataFrame API Spark pour lire des tables Teradata. Par exemple :

```
options = {
    "url": "JDBC_URL",
    "dbtable": "tableName",
    "user": "teradataUsername", # or use "username" as key here
    "password": "teradataPassword",
```

```
    "staging_fs_url": "stagingFsUrl"
}
teradata_read_table = spark.read.format("teradatanos").option(**options).load()
```

Écrire dans les tables Teradata

Conditions préalables

- Une table Teradata dans laquelle vous souhaitez écrire :*tableName*.
- L'environnement Teradata dispose d'un accès en lecture à l'emplacement Amazon S3 spécifié par l'`staging_fs_url`option, *stagingFsUrl* .
- Le rôle IAM associé à la AWS Glue tâche dispose d'un accès en écriture à l'emplacement Amazon S3 spécifié par `staging_fs_url` option.
- Une connexion AWS Glue Teradata configurée pour fournir des informations d'authentification. Suivez la procédure indiquée dans Pour configurer une connexion AWS Glue Teradata NOS : pour configurer vos informations d'authentification. Vous aurez besoin du nom de la AWS Glue connexion,*connectionName*.

  Par exemple :

```
teradata_write = glueContext.write_dynamic_frame.from_options(
    frame=dynamicFrame,
    connection_type= "teradatanos",
    connection_options={
        "connectionName": "connectionName",
        "dbtable": "tableName",
        "staging_fs_url": "stagingFsUrl"
    }
)
```

Référence des options de connexion Teradata

Options de connexion et d'opération :

- `connectionName` — Obligatoire. Utilisé pour la lecture/l'écriture. Nom d'une connexion AWS Glue Teradata configurée pour fournir des informations d'authentification et de mise en réseau à votre méthode de connexion.
- `staging_fs_url` — Obligatoire. Utilisé pour la lecture/l'écriture. Emplacement accessible en écriture dans Amazon S3, à utiliser pour les données déchargées lors de la lecture depuis

Teradata, et pour les données Parquet à charger dans Redshift lors de l'écriture sur Teradata. Le compartiment S3 doit se trouver dans la même région que celle de vos AWS Glue tâches.

- `dbtable` — obligatoire pour l'écriture, obligatoire pour la lecture à moins qu'une `query` ne soit fournie. Utilisé pour la lecture/l'écriture. Le nom de la table avec lequelle votre méthode de connexion va interagir.
- `query` : utilisé pour la lecture. Une requête SELECT SQL définissant ce qui doit être récupéré lors de la lecture à partir de Teradata. Vous ne pouvez pas réussir si l'option `dbtable` est fournie.
- `clean_staging_s3_dir`— Facultatif. Utilisé pour la lecture/l'écriture. Si la valeur est true, nettoyez les objets Amazon S3 intermédiaires après une lecture ou une écriture. La valeur par défaut est True.
- `pre_actions`— Facultatif. Utilisé pour écrire. Liste séparée par des points-virgules des commandes SQL exécutées avant le transfert des données entre Spark et Teradata Vantage.
- `post_actions`— Facultatif. Utilisé pour écrire. Liste séparée par des points-virgules des commandes SQL exécutées après le transfert des données entre Spark et Teradata Vantage.
- `truncate`— Facultatif. Utilisé pour écrire. Si la valeur est true, le connecteur tronque la table lors de l'écriture en mode de réécriture. Si la valeur est false, le connecteur supprime la table lors de l'écriture en mode de réécriture. La valeur par défaut est false.
- `create_table_script`— Facultatif. Utilisé pour écrire. Une instruction SQL pour créer une table lors de l'écriture sur Teradata Vantage. Cette option est utile lorsque vous souhaitez créer une table avec des métadonnées personnalisées (par exemple, table CREATE MULTISET ou SET ou modifier l'index principal). Notez que le nom de table utilisé dans le script de création de table doit correspondre au nom de table spécifié dans l'option `dbtable`.
- `partition_size_in_mb`— Facultatif. Utilisé pour la lecture. Taille maximale d'une partition Spark en mégaoctets lors de la lecture d'objets Amazon S3 intermédiaires. La valeur par défaut est 128.

Vous pouvez fournir des options avancées lors de la création d'un nœud Teradata. Ces options sont les mêmes que celles disponibles lors de la programmation AWS Glue de scripts Spark.

Consultez the section called "Connexions Teradata Vantage".

Options d'autorisation :

Vous trouverez ci-dessous les options utilisées pour fournir les informations d'identification du AWS compte que le connecteur utilise pour accéder au compartiment Amazon S3 intermédiaire.

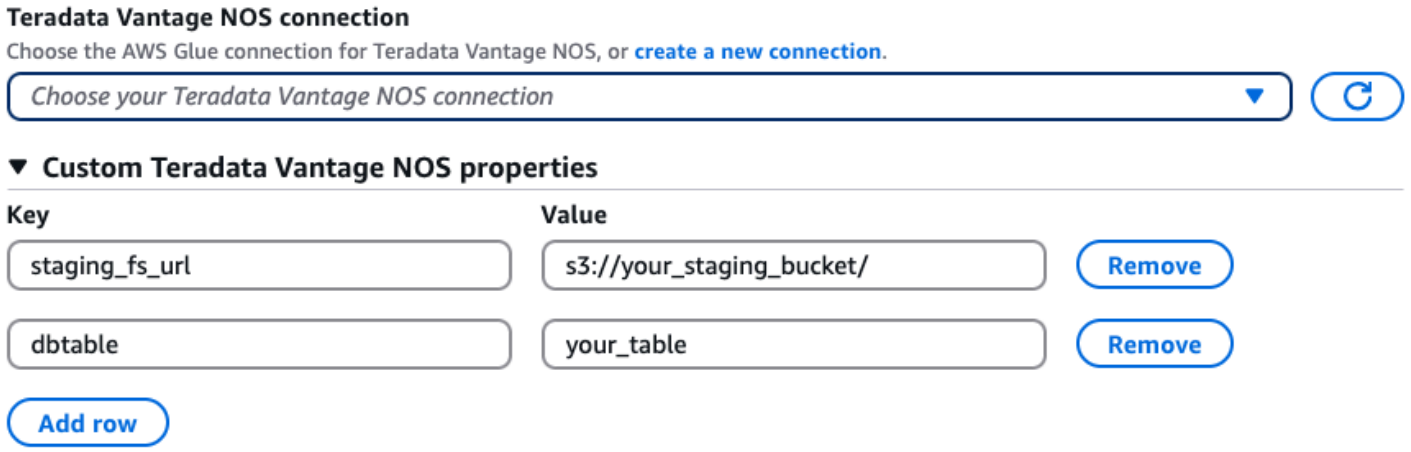
Vous pouvez choisir (1) de ne fournir aucune option d'autorisation et d'utiliser des informations d'identification temporaires générées à partir de votre rôle d' AWS Glue exécution ; ou (2) de fournir un objet d'autorisation `auth_object` que vous avez créé ; ou (3) de le fournir `aws_access_key_id and aws_secret_access_key` si vous utilisez des informations d'identification à long terme, ou de fournir `aws_access_keyaws_secret_access_key`, et `aws_session_token` si vous utilisez des informations d'identification temporaires.

- `auth_object` : facultatif. Utilisé pour accéder au compartiment Amazon S3 intermédiaire. Chaîne d'objets d'autorisation créée dans une instance Teradata. Si elle est fournie, le connecteur utilisera cet objet d'autorisation pour accéder au compartiment Amazon S3 intermédiaire. Si elles ne sont pas fournies, `aws_access_key_id` et qu'elles ne le `aws_secret_access_key` sont pas non plus, une information d'identification temporaire sera extraite du rôle AWS Glue d'exécution et utilisée par le connecteur. Le AWS compte associé à cet objet d'autorisation doit se trouver dans la même région que vos AWS Glue tâches et votre compartiment Amazon S3 intermédiaire ou être configuré avec la confiance entre comptes.
- `aws_access_key_id` : facultatif. Utilisé pour accéder au compartiment Amazon S3 intermédiaire. Fait partie des informations d'identification de sécurité d'un AWS compte. Si `auth_object` n'est pas fourni et si `aws_access_key_id` est fourni avec `aws_secret_access_key`, le connecteur les utilisera pour accéder au compartiment Amazon S3 intermédiaire. Le AWS compte associé à cette clé d'accès doit se trouver dans la même région que vos AWS Glue tâches et votre compartiment Amazon S3 intermédiaire ou être configuré avec la confiance entre comptes.
- `aws_secret_access_key` : facultatif. Utilisé pour accéder au compartiment Amazon S3 intermédiaire. Fait partie des informations d'identification de sécurité d'un AWS compte. Si `auth_object` n'est pas fourni et si `aws_secret_access_key` est fourni avec `aws_access_key_id`, le connecteur les utilisera pour accéder au compartiment Amazon S3 intermédiaire. Le AWS compte associé à cette clé secrète doit se trouver dans la même région que vos AWS Glue tâches et votre compartiment Amazon S3 intermédiaire ou être configuré avec la confiance entre comptes.
- `aws_session_token` : facultatif. Utilisé pour accéder au compartiment Amazon S3 intermédiaire. Fait partie d'un identifiant de sécurité temporaire d'un AWS compte. Doit être fourni avec `aws_access_key_id` et `aws_secret_access_key`.

Fournir des options dans l'interface utilisateur AWS Glue Visual ETL

Vous pouvez indiquer toutes les options ci-dessus dans l'interface utilisateur de votre tâche ETL visuel. Pour l'option connectionName, vous devez choisir dans la liste déroulante des connexions

Teradata Vantage NOS. Pour toutes les autres options, vous devez les fournir via les propriétés personnalisées de Teradata Vantage NOS sous forme de paires clé-valeur.



Connexions Vertica

Vous pouvez utiliser AWS Glue for Spark pour lire et écrire dans des tables dans Vertica dans AWS Glue 4.0 et versions ultérieures. Vous pouvez définir ce qu'il faut lire dans Vertica à l'aide d'une requête SQL. Vous vous connectez à Vertica à l'aide du nom d'utilisateur et du mot de passe enregistrés AWS Secrets Manager via une connexion AWS Glue.

Pour de plus amples informations sur Vertica, consultez la documentation Vertica.

Configuration de connexions Vertica

Pour vous connecter à Vertica depuis AWS Glue, vous devez créer et stocker vos informations d'identification Vertica dans un AWS Secrets Manager secret, puis associer ce secret à une connexion Vertica AWS Glue. Si votre instance Vertica se trouve dans un Amazon VPC, vous devrez également fournir des options réseau à votre connexion AWS Glue Vertica. Vous aurez besoin d'un compartiment Amazon S3 ou d'un dossier à utiliser pour le stockage temporaire lors de la lecture et de l'écriture dans la base de données.

Pour vous connecter à Vertica depuis AWS Glue, vous aurez besoin de certaines conditions préalables :

- Un compartiment ou un dossier Amazon S3 à utiliser pour le stockage temporaire lors de la lecture et de l'écriture dans la base de données, désigné par `tempS3Path`.

> **Note**
>
> Lorsque vous utilisez Vertica dans les aperçus des données des tâches AWS Glue, il est possible que les fichiers temporaires ne soient pas automatiquement supprimés. *tempS3Path* Pour garantir la suppression des fichiers temporaires, mettez directement fin à la session de prévisualisation des données en choisissant Mettre fin à la session dans le volet Prévisualisation des données.
> Si vous ne pouvez pas garantir la fin directe de la session de prévisualisation des données, pensez à configurer le cycle de vie d'Amazon S3 pour supprimer les anciennes données. Nous recommandons de supprimer les données de plus de 49 heures, sur la base de la durée d'exécution maximale des tâches plus une marge. Pour de plus amples informations sur la configuration du cycle de vie Amazon S3, consultez Gestion du cycle de vie de votre stockage dans la documentation Amazon S3.

- Une politique IAM avec les autorisations appropriées pour votre chemin Amazon S3 que vous pouvez associer à votre rôle de travail AWS Glue.
- Si votre instance Vertica se trouve dans un Amazon VPC, configurez Amazon VPC pour permettre à votre tâche AWS Glue de communiquer avec l'instance Vertica sans que le trafic ne transite par l'Internet public.

  Dans Amazon VPC, identifiez ou créez un VPC, un sous-réseau et un groupe de sécurité que AWS Glue utilisera lors de l'exécution de la tâche. En outre, vous devez vous assurer qu'Amazon VPC est configuré pour autoriser le trafic réseau entre votre instance Vertica et cet emplacement. Votre tâche devra établir une connexion TCP avec votre port client Vertica (par défaut 5433). Selon la configuration de votre réseau, cela peut nécessiter des modifications des règles du groupe de sécurité, du réseau ACLs, des passerelles NAT et des connexions d'appairage.

Vous pouvez ensuite configurer AWS Glue pour l'utiliser avec Vertica.

Pour configurer une connexion à Vertica :

1. Dans AWS Secrets Manager, créez un secret à l'aide de vos informations d'identification Vertica, *verticaUsername* et*verticaPassword*. Pour créer un secret dans Secrets Manager, suivez le didacticiel disponible dans la section Créer un AWS Secrets Manager secret dans la AWS Secrets Manager documentation. Après avoir créé le secret, conservez le nom du secret *secretName* pour l'étape suivante.

- Lorsque vous sélectionnez des paires clé/valeur, créez une paire pour la clé `user` contenant la valeur. *verticaUsername*
- Lorsque vous sélectionnez des paires clé/valeur, créez une paire pour la clé `password` contenant la valeur. *verticaPassword*

2. Dans la console AWS Glue, créez une connexion en suivant les étapes décrites dansthe section called "Ajouter une AWS Glue connexion". Après avoir créé la connexion, conservez le nom de la connexion pour l'étape suivante. *connectionName*

   - Lorsque vous sélectionnez un type de connexion, sélectionnez Vertica.
   - Lorsque vous sélectionnez l'hôte Vertica, indiquez le nome d'hôte de votre installation Vertica.
   - Lorsque vous sélectionnez le port Vertica, le port via lequel votre installation Vertica est disponible.
   - Lorsque vous sélectionnez un AWS secret, indiquez-le*secretName*.

3. Dans les situations suivantes, vous pouvez avoir besoin d'une configuration supplémentaire :

   - Pour les instances Vertica hébergées AWS dans un Amazon VPC
     - Fournissez les informations de connexion Amazon VPC à la connexion AWS Glue qui définit vos informations de sécurité Vertica. Lorsque vous créez ou mettez à jour votre connexion, définissez le VPC, le sous-réseau et les groupes de sécurité dans les options réseau.

Après avoir créé une connexion AWS Glue Vertica, vous devez effectuer les étapes suivantes avant d'appeler votre méthode de connexion.

- Accordez au rôle IAM associé à votre tâche AWS Glue des autorisations à*tempS3Path*.
- Accordez au rôle IAM associé à votre tâche AWS Glue l'autorisation de lire*secretName*.
- Dans la configuration de votre tâche AWS Glue*connectionName*, fournissez une connexion réseau supplémentaire.

Lecture à partir de Vertica

Prérequis :

- Une table Vertica à partir de laquelle vous souhaitez lire. Vous aurez besoin du nom de la base de données Vertica *dbName* et du nom de la table. *tableName*

- Une connexion AWS Glue Vertica configurée pour fournir des informations d'authentification. Suivez les étapes de la procédure précédente Pour configurer une connexion à Vertica afin de configurer vos informations d'authentification. Vous aurez besoin du nom de la connexion AWS Glue,*connectionName*.
- Un compartiment ou un dossier Amazon S3 à utiliser pour le stockage temporaire, mentionné précédemment. Vous aurez besoin du nom,*tempS3Path*. Vous devrez vous connecter à cet emplacement à l'aide du protocole `s3a`.

Par exemple :

```
dynamicFrame = glueContext.create_dynamic_frame.from_options(
    connection_type="vertica",
    connection_options={
        "connectionName": "connectionName",
        "staging_fs_url": "s3a://tempS3Path",
        "db": "dbName",
        "table": "tableName",
    }
)
```

Vous pouvez également fournir une requête SQL SELECT pour filtrer les résultats qui vous sont renvoyés DynamicFrame ou pour accéder à un ensemble de données à partir de plusieurs tables.

Par exemple :

```
dynamicFrame = glueContext.create_dynamic_frame.from_options(
    connection_type="vertica",
    connection_options={
        "connectionName": "connectionName",
        "staging_fs_url": "s3a://tempS3Path",
        "db": "dbName",
        "query": "select * FROM tableName",
    },
)
```

Écrire dans les tables Vertica

Cet exemple écrit les informations d'un fichier existant DynamicFrame *dynamicFrame* dans Vertica. Si le tableau contient déjà des informations, AWS Glue ajoutera les données de votre DynamicFrame.

Prérequis :

- Nom de table actuel ou souhaité*tableName*, dans lequel vous souhaitez écrire. Vous aurez également besoin du nom de base de données Vertica correspondant,*dbName*.
- Une connexion AWS Glue Vertica configurée pour fournir des informations d'authentification. Suivez les étapes de la procédure précédente Pour configurer une connexion à Vertica afin de configurer vos informations d'authentification. Vous aurez besoin du nom de la connexion AWS Glue,*connectionName*.
- Un compartiment ou un dossier Amazon S3 à utiliser pour le stockage temporaire, mentionné précédemment. Vous aurez besoin du nom,*tempS3Path*. Vous devrez vous connecter à cet emplacement à l'aide du protocole `s3a`.

Par exemple :

```
glueContext.write_dynamic_frame.from_options(
    frame=dynamicFrame,
    connection_type="vertica",
    connection_options={
        "connectionName": "connectionName",
        "staging_fs_url": "s3a://tempS3Path",
        "db": "dbName",
        "table": "tableName",
    }
)
```

Référence des options de connexion Vertica

- `connectionName` — Obligatoire. Utilisé pour la lecture/l'écriture. Le nom d'une connexion AWS Glue Vertica configurée pour fournir des informations d'authentification et de réseau à votre méthode de connexion.
- db — Obligatoire. Utilisé pour la lecture/l'écriture. Le nom d'une base de données dans Vertica avec lequelle votre méthode de connexion va interagir.
- `dbSchema` — obligatoire si nécessaire pour identifier votre table. Utilisé pour la lecture/l'écriture. Valeur par défaut : `public`. Le nom d'un schéma avec lequel votre méthode de connexion va interagir.

- `table` — obligatoire pour l'écriture, obligatoire pour la lecture à moins qu'une `query` ne soit fournie. Utilisé pour la lecture/l'écriture. Le nom de la table avec lequelle votre méthode de connexion va interagir.
- `query` : utilisé pour la lecture. Une requête SELECT SQL définissant ce qui doit être récupéré lors de la lecture à partir de Teradata.
- `staging_fs_url` — Obligatoire. Utilisé pour la lecture/l'écriture. Valeurs valides : `s3a` URLs. L'URL d'un compartiment ou d'un dossier Amazon S3 à utiliser pour le stockage temporaire.

DataFrame options pour l'ETL dans AWS Glue 5.0 pour Spark

A DataFrame est un ensemble de données organisé en colonnes nommées semblables à une table et prenant en charge les opérations de style fonctionnel (map/reduce/filter/etc.) et les opérations SQL (sélection, projet, agrégation).

Pour créer un DataFrame pour une source de données prise en charge par Glue, les éléments suivants sont requis :

- connecteur de source de données `ClassName`
- connexion de source de données `Options`

De même, pour écrire un dans DataFrame un récepteur de données supporté par Glue, il en va de même :

- connecteur de récepteur de données `ClassName`
- connexion au récepteur de données `Options`

Notez que les fonctionnalités de AWS Glue telles que les signets de tâches et les DynamicFrame options telles que celles-ci ne `connectionName` sont pas prises en charge dans DataFrame. Pour plus de détails sur les opérations prises en charge DataFrame et pour en savoir plus, consultez la documentation de Spark pour DataFrame.

Spécification du connecteur ClassName

Pour spécifier le `ClassName` d'une source de données/d'un récepteur, utilisez l'option `.format` permettant de fournir le `ClassName` du connecteur correspondant qui définit la source de données/le récepteur.

Connecteurs JDBC

Pour les connecteurs JDBC, indiquez `jdbc` en tant que valeur de l'option `.format` et fournissez le pilote JDBC `ClassName` dans l'option `driver`.

```
df = spark.read.format("jdbc").option("driver", "<DATA SOURCE JDBC DRIVER
 CLASSNAME>")...

df.write.format("jdbc").option("driver", "<DATA SINK JDBC DRIVER CLASSNAME>")...
```

Le tableau suivant répertorie le pilote JDBC de la source `ClassName` de données prise en charge dans AWS Glue for. DataFrames

Source de données	Chauffeur ClassName
PostgreSQL	pilote org.postgreSQL
Oracle	oracle.jdbc.driver. OracleDriver
SQLServer	com.microsoft.sqlserver.jdbc. SQLServer Chauffeur
MySQL	com.mysql.jdbc.driver
SAPHana	com.sap.db.jdbc.driver
Teradata	com.teradata.jdbc. TeraDriver

Connecteurs Spark

Pour les connecteurs Spark, spécifiez le `ClassName` du connecteur en tant que valeur de l'option `.format`.

```
df = spark.read.format("<DATA SOURCE CONNECTOR CLASSNAME>")...

df.write.format("<DATA SINK CONNECTOR CLASSNAME>")...
```

Le tableau suivant répertorie le connecteur Spark `ClassName` de la source de données prise en charge dans AWS Glue for DataFrames.

Source de données	ClassName

MongoDB/DocumentDB	glue.spark.mongodb
Redshift	io.github.spark_redshift_community.spark.redshift
AzureCosmos	cosmos.oltp
AzureSQL	com.microsoft.sqlserver.jdbc.spark
BigQuery	com.google.cloud.spark.bigquery
OpenSearch	org.opensearch.spark.sql
Snowflake	net.snowflake.spark.snowflake
Vertica	com.vertica.spark.datasource. VerticaSource

Spécification des options de connexion

Pour spécifier les `Options` de la connexion à une source de données/un récepteur, utilisez `.option(<KEY>, <VALUE>)` pour fournir des options individuelles ou `.options(<MAP>)` pour fournir plusieurs options sous forme de carte clé-valeur.

Chaque donnée source/sink prend en charge son propre ensemble de connexions`Options`. Pour plus de détails sur les connecteurs disponibles`Options`, reportez-vous à la documentation publique du connecteur de données source/sink Spark spécifique répertorié dans le tableau suivant.

- JDBC
- MongoDB/DocumentDB
- Redshift
- AzureCosmos
- AzureSQL
- BigQuery
- OpenSearch
- Snowflake
- Vertica

Exemples

Les exemples suivants sont extraits de PostgreSQL et écrivent dans : SnowFlake

Python

Exemple :

```
from awsglue.context import GlueContext
from pyspark.sql import SparkSession

spark = SparkSession.builder.getOrCreate()

dataSourceClassName = "jdbc"
dataSourceOptions = {
  "driver": "org.postgresql.Driver",
  "url": "<url>",
  "user": "<user>",
  "password": "<password>",
  "dbtable": "<dbtable>",
}

dataframe = spark.read.format(className).options(**options).load()

dataSinkClassName = "net.snowflake.spark.snowflake"
dataSinkOptions = {
  "sfUrl": "<url>",
  "sfUsername": "<username>",
  "sfPassword": "<password>",
  "sfDatabase" -> "<database>",
  "sfSchema" -> "<schema>",
  "sfWarehouse" -> "<warehouse>"
}

dataframe.write.format(dataSinkClassName).options(**dataSinkOptions).save()
```

Scala

Exemple :

```
import org.apache.spark.sql.SparkSession

val spark = SparkSession.builder().getOrCreate()
```

```
val dataSourceClassName = "jdbc"
val dataSourceOptions = Map(
  "driver" -> "org.postgresql.Driver",
  "url" -> "<url>",
  "user" -> "<user>",
  "password" -> "<password>",
  "dbtable" -> "<dbtable>"
)

val dataframe =
 spark.read.format(dataSourceClassName).options(dataSourceOptions).load()

val dataSinkClassName = "net.snowflake.spark.snowflake"
val dataSinkOptions = Map(
  "sfUrl" -> "<url>",
  "sfUsername" -> "<username>",
  "sfPassword" -> "<password>",
  "sfDatabase" -> "<database>",
  "sfSchema" -> "<schema>",
  "sfWarehouse" -> "<warehouse>"
)

dataframe.write.format(dataSinkClassName).options(dataSinkOptions).save()
```

Valeurs personnalisées et AWS Marketplace ConnectionType

Cela inclut les éléments suivants :

- `"connectionType": "marketplace.athena"` : désigne une connexion à un magasin de données Amazon Athena. La connexion utilise un connecteur provenant de AWS Marketplace.
- `"connectionType": "marketplace.spark"` : désigne une connexion à un magasin de données Apache Spark. La connexion utilise un connecteur provenant de AWS Marketplace.
- `"connectionType": "marketplace.jdbc"` : désigne une connexion à un magasin de données JDBC. La connexion utilise un connecteur provenant de AWS Marketplace.
- `"connectionType": "custom.athena"` : désigne une connexion à un magasin de données Amazon Athena. La connexion utilise un connecteur personnalisé que vous chargez sur AWS Glue Studio.
- `"connectionType": "custom.spark"` : désigne une connexion à un magasin de données Apache Spark. La connexion utilise un connecteur personnalisé que vous chargez sur AWS Glue Studio.

- `"connectionType": "custom.jdbc"` : désigne une connexion à un magasin de données JDBC. La connexion utilise un connecteur personnalisé que vous chargez sur AWS Glue Studio.

Options de connexion pour le type custom.jdbc ou marketplace.jdbc

- `className` — chaîne, obligatoire, nom de la classe du pilote.
- `connectionName` — chaîne, obligatoire, nom de la connexion associée au connecteur.
- `url` — chaîne, obligatoire, URL JDBC avec des espaces réservés (`${}`) qui sont utilisés pour créer la connexion à la source de données. L'espace réservé `${secretKey}` est remplacé par le secret du même nom dans AWS Secrets Manager. Reportez-vous à la documentation du magasin de données pour plus d'informations sur la construction de l'URL.
- `secretId` ou `user/password` — chaîne, obligatoire, utilisée pour récupérer les informations d'identification de l'URL.
- `dbTable` ou `query` — chaîne, obligatoire, la table ou la requête SQL à partir de laquelle obtenir les données. Vous pouvez préciser `dbTable` ou `query`, mais pas les deux.
- `partitionColumn` — chaîne, facultatif, nom d'une colonne entière utilisée pour le partitionnement. Cette option fonctionne uniquement lorsqu'elle est incluse dans `lowerBound`, `upperBound` et `numPartitions`. Cette option fonctionne de la même manière que dans le lecteur JDBC SQL Spark. Pour plus d'informations, consultez la section JDBC vers d'autres bases de données dans le guide Apache Spark SQL DataFrames et Datasets.

  Les valeurs `lowerBound` et `upperBound` sont utilisées pour décider de la progression de la partition, pas pour filtrer les lignes de la table. Toutes les lignes de la table sont partitionnées et renvoyées.

  > **Note**
  >
  > Lorsque vous utilisez une requête au lieu d'un nom de table, vous devez valider que la requête fonctionne avec la condition de partitionnement spécifiée. Par exemple :
  >
  > - Si le format de votre requête est `"SELECT col1 FROM table1"`, testez la requête en ajoutant une clause `WHERE` à la fin de la requête qui utilise la colonne de partition.
  > - Si le format de votre requête est `SELECT col1 FROM table1 WHERE col2=val"`, testez la requête en étendant la clause `WHERE` avec `AND` et une expression qui utilise la colonne de partition.

- `lowerBound` — entier, facultatif, valeur minimale de `partitionColumn` qui est utilisée pour décider de la progression de la partition.
- `upperBound` — entier, facultatif, valeur maximale de `partitionColumn` qui est utilisée pour décider de la progression de la partition.
- `numPartitions` — entier, facultatif, nombre de partitions. Cette valeur, ainsi que `lowerBound` (inclusive) et `upperBound` (exclusive) forment les progressions de partition pour les expressions de clause `WHERE` générées qui sont utilisées pour diviser le fichier `partitionColumn`.

  > **Important**
  >
  > Soyez prudent avec le nombre de partitions, car avoir un trop grand nombre de partitions peut causer des problèmes sur vos systèmes de bases de données externes.

- `filterPredicate` — chaîne, facultative, clause de condition supplémentaire pour filtrer les données à partir de la source. Par exemple :

  ```
  BillingCity='Mountain View'
  ```

  Lorsque vous utilisez une requête au lieu d'un nom de table, vous devez vérifier que la requête fonctionne avec le `filterPredicate` spécifié. Par exemple :

  - Si le format de votre requête est `"SELECT col1 FROM table1"`, testez la requête en ajoutant une clause `WHERE` à la fin de la requête qui utilise le prédicat de filtre.
  - Si le format de votre requête est `"SELECT col1 FROM table1 WHERE col2=val"`, testez la requête en étendant la clause `WHERE` avec `AND` et une expression qui utilise le prédicat de filtre.

- `dataTypeMapping` — dictionnaire, facultatif, mappage de type de données personnalisé qui crée un mappage d'un type de données JDBC à un type de données Glue. Par exemple, l'option `"dataTypeMapping":{"FLOAT":"STRING"}` mappe les champs de données du type JDBC `FLOAT` sur le type Java `String` en appelant la méthode `ResultSet.getString()` du pilote, et l'utilise pour créer les enregistrements AWS Glue. L'objet est `ResultSet` implémenté par chaque pilote, donc le comportement est spécifique au pilote que vous utilisez. Reportez-vous à la documentation de votre pilote JDBC pour comprendre comment le pilote effectue les conversions.
- Les types de données AWS Glue actuellement pris en charge sont les suivants :
  - DATE
  - CHAÎNE
  - TIMESTAMP

- INT
- FLOAT
- LONG
- BIGDECIMAL
- BYTE
- SHORT
- DOUBLE

Les types de données JDBC pris en charge sont Java8 java.sql.types.

Les mappages de type de données par défaut (de JDBC à AWS Glue) sont les suivants :

- DATE -> DATE
- VARCHAR -> STRING
- CHAR -> STRING
- LONGNVARCHAR -> STRING
- TIMESTAMP -> TIMESTAMP
- INTEGER -> INT
- FLOAT -> FLOAT
- REAL -> FLOAT
- BIT -> BOOLEAN
- BOOLEAN -> BOOLEAN
- BIGINT -> LONG
- DECIMAL -> BIGDECIMAL
- NUMERIC -> BIGDECIMAL
- TINYINT -> SHORT
- SMALLINT -> SHORT
- DOUBLE -> DOUBLE

Si vous utilisez un mappage de type de données personnalisé avec l'option `dataTypeMapping`, vous pouvez remplacer un mappage de type de données par défaut. Seuls les types de données JDBC répertoriés dans l'option `dataTypeMapping` sont affectés ; le mappage par défaut est utilisé pour tous les autres types de données JDBC. Vous pouvez ajouter des mappages pour des types de données JDBC supplémentaires si nécessaire. Si un type de données JDBC n'est inclus

ni dans le mappage par défaut ni dans un mappage personnalisé, le type de données est converti en type de données AWS Glue `STRING` par défaut.

L'exemple de code Python suivant montre comment lire à partir de bases de données JDBC avec des pilotes AWS Marketplace JDBC. Il montre la lecture à partir d'une base de données et l'écriture dans un emplacement S3.

```
    import sys
    from awsglue.transforms import *
    from awsglue.utils import getResolvedOptions
    from pyspark.context import SparkContext
    from awsglue.context import GlueContext
    from awsglue.job import Job

    ## @params: [JOB_NAME]
    args = getResolvedOptions(sys.argv, ['JOB_NAME'])

    sc = SparkContext()
    glueContext = GlueContext(sc)
    spark = glueContext.spark_session
    job = Job(glueContext)
    job.init(args['JOB_NAME'], args)
    ## @type: DataSource
    ## @args: [connection_type = "marketplace.jdbc", connection_options =
     {"dataTypeMapping":{"INTEGER":"STRING"},"upperBound":"200","query":"select id,
       name, department from department where id < 200","numPartitions":"4",
       "partitionColumn":"id","lowerBound":"0","connectionName":"test-connection-
jdbc"},
        transformation_ctx = "DataSource0"]
    ## @return: DataSource0
    ## @inputs: []
    DataSource0 = glueContext.create_dynamic_frame.from_options(connection_type =
      "marketplace.jdbc", connection_options = {"dataTypeMapping":{"INTEGER":"STRING"},
      "upperBound":"200","query":"select id, name, department from department where
       id < 200","numPartitions":"4","partitionColumn":"id","lowerBound":"0",
       "connectionName":"test-connection-jdbc"}, transformation_ctx = "DataSource0")
    ## @type: ApplyMapping
    ## @args: [mappings = [("department", "string", "department", "string"), ("name",
 "string",
     "name", "string"), ("id", "int", "id", "int")], transformation_ctx =
 "Transform0"]
    ## @return: Transform0
```

```
    ## @inputs: [frame = DataSource0]
    Transform0 = ApplyMapping.apply(frame = DataSource0, mappings = [("department",
"string",
     "department", "string"), ("name", "string", "name", "string"), ("id", "int",
"id", "int")],
      transformation_ctx = "Transform0")
    ## @type: DataSink
    ## @args: [connection_type = "s3", format = "json", connection_options = {"path":
     "s3://<S3 path>/", "partitionKeys": []}, transformation_ctx = "DataSink0"]
    ## @return: DataSink0
    ## @inputs: [frame = Transform0]
    DataSink0 = glueContext.write_dynamic_frame.from_options(frame = Transform0,
      connection_type = "s3", format = "json", connection_options = {"path":
      "s3://<S3 path>/", "partitionKeys": []}, transformation_ctx = "DataSink0")
    job.commit()
```

Options de connexion pour le type custom.athena ou marketplace.athena

- `className` — chaîne, obligatoire, nom de la classe du pilote. Lorsque vous utilisez le CloudWatch connecteur Athena, cette valeur de paramètre est le préfixe de la classe Name (par exemple,). `"com.amazonaws.athena.connectors"` Le CloudWatch connecteur Athena est composé de deux classes : un gestionnaire de métadonnées et un gestionnaire d'enregistrements. Si vous fournissez le préfixe commun ici, l'API charge les classes correctes en fonction de ce préfixe.
- `tableName`— Chaîne, obligatoire, nom du flux de CloudWatch journal à lire. Cet extrait de code utilise le nom de vue spécial `all_log_streams`, ce qui signifie que la trame de données dynamique renvoyée contiendra les données de tous les flux de journaux du groupe de journaux.
- `schemaName`— Chaîne, obligatoire, nom du groupe de CloudWatch journaux à partir duquel la lecture doit être effectuée. Par exemple, `/aws-glue/jobs/output`.
- `connectionName` — chaîne, obligatoire, nom de la connexion associée au connecteur.

Pour des options supplémentaires pour ce connecteur, consultez le fichier README du CloudWatch connecteur Amazon Athena sur. GitHub

L'exemple de code Python suivant montre comment lire à partir d'un magasin de données Athena à l'aide d'un connecteur AWS Marketplace . Il montre la lecture à partir d'Athena et l'écriture dans un emplacement S3.

```
    import sys
    from awsglue.transforms import *
```

```
    from awsglue.utils import getResolvedOptions
    from pyspark.context import SparkContext
    from awsglue.context import GlueContext
    from awsglue.job import Job

    ## @params: [JOB_NAME]
    args = getResolvedOptions(sys.argv, ['JOB_NAME'])

    sc = SparkContext()
    glueContext = GlueContext(sc)
    spark = glueContext.spark_session
    job = Job(glueContext)
    job.init(args['JOB_NAME'], args)
    ## @type: DataSource
    ## @args: [connection_type = "marketplace.athena", connection_options =
     {"tableName":"all_log_streams","schemaName":"/aws-glue/jobs/output",
      "connectionName":"test-connection-athena"}, transformation_ctx = "DataSource0"]
    ## @return: DataSource0
    ## @inputs: []
    DataSource0 = glueContext.create_dynamic_frame.from_options(connection_type =
      "marketplace.athena", connection_options = {"tableName":"all_log_streams",,
      "schemaName":"/aws-glue/jobs/output","connectionName":
      "test-connection-athena"}, transformation_ctx = "DataSource0")
    ## @type: ApplyMapping
    ## @args: [mappings = [("department", "string", "department", "string"), ("name",
"string",
     "name", "string"), ("id", "int", "id", "int")], transformation_ctx =
"Transform0"]
    ## @return: Transform0
    ## @inputs: [frame = DataSource0]
    Transform0 = ApplyMapping.apply(frame = DataSource0, mappings = [("department",
"string",
     "department", "string"), ("name", "string", "name", "string"), ("id", "int",
"id", "int")],
      transformation_ctx = "Transform0")
    ## @type: DataSink
    ## @args: [connection_type = "s3", format = "json", connection_options = {"path":
     "s3://<S3 path>/", "partitionKeys": []}, transformation_ctx = "DataSink0"]
    ## @return: DataSink0
    ## @inputs: [frame = Transform0]
    DataSink0 = glueContext.write_dynamic_frame.from_options(frame = Transform0,
      connection_type = "s3", format = "json", connection_options = {"path":
      "s3://<S3 path>/", "partitionKeys": []}, transformation_ctx = "DataSink0")
```

```
    job.commit()
```

Options de connexion pour le type custom.spark ou marketplace.spark

- `className` — chaîne, obligatoire, nom de la classe du connecteur.
- `secretId` — chaîne, facultative, utilisée pour récupérer les informations d'identification pour la connexion du connecteur.
- `connectionName` — chaîne, obligatoire, nom de la connexion associée au connecteur.
- D'autres options dépendent du magasin de données. Par exemple, les options OpenSearch de configuration commencent par le préfixe`es`, comme décrit dans la documentation d'Elasticsearch pour Apache Hadoop. Les connexions Spark à Snowflake utilisent des options telles que `sfUser` et `sfPassword`, comme décrit dans Using the Spark Connector dans le guide Connecting to Snowflake.

L'exemple de code Python suivant montre comment lire depuis un magasin de OpenSearch données à l'aide d'une `marketplace.spark` connexion.

```
    import sys
    from awsglue.transforms import *
    from awsglue.utils import getResolvedOptions
    from pyspark.context import SparkContext
    from awsglue.context import GlueContext
    from awsglue.job import Job

    ## @params: [JOB_NAME]
    args = getResolvedOptions(sys.argv, ['JOB_NAME'])

    sc = SparkContext()
    glueContext = GlueContext(sc)
    spark = glueContext.spark_session
    job = Job(glueContext)
    job.init(args['JOB_NAME'], args)
    ## @type: DataSource
    ## @args: [connection_type = "marketplace.spark", connection_options =
{"path":"test",
     "es.nodes.wan.only":"true","es.nodes":"https://<AWS endpoint>",
     "connectionName":"test-spark-es","es.port":"443"}, transformation_ctx =
"DataSource0"]
    ## @return: DataSource0
    ## @inputs: []
```

```
    DataSource0 = glueContext.create_dynamic_frame.from_options(connection_type =
      "marketplace.spark", connection_options = {"path":"test","es.nodes.wan.only":
      "true","es.nodes":"https://<AWS endpoint>","connectionName":
      "test-spark-es","es.port":"443"}, transformation_ctx = "DataSource0")
    ## @type: DataSink
    ## @args: [connection_type = "s3", format = "json", connection_options = {"path":
         "s3://<S3 path>/", "partitionKeys": []}, transformation_ctx = "DataSink0"]
    ## @return: DataSink0
    ## @inputs: [frame = DataSource0]
    DataSink0 = glueContext.write_dynamic_frame.from_options(frame = DataSource0,
       connection_type = "s3", format = "json", connection_options = {"path":
       "s3://<S3 path>/", "partitionKeys": []}, transformation_ctx = "DataSink0")
    job.commit()
```

Options générales

Les options de cette section sont fournies en tant que `connection_options`, mais ne s'appliquent pas à un connecteur en particulier.

Les paramètres suivants sont généralement utilisés lors de la configuration de signets. Ils peuvent s'appliquer aux flux de travail Amazon S3 ou JDBC. Pour de plus amples informations, veuillez consulter the section called “Utilisation des marque-pages de tâche”.

- `jobBookmarkKeys` : un tableau des noms des colonnes.
- `jobBookmarkKeysSortOrder` : chaîne définissant comment comparer les valeurs en fonction de l'ordre de tri. Valeurs valides : `"asc"`, `"desc"`.
- `useS3ListImplementation` : utilisé pour gérer les performances de la mémoire lors de la liste du contenu du compartiment Amazon S3. Pour plus d'informations, consultez Optimiser la gestion de la mémoire dans AWS Glue.

Options de format pour les entrées et sorties dans AWS Glue pour Spark

Ces pages fournissent des informations sur la prise en charge des fonctionnalités et les paramètres de configuration pour les formats de données pris en charge par AWS Glue pour Spark. Reportez-vous à la section suivante pour une description de l'utilisation et de l'applicabilité de ces informations.

Prise en charge des fonctionnalités de tous les formats de données dans AWS Glue

Chaque format de données peut prendre en charge différentes fonctionnalités AWS de Glue. Les fonctionnalités communes suivantes peuvent être prises en charge ou non en fonction de votre type

de format. Reportez-vous à la documentation de votre format de données pour comprendre comment tirer parti de nos caractéristiques pour répondre à vos besoins.

Lecture	AWS Glue peut reconnaître et interpréter ce format de données sans ressources supplémentaires, telles que des connecteurs.
Écrire	AWS Glue peut écrire des données dans ce format sans ressources supplémentaires. Vous pouvez inclure des bibliothèques tierces dans votre travail et utiliser les fonctions standard d'Apache Spark pour écrire des données, comme vous le feriez dans d'autres environnements Spark. Pour plus d'informations sur ces bibliothèques, consultez the section called "Bibliothèques Python".
Lecture en streaming	AWS Glue peut reconnaître et interpréter ce format de données à partir d'un flux de messages Apache Kafka, Amazon Managed Streaming for Apache Kafka ou Amazon Kinesis. Nous nous attendons à ce que les flux présentent les données dans un format cohérent, afin qu'elles soient lues comme `DataFrames` .
Groupement des petits fichiers	AWS Glue peut regrouper des fichiers pour créer un travail par lots envoyé à chaque nœud lors de transformations avec AWS Glue. Cela peut améliorer considérablement les performances pour les charges de travail impliquant de grandes quantités de petits fichiers. Pour de plus amples informations, veuillez consulter the section called "Groupement des fichiers d'entrée".

Signets de tâche	AWS Glue peut suivre la progression des transformations effectuant le même travail sur le même jeu de données d'une exécution à l'autre à l'aide de signets de tâches. Cela peut améliorer les performances des charges de travail impliquant des jeux de données pour lesquels le travail doit uniquement être effectué sur les nouvelles données depuis la dernière exécution de la tâche. Pour de plus amples informations, veuillez consulter the section called "Suivi des données traitées à l'aide de signets de tâche".

Paramètres utilisés pour interagir avec les formats de données dans AWS Glue

Certains types de connexion AWS Glue prennent en charge plusieurs `format` types, ce qui vous oblige à spécifier des informations sur le format de vos données avec un `format_options` objet lorsque vous utilisez des méthodes telles que`GlueContext.write_dynamic_frame.from_options`.

- `s3`— Pour plus d'informations, consultez Types de connexion et options pour l'ETL dans AWS Glue :Paramètres de connexion S3. Vous pouvez également afficher la documentation des méthodes facilitant ce type de connexion :the section called "create_dynamic_frame_from_options" et the section called "write_dynamic_frame_from_options" en Python et dans les méthodes Scala correspondantes the section called "getSourceWithFormater" et the section called "getSinkWithFormater".

- `kinesis`— Pour plus d'informations, consultez Types de connexion et options pour l'ETL dans AWS Glue :Paramètres de connexion Kinesis. Vous pouvez également afficher la documentation de la méthode facilitant ce type de connexion :the section called "create_data_frame_from_options" et la méthode Scala correspondante the section called "createDataFrameFromOptions".
- `kafka`— Pour plus d'informations, consultez Types de connexion et options pour l'ETL dans AWS Glue :Paramètres de connexion de Kafka. Vous pouvez également afficher la documentation de la méthode facilitant ce type de connexion :the section called "create_data_frame_from_options" et la méthode Scala correspondante the section called "createDataFrameFromOptions".

Certains types de connexion ne nécessitent pas`format_options`. Par exemple, dans le cadre d'une utilisation normale, une connexion JDBC à une base de données relationnelle récupère les données dans un format de données tabulaire cohérent. Par conséquent, la lecture à partir d'une connexion JDBC ne nécessiterait pas `format_options`.

Certaines méthodes pour lire et écrire des données dans de la colle ne nécessitent pas`format_options`. Par exemple, en utilisant `GlueContext.create_dynamic_frame.from_catalog` les crawlers AWS Glue. Les robots d'exploration déterminent la forme de vos données. Lorsque vous utilisez des robots d'exploration, un classificateur AWS Glue examinera vos données afin de prendre des décisions éclairées quant à la manière de représenter le format de vos données. Il stockera ensuite une représentation de vos données dans le catalogue de données AWS Glue, qui peut être utilisé dans un script AWS Glue ETL pour récupérer vos données avec la `GlueContext.create_dynamic_frame.from_catalog` méthode. Les robots d'exploration éliminent la nécessité de spécifier manuellement des informations sur le format de vos données.

Pour les tâches qui accèdent aux tables AWS Lake Formation gouvernées, AWS Glue prend en charge la lecture et l'écriture dans tous les formats pris en charge par les tables régies par Lake Formation. Pour consulter la liste actuelle des formats pris en charge pour les tables AWS Lake Formation gouvernées, consultez la section Remarques et restrictions relatives aux tables gouvernées dans le guide du AWS Lake Formation développeur.

> **Note**
>
> Pour écrire Apache Parquet, AWS Glue ETL prend uniquement en charge l'écriture dans une table régie en spécifiant une option pour un type de dispositif d'écriture Parquet personnalisé optimisé pour les cadres dynamiques. Lorsque vous écrivez sur une table régie avec le format `parquet`, vous devez ajouter la clé `useGlueParquetWriter` avec une valeur de `true` dans les paramètres de la table.

Rubriques

- Utilisation du format CSV dans AWS Glue
- Utilisation du format Parquet dans AWS Glue
- Utilisation du format XML dans AWS Glue
- Utilisation du format Avro dans Glue AWS
- Utilisation du format GrokLog dans Glue AWS

- Utilisation du format Ion dans AWS Glue
- Utilisation du format JSON dans AWS Glue
- Utilisation du format ORC dans Glue AWS
- Utilisation de frameworks de lacs de données avec des tâches AWS Glue ETL
- Référence de configuration partagée

Utilisation du format CSV dans AWS Glue

AWS Glue récupère les données des sources et écrit les données sur des cibles stockées et transportées dans différents formats de données. Si vos données sont stockées ou transportées au format CSV, ce document présente les fonctionnalités disponibles pour utiliser vos données dans AWS Glue.

AWS Glue prend en charge l'utilisation du format CSV (valeurs séparées par des virgules). Il s'agit d'un format de données minimal basé sur des lignes. CSVs ne sont souvent pas strictement conformes à une norme, mais vous pouvez vous référer aux RFC 4180 et RFC 7111 pour plus d'informations.

Vous pouvez utiliser AWS Glue pour lire CSVs depuis Amazon S3 et depuis des sources de streaming, ainsi que CSVs pour écrire sur Amazon S3. Vous pouvez lire et écrire des archives `bzip` et `gzip` contenant des fichiers CSV provenant de S3. Vous configurez le comportement de compression sur Paramètres de connexion S3 plutôt que dans la configuration décrite sur cette page.

Le tableau suivant indique les fonctionnalités courantes de AWS Glue compatibles avec l'option de format CSV.

Lecture	Écrire	Lecture en streaming	Groupement des petits fichiers	Signets de tâche
Pris en charge	Pris en charge	Pris en charge	Pris en charge	Pris en charge

Exemple : lecture de fichiers ou de dossiers CSV depuis S3

Prérequis : vous aurez besoin des chemins S3 (`s3path`) vers des fichiers ou dossiers CSV que vous souhaitez lire.

Configuration : dans vos options de fonction, spécifiez `format="csv"`. Dans vos `connection_options`, utilisez la clé `paths` pour spécifier `s3path`. Vous pouvez configurer

la manière dont le lecteur interagit avec S3 dans `connection_options`. Pour plus de détails, voir Types de connexion et options pour l'ETL dans AWS Glue :Paramètres de connexion S3. Vous pouvez configurer la manière dont le lecteur interprète les fichiers CSV dans votre `format_options`. Pour plus d'informations, consultez CSV Configuration Reference (Référence de configuration CSV).

Le script AWS Glue ETL suivant montre le processus de lecture de fichiers ou de dossiers CSV à partir de S3.

Nous fournissons un lecteur CSV personnalisé avec des optimisations de performances pour les flux de travail courants à travers la clé de configuration `optimizePerformance`. Pour déterminer si ce lecteur est adapté à votre charge de travail, consultez the section called “Utilisation d'un lecteur CSV optimisé”.

Python

Pour cet exemple, utilisez la méthode create_dynamic_frame.from_options.

```
# Example: Read CSV from S3
# For show, we handle a CSV with a header row.  Set the withHeader option.
# Consider whether optimizePerformance is right for your workflow.

from pyspark.context import SparkContext
from awsglue.context import GlueContext

sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)
spark = glueContext.spark_session

dynamicFrame = glueContext.create_dynamic_frame.from_options(
    connection_type="s3",
    connection_options={"paths": ["s3://s3path"]},
    format="csv",
    format_options={
        "withHeader": True,
        # "optimizePerformance": True,
    },
)
```

Vous pouvez également l'utiliser DataFrames dans un script (`pyspark.sql.DataFrame`).

```
dataFrame = spark.read\
```

```
    .format("csv")\
    .option("header", "true")\
    .load("s3://s3path")
```

Scala

Pour cet exemple, utilisez l'opération getSourceWithFormat.

```
// Example: Read CSV from S3
// For show, we handle a CSV with a header row.  Set the withHeader option.
// Consider whether optimizePerformance is right for your workflow.

import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.{DynamicFrame, GlueContext}
import org.apache.spark.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)

    val dynamicFrame = glueContext.getSourceWithFormat(
      formatOptions=JsonOptions("""{"withHeader": true}"""),
      connectionType="s3",
      format="csv",
      options=JsonOptions("""{"paths": ["s3://s3path"], "recurse": true}""")
    ).getDynamicFrame()
  }
}
```

Vous pouvez également l'utiliser DataFrames dans un script (`org.apache.spark.sql.DataFrame`).

```
val dataFrame = spark.read
  .option("header","true")
  .format("csv")
  .load("s3://s3path“)
```

Exemple : écriture de fichiers et dossiers CSV dans S3

Prérequis : Vous aurez besoin d'un initialisé DataFrame (`dataFrame`) ou d'un DynamicFrame (`dynamicFrame`). Vous aurez également besoin de votre chemin de sortie S3, `s3path`.

Configuration : dans vos options de fonction, spécifiez `format="csv"`. Dans vos `connection_options`, utilisez la clé `paths` pour spécifier `s3path`. Vous pouvez configurer la manière dont le scripteur interagit avec S3 dans `connection_options`. Pour plus de détails, voir Types de connexion et options pour l'ETL dans AWS Glue :Paramètres de connexion S3. Vous pouvez configurer la manière dont votre opération écrit le contenu de vos fichiers dans `format_options`. Pour plus d'informations, consultez CSV Configuration Reference (Référence de configuration CSV). Le script ETL AWS Glue suivant montre le processus d'écriture de fichiers et dossiers CSV vers S3.

Python

Pour cet exemple, utilisez la méthode write_dynamic_frame.from_options.

```
# Example: Write CSV to S3
# For show, customize how we write string type values.  Set quoteChar to -1 so our
 values are not quoted.

from pyspark.context import SparkContext
from awsglue.context import GlueContext

sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

glueContext.write_dynamic_frame.from_options(
    frame=dynamicFrame,
    connection_type="s3",
    connection_options={"path": "s3://s3path"},
    format="csv",
    format_options={
        "quoteChar": -1,
    },
)
```

Vous pouvez également l'utiliser DataFrames dans un script (`pyspark.sql.DataFrame`).

```
dataFrame.write\
```

```
    .format("csv")\
    .option("quote", None)\
    .mode("append")\
    .save("s3://s3path")
```

Scala

Pour cet exemple, utilisez la méthode getSinkWithFormat.

```
// Example: Write CSV to S3
// For show, customize how we write string type values. Set quoteChar to -1 so our
 values are not quoted.

import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.{DynamicFrame, GlueContext}
import org.apache.spark.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)

    glueContext.getSinkWithFormat(
        connectionType="s3",
        options=JsonOptions("""{"path": "s3://s3path"}"""),
        format="csv"
    ).writeDynamicFrame(dynamicFrame)
  }
}
```

Vous pouvez également l'utiliser DataFrames dans un script (`org.apache.spark.sql.DataFrame`).

```
dataFrame.write
    .format("csv")
    .option("quote", null)
    .mode("Append")
    .save("s3://s3path")
```

Référence de configuration CSV

Vous pouvez utiliser ce qui suit `format_options` partout où les bibliothèques AWS Glue le spécifient `format="csv"` :

- `separator` – spécifie le caractère délimiteur. La valeur par défaut est une virgule, mais tout autre caractère peut être spécifié.
  - Type : texte, Valeur par défaut : `","`
- `escaper` – spécifie le caractère à utiliser pour l'échappement. Cette option est utilisée uniquement lors de la lecture de fichiers CSV, pas l'écriture. Si cette option est activée, le caractère qui suit immédiatement est utilisé tel quel, sauf pour un petit ensemble d'échappements connus (`\n`, `\r`, `\t` et `\0`).
  - Type : texte, Valeur par défaut : aucune
- `quoteChar` – spécifie le caractère à utiliser pour les guillemets. La valeur par défaut est les guillemets doubles. Définissez ce champ sur `-1` pour désactiver entièrement les guillemets.
  - Type : texte, Valeur par défaut : `'"'`
- `multiLine` – spécifie si un même enregistrement peut couvrir plusieurs lignes. Cela peut se produire lorsqu'un champ contient un caractère de nouvelle ligne. Vous devez définir cette option sur `True` si aucun enregistrement ne s'étend sur plusieurs lignes. L'activation de `multiLine` peut réduire les performances, car elle nécessite un fractionnement plus prudent des fichiers lors de l'analyse.
  - Type : Booléen, Valeur par défaut : `false`
- `withHeader` – spécifie spécifie s'il convient de traiter la première ligne comme un en-tête. Cette option peut être utilisée dans la classe `DynamicFrameReader`.
  - Type : Booléen, Valeur par défaut : `false`
- `writeHeader` – spécifie s'il convient d'écrire l'en-tête dans la sortie. Cette option peut être utilisée dans la classe `DynamicFrameWriter`.
  - Type : Booléen, Valeur par défaut : `true`
- `skipFirst` – spécifie s'il convient d'ignorer la première ligne de données.
  - Type : Booléen, Valeur par défaut : `false`
- `optimizePerformance` –spécifie s'il faut utiliser le lecteur CSV SIMD avancé avec les formats de mémoire en colonnes basés sur Apache Arrow. Disponible uniquement dans les versions 3.0 et ultérieures d'AWS Glue.
  - Type : Booléen, Valeur par défaut : `false`

- `strictCheckForQuoting`— Lors de l'écriture CSVs, Glue peut ajouter des guillemets aux valeurs qu'elle interprète comme des chaînes. Cela est fait pour éviter toute ambiguïté dans ce qui est écrit. Pour gagner du temps au moment de décider quoi écrire, Glue peut ajouter des guillemets dans certaines situations où ils ne sont pas nécessaires. L'activation d'une vérification stricte effectuera un calcul plus intensif et n'ajoutera de guillemets qu'en cas de nécessité absolue. Disponible uniquement dans les versions 3.0 et ultérieures d'AWS Glue.
  - Type : Booléen, Valeur par défaut : `false`

Optimiser les performances de lecture avec un lecteur CSV SIMD vectorisé

La version 3.0 d'AWS Glue ajoute un lecteur CSV optimisé qui peut considérablement accélérer les performances globales des tâches par rapport aux lecteurs CSV basés sur des lignes.

Le lecteur optimisé :

- utilise les instructions SIMD du processeur pour lire sur le disque
- ecrit immédiatement les enregistrements en mémoire dans un format en colonnes (Apache Arrow)
- divise les enregistrements en lots

Cette méthode permet de gagner du temps de traitement lorsque les enregistrements sont ultérieurement regroupés ou convertis en format en colonnes. Certains exemples concernent la modification de schémas ou la récupération de données par colonne.

Pour utiliser le lecteur optimisé, définissez `"optimizePerformance"` sur `true` dans le `format_options` ou la propriété table.

```
glueContext.create_dynamic_frame.from_options(
    frame = datasource1,
    connection_type = "s3",
    connection_options = {"paths": ["s3://s3path"]},
    format = "csv",
    format_options={
        "optimizePerformance": True,
        "separator": ","
        },
    transformation_ctx = "datasink2")
```

Limitations du lecteur CSV vectorisé

Notez les limitations suivantes du lecteur CSV vectorisé :

- Il ne prend pas en charge les options de formatage `multiLine` et `escaper`. Il utilise la valeur par défaut `escaper` du caractère guillemets doubles `'"'`. Lorsque ces options sont définies, AWS Glue revient automatiquement à l'utilisation du lecteur CSV basé sur les lignes.
- Il ne prend pas en charge la création d'un DynamicFrame avec ChoiceType.
- Il ne prend pas en charge la création d'un DynamicFrame avec des enregistrements d'erreur.
- Il ne prend pas en charge la lecture de fichiers CSV contenant des caractères multioctets tels que des caractères japonais ou chinois.

Utilisation du format Parquet dans AWS Glue

AWS Glue récupère les données des sources et écrit les données sur des cibles stockées et transportées dans différents formats de données. Si vos données sont stockées ou transportées au format de données Parquet, ce document présente les fonctionnalités disponibles pour utiliser vos données dans AWS Glue.

AWS Supports à colle au format Parquet. Ce format est un format de données en colonnes, orienté vers la performance. Pour accéder à une présentation du format par l'autorité standard, consultez Apache Parquet Documentation Overview (Présentation de la documentation Apache Parquet).

Vous pouvez utiliser AWS Glue pour lire des fichiers Parquet depuis Amazon S3 et depuis des sources de streaming, ainsi que pour écrire des fichiers Parquet sur Amazon S3. Vous pouvez lire et écrire des archives `bzip` et `gzip` contenant des fichiers Parquet provenant de S3. Vous configurez le comportement de compression sur Paramètres de connexion S3 plutôt que dans la configuration décrite sur cette page.

Le tableau suivant indique les fonctionnalités courantes de AWS Glue compatibles avec l'option de format Parquet.

Lecture	Écrire	Lecture en streaming	Groupement des petits fichiers	Signets de tâche
Pris en charge	Pris en charge	Pris en charge	Non pris en charge	Pris en charge*

* Supporté dans la version 1.0+ de AWS Glue

Exemple : lecture de fichiers ou dossiers Parquet depuis S3

Prérequis : vous aurez besoin des chemins S3 (`s3path`) vers des fichiers ou dossiers Parquet que vous souhaitez lire.

Configuration : dans vos options de fonction, spécifiez `format="parquet"`. Dans vos `connection_options`, utilisez la clé `paths` pour spécifier votre `s3path`.

Vous pouvez configurer la manière dont le lecteur interagit avec S3 dans les `connection_options`. Pour plus de détails, voir Types de connexion et options pour l'ETL dans AWS Glue :Paramètres de connexion S3.

Vous pouvez configurer la manière dont le lecteur interprète les fichiers Parquet dans votre `format_options`. Pour plus d'informations, consultez Parquet Configuration Reference (Référence de configuration Parquet).

Le script AWS Glue ETL suivant montre le processus de lecture des fichiers ou dossiers Parquet depuis S3 :

Python

Pour cet exemple, utilisez la méthode create_dynamic_frame.from_options.

```
# Example: Read Parquet from S3

from pyspark.context import SparkContext
from awsglue.context import GlueContext

sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)
spark = glueContext.spark_session

dynamicFrame = glueContext.create_dynamic_frame.from_options(
    connection_type = "s3",
    connection_options = {"paths": ["s3://s3path/"]},
    format = "parquet"
)
```

Vous pouvez également l'utiliser DataFrames dans un script (`pyspark.sql.DataFrame`).

```
dataFrame = spark.read.parquet("s3://s3path/")
```

Scala

Pour cet exemple, utilisez la méthode getSourceWithFormat.

```
// Example: Read Parquet from S3

import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.{DynamicFrame, GlueContext}
import org.apache.spark.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)

    val dynamicFrame = glueContext.getSourceWithFormat(
      connectionType="s3",
      format="parquet",
      options=JsonOptions("""{"paths": ["s3://s3path"]}""")
    ).getDynamicFrame()
  }
}
```

Vous pouvez également l'utiliser DataFrames dans un script (`org.apache.spark.sql.DataFrame`).

```
spark.read.parquet("s3://s3path/")
```

Exemple : écriture de fichiers et dossiers Parquet dans S3

Prérequis : Vous aurez besoin d'un initialisé DataFrame (`dataFrame`) ou DynamicFrame (`dynamicFrame`). Vous aurez également besoin de votre chemin de sortie S3, `s3path`.

Configuration : dans vos options de fonction, spécifiez `format="parquet"`. Dans vos `connection_options`, utilisez la clé `paths` pour spécifier `s3path`.

Vous pouvez modifier davantage la manière dont le scripteur interagit avec S3 dans les `connection_options`. Pour plus de détails, voir Types de connexion et options pour l'ETL dans AWS Glue :Paramètres de connexion S3. Vous pouvez configurer la manière dont votre opération écrit le contenu de vos fichiers dans `format_options`. Pour plus d'informations, consultez Parquet Configuration Reference (Référence de configuration Parquet).

Le script AWS Glue ETL suivant montre le processus d'écriture de fichiers et de dossiers Parquet dans S3.

Nous fournissons un rédacteur de parquet personnalisé avec des optimisations de performances DynamicFrames, grâce à la clé de `useGlueParquetWriter` configuration. Pour déterminer si ce scripteur est adapté à votre charge de travail, consultez Glue Parquet Writer (Scripteur Parquet Glue).

Python

Pour cet exemple, utilisez la méthode write_dynamic_frame.from_options.

```
# Example: Write Parquet to S3
# Consider whether useGlueParquetWriter is right for your workflow.

from pyspark.context import SparkContext
from awsglue.context import GlueContext

sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

glueContext.write_dynamic_frame.from_options(
    frame=dynamicFrame,
    connection_type="s3",
    format="parquet",
    connection_options={
        "path": "s3://s3path",
    },
    format_options={
        # "useGlueParquetWriter": True,
    },
)
```

Vous pouvez également l'utiliser DataFrames dans un script (`pyspark.sql.DataFrame`).

```
df.write.parquet("s3://s3path/")
```

Scala

Pour cet exemple, utilisez la méthode getSinkWithFormat.

```
// Example: Write Parquet to S3
```

```
// Consider whether useGlueParquetWriter is right for your workflow.

import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.{DynamicFrame, GlueContext}
import org.apache.spark.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)

    glueContext.getSinkWithFormat(
        connectionType="s3",
        options=JsonOptions("""{"path": "s3://s3path"}"""),
        format="parquet"
    ).writeDynamicFrame(dynamicFrame)
  }
}
```

Vous pouvez également l'utiliser DataFrames dans un script (`org.apache.spark.sql.DataFrame`).

```
df.write.parquet("s3://s3path/")
```

Référence de configuration Parquet

Vous pouvez utiliser ce qui suit `format_options` partout où les bibliothèques AWS Glue le spécifient `format="parquet"` :

- `useGlueParquetWriter`— Spécifie l'utilisation d'un rédacteur Parquet personnalisé doté d'optimisations des performances pour DynamicFrame les flux de travail. Pour plus d'informations, consultez Glue Parquet Writer (Scripteur Parquet Glue).
  - Type : Booléen, Valeur par défaut :`false`
- `compression` – spécifie le codec de compression utilisé. Les valeurs sont entièrement compatibles avec `org.apache.parquet.hadoop.metadata.CompressionCodecName`.
  - Type : Texte énuméré, Valeur par défaut : `"snappy"`
  - Valeurs : `"uncompressed"`, `"snappy"`, `"gzip"`, et `"lzo"`

- `blockSize` – spécifie la taille en octets d'un groupe de lignes mis en mémoire. Vous l'utilisez pour régler les performances. La taille doit être divisée exactement en un certain nombre de mégaoctets.
  - Type : Numérique, Valeur par défaut :`134217728`
  - La valeur par défaut est égale à 128 Mo.
- `pageSize` – spécifie la taille d'une page en octets. Vous l'utilisez pour régler les performances. Une page est la plus petite unité qui doit être lue entièrement pour accéder à un enregistrement unique.
  - Type : Numérique, Valeur par défaut :`1048576`
  - La valeur par défaut est égale à 1 Mo.

> **Note**
>
> De plus, toutes les options acceptées par le code SparkSQL sous-jacent peuvent être transmises à ce format via le paramètre map `connection_options`. Par exemple, vous pouvez définir une configuration Spark telle que mergeSchema pour le dispositif d'écriture AWS Glue Spark pour fusionner le schéma de tous les fichiers.

Optimisez les performances d'écriture avec AWS Glue Parquet Writer

> **Note**
>
> Le rédacteur AWS Glue Parquet a toujours été accessible via le type de `glueparquet` format. Ce modèle d'accès n'est plus préconisé. Utilisez plutôt le type `parquet` avec `useGlueParquetWriter` activé.

Le graveur AWS Glue Parquet présente des améliorations de performances qui permettent d'écrire plus rapidement les fichiers Parquet. Le scripteur traditionnel calcule un schéma avant d'écrire. Le format Parquet ne stocke pas le schéma de manière rapidement récupérable, ce qui peut prendre un certain temps. Avec le rédacteur AWS Glue Parquet, aucun schéma précalculé n'est requis. Au fur et à mesure que les données sont introduites, le scripteur calcule et modifie le schéma de manière dynamique.

Notez les limitations suivantes lorsque vous spécifiez `useGlueParquetWriter` :

- Le dispositif d'écriture prend uniquement en charge l'évolution du schéma, comme l'ajout ou la suppression de colonnes, mais pas la modification des types de colonnes, par exemple avec `ResolveChoice`.
- Le rédacteur ne prend pas en charge l'écriture vide DataFrames, par exemple pour écrire un fichier contenant uniquement des schémas. Lors de l'intégration au catalogue de données AWS Glue via un paramètre`enableUpdateCatalog=True`, toute tentative d'écriture d'un fichier vide ne DataFrame mettra pas à jour le catalogue de données. Cela entraînera la création d'une table sans schéma dans le catalogue de données.

Si votre transformation ne nécessite pas ces limitations, l'activation du graveur AWS Glue Parquet devrait améliorer les performances.

Utilisation du format XML dans AWS Glue

AWS Glue récupère les données des sources et écrit les données sur des cibles stockées et transportées dans différents formats de données. Si vos données sont stockées ou transportées au format de données XML, ce document présente les fonctionnalités disponibles pour utiliser vos données dans AWS Glue.

AWS Glue prend en charge l'utilisation du format XML. Ce format représente des structures de données hautement configurables et définies de manière rigide qui ne sont pas basées sur des lignes ou des colonnes. Le XML est hautement standardisé. Pour accéder à une présentation du format par l'autorité standard, consultez XML Essentials (Essentiels XML).

Vous pouvez utiliser AWS Glue pour lire des fichiers XML depuis Amazon S3, ainsi que `bzip` des `gzip` archives contenant des fichiers XML. Vous configurez le comportement de compression sur Paramètres de connexion S3 plutôt que dans la configuration décrite sur cette page.

Le tableau suivant indique les fonctionnalités courantes de AWS Glue compatibles avec l'option de format XML.

Lecture	Écrire	Lecture en streaming	Groupement des petits fichiers	Signets de tâche
Pris en charge	Non pris en charge	Non pris en charge	Pris en charge	Pris en charge

Exemple : lecture de XML depuis S3

Le lecteur XML prend un nom de balise XML. Il examine les éléments comportant cette balise dans son entrée pour en déduire un schéma et remplit un DynamicFrame avec les valeurs correspondantes. La fonctionnalité AWS Glue XML se comporte de la même manière que la source de données XML pour Apache Spark. Vous pourriez être en mesure de mieux comprendre le comportement de base en comparant ce lecteur à la documentation de ce projet.

Prérequis :Vous aurez besoin des chemins S3 (`s3path`) vers les fichiers ou dossiers XML que vous souhaitez lire, ainsi qu'à certaines informations concernant votre fichier XML. Vous aurez également besoin de la balise pour l'élément XML que vous souhaitez lire, `xmlTag`.

Configuration : dans vos options de fonction, spécifiez `format="xml"`. Dans vos `connection_options`, utilisez la clé `paths` pour spécifier `s3path`. Vous pouvez configurer plus en détail la manière dont le lecteur interagit avec S3 dans le `connection_options`. Pour plus de détails, voir Types de connexion et options pour l'ETL dans AWS Glue :Paramètres de connexion S3. Dans vos `format_options`, utilisez la clé `rowTag` pour spécifier `xmlTag`. Vous pouvez configurer plus en détail la façon dont le lecteur interprète les fichiers XML dans votre `format_options`. Pour plus d'informations, consultez .XML Configuration Reference (Référence de configuration XML).

Le script AWS Glue ETL suivant montre le processus de lecture de fichiers ou dossiers XML à partir de S3.

Python

Pour cet exemple, utilisez la méthode create_dynamic_frame.from_options.

```
# Example: Read XML from S3
# Set the rowTag option to configure the reader.

from awsglue.context import GlueContext
from pyspark.context import SparkContext

sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

dynamicFrame = glueContext.create_dynamic_frame.from_options(
    connection_type="s3",
    connection_options={"paths": ["s3://s3path"]},
    format="xml",
    format_options={"rowTag": "xmlTag"},
```

```
)
```

Vous pouvez également l'utiliser DataFrames dans un script (`pyspark.sql.DataFrame`).

```
dataFrame = spark.read\
    .format("xml")\
    .option("rowTag", "xmlTag")\
    .load("s3://s3path")
```

Scala

Pour cet exemple, utilisez l'opération getSourceWithFormat.

```
// Example: Read XML from S3
// Set the rowTag option to configure the reader.

import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.GlueContext
import org.apache.spark.sql.SparkSession

val glueContext = new GlueContext(SparkContext.getOrCreate())
val sparkSession: SparkSession = glueContext.getSparkSession

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val dynamicFrame = glueContext.getSourceWithFormat(
      formatOptions=JsonOptions("""{"rowTag": "xmlTag"}"""),
      connectionType="s3",
      format="xml",
      options=JsonOptions("""{"paths": ["s3://s3path"], "recurse": true}""")
    ).getDynamicFrame()
}
```

Vous pouvez également l'utiliser DataFrames dans un script (`org.apache.spark.sql.DataFrame`).

```
val dataFrame = spark.read
  .option("rowTag", "xmlTag")
  .format("xml")
  .load("s3://s3path“)
```

Référence de configurations XML

Vous pouvez utiliser ce qui suit `format_options` partout où les bibliothèques AWS Glue le spécifient `format="xml"` :

- `rowTag` – spécifie la balise XML du fichier à traiter comme ligne. Les balises de ligne ne se ferment pas automatiquement.
  - Type : Texte, Obligatoire
- `encoding` – spécifie le codage de caractère. Il peut s'agir du nom ou de l'alias d'unCharsetpris en charge par notre environnement d'exécution. Nous ne donnons pas de garanties spécifiques concernant la prise en charge de l'encodage, mais les principaux codages devraient fonctionner.
  - Type : texte, Valeur par défaut : `"UTF-8"`
- `excludeAttribute` – spécifie si vous souhaitez ou pas exclure les attributs dans les éléments.
  - Type : Booléen, Valeur par défaut : `false`
- `treatEmptyValuesAsNulls` – pécifie s'il convient de traiter un espace blanc comme valeur nulle.
  - Type : Booléen, Valeur par défaut : `false`
- `attributePrefix` – Un préfixe des attributs pour les différencier des éléments. Ce préfixe est utilisé pour les noms de champs.
  - Type : texte, Valeur par défaut : `"_"`
- `valueTag` – La balise utilisée pour une valeur lorsqu'il y a des attributs de l'élément qui n'ont pas d'enfant.
  - Type : texte, Valeur par défaut : `"_VALUE"`
- `ignoreSurroundingSpaces` – spécifie si les espaces blancs qui entourent les valeurs doivent être ignorés.
  - Type : Booléen, Valeur par défaut : `false`
- `withSchema` – Contient le schéma attendu, dans les cas où vous souhaitez remplacer le schéma déduit. Si vous n'utilisez pas cette option, AWS Glue déduit le schéma des données XML.
  - Type :Texte, Par défaut: Non applicable
  - La valeur doit être un objet JSON qui représente un `StructType`.

Spécifier manuellement le schéma XML

Exemple de schéma XML manuel

Il s'agit d'un exemple d'utilisation de l'option format `withSchema` pour spécifier le schéma des données XML.

```
from awsglue.gluetypes import *

schema = StructType([
  Field("id", IntegerType()),
  Field("name", StringType()),
  Field("nested", StructType([
    Field("x", IntegerType()),
    Field("y", StringType()),
    Field("z", ChoiceType([IntegerType(), StringType()]))
  ]))
])

datasource0 = create_dynamic_frame_from_options(
    connection_type,
    connection_options={"paths": ["s3://xml_bucket/someprefix"]},
    format="xml",
    format_options={"withSchema": json.dumps(schema.jsonValue())},
    transformation_ctx = ""
)
```

Utilisation du format Avro dans Glue AWS

AWS Glue récupère les données des sources et écrit les données sur des cibles stockées et transportées dans différents formats de données. Si vos données sont stockées ou transportées au format de données Avro, ce document présente les fonctionnalités disponibles pour utiliser vos données dans AWS Glue.

AWS Glue supporte l'utilisation du format Avro. Ce format est un format de données en colonnes et orienté vers la performance. Pour accéder à une présentation du format par l'autorité standard, consultez documentation Apache Avro 1.8.2 .

Vous pouvez utiliser AWS Glue pour lire des fichiers Avro depuis Amazon S3 et depuis des sources de streaming, ainsi que pour écrire des fichiers Avro sur Amazon S3. Vous pouvez lire et écrire `bzip2` et `gzip`des archives contenant des fichiers Avro provenant de S3. De plus, vous pouvez écrire des archives `deflate`, `snappy` et `xz` contenant des fichiers Avro. Vous configurez le comportement de compression sur Paramètres de connexion S3 plutôt que dans la configuration décrite sur cette page.

Le tableau suivant indique quelles opérations AWS Glue courantes prennent en charge l'option de format Avro.

Lecture	Écrire	Lecture en streaming	Groupement des petits fichiers	Signets de tâche
Pris en charge	Pris en charge	Pris en charge*	Non pris en charge	Pris en charge

* Pris en charge avec des restrictions. Pour de plus amples informations, veuillez consulter the section called "Notes et restrictions pour les sources en streaming Avro".

Exemple : lecture de fichiers Avro ou de dossiers de S3

Prérequis : vous aurez besoin des chemins S3 (`s3path`) vers des fichiers ou dossiers Avro que vous souhaitez lire.

Configuration : dans vos options de fonction, spécifiez `format="avro"`. Dans vos `connection_options`, utilisez la clé `paths` pour spécifier `s3path`. Vous pouvez configurer la manière dont le lecteur interagit avec S3 dans les `connection_options`. Pour plus de détails, voir Options de format de données pour les entrées et sorties ETL dans AWS Glue :the section called "Paramètres de connexion S3". Vous pouvez configurer la manière dont le lecteur interprète les fichiers Avro dans votre `format_options`. Pour plus d'informations, consultez .Référence de configuration Avro.

Le script AWS Glue ETL suivant montre le processus de lecture de fichiers ou de dossiers Avro depuis S3 :

Python

Pour cet exemple, utilisez la méthode create_dynamic_frame.from_options.

```
from pyspark.context import SparkContext
from awsglue.context import GlueContext

sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

dynamicFrame = glueContext.create_dynamic_frame.from_options(
```

```
    connection_type="s3",
    connection_options={"paths": ["s3://s3path"]},
    format="avro"
)
```

Scala

Pour cet exemple, utilisez l'opération getSourceWithFormat.

```
import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.GlueContext
import org.apache.spark.sql.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)

    val dynamicFrame = glueContext.getSourceWithFormat(
      connectionType="s3",
      format="avro",
      options=JsonOptions("""{"paths": ["s3://s3path"]}""")
    ).getDynamicFrame()
  }
```

Exemple : écriture de fichiers et dossiers Avro dans S3

Prérequis : Vous aurez besoin d'un initialisé DataFrame (`dataFrame`) ou DynamicFrame (`dynamicFrame`). Vous aurez également besoin de votre chemin de sortie S3, `s3path`.

Configuration : dans vos options de fonction, spécifiez `format="avro"`. Dans vos `connection_options`, utilisez la clé `paths` pour spécifier votre `s3path`. Vous pouvez modifier davantage la manière dont le scripteur interagit avec S3 dans les `connection_options`. Pour plus de détails, voir Options de format de données pour les entrées et sorties ETL dans AWS Glue :the section called "Paramètres de connexion S3". Vous pouvez modifier la façon dont le rédacteur interprète les fichiers Avro dans votre `format_options`. Pour plus d'informations, consultez .Référence de configuration Avro.

Le script AWS Glue ETL suivant montre le processus d'écriture de fichiers ou de dossiers Avro dans S3.

Python

Pour cet exemple, utilisez la méthode write_dynamic_frame.from_options.

```
from pyspark.context import SparkContext
from awsglue.context import GlueContext

sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

glueContext.write_dynamic_frame.from_options(
    frame=dynamicFrame,
    connection_type="s3",
    format="avro",
    connection_options={
        "path": "s3://s3path"
    }
)
```

Scala

Pour cet exemple, utilisez la méthode getSinkWithFormat.

```
import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.{DynamicFrame, GlueContext}
import org.apache.spark.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)

    glueContext.getSinkWithFormat(
      connectionType="s3",
      options=JsonOptions("""{"path": "s3://s3path"}"""),
      format="avro"
    ).writeDynamicFrame(dynamicFrame)
  }
}
```

Référence de configurations Avro

Vous pouvez utiliser les `format_options` valeurs suivantes partout où les bibliothèques AWS Glue le spécifient `format="avro"` :

- `version`— Spécifie la version du reader/writer format Apache Avro à prendre en charge. La valeur par défaut est 1.7. Vous pouvez spécifier `format_options={"version": “1.8”}` pour activer la lecture et l'écriture d'un type logique Avro. Pour plus d'informations, consultez les spécifications Apache Avro 1.7.7 et les spécifications Apache Avro 1.8.2.

  Le connecteur Apache Avro 1.8 prend en charge les conversions de type logique suivantes :

Pour le lecteur : ce tableau montre la conversion entre le type de données Avro (type logique et type primitif Avro) et le type de données AWS Glue `DynamicFrame` pour le lecteur Avro 1.7 et 1.8.

Type de données Avro : Type logique	Type de données Avro : Type Avro primitif	GlueDynamicFrame Type de données : Avro Reader 1.7	GlueDynamicFrame Type de données : Avro Reader 1.8
Décimal	octets	BINAIRE	Décimal
Décimal	corrigé(e)(s)	BINAIRE	Décimal
Date	int	INT	Date
Temps (milliseconde)	int	INT	INT
Temps (microseconde)	long	LONG	LONG
Horodatage (milliseconde)	long	LONG	Horodatage
Horodatage (microseconde)	long	LONG	LONG
Durée (pas un type logique)	corrigé(e)(s) sur 12	BINAIRE	BINAIRE

Pour le dispositif d'écriture : ce tableau affiche la conversion entre le type de données AWS Glue `DynamicFrame` et le type de données Avro pour le dispositif d'écriture Avro 1.7 et 1.8.

Type de données AWS Glue **`DynamicFrame`**	Type de données Avro : Avro Writer 1.7	Type de données Avro : Avro Writer 1.8
Décimal	String	decimal
Date	String	date
Horodatage	String	timestamp-micros

Assistance Avro Spark DataFrame

Pour utiliser Avro depuis l' DataFrame API Spark, vous devez installer le plugin Spark Avro pour la version de Spark correspondante. La version de Spark disponible dans votre tâche est déterminée par votre version de AWS Glue. Pour plus d'informations sur les versions Spark, consultez the section called “AWS Glue versions”. Ce plugin est maintenu par Apache, nous ne donnons pas de garanties spécifiques de support.

Dans AWS Glue 2.0, utilisez la version 2.4.3 du plugin Spark Avro. Vous pouvez trouver ce fichier JAR sur Maven Central, voirorg.apache.spark:spark-avro_2.12:2.4.3.

Dans AWS Glue 3.0, utilisez la version 3.1.1 du plugin Spark Avro. Vous pouvez trouver ce fichier JAR sur Maven Central, voirorg.apache.spark : spark-avro_2.12:3.1.1.

Pour inclure des éléments supplémentaires JARs dans une tâche AWS Glue ETL, utilisez le paramètre `--extra-jars` job. Pour de plus amples informations sur la définition des paramètres de la tâche, consultez the section called “Paramètres du job”. Vous pouvez également configurer ce paramètre dans AWS Management Console.

Utilisation du format GrokLog dans Glue AWS

AWS Glue récupère les données des sources et écrit les données sur des cibles stockées et transportées dans différents formats de données. Si vos données sont stockées ou transportées dans un format de texte brut peu structuré, ce document présente les fonctionnalités disponibles pour utiliser vos données dans les modèles AWS Glue through Grok.

AWS Supports à colle utilisant des motifs Grok. Les modèles Grok sont similaires aux groupes de capture d'expressions régulières. Ils reconnaissent les modèles de séquences de caractères dans un fichier texte brut et leur donnent un type et un objectif. Dans AWS Glue, leur objectif principal est de lire les journaux. Pour une introduction au Grok par les auteurs, voirRéférence Logstash : plug-in de filtre Grok.

Lecture	Écrire	Lecture en streaming	Groupement des petits fichiers	Signets de tâche
Pris en charge	Ne s'applique pas	Pris en charge	Pris en charge	Non pris en charge

Référence de configuration grokLog

Vous pouvez utiliser les valeurs `format_options` suivantes avec `format="grokLog"` :

- `logFormat` — spécifie le modèle Grok correspondant au format du journal.
- `customPatterns` — spécifie les modèles Grok supplémentaires utilisés ici.
- `MISSING` — spécifie le signal à utiliser lors de l'identification des valeurs manquantes. La valeur par défaut est `'-'`.
- `LineCount` — spécifie le nombre de lignes de chaque enregistrement de journal. La valeur par défaut est `'1'`, et actuellement seuls les enregistrements d'une ligne sont pris en charge.
- `StrictMode` — valeur booléenne indiquant si le mode strict est activé. En mode strict, le lecteur n'exécute pas de conversion ou récupération de type automatique. La valeur par défaut est `"false"`.

Utilisation du format Ion dans AWS Glue

AWS Glue récupère les données des sources et écrit les données sur des cibles stockées et transportées dans différents formats de données. Si vos données sont stockées ou transportées au format de données Ion, ce document présente les fonctionnalités disponibles pour utiliser vos données dans AWS Glue.

AWS Glue supporte l'utilisation du format Ion. Ce format représente des structures de données (qui ne sont pas basées sur des lignes ou des colonnes) dans des représentations binaires et en texte brut interchangeables. Pour accéder à une présentation du format par les auteurs, consultez Amazon Ion. (Pour plus d'informations, consultez la spécification Amazon Ion.)

Vous pouvez utiliser AWS Glue pour lire les fichiers Ion depuis Amazon S3. Vous pouvez lire`bzip` `gzip` et des archives contenant des fichiers Ion provenant de S3. Vous configurez le comportement de compression sur Paramètres de connexion S3 plutôt que dans la configuration décrite sur cette page.

Le tableau suivant indique quelles opérations AWS Glue courantes prennent en charge l'option de format Ion.

Lecture	Écrire	Lecture en streaming	Groupement des petits fichiers	Signets de tâche
Pris en charge	Non pris en charge	Non pris en charge	Pris en charge	Non pris en charge

Exemple : lecture de fichiers ou de dossiers Ion de S3

Prérequis : vous aurez besoin des chemins S3 (`s3path`) vers des fichiers ou dossiers Ion que vous souhaitez lire.

Configuration : dans vos options de fonction, spécifiez `format="json"`. Dans vos `connection_options`, utilisez la clé `paths` pour spécifier votre `s3path`. Vous pouvez configurer la manière dont le lecteur interagit avec S3 dans les `connection_options`. Pour plus de détails, voir Types de connexion et options pour l'ETL dans AWS Glue :the section called “Paramètres de connexion S3”.

Le script AWS Glue ETL suivant montre le processus de lecture de fichiers ou de dossiers Ion depuis S3 :

Python

Pour cet exemple, utilisez la méthode create_dynamic_frame.from_options.

```
# Example: Read ION from S3

from pyspark.context import SparkContext
from awsglue.context import GlueContext

sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)
```

```
dynamicFrame = glueContext.create_dynamic_frame.from_options(
    connection_type="s3",
    connection_options={"paths": ["s3://s3path"]},
    format="ion"
)
```

Scala

Pour cet exemple, utilisez l'opération getSourceWithFormat.

```
// Example: Read ION from S3

import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.GlueContext
import org.apache.spark.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)

    val dynamicFrame = glueContext.getSourceWithFormat(
      connectionType="s3",
      format="ion",
      options=JsonOptions("""{"paths": ["s3://s3path"], "recurse": true}""")
    ).getDynamicFrame()
  }
}
```

Référence de configurations Ion

Il n'y a aucune valeur `format_options` pour `format="ion"`.

Utilisation du format JSON dans AWS Glue

AWS Glue récupère les données des sources et écrit les données sur des cibles stockées et transportées dans différents formats de données. Si vos données sont stockées ou transportées au format JSON, ce document vous présente les fonctionnalités disponibles pour utiliser vos données dans AWS Glue.

AWS Glue prend en charge l'utilisation du format JSON. Ce format représente des structures de données avec une forme uniforme mais un contenu flexible qui ne sont pas basés sur des lignes ou

des colonnes. Le JSON est défini par des normes parallelles émises par plusieurs autorités, dont l'ECMA-404. Pour accéder à une présentation du format par une source fréquemment référencée, consultez Introducing JSON (Présentation de JSON).

Vous pouvez utiliser AWS Glue pour lire des fichiers JSON depuis Amazon S3, ainsi que `bzip` des fichiers JSON `gzip` compressés. Vous configurez le comportement de compression sur Paramètres de connexion S3 plutôt que dans la configuration décrite sur cette page.

Lecture	Écrire	Lecture en streaming	Groupement des petits fichiers	Signets de tâche	
Pris en charge	Pris en charge	Pris en charge	Pris en charge	Pris en charge	

Exemple : lecture de fichiers ou de dossiers JSON de S3

Prérequis : vous aurez besoin des chemins S3 (`s3path`) vers des fichiers ou dossiers JSON que vous souhaitez lire.

Configuration : dans vos options de fonction, spécifiez `format="json"`. Dans vos `connection_options`, utilisez la clé `paths` pour spécifier votre `s3path`. Vous pouvez également modifier la façon dont votre opération de lecture traversera s3 dans les options de connexion, consultez the section called "Paramètres de connexion S3" pour plus d'informations. Vous pouvez configurer la manière dont le lecteur interprète les fichiers JSON dans votre `format_options`. Pour plus d'informations, consultez JSON Configuration Reference (Référence de configuration JSON).

Le script AWS Glue ETL suivant montre le processus de lecture de fichiers ou de dossiers JSON depuis S3 :

Python

Pour cet exemple, utilisez la méthode create_dynamic_frame.from_options.

```
# Example: Read JSON from S3
# For show, we handle a nested JSON file that we can limit with the JsonPath
 parameter
# For show, we also handle a JSON where a single entry spans multiple lines
# Consider whether optimizePerformance is right for your workflow.
```

```
from pyspark.context import SparkContext
from awsglue.context import GlueContext

sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)
spark = glueContext.spark_session

dynamicFrame = glueContext.create_dynamic_frame.from_options(
    connection_type="s3",
    connection_options={"paths": ["s3://s3path"]},
    format="json",
    format_options={
        "jsonPath": "$.id",
        "multiline": True,
        # "optimizePerformance": True, -> not compatible with jsonPath, multiline
    }
)
```

Vous pouvez également l'utiliser DataFrames dans un script (`pyspark.sql.DataFrame`).

```
dataFrame = spark.read\
    .option("multiline", "true")\
    .json("s3://s3path")
```

Scala

Pour cet exemple, utilisez l'opération getSourceWithFormat.

```
// Example: Read JSON from S3
// For show, we handle a nested JSON file that we can limit with the JsonPath
 parameter
// For show, we also handle a JSON where a single entry spans multiple lines
// Consider whether optimizePerformance is right for your workflow.

import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.{DynamicFrame, GlueContext}
import org.apache.spark.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
```

```
    val dynamicFrame = glueContext.getSourceWithFormat(
      formatOptions=JsonOptions("""{"jsonPath": "$.id", "multiline": true,
 "optimizePerformance":false}"""),
      connectionType="s3",
      format="json",
      options=JsonOptions("""{"paths": ["s3://s3path"], "recurse": true}""")
    ).getDynamicFrame()
  }
}
```

Vous pouvez également l'utiliser DataFrames dans un script (`pyspark.sql.DataFrame`).

```
val dataFrame = spark.read
    .option("multiline", "true")
    .json("s3://s3path")
```

Exemple : écriture de fichiers et dossiers JSON dans S3

Prérequis : Vous aurez besoin d'un initialisé DataFrame (`dataFrame`) ou DynamicFrame (`dynamicFrame`). Vous aurez également besoin de votre chemin de sortie S3, `s3path`.

Configuration : dans vos options de fonction, spécifiez `format="json"`. Dans vos `connection_options`, utilisez la clé `paths` pour spécifier `s3path`. Vous pouvez modifier davantage la manière dont le scripteur interagit avec S3 dans les `connection_options`. Pour plus de détails, voir Options de format de données pour les entrées et sorties ETL dans AWS Glue :the section called "Paramètres de connexion S3". Vous pouvez configurer la manière dont le lecteur interprète les fichiers JSON dans votre `format_options`. Pour plus d'informations, consultez JSON Configuration Reference (Référence de configuration JSON).

Le script AWS Glue ETL suivant montre le processus d'écriture de fichiers ou de dossiers JSON à partir de S3 :

Python

Pour cet exemple, utilisez la méthode write_dynamic_frame.from_options.

```
# Example: Write JSON to S3

from pyspark.context import SparkContext
```

```
from awsglue.context import GlueContext

sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

glueContext.write_dynamic_frame.from_options(
    frame=dynamicFrame,
    connection_type="s3",
    connection_options={"path": "s3://s3path"},
    format="json"
)
```

Vous pouvez également l'utiliser DataFrames dans un script (`pyspark.sql.DataFrame`).

```
df.write.json("s3://s3path/")
```

Scala

Pour cet exemple, utilisez la méthode getSinkWithFormat.

```
// Example: Write JSON to S3

import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.{DynamicFrame, GlueContext}
import org.apache.spark.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)

    glueContext.getSinkWithFormat(
        connectionType="s3",
        options=JsonOptions("""{"path": "s3://s3path"}"""),
        format="json"
    ).writeDynamicFrame(dynamicFrame)
  }
}
```

Vous pouvez également l'utiliser DataFrames dans un script (`pyspark.sql.DataFrame`).

```
df.write.json("s3://s3path")
```

Référence de configurations Json

Vous pouvez utiliser les valeurs `format_options` suivantes avec `format="json"` :

- `jsonPath`— JsonPathExpression qui identifie un objet à lire dans des enregistrements. Cette expression est particulièrement utile lorsqu'un fichier contient des enregistrements imbriqués à l'intérieur d'un tableau externe. Par exemple, l' JsonPath expression suivante cible le `id` champ d'un objet JSON.

  ```
  format="json", format_options={"jsonPath": "$.id"}
  ```

- `multiline` — valeur booléenne qui spécifie si un même enregistrement peut couvrir plusieurs lignes. Cela peut se produire lorsqu'un champ contient un caractère de nouvelle ligne. Vous devez définir cette option sur `"true"` si aucun enregistrement ne s'étend sur plusieurs lignes. La valeur par défaut est `"false"`, qui permet un fractionnement en fichiers plus agressif pendant l'analyse.
- `optimizePerformance` — valeur booléenne qui spécifie s'il faut utiliser le lecteur CSV SIMD avancé avec les formats de mémoire en colonnes basés sur Apache Arrow. Disponible uniquement dans AWS Glue 3.0. Non compatible avec `multiline` ou `jsonPath`. Si vous fournissez l'une ou l'autre de ces options, AWS Glue sera obligée de revenir au lecteur standard.
- `withSchema` — Une valeur String qui spécifie un schéma de table au format décrit dans the section called "Spécifier le schéma XML". Utilisé uniquement avec `optimizePerformance` lors de la lecture à partir de connexions hors catalogue.

Utilisation d'un lecteur CSV SIMD vectorisé avec le format en colonnes Apache Arrow

AWS Glue La version 3.0 ajoute un lecteur vectorisé pour les données JSON. Il fonctionne deux fois plus vite dans certaines conditions que le lecteur standard. Ce lecteur est livré avec certaines limitations dont les utilisateurs doivent être conscients avant utilisation, décrites dans cette section.

Pour utiliser le lecteur optimisé, définissez `"optimizePerformance"` sur True dans le `format_options` ou la propriété table. Vous devrez également fournir `withSchema` sauf si vous lisez le catalogue. `withSchema` attend une entrée comme décrit dans le the section called "Spécifier le schéma XML".

```
// Read from S3 data source
glueContext.create_dynamic_frame.from_options(
    connection_type = "s3",
```

```
    connection_options = {"paths": ["s3://s3path"]},
    format = "json",
    format_options={
        "optimizePerformance": True,
        "withSchema": SchemaString
        })

// Read from catalog table
glueContext.create_dynamic_frame.from_catalog(
    database = database,
    table_name = table,
    additional_options = {
    // The vectorized reader for JSON can read your schema from a catalog table
 property.
        "optimizePerformance": True,
        })
```

Pour plus d'informations sur la création d'un *SchemaString* dans la bibliothèque AWS Glue, consultezthe section called "Types".

Limitations du lecteur CSV vectorisé

Prenez en compte les limitations suivantes :

- Les éléments JSON avec des objets imbriqués ou des valeurs de tableau ne sont pas pris en charge. S'il est fourni, AWS Glue retournera au lecteur standard.
- Un schéma doit être fourni, soit à partir du catalogue, soit avec `withSchema`.
- Non compatible avec `multiline` ou `jsonPath`. Si vous fournissez l'une ou l'autre de ces options, AWS Glue sera obligée de revenir au lecteur standard.
- Le fait de fournir des enregistrements d'entrée qui ne correspondent pas au schéma d'entrée entraînera un dysfonctionnement du lecteur.
- Les enregistrements d'erreurs ne seront pas créés.
- Les fichiers JSON contenant des caractères à plusieurs octets (tels que les caractères japonais ou chinois) ne sont pas pris en charge.

Utilisation du format ORC dans Glue AWS

AWS Glue récupère les données des sources et écrit les données sur des cibles stockées et transportées dans différents formats de données. Si vos données sont stockées ou transportées au

format ORC, ce document présente les fonctionnalités disponibles pour utiliser vos données dans AWS Glue.

AWS Glue supporte l'utilisation du format ORC. Ce format est un format de données en colonnes, orienté vers la performance. Pour accéder à une présentation du format par l'autorité standard, consultez Apache Orc.

Vous pouvez utiliser AWS Glue pour lire des fichiers ORC depuis Amazon S3 et depuis des sources de streaming, ainsi que pour écrire des fichiers ORC sur Amazon S3. Vous pouvez lire et écrire des archives `bzip` et `gzip` contenant des fichiers CSV provenant de S3. Vous configurez le comportement de compression sur Paramètres de connexion S3 plutôt que dans la configuration décrite sur cette page.

Le tableau suivant indique quelles opérations AWS Glue courantes prennent en charge l'option de format ORC.

Lecture	Écrire	Lecture en streaming	Groupement des petits fichiers	Signets de tâche
Pris en charge	Pris en charge	Pris en charge	Non pris en charge	Pris en charge*

* Supporté dans la version 1.0+ de AWS Glue

Exemple : lecture de fichiers ou de dossiers ORC à partir de S3

Prérequis : vous aurez besoin des chemins S3 (`s3path`) vers des fichiers ou dossiers ORC que vous souhaitez lire.

Configuration : dans vos options de fonction, spécifiez `format="orc"`. Dans vos `connection_options`, utilisez la clé `paths` pour spécifier votre `s3path`. Vous pouvez configurer la manière dont le lecteur interagit avec S3 dans les `connection_options`. Pour plus de détails, voir Types de connexion et options pour l'ETL dans AWS Glue :the section called “Paramètres de connexion S3”.

Le script AWS Glue ETL suivant montre le processus de lecture de fichiers ou de dossiers ORC depuis S3 :

Python

Pour cet exemple, utilisez la méthode create_dynamic_frame.from_options.

```
from pyspark.context import SparkContext
from awsglue.context import GlueContext

sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

dynamicFrame = glueContext.create_dynamic_frame.from_options(
    connection_type="s3",
    connection_options={"paths": ["s3://s3path"]},
    format="orc"
)
```

Vous pouvez également l'utiliser DataFrames dans un script (`pyspark.sql.DataFrame`).

```
dataFrame = spark.read\
    .orc("s3://s3path")
```

Scala

Pour cet exemple, utilisez l'opération getSourceWithFormat.

```
import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.GlueContext
import org.apache.spark.sql.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)

    val dynamicFrame = glueContext.getSourceWithFormat(
      connectionType="s3",
      format="orc",
      options=JsonOptions("""{"paths": ["s3://s3path"]}""")
    ).getDynamicFrame()
  }
}
```

Vous pouvez également l'utiliser DataFrames dans un script (`pyspark.sql.DataFrame`).

```
val dataFrame = spark.read
    .orc("s3://s3path")
```

Exemple : écriture de fichiers et dossiers ORC dans S3

Prérequis : Vous aurez besoin d'un initialisé DataFrame (`dataFrame`) ou DynamicFrame (`dynamicFrame`). Vous aurez également besoin de votre chemin de sortie S3, `s3path`.

Configuration : dans vos options de fonction, spécifiez `format="orc"`. Dans vos options de connexion, utilisez`paths`touche à spécifier`s3path`. Vous pouvez modifier davantage la manière dont le scripteur interagit avec S3 dans les `connection_options`. Pour plus de détails, voir Options de format de données pour les entrées et sorties ETL dans AWS Glue :the section called "Paramètres de connexion S3". L'exemple de code suivant montre le processus :

Python

Pour cet exemple, utilisez la méthode write_dynamic_frame.from_options.

```
from pyspark.context import SparkContext
from awsglue.context import GlueContext

sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

glueContext.write_dynamic_frame.from_options(
    frame=dynamicFrame,
    connection_type="s3",
    format="orc",
    connection_options={
        "path": "s3://s3path"
    }
)
```

Vous pouvez également l'utiliser DataFrames dans un script (`pyspark.sql.DataFrame`).

```
df.write.orc("s3://s3path/")
```

Scala

Pour cet exemple, utilisez la méthode getSinkWithFormat.

```
import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.{DynamicFrame, GlueContext}
import org.apache.spark.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)

    glueContext.getSinkWithFormat(
      connectionType="s3",
      options=JsonOptions("""{"path": "s3://s3path"}"""),
      format="orc"
    ).writeDynamicFrame(dynamicFrame)
  }
}
```

Vous pouvez également l'utiliser DataFrames dans un script (`pyspark.sql.DataFrame`).

```
df.write.orc("s3://s3path/")
```

Référence de configuration ORC

Il n'y a aucune valeur `format_options` pour `format="orc"`. Cependant, les options acceptées par le code SparkSQL sous-jacent peuvent lui être transmises au moyen du paramètre de carte `connection_options`.

Utilisation de frameworks de lacs de données avec des tâches AWS Glue ETL

Les frameworks de lacs de données open source simplifient le traitement incrémentiel des données pour les fichiers que vous stockez dans des lacs de données basés sur Amazon S3. AWS Glue La version 3.0 et les versions ultérieures prennent en charge les frameworks de lacs de données open source suivants :

- Apache Hudi
- Linux Foundation Delta Lake
- Apache Iceberg

Nous fournissons une prise en charge native de ces cadres afin de vous permettre de lire et écrire les données à stocker dans Amazon S3 de manière cohérente sur le plan transactionnel. Il n'est pas nécessaire d'installer un connecteur distinct ou d'effectuer des étapes de configuration supplémentaires pour utiliser ces cadres dans les tâches AWS Glue ETL.

Lorsque vous gérez des ensembles de données via le AWS Glue Data Catalog, vous pouvez utiliser AWS Glue des méthodes pour lire et écrire des tables de lacs de données avec Spark DataFrames. Vous pouvez également lire et écrire des données Amazon S3 à l'aide de l' DataFrame API Spark.

Dans cette vidéo, vous découvrirez les principes de base du fonctionnement d'Apache Hudi, d'Apache Iceberg et de Delta Lake. Vous découvrirez comment insérer, mettre à jour et supprimer des données dans votre lac de données et comment chacun de ces cadres fonctionne.

Rubriques

- Limitations
- Utilisation du framework Hudi dans Glue AWS
- Utilisation du framework Delta Lake dans AWS Glue
- Utilisation du framework Iceberg dans Glue AWS

Limitations

Tenez compte des limites suivantes avant d'utiliser des frameworks de lacs de données avec AWS Glue.

- Les AWS Glue `GlueContext` méthodes suivantes DynamicFrame ne prennent pas en charge la lecture et l'écriture de tables du framework Data Lake. Utilisez plutôt les `GlueContext` méthodes pour DataFrame ou DataFrame l'API Spark.
  - `create_dynamic_frame.from_catalog`
  - `write_dynamic_frame.from_catalog`
  - `getDynamicFrame`
  - `writeDynamicFrame`
- Les `GlueContext` méthodes suivantes DataFrame sont prises en charge par le contrôle des autorisations de Lake Formation :
  - `create_data_frame.from_catalog`
  - `write_data_frame.from_catalog`
  - `getDataFrame`

  - `writeDataFrame`
- Le regroupement de petits fichiers n'est pas pris en charge.
- Les signets de tâche ne sont pas pris en charge.
- Apache Hudi 0.10.1 pour AWS Glue 3.0 ne prend pas en charge les tables Hudi Merge on Read (MoR).
- `ALTER TABLE … RENAME TO`n'est pas disponible pour Apache Iceberg 0.13.1 pour 3.0. AWS Glue

Limitations des tables au format de lac de données gérées par les autorisations de Lake Formation

Les formats de lac de données sont intégrés à l' AWS Glue ETL via les autorisations Lake Formation. La création d'un DynamicFrame utilisateur n'`create_dynamic_frame`est pas prise en charge. Pour plus d'informations, consultez les exemples suivants :

- Exemple : lecture et écriture d'une table Iceberg avec contrôle des autorisations de Lake Formation
- Exemple : lecture et écriture d'une table Hudi avec contrôle des autorisations de Lake Formation
- Exemple : lecture et écriture d'une table Delta Lake avec contrôle des autorisations de Lake Formation

> **Note**
>
> L'intégration avec les autorisations AWS Glue ETL via Lake Formation pour Apache Hudi, Apache Iceberg et Delta Lake n'est prise en charge que dans la AWS Glue version 4.0.

Apache Iceberg offre la meilleure intégration avec l' AWS Glue ETL via les autorisations Lake Formation. Il prend en charge presque toutes les opérations et inclut la prise en charge de SQL.

Hudi prend en charge la plupart des opérations de base à l'exception des opérations administratives. C'est parce que ces options sont généralement effectuées via l'écriture de dataframes et spécifiées via `additional_options`. Vous devez l'utiliser pour créer AWS Glue APIs DataFrames pour vos opérations car SparkSQL n'est pas supporté.

Delta Lake prend uniquement en charge la lecture, l'ajout et le remplacement de données de table. Delta Lake nécessite l'utilisation de ses propres bibliothèques pour pouvoir effectuer diverses tâches telles que les mises à jour.

Les fonctionnalités suivantes ne sont pas disponibles pour les tables Iceberg gérées par les autorisations de Lake Formation.

- Compaction à l'aide AWS Glue de l'ETL
- Support de Spark SQL via AWS Glue ETL

Les limites des tables Hudi gérées par les autorisations de Lake Formation sont les suivantes :

- Suppression de fichiers orphelins

Les limites des tables Delta Lake gérées par les autorisations de Lake Formation sont les suivantes :

- Toutes les fonctionnalités autres que l'insertion et la lecture à partir des tables de Delta Lake.

Utilisation du framework Hudi dans Glue AWS

AWS Glue 3.0 et versions ultérieures prennent en charge le framework Apache Hudi pour les lacs de données. Hudi est un cadre de stockage de lac de données open source qui simplifie le traitement incrémentiel des données et le développement de pipelines de données. Cette rubrique décrit les fonctionnalités disponibles pour utiliser vos données dans AWS Glue lorsque vous transportez ou stockez vos données dans une table Hudi. Pour en savoir plus sur Hudi, consultez la documentation officielle d'Apache Hudi.

Vous pouvez utiliser AWS Glue pour effectuer des opérations de lecture et d'écriture sur des tables Hudi dans Amazon S3, ou travailler avec des tables Hudi à l'aide du catalogue de données AWS Glue. Des opérations supplémentaires, notamment l'insertion, la mise à jour et toutes les opérations d'Apache Spark, sont également prises en charge.

> **Note**
>
> L'implémentation d'Apache Hudi 0.15.0 dans AWS Glue 5.0 rétablit en interne le HUDI-7001. Elle ne présente pas la régression liée à la génération de clés complexes lorsque la clé d'enregistrement est constituée d'un seul champ. Toutefois, ce comportement est différent de celui d'OSS Apache Hudi 0.15.0.
> Apache Hudi 0.10.1 pour AWS Glue 3.0 ne prend pas en charge les tables Hudi Merge on Read (MoR).

Le tableau suivant répertorie la version Hudi incluse dans chaque version de AWS Glue.

AWS Version Glue	Version de Hudi prise en charge
5.1	1.0.2
5.0	0,15,0
4.0	0.12.1
3.0	0,1,1

Pour en savoir plus sur les frameworks de lacs de données pris en charge par AWS Glue, consultezUtilisation de frameworks de lacs de données avec des tâches AWS Glue ETL.

Activation de Hudi

Pour activer Hudi for AWS Glue, effectuez les tâches suivantes :

- Spécifiez `hudi` comme valeur pour le paramètre de tâche `--datalake-formats`. Pour de plus amples informations, veuillez consulter Utilisation des paramètres des tâches dans les tâches AWS Glue.
- Créez une clé nommée `--conf` d'après votre tâche AWS Glue et définissez-la sur la valeur suivante. Vous pouvez également définir la configuration suivante à l'aide de `SparkConf` dans votre script. Ces paramètres permettent à Apache Spark de gérer correctement les tables Hudi.

```
spark.serializer=org.apache.spark.serializer.KryoSerializer
```

- La prise en charge des autorisations de Lake Formation pour Hudi est activée par défaut pour AWS Glue 4.0. Aucune configuration supplémentaire n'est nécessaire pour les tables Hudi enregistrées pour reading/writing To Lake Formation. Pour lire une table Hudi enregistrée, le rôle IAM de la tâche AWS Glue doit disposer de l'autorisation SELECT. Pour écrire dans une table Hudi enregistrée, le rôle IAM de la tâche AWS Glue doit disposer de l'autorisation SUPER. Pour en savoir plus sur la gestion des autorisations de Lake Formation, consultez la section Octroi et révocation d'autorisations liées aux ressources du catalogue de données.

Utilisation d'une autre version de Hudi

Pour utiliser une version de Hudi non prise en charge par AWS Glue, spécifiez vos propres fichiers JAR Hudi à l'aide du paramètre `--extra-jars` job. N'incluez pas `hudi` comme valeur du paramètre de tâche `--datalake-formats`. Si vous utilisez AWS Glue 5.0 ou une version ultérieure, vous devez définir le paramètre de `--user-jars-first true` tâche.

Exemple : écrire une table Hudi sur Amazon S3 et l'enregistrer dans le catalogue de données AWS Glue

Cet exemple de script montre comment écrire une table Hudi sur Amazon S3 et enregistrer la table dans le catalogue de données AWS Glue. L'exemple utilise l'outil de synchronisation Hive de Hudi pour enregistrer la table.

> **Note**
>
> Dans cet exemple, vous devez définir le paramètre `--enable-glue-datacatalog` job afin d'utiliser le catalogue de données AWS Glue comme métastore Apache Spark Hive. Pour en savoir plus, veuillez consulter la section Utilisation des paramètres des tâches dans les tâches AWS Glue.

Python

```
# Example: Create a Hudi table from a DataFrame
# and register the table to Glue Data Catalog

additional_options={
    "hoodie.table.name": "<your_table_name>",
    "hoodie.database.name": "<your_database_name>",
    "hoodie.datasource.write.storage.type": "COPY_ON_WRITE",
    "hoodie.datasource.write.operation": "upsert",
    "hoodie.datasource.write.recordkey.field": "<your_recordkey_field>",
    "hoodie.datasource.write.precombine.field": "<your_precombine_field>",
    "hoodie.datasource.write.partitionpath.field": "<your_partitionkey_field>",
    "hoodie.datasource.write.hive_style_partitioning": "true",
    "hoodie.datasource.hive_sync.enable": "true",
    "hoodie.datasource.hive_sync.database": "<your_database_name>",
    "hoodie.datasource.hive_sync.table": "<your_table_name>",
    "hoodie.datasource.hive_sync.partition_fields": "<your_partitionkey_field>",
    "hoodie.datasource.hive_sync.partition_extractor_class":
 "org.apache.hudi.hive.MultiPartKeysValueExtractor",
    "hoodie.datasource.hive_sync.use_jdbc": "false",
```

```
    "hoodie.datasource.hive_sync.mode": "hms",
    "path": "s3://<s3Path/>"
}

dataFrame.write.format("hudi") \
    .options(**additional_options) \
    .mode("overwrite") \
    .save()
```

Scala

```
// Example: Example: Create a Hudi table from a DataFrame
// and register the table to Glue Data Catalog

val additionalOptions = Map(
  "hoodie.table.name" -> "<your_table_name>",
  "hoodie.database.name" -> "<your_database_name>",
  "hoodie.datasource.write.storage.type" -> "COPY_ON_WRITE",
  "hoodie.datasource.write.operation" -> "upsert",
  "hoodie.datasource.write.recordkey.field" -> "<your_recordkey_field>",
  "hoodie.datasource.write.precombine.field" -> "<your_precombine_field>",
  "hoodie.datasource.write.partitionpath.field" -> "<your_partitionkey_field>",
  "hoodie.datasource.write.hive_style_partitioning" -> "true",
  "hoodie.datasource.hive_sync.enable" -> "true",
  "hoodie.datasource.hive_sync.database" -> "<your_database_name>",
  "hoodie.datasource.hive_sync.table" -> "<your_table_name>",
  "hoodie.datasource.hive_sync.partition_fields" -> "<your_partitionkey_field>",
  "hoodie.datasource.hive_sync.partition_extractor_class" ->
 "org.apache.hudi.hive.MultiPartKeysValueExtractor",
  "hoodie.datasource.hive_sync.use_jdbc" -> "false",
  "hoodie.datasource.hive_sync.mode" -> "hms",
  "path" -> "s3://<s3Path/>")

dataFrame.write.format("hudi")
  .options(additionalOptions)
  .mode("append")
  .save()
```

Exemple : lecture d'une table Hudi depuis Amazon S3 à l'aide du catalogue de données AWS Glue

Cet exemple lit la table Hudi que vous avez créée dans Exemple : écrire une table Hudi sur Amazon S3 et l'enregistrer dans le catalogue de données AWS Glue depuis Amazon S3.

> **Note**
>
> Dans cet exemple, vous devez définir le paramètre `--enable-glue-datacatalog` job afin d'utiliser le catalogue de données AWS Glue comme métastore Apache Spark Hive. Pour en savoir plus, veuillez consulter la section Utilisation des paramètres des tâches dans les tâches AWS Glue.

Python

Pour cet exemple, utilisez la méthode `GlueContext.create_data_frame.from_catalog()`.

```
# Example: Read a Hudi table from Glue Data Catalog

from awsglue.context import GlueContext
from pyspark.context import SparkContext

sc = SparkContext()
glueContext = GlueContext(sc)

dataFrame = glueContext.create_data_frame.from_catalog(
    database = "<your_database_name>",
    table_name = "<your_table_name>"
)
```

Scala

Pour cet exemple, utilisez la méthode getCatalogSource.

```
// Example: Read a Hudi table from Glue Data Catalog

import com.amazonaws.services.glue.GlueContext
import org.apache.spark.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)

    val dataFrame = glueContext.getCatalogSource(
      database = "<your_database_name>",
```

```
      tableName = "<your_table_name>"
    ).getDataFrame()
  }
}
```

Exemple : mise à jour et insertion d'un **`DataFrame`** dans une table Hudi d'Amazon S3

Cet exemple utilise le catalogue de données AWS Glue pour DataFrame insérer un dans la table Hudi que vous avez créée dansExemple : écrire une table Hudi sur Amazon S3 et l'enregistrer dans le catalogue de données AWS Glue.

> **Note**
>
> Dans cet exemple, vous devez définir le paramètre `--enable-glue-datacatalog` job afin d'utiliser le catalogue de données AWS Glue comme métastore Apache Spark Hive. Pour en savoir plus, veuillez consulter la section Utilisation des paramètres des tâches dans les tâches AWS Glue.

Python

Pour cet exemple, utilisez la méthode `GlueContext.write_data_frame.from_catalog()`.

```
# Example: Upsert a Hudi table from Glue Data Catalog

from awsglue.context import GlueContext
from pyspark.context import SparkContext

sc = SparkContext()
glueContext = GlueContext(sc)

glueContext.write_data_frame.from_catalog(
    frame = dataFrame,
    database = "<your_database_name>",
    table_name = "<your_table_name>",
    additional_options={
        "hoodie.table.name": "<your_table_name>",
        "hoodie.database.name": "<your_database_name>",
        "hoodie.datasource.write.storage.type": "COPY_ON_WRITE",
        "hoodie.datasource.write.operation": "upsert",
        "hoodie.datasource.write.recordkey.field": "<your_recordkey_field>",
```

```
        "hoodie.datasource.write.precombine.field": "<your_precombine_field>",
        "hoodie.datasource.write.partitionpath.field": "<your_partitionkey_field>",
        "hoodie.datasource.write.hive_style_partitioning": "true",
        "hoodie.datasource.hive_sync.enable": "true",
        "hoodie.datasource.hive_sync.database": "<your_database_name>",
        "hoodie.datasource.hive_sync.table": "<your_table_name>",
        "hoodie.datasource.hive_sync.partition_fields": "<your_partitionkey_field>",
        "hoodie.datasource.hive_sync.partition_extractor_class":
 "org.apache.hudi.hive.MultiPartKeysValueExtractor",
        "hoodie.datasource.hive_sync.use_jdbc": "false",
        "hoodie.datasource.hive_sync.mode": "hms"
    }
)
```

Scala

Pour cet exemple, utilisez la méthode getCatalogSink.

```
// Example: Upsert a Hudi table from Glue Data Catalog

import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.util.JsonOptions
import org.apacke.spark.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    glueContext.getCatalogSink("<your_database_name>", "<your_table_name>",
      additionalOptions = JsonOptions(Map(
        "hoodie.table.name" -> "<your_table_name>",
        "hoodie.database.name" -> "<your_database_name>",
        "hoodie.datasource.write.storage.type" -> "COPY_ON_WRITE",
        "hoodie.datasource.write.operation" -> "upsert",
        "hoodie.datasource.write.recordkey.field" -> "<your_recordkey_field>",
        "hoodie.datasource.write.precombine.field" -> "<your_precombine_field>",
        "hoodie.datasource.write.partitionpath.field" ->
 "<your_partitionkey_field>",
        "hoodie.datasource.write.hive_style_partitioning" -> "true",
        "hoodie.datasource.hive_sync.enable" -> "true",
        "hoodie.datasource.hive_sync.database" -> "<your_database_name>",
        "hoodie.datasource.hive_sync.table" -> "<your_table_name>",
        "hoodie.datasource.hive_sync.partition_fields" ->
 "<your_partitionkey_field>",
```

```
        "hoodie.datasource.hive_sync.partition_extractor_class" ->
 "org.apache.hudi.hive.MultiPartKeysValueExtractor",
        "hoodie.datasource.hive_sync.use_jdbc" -> "false",
        "hoodie.datasource.hive_sync.mode" -> "hms"
      )))
      .writeDataFrame(dataFrame, glueContext)
  }
}
```

Exemple : lecture d'une table Hudi depuis Amazon S3 à l'aide de Spark

Cet exemple lit une table Hudi depuis Amazon S3 à l'aide de l' DataFrameAPI Spark.

Python

```
# Example: Read a Hudi table from S3 using a Spark DataFrame

dataFrame = spark.read.format("hudi").load("s3://<s3path/>")
```

Scala

```
// Example: Read a Hudi table from S3 using a Spark DataFrame

val dataFrame = spark.read.format("hudi").load("s3://<s3path/>")
```

Exemple : écriture d'une table Hudi sur Amazon S3 à l'aide de Spark

Cet exemple écrit une table Hudi sur Amazon S3 à l'aide de Spark.

Python

```
# Example: Write a Hudi table to S3 using a Spark DataFrame

dataFrame.write.format("hudi") \
    .options(**additional_options) \
    .mode("overwrite") \
    .save("s3://<s3Path/>")
```

Scala

```
// Example: Write a Hudi table to S3 using a Spark DataFrame
```

```
dataFrame.write.format("hudi")
  .options(additionalOptions)
  .mode("overwrite")
  .save("s3://<s3path/>")
```

Exemple : lecture et écriture d'une table Hudi avec contrôle des autorisations de Lake Formation

Cet exemple lit et écrit dans une table Hudi avec contrôle des autorisations de Lake Formation.

1. Créez une table Hudi et enregistrez-la dans Lake Formation.
   a. Pour activer le contrôle des autorisations de Lake Formation, vous devez d'abord enregistrer le chemin d'accès Amazon S3 de la table sur Lake Formation. Pour plus d'informations, consultez la rubrique Enregistrement d'un emplacement Amazon S3. Vous pouvez l'enregistrer depuis la console Lake Formation ou à l'aide de la AWS CLI :

   ```
   aws lakeformation register-resource --resource-arn arn:aws:s3:::<s3-bucket>/<s3-
   folder> --use-service-linked-role --region <REGION>
   ```

   Une fois que vous avez enregistré un emplacement Amazon S3, toute table AWS Glue pointant vers cet emplacement (ou l'un de ses emplacements enfants) renverra la valeur du `IsRegisteredWithLakeFormation` paramètre comme vraie lors de l'`GetTable`appel.

   b. Créez une table Hudi qui pointe vers le chemin Amazon S3 enregistré via l'API Spark dataframe :

   ```
   hudi_options = {
       'hoodie.table.name': table_name,
       'hoodie.database.name': database_name,
       'hoodie.datasource.write.storage.type': 'COPY_ON_WRITE',
       'hoodie.datasource.write.recordkey.field': 'product_id',
       'hoodie.datasource.write.table.name': table_name,
       'hoodie.datasource.write.operation': 'upsert',
       'hoodie.datasource.write.precombine.field': 'updated_at',
       'hoodie.datasource.write.hive_style_partitioning': 'true',
       'hoodie.upsert.shuffle.parallelism': 2,
       'hoodie.insert.shuffle.parallelism': 2,
       'path': <S3_TABLE_LOCATION>,
       'hoodie.datasource.hive_sync.enable': 'true',
       'hoodie.datasource.hive_sync.database': database_name,
       'hoodie.datasource.hive_sync.table': table_name,
   ```

```
    'hoodie.datasource.hive_sync.use_jdbc': 'false',
    'hoodie.datasource.hive_sync.mode': 'hms'
}

df_products.write.format("hudi")  \
    .options(**hudi_options)  \
    .mode("overwrite")  \
    .save()
```

2. Accordez à Lake Formation l'autorisation d'accéder au rôle IAM AWS Glue Job. Vous pouvez accorder des autorisations à partir de la console Lake Formation ou à l'aide de la AWS CLI. Pour plus d'informations, consultez la rubrique Octroi d'autorisations de table via la console Lake Formation et la méthode de ressource nommée
3. Lisez la table Hudi enregistrée dans Lake Formation. Le code est le même que celui de la lecture d'une table Hudi non enregistrée. Notez que le rôle IAM de la tâche AWS Glue doit disposer de l'autorisation SELECT pour que la lecture réussisse.

```
val dataFrame = glueContext.getCatalogSource(
     database = "<your_database_name>",
     tableName = "<your_table_name>"
   ).getDataFrame()
```

4. Écrire dans une table Hudi enregistrée dans Lake Formation. Le code est le même que celui de l'écriture dans une table Hudi non enregistrée. Notez que le rôle IAM de la tâche AWS Glue doit disposer de l'autorisation SUPER pour que l'écriture réussisse.

```
glueContext.getCatalogSink("<your_database_name>", "<your_table_name>",
      additionalOptions = JsonOptions(Map(
        "hoodie.table.name" -> "<your_table_name>",
        "hoodie.database.name" -> "<your_database_name>",
        "hoodie.datasource.write.storage.type" -> "COPY_ON_WRITE",
        "hoodie.datasource.write.operation" -> "<write_operation>",
        "hoodie.datasource.write.recordkey.field" -> "<your_recordkey_field>",
        "hoodie.datasource.write.precombine.field" -> "<your_precombine_field>",
        "hoodie.datasource.write.partitionpath.field" -> "<your_partitionkey_field>",
        "hoodie.datasource.write.hive_style_partitioning" -> "true",
        "hoodie.datasource.hive_sync.enable" -> "true",
        "hoodie.datasource.hive_sync.database" -> "<your_database_name>",
        "hoodie.datasource.hive_sync.table" -> "<your_table_name>",
        "hoodie.datasource.hive_sync.partition_fields" ->
 "<your_partitionkey_field>",
```

```
        "hoodie.datasource.hive_sync.partition_extractor_class" ->
"org.apache.hudi.hive.MultiPartKeysValueExtractor",
        "hoodie.datasource.hive_sync.use_jdbc" -> "false",
        "hoodie.datasource.hive_sync.mode" -> "hms"
      )))
      .writeDataFrame(dataFrame, glueContext)
```

Utilisation du framework Delta Lake dans AWS Glue

AWS Glue 3.0 et versions ultérieures prennent en charge le framework Delta Lake de la Linux Foundation. Delta Lake est un cadre de stockage de lac de données open source qui vous permet d'effectuer des transactions ACID, d'adapter la gestion des métadonnées et d'unifier le streaming et le traitement des données par lots. Cette rubrique décrit les fonctionnalités disponibles pour utiliser vos données dans AWS Glue lorsque vous transportez ou stockez vos données dans une table Delta Lake. Pour en savoir plus sur Delta Lake, consultez la documentation officielle de Delta Lake.

Vous pouvez utiliser AWS Glue pour effectuer des opérations de lecture et d'écriture sur des tables Delta Lake dans Amazon S3, ou travailler avec des tables Delta Lake à l'aide du catalogue de données AWS Glue. Des opérations supplémentaires telles que l'insertion, la mise à jour et la lecture et l'écriture de tables par lots sont également prises en charge. Lorsque vous utilisez des tables Delta Lake, vous avez également la possibilité d'avoir recours à des méthodes de la bibliothèque Python de Delta Lake, telles que `DeltaTable.forPath`. Pour plus d'informations sur la bibliothèque Python de Delta Lake, consultez la documentation Python de Delta Lake.

Le tableau suivant répertorie les versions de Delta Lake incluses dans chaque version de AWS Glue.

AWS Version Glue	Version de Delta Lake prise en charge
5.1	3.3.2
5.0	3.3.0
4.0	2.1.0
3.0	1.0.0

Pour en savoir plus sur les frameworks de lacs de données pris en charge par AWS Glue, consultezUtilisation de frameworks de lacs de données avec des tâches AWS Glue ETL.

Activer Delta Lake pour AWS Glue

Pour activer Delta Lake for AWS Glue, effectuez les tâches suivantes :

- Spécifiez `delta` comme valeur pour le paramètre de tâche `--datalake-formats`. Pour de plus amples informations, veuillez consulter Utilisation des paramètres des tâches dans les tâches AWS Glue.
- Créez une clé nommée `--conf` d'après votre tâche AWS Glue et définissez-la sur la valeur suivante. Vous pouvez également définir la configuration suivante à l'aide de `SparkConf` dans votre script. Ces paramètres permettent à Apache Spark de gérer correctement les tables Delta Lake.

```
spark.sql.extensions=io.delta.sql.DeltaSparkSessionExtension --conf
 spark.sql.catalog.spark_catalog=org.apache.spark.sql.delta.catalog.DeltaCatalog --
conf
 spark.delta.logStore.class=org.apache.spark.sql.delta.storage.S3SingleDriverLogStore
```

- La prise en charge des autorisations Lake Formation pour les tables Delta est activée par défaut pour AWS Glue 4.0. Aucune configuration supplémentaire n'est nécessaire reading/writing pour les tables Delta enregistrées par To Lake Formation. Pour lire une table Delta enregistrée, le rôle IAM de la tâche AWS Glue doit disposer de l'autorisation SELECT. Pour écrire dans une table Delta enregistrée, le rôle IAM de la tâche AWS Glue doit disposer de l'autorisation SUPER. Pour en savoir plus sur la gestion des autorisations de Lake Formation, consultez la section Octroi et révocation d'autorisations liées aux ressources du catalogue de données.

Utilisation d'une autre version de Delta Lake

Pour utiliser une version de Delta Lake non prise en charge par AWS Glue, spécifiez vos propres fichiers JAR Delta Lake à l'aide du paramètre `--extra-jars` job. N'incluez pas `delta` comme valeur du paramètre de tâche `--datalake-formats`. Si vous utilisez AWS Glue 5.0 ou une version ultérieure, vous devez définir le paramètre de `--user-jars-first true` tâche. Dans cette situation, pour utiliser la bibliothèque Python de Delta Lake, vous devez spécifier les fichiers JAR de la bibliothèque à l'aide du paramètre de tâche `--extra-py-files`. La bibliothèque Python est fournie dans les fichiers JAR de Delta Lake.

Exemple : écrire une table Delta Lake sur Amazon S3 et l'enregistrer dans le catalogue de données AWS Glue

Le script AWS Glue ETL suivant montre comment écrire une table Delta Lake sur Amazon S3 et enregistrer la table dans le catalogue de données AWS Glue.

Python

```
# Example: Create a Delta Lake table from a DataFrame
# and register the table to Glue Data Catalog

additional_options = {
    "path": "s3://<s3Path>"
}
dataFrame.write \
    .format("delta") \
    .options(**additional_options) \
    .mode("append") \
    .partitionBy("<your_partitionkey_field>") \
    .saveAsTable("<your_database_name>.<your_table_name>")
```

Scala

```
// Example: Example: Create a Delta Lake table from a DataFrame
// and register the table to Glue Data Catalog

val additional_options = Map(
  "path" -> "s3://<s3Path>"
)
dataFrame.write.format("delta")
  .options(additional_options)
  .mode("append")
  .partitionBy("<your_partitionkey_field>")
  .saveAsTable("<your_database_name>.<your_table_name>")
```

Exemple : lecture d'une table Delta Lake depuis Amazon S3 à l'aide du catalogue de données AWS Glue

Le script AWS Glue ETL suivant lit la table Delta Lake que vous avez créée dansExemple : écrire une table Delta Lake sur Amazon S3 et l'enregistrer dans le catalogue de données AWS Glue.

Python

Pour cet exemple, utilisez la méthode create_data_frame.from_catalog.

```
# Example: Read a Delta Lake table from Glue Data Catalog

from awsglue.context import GlueContext
from pyspark.context import SparkContext

sc = SparkContext()
glueContext = GlueContext(sc)

df = glueContext.create_data_frame.from_catalog(
    database="<your_database_name>",
    table_name="<your_table_name>",
    additional_options=additional_options
)
```

Scala

Pour cet exemple, utilisez la méthode getCatalogSource.

```
// Example: Read a Delta Lake table from Glue Data Catalog

import com.amazonaws.services.glue.GlueContext
import org.apacke.spark.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    val df = glueContext.getCatalogSource("<your_database_name>",
 "<your_table_name>",
      additionalOptions = additionalOptions)
      .getDataFrame()
  }
}
```

Exemple : insérez un **`DataFrame`** dans une table Delta Lake dans Amazon S3 à l'aide du catalogue de données AWS Glue

Cet exemple insère des données dans la table Delta Lake que vous avez créée dans Exemple : écrire une table Delta Lake sur Amazon S3 et l'enregistrer dans le catalogue de données AWS Glue.

> **Note**
>
> Dans cet exemple, vous devez définir le paramètre `--enable-glue-datacatalog` job afin d'utiliser le catalogue de données AWS Glue comme métastore Apache Spark Hive. Pour en savoir plus, veuillez consulter la section Utilisation des paramètres des tâches dans les tâches AWS Glue.

Python

Pour cet exemple, utilisez la méthode write_data_frame.from_catalog.

```
# Example: Insert into a Delta Lake table in S3 using Glue Data Catalog

from awsglue.context import GlueContext
from pyspark.context import SparkContext

sc = SparkContext()
glueContext = GlueContext(sc)

glueContext.write_data_frame.from_catalog(
    frame=dataFrame,
    database="<your_database_name>",
    table_name="<your_table_name>",
    additional_options=additional_options
)
```

Scala

Pour cet exemple, utilisez la méthode getCatalogSink.

```
// Example: Insert into a Delta Lake table in S3 using Glue Data Catalog

import com.amazonaws.services.glue.GlueContext
import org.apacke.spark.SparkContext
```

```
object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    glueContext.getCatalogSink("<your_database_name>", "<your_table_name>",
      additionalOptions = additionalOptions)
      .writeDataFrame(dataFrame, glueContext)
  }
}
```

Exemple : lecture d'une table Delta Lake depuis Amazon S3 à l'aide de l'API Spark

Cet exemple lit une table Delta Lake depuis Amazon S3 à l'aide de l'API Spark.

Python

```
# Example: Read a Delta Lake table from S3 using a Spark DataFrame

dataFrame = spark.read.format("delta").load("s3://<s3path/>")
```

Scala

```
// Example: Read a Delta Lake table from S3 using a Spark DataFrame

val dataFrame = spark.read.format("delta").load("s3://<s3path/>")
```

Exemple : écriture d'une table Delta Lake sur Amazon S3 à l'aide de Spark

Cet exemple écrit une table Delta Lake sur Amazon S3 à l'aide de Spark.

Python

```
# Example: Write a Delta Lake table to S3 using a Spark DataFrame

dataFrame.write.format("delta") \
    .options(**additional_options) \
    .mode("overwrite") \
    .partitionBy("<your_partitionkey_field>")
    .save("s3://<s3Path>")
```

Scala

```
// Example: Write a Delta Lake table to S3 using a Spark DataFrame

dataFrame.write.format("delta")
  .options(additionalOptions)
  .mode("overwrite")
  .partitionBy("<your_partitionkey_field>")
  .save("s3://<s3path/>")
```

Exemple : lecture et écriture d'une table Delta Lake avec contrôle des autorisations de Lake Formation

Cet exemple lit et écrit dans une table Delta Lake avec contrôle des autorisations de Lake Formation.

1. Créez une table Delta et enregistrez-la dans Lake Formation

   a. Pour activer le contrôle des autorisations de Lake Formation, vous devez d'abord enregistrer le chemin d'accès Amazon S3 de la table sur Lake Formation. Pour plus d'informations, consultez la rubrique Enregistrement d'un emplacement Amazon S3. Vous pouvez l'enregistrer depuis la console Lake Formation ou à l'aide de la AWS CLI :

   ```
   aws lakeformation register-resource --resource-arn arn:aws:s3:::<s3-bucket>/<s3-
   folder> --use-service-linked-role --region <REGION>
   ```

   Une fois que vous avez enregistré un emplacement Amazon S3, toute table AWS Glue pointant vers cet emplacement (ou l'un de ses emplacements enfants) renverra la valeur du `IsRegisteredWithLakeFormation` paramètre comme vraie lors de l'`GetTable`appel.

   b. Créez une table Delta qui pointe vers le chemin Amazon S3 enregistré via Spark :

   > **Note**
   >
   > Voici des exemples Python.

   ```
   dataFrame.write \
    .format("delta") \
    .mode("overwrite") \
    .partitionBy("<your_partitionkey_field>") \
   ```

```
  .save("s3://<the_s3_path>")
```

Une fois les données écrites sur Amazon S3, utilisez le robot AWS Glue pour créer une nouvelle table de catalogue Delta. Pour plus d'informations, voir Présentation du support de table natif de Delta Lake avec les robots AWS Glue.

Vous pouvez également créer le tableau manuellement via l'`CreateTable`API AWS Glue.

2. Accordez à Lake Formation l'autorisation d'accéder au rôle IAM AWS Glue Job. Vous pouvez accorder des autorisations depuis la console Lake Formation ou utiliser la AWS CLI. Pour plus d'informations, consultez la rubrique Octroi d'autorisations de table via la console Lake Formation et la méthode de ressource nommée
3. Lisez la table Delta enregistrée dans Lake Formation. Le code est le même que celui de la lecture d'une table Delta non enregistrée. Notez que le rôle IAM de la tâche AWS Glue doit disposer de l'autorisation SELECT pour que la lecture réussisse.

```
# Example: Read a Delta Lake table from Glue Data Catalog

df = glueContext.create_data_frame.from_catalog(
    database="<your_database_name>",
    table_name="<your_table_name>",
    additional_options=additional_options
)
```

4. Écrire dans une table Delta enregistrée dans Lake Formation. Le code est le même que celui de l'écriture dans une table Delta non enregistrée. Notez que le rôle IAM de la tâche AWS Glue doit disposer de l'autorisation SUPER pour que l'écriture réussisse.

   Par défaut, AWS Glue utilise `Append` le mode SaveMode. Vous pouvez le modifier en définissant l'option saveMode dans `additional_options`. Pour plus d'informations sur la prise en charge du mode SaveMode dans les tables Delta, consultez Écrire dans une table.

```
glueContext.write_data_frame.from_catalog(
    frame=dataFrame,
    database="<your_database_name>",
    table_name="<your_table_name>",
    additional_options=additional_options
)
```

Utilisation du framework Iceberg dans Glue AWS

AWS Glue 3.0 et versions ultérieures prennent en charge le framework Apache Iceberg pour les lacs de données. Iceberg fournit un format de table hautement performant qui fonctionne exactement comme une table SQL. Cette rubrique décrit les fonctionnalités disponibles pour utiliser vos données dans AWS Glue lorsque vous transportez ou stockez vos données dans une table Iceberg. Pour en savoir plus sur Iceberg, consultez la documentation officielle d'Apache Iceberg.

Vous pouvez utiliser AWS Glue pour effectuer des opérations de lecture et d'écriture sur des tables Iceberg dans Amazon S3, ou travailler avec des tables Iceberg à l'aide du catalogue de données AWS Glue. Des opérations supplémentaires, notamment l'insertion et toutes les requêtes Spark et écritures Spark, sont également prises en charge. La mise à jour n'est pas prise en charge pour les tables Iceberg.

> **Note**
>
> `ALTER TABLE … RENAME TO` n'est pas disponible pour Apache Iceberg 0.13.1 pour AWS Glue 3.0.

Le tableau suivant répertorie les versions d'Iceberg incluses dans chaque version de AWS Glue.

AWS Version Glue	Version d'Iceberg prise en charge
5.1	1.10.0
5.0	1.7.1
4.0	1.0.0
3.0	0.13.1

Pour en savoir plus sur les frameworks de lacs de données pris en charge par AWS Glue, consultezUtilisation de frameworks de lacs de données avec des tâches AWS Glue ETL.

Activation du cadre Iceberg

Pour activer Iceberg for AWS Glue, effectuez les tâches suivantes :

- Spécifiez `iceberg` comme valeur pour le paramètre de tâche `--datalake-formats`. Pour de plus amples informations, veuillez consulter Utilisation des paramètres des tâches dans les tâches AWS Glue.
- Créez une clé nommée `--conf` d'après votre tâche AWS Glue et définissez-la sur la valeur suivante. Vous pouvez également définir la configuration suivante à l'aide de `SparkConf` dans votre script. Ces paramètres permettent à Apache Spark de gérer correctement les tables Iceberg.

```
spark.sql.extensions=org.apache.iceberg.spark.extensions.IcebergSparkSessionExtensions
--conf spark.sql.catalog.glue_catalog=org.apache.iceberg.spark.SparkCatalog
--conf spark.sql.catalog.glue_catalog.warehouse=s3://<your-warehouse-dir>/
--conf spark.sql.catalog.glue_catalog.catalog-
impl=org.apache.iceberg.aws.glue.GlueCatalog
--conf spark.sql.catalog.glue_catalog.io-impl=org.apache.iceberg.aws.s3.S3FileIO
```

  Si vous lisez ou écrivez sur des tables Iceberg enregistrées auprès de Lake Formation, suivez les instructions de AWS Glue 5.0 et versions ultérieures. the section called “Lake Formation pour FGAC” Dans AWS Glue 4.0, ajoutez la configuration suivante pour activer la prise en charge de Lake Formation.

```
--conf spark.sql.catalog.glue_catalog.glue.lakeformation-enabled=true
--conf spark.sql.catalog.glue_catalog.glue.id=<table-catalog-id>
```

  Si vous utilisez AWS Glue 3.0 avec Iceberg 0.13.1, vous devez définir les configurations supplémentaires suivantes pour utiliser le gestionnaire de verrouillage Amazon DynamoDB afin de garantir une transaction atomique. AWS Glue 4.0 ou version ultérieure utilise le verrouillage optimiste par défaut. Pour plus d'informations, consultez Iceberg AWS Integrations dans la documentation officielle d'Apache Iceberg.

```
--conf spark.sql.catalog.glue_catalog.lock-
impl=org.apache.iceberg.aws.glue.DynamoLockManager
--conf spark.sql.catalog.glue_catalog.lock.table=<your-dynamodb-table-name>
```

Utilisation d'une autre version d'Iceberg

Pour utiliser une version d'Iceberg non prise en charge par AWS Glue, spécifiez vos propres fichiers JAR Iceberg à l'aide du paramètre `--extra-jars` job. N'incluez pas `iceberg` comme valeur du

paramètre `--datalake-formats`. Si vous utilisez AWS Glue 5.0 ou une version ultérieure, vous devez définir le paramètre de `--user-jars-first true` tâche.

Activation du chiffrement pour les tables Iceberg

> **Note**
>
> Les tables Iceberg possèdent leurs propres mécanismes pour activer le chiffrement côté serveur. Vous devez activer cette configuration en plus de la configuration de sécurité de AWS Glue.

Pour activer le chiffrement côté serveur sur les tables Iceberg, consultez les conseils de la documentation d'Iceberg.

Ajouter une configuration Spark pour les différentes régions Iceberg

Pour ajouter une configuration Spark supplémentaire pour l'accès aux tables interrégionales d'Iceberg avec le AWS Glue Data Catalog AWS Lake Formation, suivez les étapes ci-dessous :

1. Créez un point d'accès multi-régions.
2. Définissez les propriétés Spark suivantes :

```
-----
    --conf spark.sql.catalog.my_catalog.s3.use-arn-region-enabled=true \
    --conf spark.sql.catalog.{CATALOG}.s3.access-points.bucket1",
 "arn:aws:s3::<account-id>:accesspoint/<mrap-id>.mrap \
    --conf spark.sql.catalog.{CATALOG}.s3.access-points.bucket2",
 "arn:aws:s3::<account-id>:accesspoint/<mrap-id>.mrap
-----
```

Exemple : écrire une table Iceberg sur Amazon S3 et l'enregistrer dans le catalogue de données AWS Glue

Cet exemple de script montre comment écrire dans une table Iceberg dans Amazon S3. L'exemple utilise Iceberg AWS Integrations pour enregistrer la table dans le catalogue de données AWS Glue.

Python

```
# Example: Create an Iceberg table from a DataFrame
```

```
# and register the table to Glue Data Catalog

dataFrame.createOrReplaceTempView("tmp_<your_table_name>")

query = f"""
CREATE TABLE glue_catalog.<your_database_name>.<your_table_name>
USING iceberg
TBLPROPERTIES ("format-version"="2")
AS SELECT * FROM tmp_<your_table_name>
"""
spark.sql(query)
```

Scala

```
// Example: Example: Create an Iceberg table from a DataFrame
// and register the table to Glue Data Catalog

dataFrame.createOrReplaceTempView("tmp_<your_table_name>")

val query = """CREATE TABLE glue_catalog.<your_database_name>.<your_table_name>
USING iceberg
TBLPROPERTIES ("format-version"="2")
AS SELECT * FROM tmp_<your_table_name>
"""
spark.sql(query)
```

Vous pouvez également écrire dans une table Iceberg sur Amazon S3 et dans le catalogue de données à l'aide des méthodes Spark.

Conditions préalables : vous devez créer un catalogue pour que la bibliothèque Iceberg puisse l'utiliser. Lorsque vous utilisez le AWS catalogue de données AWS Glue, Glue simplifie les choses. Le catalogue de données AWS Glue est préconfiguré pour être utilisé par les bibliothèques Spark en tant que`glue_catalog`. Les tables du catalogue de données sont identifiées par a *`databaseName`* et *`tableName`* a. Pour plus d'informations sur le catalogue de données AWS Glue, consultezDécouverte et catalogage des données.

Si vous n'utilisez pas le catalogue de données AWS Glue, vous devrez approvisionner un catalogue via le Spark APIs. Pour plus d'informations, consultez Spark Configuration dans la documentation Iceberg.

Cet exemple écrit une table Iceberg dans Amazon S3 et le catalogue de données à l'aide de Spark.

Python

```
# Example: Write an Iceberg table to S3 on the Glue Data Catalog

# Create (equivalent to CREATE TABLE AS SELECT)
dataFrame.writeTo("glue_catalog.databaseName.tableName") \
    .tableProperty("format-version", "2") \
    .create()

# Append (equivalent to INSERT INTO)
dataFrame.writeTo("glue_catalog.databaseName.tableName") \
    .tableProperty("format-version", "2") \
    .append()
```

Scala

```
// Example: Write an Iceberg table to S3 on the Glue Data Catalog

// Create (equivalent to CREATE TABLE AS SELECT)
dataFrame.writeTo("glue_catalog.databaseName.tableName")
    .tableProperty("format-version", "2")
    .create()

// Append (equivalent to INSERT INTO)
dataFrame.writeTo("glue_catalog.databaseName.tableName")
    .tableProperty("format-version", "2")
    .append()
```

Exemple : lecture d'une table Iceberg depuis Amazon S3 à l'aide du catalogue de données AWS Glue

Cet exemple lit la table Iceberg que vous avez créée dans Exemple : écrire une table Iceberg sur Amazon S3 et l'enregistrer dans le catalogue de données AWS Glue.

Python

Pour cet exemple, utilisez la méthode `GlueContext.create_data_frame.from_catalog()`.

```
# Example: Read an Iceberg table from Glue Data Catalog

from awsglue.context import GlueContext
```

```
from pyspark.context import SparkContext

sc = SparkContext()
glueContext = GlueContext(sc)

df = glueContext.create_data_frame.from_catalog(
    database="<your_database_name>",
    table_name="<your_table_name>",
    additional_options=additional_options
)
```

Scala

Pour cet exemple, utilisez la méthode getCatalogSource.

```
// Example: Read an Iceberg table from Glue Data Catalog

import com.amazonaws.services.glue.GlueContext
import org.apacke.spark.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    val df = glueContext.getCatalogSource("<your_database_name>",
 "<your_table_name>",
      additionalOptions = additionalOptions)
      .getDataFrame()
  }
}
```

Exemple : **`DataFrame`** insérer un dans une table Iceberg dans Amazon S3 à l'aide du catalogue de données AWS Glue

Cet exemple insère des données dans la table Iceberg que vous avez créée dans Exemple : écrire une table Iceberg sur Amazon S3 et l'enregistrer dans le catalogue de données AWS Glue.

> **Note**
>
> Dans cet exemple, vous devez définir le paramètre `--enable-glue-datacatalog` job afin d'utiliser le catalogue de données AWS Glue comme métastore Apache Spark Hive. Pour

en savoir plus, veuillez consulter la section Utilisation des paramètres des tâches dans les tâches AWS Glue.

Python

Pour cet exemple, utilisez la méthode `GlueContext.write_data_frame.from_catalog()`.

```
# Example: Insert into an Iceberg table from Glue Data Catalog

from awsglue.context import GlueContext
from pyspark.context import SparkContext

sc = SparkContext()
glueContext = GlueContext(sc)

glueContext.write_data_frame.from_catalog(
    frame=dataFrame,
    database="<your_database_name>",
    table_name="<your_table_name>",
    additional_options=additional_options
)
```

Scala

Pour cet exemple, utilisez la méthode getCatalogSink.

```
// Example: Insert into an Iceberg table from Glue Data Catalog

import com.amazonaws.services.glue.GlueContext
import org.apacke.spark.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    glueContext.getCatalogSink("<your_database_name>", "<your_table_name>",
      additionalOptions = additionalOptions)
      .writeDataFrame(dataFrame, glueContext)
  }
}
```

Exemple : lecture d'une table Iceberg depuis Amazon S3 à l'aide de Spark

Conditions préalables : vous devez créer un catalogue pour que la bibliothèque Iceberg puisse l'utiliser. Lorsque vous utilisez le AWS catalogue de données AWS Glue, Glue simplifie les choses. Le catalogue de données AWS Glue est préconfiguré pour être utilisé par les bibliothèques Spark en tant que`glue_catalog`. Les tables du catalogue de données sont identifiées par a *`databaseName`* et *`tableName`* a. Pour plus d'informations sur le catalogue de données AWS Glue, consultezDécouverte et catalogage des données.

Si vous n'utilisez pas le catalogue de données AWS Glue, vous devrez approvisionner un catalogue via le Spark APIs. Pour plus d'informations, consultez Spark Configuration dans la documentation Iceberg.

Cet exemple lit une table Iceberg dans Amazon S3 à partir du catalogue de données à l'aide de Spark.

Python

```
# Example: Read an Iceberg table on S3 as a DataFrame from the Glue Data Catalog

dataFrame = spark.read.format("iceberg").load("glue_catalog.databaseName.tableName")
```

Scala

```
// Example: Read an Iceberg table on S3 as a DataFrame from the Glue Data Catalog

val dataFrame =
 spark.read.format("iceberg").load("glue_catalog.databaseName.tableName")
```

Exemple : lecture et écriture d'une table Iceberg avec contrôle des autorisations de Lake Formation

Cet exemple lit et écrit dans une table Iceberg avec contrôle des autorisations de Lake Formation.

> **Note**
>
> Cet exemple ne fonctionne que dans AWS Glue 4.0. Dans AWS Glue 5.0 et versions ultérieures, suivez les instructions fournies dansthe section called “Lake Formation pour FGAC”.

1. Créez une table Iceberg et enregistrez-la dans Lake Formation :

   a. Pour activer le contrôle des autorisations de Lake Formation, vous devez d'abord enregistrer le chemin d'accès Amazon S3 de la table sur Lake Formation. Pour plus d'informations, consultez la rubrique Enregistrement d'un emplacement Amazon S3. Vous pouvez l'enregistrer depuis la console Lake Formation ou à l'aide de la AWS CLI :

   ```
   aws lakeformation register-resource --resource-arn arn:aws:s3:::<s3-bucket>/<s3-folder> --use-service-linked-role --region <REGION>
   ```

   Une fois que vous avez enregistré un emplacement Amazon S3, toute table AWS Glue pointant vers cet emplacement (ou l'un de ses emplacements enfants) renverra la valeur du `IsRegisteredWithLakeFormation` paramètre comme vraie lors de l'`GetTable`appel.

   b. Créez une table Iceberg qui pointe vers le chemin enregistré via Spark SQL :

   > **Note**
   >
   > Voici des exemples Python.

   ```
   dataFrame.createOrReplaceTempView("tmp_<your_table_name>")

   query = f"""
   CREATE TABLE glue_catalog.<your_database_name>.<your_table_name>
   USING iceberg
   AS SELECT * FROM tmp_<your_table_name>
   """
   spark.sql(query)
   ```

   Vous pouvez également créer le tableau manuellement via l'`CreateTable`API AWS Glue. Pour plus d'informations, consultez la section Création de tables Apache Iceberg.

   > **Note**
   >
   > L'API `UpdateTable` ne prend actuellement pas en charge le format de table Iceberg en tant qu'entrée de l'opération.

2. Accordez à Lake Formation l'autorisation d'accéder au rôle IAM da la tâche. Vous pouvez accorder des autorisations à partir de la console Lake Formation ou à l'aide de la AWS CLI. Pour plus

d'informations, voir : https://docs.aws.amazon.com/lake-formation/ latest/dg/granting -table-permissions.html

3. Lisez une table Iceberg enregistrée dans Lake Formation. Le code est le même que celui de la lecture d'une table Iceberg non enregistrée. Notez que le rôle IAM de votre job AWS Glue doit disposer de l'autorisation SELECT pour que la lecture réussisse.

```
# Example: Read an Iceberg table from the AWS Glue Data Catalog
from awsglue.context import GlueContextfrom pyspark.context import SparkContext

sc = SparkContext()
glueContext = GlueContext(sc)

df = glueContext.create_data_frame.from_catalog(
    database="<your_database_name>",
    table_name="<your_table_name>",
    additional_options=additional_options
)
```

4. Écrivez dans une table Iceberg enregistrée dans Lake Formation. Le code est le même que celui de l'écriture dans une table Iceberg non enregistrée. Notez que le rôle IAM de votre tâche AWS Glue doit disposer de l'autorisation SUPER pour que l'écriture réussisse.

```
glueContext.write_data_frame.from_catalog(
    frame=dataFrame,
    database="<your_database_name>",
    table_name="<your_table_name>",
    additional_options=additional_options
)
```

Référence de configuration partagée

Vous pouvez utiliser les valeurs `format_options` suivantes avec tout type de format.

- `attachFilename` : une chaîne au format approprié à utiliser comme nom de colonne. Si vous fournissez cette option, le nom du fichier source de l'enregistrement sera ajouté à l'enregistrement. La valeur du paramètre sera utilisée comme nom de colonne.
- `attachTimestamp` : une chaîne au format approprié à utiliser comme nom de colonne. Si vous fournissez cette option, l'heure de la modification du fichier source de l'enregistrement sera ajoutée à l'enregistrement. La valeur du paramètre sera utilisée comme nom de colonne.

AWS Support de Glue Data Catalog pour les tâches Spark SQL

Le catalogue de données AWS Glue est un catalogue compatible avec le métastore Apache Hive. Vous pouvez configurer vos tâches et points de terminaison de développement AWS Glue de manière à utiliser Data Catalog en tant que métastore Apache Hive externe. Vous pouvez alors exécuter directement les requêtes Apache Spark SQL sur les tables stockées dans le Data Catalog. Les images dynamiques AWS Glue s'intègrent au Catalog Data par défaut. Toutefois, avec cette fonctionnalité, les tâches Spark SQL peuvent commencer à utiliser Data Catalog en tant que métastore Hive externe.

Cette fonctionnalité nécessite un accès réseau à l'API de point de terminaison AWS Glue. Pour AWS Glue avec des connexions situées dans des sous-réseaux privés, vous devez configurer un point de terminaison VPC ou une passerelle NAT pour fournir l'accès au réseau. Pour plus d'informations sur la configuration des points de terminaison de VPC, reportez-vous à Configuration de l'accès réseau aux magasins de données. Pour créer une passerelle NAT, veuillez consulter Passerelles NAT dans le Guide de l'utilisateur Amazon VPC.

Vous pouvez configurer des tâches et des points de terminaison de développement AWS Glue en ajoutant l'argument `"--enable-glue-datacatalog": ""` aux arguments de tâche et aux arguments de point de terminaison de développement, respectivement. La transmission de cet argument définit certaines configurations dans Spark qui lui permettent d'accéder à Data Catalog en tant que métastore Hive externe. Elle permet également la prise en charge de Hive dans l'objet `SparkSession` créé dans la tâche ou le point de terminaison de développement AWS Glue.

Pour activer l'accès au catalogue de données, cochez la case Utiliser le catalogue de données AWS Glue comme métastore Hive dans le groupe d'options du catalogue sur la page Ajouter une tâche ou Ajouter un point de terminaison de la console. Notez que le rôle IAM utilisé pour la tâche ou le point de terminaison de développement doit disposer des autorisations `glue:CreateDatabase`. Une base de données appelée « `default` » est créée dans Data Catalog, si elle n'existe pas déjà.

Prenons l'exemple de la façon dont vous pouvez utiliser cette fonction dans vos tâches Spark SQL. L'exemple suivant suppose que vous avez analysé l'ensemble de données des législateurs américains disponible dans `s3://awsglue-datasets/examples/us-legislators`.

Pour accéder aux serialize/deserialize données des tables définies dans le catalogue de données AWS Glue, Spark SQL a besoin de la SerDe classe Hive pour le format défini dans le catalogue de données AWS Glue dans le chemin de classe de la tâche Spark.

SerDes car certains formats courants sont distribués parAWS Glue. Voici les liens Amazon S3 pour :

- JSON
- XML
- Grok

Ajoutez le JSON SerDe en tant que JAR supplémentaire au point de terminaison de développement. Pour les tâches, vous pouvez ajouter l'`--extra-jars`argument SerDe using the dans le champ des arguments. Pour de plus amples informations, veuillez consulter Utilisation des paramètres des tâches dans les tâches AWS Glue.

Voici un exemple d'entrée JSON pour créer un point de terminaison de développement avec Data Catalog activé pour Spark SQL.

```
{
    "EndpointName": "Name",
    "RoleArn": "role_ARN",
    "PublicKey": "public_key_contents",
    "NumberOfNodes": 2,
    "Arguments": {
      "--enable-glue-datacatalog": ""
    },
    "ExtraJarsS3Path": "s3://crawler-public/json/serde/json-serde.jar"
}
```

Maintenant, interrogez les tables créées à partir de l'ensemble de données des législateurs américains à l'aide de Spark SQL.

```
>>> spark.sql("use legislators")
DataFrame[]
>>> spark.sql("show tables").show()
+-----------+-----------------+-----------+
|   database|        tableName|isTemporary|
+-----------+-----------------+-----------+
|legislators|       areas_json|      false|
|legislators|   countries_json|      false|
|legislators|      events_json|      false|
|legislators| memberships_json|      false|
|legislators|organizations_json|      false|
|legislators|     persons_json|      false|
+-----------+-----------------+-----------+
```

```
>>> spark.sql("describe memberships_json").show()
+--------------------+---------+-----------------+
|            col_name|data_type|          comment|
+--------------------+---------+-----------------+
|             area_id|   string|from deserializer|
|     on_behalf_of_id|   string|from deserializer|
|     organization_id|   string|from deserializer|
|                role|   string|from deserializer|
|           person_id|   string|from deserializer|
|legislative_perio...|   string|from deserializer|
|          start_date|   string|from deserializer|
|            end_date|   string|from deserializer|
+--------------------+---------+-----------------+
```

Si la SerDe classe correspondant au format n'est pas disponible dans le chemin de classe de la tâche, une erreur semblable à la suivante s'affichera.

```
>>> spark.sql("describe memberships_json").show()

Caused by: MetaException(message:java.lang.ClassNotFoundException Class
 org.openx.data.jsonserde.JsonSerDe not found)
    at
 org.apache.hadoop.hive.metastore.MetaStoreUtils.getDeserializer(MetaStoreUtils.java:399)
    at
 org.apache.hadoop.hive.ql.metadata.Table.getDeserializerFromMetaStore(Table.java:276)
    ... 64 more
```

Pour afficher uniquement les éléments `organization_id` distincts à partir de la table `memberships`, exécutez la requête SQL suivante.

```
>>> spark.sql("select distinct organization_id from memberships_json").show()
+--------------------+
|     organization_id|
+--------------------+
|d56acebe-8fdc-47b...|
|8fa6c3d2-71dc-478...|
+--------------------+
```

Si vous devez faire de même avec des images dynamiques, exécutez le code suivant.

```
>>> memberships = glueContext.create_dynamic_frame.from_catalog(database="legislators",
 table_name="memberships_json")
```

```
>>> memberships.toDF().createOrReplaceTempView("memberships")
>>> spark.sql("select distinct organization_id from memberships").show()
+--------------------+
|     organization_id|
+--------------------+
|d56acebe-8fdc-47b...|
|8fa6c3d2-71dc-478...|
+--------------------+
```

Bien qu' DynamicFrames ils soient optimisés pour les opérations ETL, le fait de permettre à Spark SQL d'accéder directement au catalogue de données fournit un moyen concis d'exécuter des instructions SQL complexes ou de porter des applications existantes.

Utilisation des marque-pages de tâche

AWS Glue pour Spark utilise les signets de tâche pour suivre les données qui ont déjà été traitées. Pour un résumé de la fonctionnalité des signets de tâches et de ce qu'elle prend en charge, consultez the section called “Suivi des données traitées à l'aide de signets de tâche”. Lorsque vous programmez une AWS Glue tâche avec des signets, vous avez accès à une flexibilité qui n'est pas disponible dans les tâches visuelles.

- Lorsque vous lisez à partir de JDBC, vous pouvez spécifier la ou les colonnes à utiliser comme clés de signet dans votre script. AWS Glue
- Vous pouvez choisir la `transformation_ctx` à appliquer à chaque appel de méthode.

Appelez toujours *`job.init`* au début du script et *`job.commit`* à la fin du script avec les paramètres correctement configurés. Ces deux fonctions initialisent le service de signet et mettent à jour le changement d'état vers le service. Les marque-pages ne fonctionnent pas s’ils ne sont pas appelés.

Spécifier les clés des signets

Pour les flux de travail JDBC, le signet assure le suivi des lignes lues par votre tâche en comparant les valeurs des champs clés à la valeur d'un signet. Cela n'est ni nécessaire ni applicable aux flux de travail Amazon S3. Lorsque vous écrivez un AWS Glue script sans l'éditeur visuel, vous pouvez spécifier la colonne à suivre à l'aide de signets. Vous pouvez aussi spécifier plusieurs colonnes. Des écarts dans la séquence de valeurs sont autorisés lorsque vous spécifiez des clés de signet définies par l'utilisateur.

> **Warning**
>
> Si des clés de signets définies par l'utilisateur sont utilisées, elles doivent toutes croître ou diminuer de façon monotone. Lorsque vous sélectionnez des champs supplémentaires pour une clé composée, les champs correspondants à des concepts tels que les « versions mineures » ou « numéros de révision » ne répondent pas à ce critère, car leurs valeurs sont réutilisées dans l'ensemble de votre jeu de données.

Vous pouvez spécifier `jobBookmarkKeys` et `jobBookmarkKeysSortOrder` de la manière suivante :

- `create_dynamic_frame.from_catalog` – Utiliser `additional_options`.
- `create_dynamic_frame.from_options` – Utiliser `connection_options`.

Contexte de transformation

De nombreuses méthodes de trame AWS Glue PySpark dynamique incluent un paramètre facultatif nommé`transformation_ctx`, qui est un identifiant unique pour l'instance d'opérateur ETL. Le paramètre `transformation_ctx` permet d'identifier les informations d'état dans un marque-page de tâche pour l'opérateur donné. Plus précisément, AWS Glue utilise `transformation_ctx` pour indexer la clé sur l'état du marque-page.

> **Warning**
>
> Le paramètre `transformation_ctx` sert de clé pour rechercher dans l'état du marque-page une source spécifique dans votre script. Pour que le marque-page fonctionne correctement, vous devez toujours conserver la source et la cohérence du paramètre `transformation_ctx` associée. La modification de la propriété source ou le renommage du paramètre `transformation_ctx` peut rendre le marque-page précédent invalide et le filtrage basé sur l'horodatage peut ne pas donner le bon résultat.

Pour que les marque-pages de tâche fonctionnent correctement, activez le paramètre de signet de tâche et définissez le paramètre `transformation_ctx`. Si vous ne transmettez pas le paramètre `transformation_ctx`, les marque-pages de tâche ne sont pas activés pour une trame dynamique ou un tableau utilisé dans la méthode. Par exemple, si vous avez une tâche ETL qui lit et joint deux

sources Amazon S3, vous pouvez choisir de transmettre le paramètre `transformation_ctx` uniquement aux méthodes qui doivent activer les marque-pages. Si vous réinitialisez le marque-page de tâche d'une tâche, toutes les transformations associées à cette tâche sont réinitialisées, quel que soit le paramètre `transformation_ctx` utilisé.

Pour plus d'informations sur la classe `DynamicFrameReader`, consultez DynamicFrameReader classe. Pour plus d'informations sur les PySpark extensions, consultezAWS Référence des PySpark extensions Glue.

Exemples

Example

Voici un exemple de script généré pour une source de données Amazon S3. Les parties du script nécessaires à l'utilisation des signets de tâche sont indiquées en italique. Pour plus d'informations sur ces éléments, consultez l'API GlueContext classe et l'API DynamicFrameWriter classe.

```
# Sample Script
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job

args = getResolvedOptions(sys.argv, ['JOB_NAME'])
sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
job = Job(glueContext)
job.init(args['JOB_NAME'], args)

datasource0 = glueContext.create_dynamic_frame.from_catalog(
    database = "database",
    table_name = "relatedqueries_csv",
    transformation_ctx = "datasource0"
)

applymapping1 = ApplyMapping.apply(
    frame = datasource0,
    mappings = [("col0", "string", "name", "string"), ("col1", "string", "number",
 "string")],
    transformation_ctx = "applymapping1"
```

```
)

datasink2 = glueContext.write_dynamic_frame.from_options(
    frame = applymapping1,
    connection_type = "s3",
    connection_options = {"path": "s3://input_path"},
    format = "json",
    transformation_ctx = "datasink2"
)


job.commit()
```

Example

Voici un exemple de script généré pour une source JDBC. La table source est une table d'employés avec la colonne empno comme clé principale. Bien que par défaut la tâche utilise une clé principale séquentielle comme clé de marque-page si aucune clé de marque-page n'est spécifiée, car empno n'est pas nécessairement séquentiel, il peut y avoir des lacunes dans les valeurs, elle n'est pas considérée comme une clé de marque-page par défaut. Par conséquent, le script désigne explicitement empno comme clé de marque-page. Cette partie du code est affichée en italique.

```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job

args = getResolvedOptions(sys.argv, ['JOB_NAME'])

sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
job = Job(glueContext)
job.init(args['JOB_NAME'], args)

datasource0 = glueContext.create_dynamic_frame.from_catalog(
    database = "hr",
    table_name = "emp",
    transformation_ctx = "datasource0",
    additional_options = {"jobBookmarkKeys":["empno"],"jobBookmarkKeysSortOrder":"asc"}
)
```

```
applymapping1 = ApplyMapping.apply(
    frame = datasource0,
    mappings = [("ename", "string", "ename", "string"), ("hrly_rate", "decimal(38,0)",
 "hrly_rate", "decimal(38,0)"), ("comm", "decimal(7,2)", "comm", "decimal(7,2)"),
 ("hiredate", "timestamp", "hiredate", "timestamp"), ("empno", "decimal(5,0)", "empno",
 "decimal(5,0)"), ("mgr", "decimal(5,0)", "mgr", "decimal(5,0)"), ("photo", "string",
 "photo", "string"), ("job", "string", "job", "string"), ("deptno", "decimal(3,0)",
 "deptno", "decimal(3,0)"), ("ssn", "decimal(9,0)", "ssn", "decimal(9,0)"), ("sal",
 "decimal(7,2)", "sal", "decimal(7,2)")],
    transformation_ctx = "applymapping1"
)

datasink2 = glueContext.write_dynamic_frame.from_options(
    frame = applymapping1,
    connection_type = "s3",
    connection_options = {"path": "s3://hr/employees"},
    format = "csv",
    transformation_ctx = "datasink2"
)

job.commit()
```

Utilisation de la détection des données sensibles en dehors de AWS Glue Studio

AWS Glue Studio vous permet de détecter les données sensibles, mais vous pouvez également utiliser la fonctionnalité de détection des données sensibles en dehors de AWS Glue Studio.

Pour afficher la liste complète des types de données sensibles gérés, consultez Managed data types.

Détection de données sensibles Détection à l'aide de types d'informations personnelles AWS gérés

AWS Glue en fournit deux APIs dans une tâche AWS Glue ETL. `detect()` et `classifyColumns()` :

```
  detect(frame: DynamicFrame,
      entityTypesToDetect: Seq[String],
      outputColumnName: String = "DetectedEntities",
      detectionSensitivity: String = "LOW"): DynamicFrame

 detect(frame: DynamicFrame,
      detectionParameters: JsonOptions,
```

```
    outputColumnName: String = "DetectedEntities",
    detectionSensitivity: String = "LOW"): DynamicFrame

  classifyColumns(frame: DynamicFrame,
    entityTypesToDetect: Seq[String],
    sampleFraction: Double = 0.1,
    thresholdFraction: Double = 0.1,
    detectionSensitivity: String = "LOW")
```

Vous pouvez utiliser l'`detect()`API pour identifier les types d' AWS informations personnelles gérées et les types d'entités personnalisés. Une nouvelle colonne est automatiquement créée avec le résultat de la détection. L'API `classifyColumns()` renvoie une carte où les clés sont les noms de colonnes et les valeurs sont la liste des types d'entités détectés. `SampleFraction` indique la fraction des données à échantillonner lors de la recherche d'entités PII, tandis que `ThresholdFraction` indique la fraction des données qui doit être satisfaite pour qu'une colonne soit identifiée comme données PII.

Détection au niveau des lignes

Dans l'exemple, la tâche exécute les actions suivantes à l'aide des touches `detect()` et `classifyColumns()` APIs :

- lire les données d'un Amazon S3 bucket et les transformer en DynamicFrame
- détection des instances « Email » et « Credit Card » dans le DynamicFrame
- renvoi d'un DynamicFrame avec les valeurs d'origine plus une colonne qui englobe le résultat de la détection pour chaque ligne
- écrire le DynamicFrame renvoyé dans un autre chemin Amazon S3

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.MappingSpec
import com.amazonaws.services.glue.errors.CallSite
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._
import com.amazonaws.services.glue.ml.EntityDetector
```

```
 object GlueApp {
   def main(sysArgs: Array[String]) {
     val spark: SparkContext = new SparkContext()
     val glueContext: GlueContext = new GlueContext(spark)
     val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
     Job.init(args("JOB_NAME"), glueContext, args.asJava)
     val frame=
glueContext.getSourceWithFormat(formatOptions=JsonOptions("""{"quoteChar": "\"",
"withHeader": true, "separator": ","}"""), connectionType="s3", format="csv",
options=JsonOptions("""{"paths": ["s3://pathToSource"], "recurse": true}"""),
transformationContext="AmazonS3_node1650160158526").getDynamicFrame()

     val frameWithDetectedPII = EntityDetector.detect(frame, Seq("EMAIL",
"CREDIT_CARD"))

     glueContext.getSinkWithFormat(connectionType="s3",
options=JsonOptions("""{"path": "s3://pathToOutput/", "partitionKeys": []}"""),
transformationContext="someCtx",
format="json").writeDynamicFrame(frameWithDetectedPII)

     Job.commit()
   }
 }
```

Détection au niveau des lignes avec actions détaillées

Dans l'exemple, la tâche exécute les actions suivantes à l'aide de `detect()` APIs :

- lecture des données à partir d'un compartiment Amazon S3 et transformation de ces données en DynamicFrame
- détection des types de données sensibles pour « USA_PTIN », « BANK_ACCOUNT », « USA_SSN », « USA_PASSPORT_NUMBER » et « PHONE_NUMBER » dans le DynamicFrame
- renvoi d'un DynamicFrame avec les valeurs modifiées masquées plus une colonne qui englobe le résultat de la détection pour chaque ligne
- écriture du DynamicFrame renvoyé dans un autre chemin Amazon S3

Contrairement à l'API `detect()` ci-dessus, celle-ci utilise des actions détaillées que les types d'entités détectent. Pour de plus amples informations, veuillez consulter Paramètres de détection pour l'utilisation de `detect()`.

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.MappingSpec
import com.amazonaws.services.glue.errors.CallSite
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._
import com.amazonaws.services.glue.ml.EntityDetector

object GlueApp {
  def main(sysArgs: Array[String]) {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)
    val frame =
 glueContext.getSourceWithFormat(formatOptions=JsonOptions("""{"quoteChar": "\"",
 "withHeader": true, "separator": ","}"""), connectionType="s3", format="csv",
 options=JsonOptions("""{"paths": ["s3://pathToSource"], "recurse": true}"""),
 transformationContext="AmazonS3_node_source").getDynamicFrame()

    val detectionParameters = JsonOptions(
      """
        {
          "USA_DRIVING_LICENSE": [{
            "action": "PARTIAL_REDACT",
            "sourceColumns": ["Driving License"],
            "actionOptions": {
              "matchPattern": "[0-9]",
              "redactChar": "*"
            }
          }],
          "BANK_ACCOUNT": [{
            "action": "DETECT",
            "sourceColumns": ["*"]
          }],
          "USA_SSN": [{
            "action": "SHA256_HASH",
            "sourceColumns": ["SSN"]
          }],
          "IP_ADDRESS": [{
            "action": "REDACT",
```

```
            "sourceColumns": ["IP Address"],
            "actionOptions": {"redactText": "*****"}
          }],
          "PHONE_NUMBER": [{
            "action": "PARTIAL_REDACT",
            "sourceColumns": ["Phone Number"],
            "actionOptions": {
              "numLeftCharsToExclude": 1,
              "numRightCharsToExclude": 0,
              "redactChar": "*"
            }
          }]
        }
      """
    )

    val frameWithDetectedPII = EntityDetector.detect(frame, detectionParameters,
 "DetectedEntities", "HIGH")

    glueContext.getSinkWithFormat(connectionType="s3", options=JsonOptions("""{"path":
 "s3://pathToOutput/", "partitionKeys": []}"""),
 transformationContext="AmazonS3_node_target",
 format="json").writeDynamicFrame(frameWithDetectedPII)

    Job.commit()
  }
}
```

Détection au niveau des colonnes

Dans l'exemple, la tâche exécute les actions suivantes à l'aide de `classifyColumns()` APIs :

- lecture des données à partir d'un compartiment Amazon S3 et transformation de ces données en DynamicFrame
- détection des instances « Email » et « Credit Card » dans le DynamicFrame
- définir les paramètres pour échantillonner 100 % de la colonne, marquer une entité comme détectée si elle se trouve dans 10 % des cellules et a une sensibilité « FAIBLE »
- renvoie une carte où les clés sont les noms de colonnes et les valeurs sont une liste des types d'entités détectés

- écriture du DynamicFrame renvoyé dans un autre chemin Amazon S3

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.MappingSpec
import com.amazonaws.services.glue.errors.CallSite
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._
import com.amazonaws.services.glue.DynamicFrame
import com.amazonaws.services.glue.ml.EntityDetector

object GlueApp {
  def main(sysArgs: Array[String]) {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)
    val frame =
 glueContext.getSourceWithFormat(formatOptions=JsonOptions("""{"quoteChar":
 "\"", "withHeader": true, "separator": ",", "optimizePerformance": false}"""),
 connectionType="s3", format="csv", options=JsonOptions("""{"paths": ["s3://
pathToSource"], "recurse": true}"""), transformationContext="frame").getDynamicFrame()

    import glueContext.sparkSession.implicits._

    val detectedDataFrame = EntityDetector.classifyColumns(
        frame,
        entityTypesToDetect = Seq("CREDIT_CARD", "PHONE_NUMBER"),
        sampleFraction = 1.0,
        thresholdFraction = 0.1,
        detectionSensitivity = "LOW"
    )
    val detectedDF = (detectedDataFrame).toSeq.toDF("columnName", "entityTypes")
    val DetectSensitiveData_node = DynamicFrame(detectedDF, glueContext)

    glueContext.getSinkWithFormat(connectionType="s3", options=JsonOptions("""{"path":
 "s3://pathToOutput", "partitionKeys": []}"""), transformationContext="someCtx",
 format="json").writeDynamicFrame(DetectSensitiveData_node)

    Job.commit()
  }
```

```
}
```

Détection de données sensibles Détection à l'aide de AWS CustomEntityType types d'informations personnelles

Vous pouvez définir des entités personnalisées via AWS Studio. Toutefois, pour utiliser cette fonctionnalité en dehors de AWS Studio, vous devez d'abord définir les types d'entités personnalisés, puis ajouter les types d'entités personnalisés définis à la liste des`entityTypesToDetect`.

Si vos données contiennent des types de données sensibles spécifiques (tels que l'ID d'employé), vous pouvez créer des entités personnalisées en appelant l'API `CreateCustomEntityType()`. L'exemple suivant définit le type d'entité personnalisé « EMPLOYEE_ID » pour l'API `CreateCustomEntityType()` avec les paramètres de demande :

```
{
    "name": "EMPLOYEE_ID",
    "regexString": "\d{4}-\d{3}",
    "contextWords": ["employee"]
}
```

Modifiez ensuite la tâche pour utiliser le nouveau type de données sensibles personnalisé en ajoutant le type d'entité personnalisé (EMPLOYEE_ID) à l'API `EntityDetector()` :

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.MappingSpec
import com.amazonaws.services.glue.errors.CallSite
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._
import com.amazonaws.services.glue.ml.EntityDetector

object GlueApp {
  def main(sysArgs: Array[String]) {
    val spark: SparkContext = new SparkContext()
```

```
      val glueContext: GlueContext = new GlueContext(spark)
      val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
      Job.init(args("JOB_NAME"), glueContext, args.asJava)
      val frame=
glueContext.getSourceWithFormat(formatOptions=JsonOptions("""{"quoteChar": "\"",
"withHeader": true, "separator": ","}"""), connectionType="s3", format="csv",
options=JsonOptions("""{"paths": ["s3://pathToSource"], "recurse": true}"""),
transformationContext="AmazonS3_node1650160158526").getDynamicFrame()

      val frameWithDetectedPII = EntityDetector.detect(frame, Seq("EMAIL",
"CREDIT_CARD", "EMPLOYEE_ID"))

      glueContext.getSinkWithFormat(connectionType="s3",
options=JsonOptions("""{"path": "s3://pathToOutput/", "partitionKeys": []}"""),
transformationContext="someCtx",
format="json").writeDynamicFrame(frameWithDetectedPII)

      Job.commit()
    }
 }
```

> **Note**
>
> Si un type de données sensibles personnalisé est défini avec le même nom qu'un type d'entité géré existant, le type de données sensibles personnalisé prévaut et remplace la logique du type d'entité géré.

Paramètres de détection pour l'utilisation de **`detect()`**

Cette méthode est utilisée pour détecter des entités dans un DynamicFrame. Il renvoie une nouvelle colonne DataFrame contenant les valeurs d'origine et une colonne supplémentaire outputColumnName contenant des métadonnées de détection des informations personnelles. Un masquage personnalisé peut être effectué une fois DynamicFrame celui-ci renvoyé dans le AWS Glue script, ou l'API detect () avec actions détaillées peut être utilisée à la place.

```
detect(frame: DynamicFrame,
       entityTypesToDetect: Seq[String],
       outputColumnName: String = "DetectedEntities",
       detectionSensitivity: String = "LOW"): DynamicFrame
```

Paramètres :

- frame — (type :`DynamicFrame`) L'entrée DynamicFrame contenant les données à traiter.
- entityTypesToDétecter — (type :`[Seq[String]`) Liste des types d'entités à détecter. Il peut s'agir de types d'entités gérés ou de types d'entités personnalisés.
- outputColumnName— (type :`String`, par défaut : "DetectedEntities«) Le nom de la colonne dans laquelle les entités détectées seront stockées. S'il n'est pas fourni, le nom de colonne par défaut est « DetectedEntities ».
- detectionSensitivity — (type : `String`, options : « FAIBLE » ou « ÉLEVÉ », par défaut : « FAIBLE ») spécifie la sensibilité du processus de détection. Les options valides sont « FAIBLE » ou « ÉLEVÉ ». Si ce n'est pas le cas, la sensibilité par défaut est réglée sur « FAIBLE ».

Paramètres `outputColumnName` :

Le nom de la colonne dans laquelle les entités détectées seront stockées. S'il n'est pas fourni, le nom de colonne par défaut est « DetectedEntities ». Pour chaque ligne de la colonne de sortie, la colonne supplémentaire inclut un mappage du nom de la colonne avec les métadonnées de l'entité détectée avec les paires clé-valeur suivantes :

- entityType : le type d'entité détecté.
- début : la position de départ de l'entité détectée dans les données d'origine.
- fin : la position de fin de l'entité détectée dans les données d'origine.
- ActionUsed — Action exécutée sur l'entité détectée (par exemple, « DETECT », « REDACT », « PARTIAL_REDACT », « _HASH »). SHA256

Exemple :

```
{
   "DetectedEntities":{
      "SSN Col":[
         {
            "entityType":"USA_SSN",
            "actionUsed":"DETECT",
            "start":4,
            "end":15
```

```
         }
      ],
      "Random Data col":[
         {
            "entityType":"BANK_ACCOUNT",
            "actionUsed":"PARTIAL_REDACT",
            "start":4,
            "end":13
         },
         {
            "entityType":"IP_ADDRESS",
            "actionUsed":"REDACT",
            "start":4,
            "end":13
         }
      ]
   }
}
```

Paramètres de détection pour **`detect()`** avec des actions détaillées

Cette méthode est utilisée pour détecter des entités dans un à DynamicFrame l'aide de paramètres spécifiés. Il renvoie une nouvelle colonne DataFrame avec les valeurs d'origine remplacées par des données sensibles masquées et une colonne supplémentaire contenant des `outputColumnName` métadonnées de détection des informations personnelles.

```
detect(frame: DynamicFrame,
       detectionParameters: JsonOptions,
       outputColumnName: String = "DetectedEntities",
       detectionSensitivity: String = "LOW"): DynamicFrame
```

Paramètres :

- frame — (type :`DynamicFrame`) : entrée DynamicFrame contenant les données à traiter.
- detectionParameters — (type :`JsonOptions`) : options JSON spécifiant les paramètres du processus de détection.
- outputColumnName— (type :`String`, par défaut : "DetectedEntities«) : nom de la colonne dans laquelle les entités détectées seront stockées. S'il n'est pas fourni, le nom de colonne par défaut est « DetectedEntities ».

- detectionSensitivity — (type : `String`, options : « FAIBLE » ou « ÉLEVÉ », par défaut : « FAIBLE ») : spécifie la sensibilité du processus de détection. Les options valides sont « FAIBLE » ou « ÉLEVÉ ». Si ce n'est pas le cas, la sensibilité par défaut est réglée sur « FAIBLE ».

Paramètres `detectionParameters`

Si aucun paramètre n'est inclus, les valeurs par défaut seront utilisées.

- action — (type :`String`, options : « DETECT », « REDACT », « PARTIAL_REDACT », "SHA256_HASH ») Spécifie l'action à effectuer sur l'entité. Obligatoire. Notez que les actions qui effectuent un masquage (toutes sauf « DÉTECTER ») ne peuvent effectuer qu'une seule action par colonne. Il s'agit d'une mesure préventive pour masquer les entités fusionnées.
- sourceColumns — (type :`List[String]`, par défaut : [« * »]) liste des noms de colonnes source sur lesquels effectuer la détection pour l'entité. La valeur par défaut est [« * »] si elle n'est pas précisée. Lève une `IllegalArgumentException` si un nom de colonne non valide est utilisé.
- sourceColumnsToExclure — (type :`List[String]`) Liste des noms des colonnes sources sur lesquelles effectuer la détection pour l'entité. Utilisez `sourceColumns` ou `sourceColumnsToExclude`. Lève une `IllegalArgumentException` si un nom de colonne non valide est utilisé.
- actionOptions : options supplémentaires basées sur l'action spécifiée :
  - Pour « DETECT » et « SHA256 _HASH », aucune option n'est autorisée.
  - Pour « RÉDIGER » :
    - redactText — (type :`String`, par défaut : « ***** ») texte destiné à remplacer l'entité détectée.
  - Pour « PARTIAL_REDACT » :
    - redactChar — (type :`String`, par défaut : « * ») caractère pour remplacer chaque caractère détecté dans l'entité.
    - matchPattern — (type :`String`) modèle Regex pour la rédaction partielle. Ne peut pas être combiné avec numLeftChars ToExclude ou`numRightCharsToExclude`.
    - numLeftCharsToExclude— (type :`String, integer`) Nombre de caractères gauches à exclure. Ne peut pas être combiné avec matchPattern, mais peut être utilisé avec `numRightCharsToExclude`.
    - numRightCharsToExclude— (type :`String, integer`) Nombre de bons caractères à exclure. Ne peut pas être combiné avec matchPattern, mais peut être utilisé avec `numRightCharsToExclude`.

Paramètres `outputColumnName`

Voir les outputColumnName paramètres

Paramètres de détection pour **`classifyColumns()`**

Cette méthode est utilisée pour détecter des entités dans un DynamicFrame. Il renvoie une carte où les clés sont les noms de colonnes et les valeurs sont une liste des types d'entités détectés. Un masquage personnalisé peut être effectué une fois celui-ci renvoyé dans le script AWS Glue.

```
classifyColumns(frame: DynamicFrame,
                entityTypesToDetect: Seq[String],
                sampleFraction: Double = 0.1,
                thresholdFraction: Double = 0.1,
                detectionSensitivity: String = "LOW")
```

Paramètres :

- frame — (type :`DynamicFrame`) L'entrée DynamicFrame contenant les données à traiter.
- entityTypesToDétecter — (type :`Seq[String]`) Liste des types d'entités à détecter. Il peut s'agir de types d'entités gérés ou de types d'entités personnalisés.
- sampleFraction — (type :`Double`, par défaut : 10 %) la fraction des données à échantillonner lors de la recherche d'entités PII.
- thresholdFraction — (type :`Double`, par défaut : 10 %) :la fraction des données qui doit être satisfaite pour qu'une colonne soit identifiée comme données PII.
- detectionSensitivity — (type : `String`, options : « FAIBLE » ou « ÉLEVÉ », par défaut : « FAIBLE ») spécifie la sensibilité du processus de détection. Les options valides sont « FAIBLE » ou « ÉLEVÉ ». Si ce n'est pas le cas, la sensibilité par défaut est réglée sur « FAIBLE ».

Types de données sensibles gérés

Entités globales

Type de données	Catégorie	Description
PERSON_NAME	Universel	Le nom de la personne.

Type de données	Catégorie	Description
EMAIL	Personnel	L'adresse e-mail.
IP_ADDRESS	Ordinateur	Adresse IP
MAC_ADDRESS	Personnel	L'adresse MAC.

Types de données pour les États-Unis

Type de données	Description
BANK_ACCOUNT	Numéro du compte bancaire. Non spécifique à un pays ou à une région, cependant, seuls les comptes bancaires américains et canadiens sont détectés.
CREDIT_CARD	Le numéro de carte de crédit.
PHONE_NUMBER	Numéro de téléphone. Non spécifique à un pays ou à une région, cependant, seuls les numéros de téléphone américains et canadiens sont détectés pour le moment.
USA_ATIN	Le numéro d'identification fiscale américain délivré l'Internal Revenue Service.
USA_CPT_CODE	Le code CPT (spécifique aux États-Unis).
USA_DEA_NUMBER	Le numéro DEA (spécifique aux États-Unis).
USA_DRIVING_LICENSE	Le numéro de permis de conduire (spécifique aux États-Unis).
USA_HCPCS_CODE	Le code HCPCS (spécifique aux États-Unis).
USA_HEALTH_INSURANCE_CLAIM_NUMBER	Le numéro de sécurité sociale (spécifique aux États-Unis).

Type de données	Description
USA_ITIN	Le numéro ITIN (pour les ressortissants ou les entités des États-Unis).
USA_MEDICARE_BENEFICIARY_IDENTIFIER	L'identifiant de bénéficiaire Medicare (spécifiq ue aux États-Unis).
USA_NATIONAL_DRUG_CODE	Le code NDC (spécifique aux États-Unis).
USA_NATIONAL_PROVIDER_IDENTIFIER	Le numéro National Provider Identifier (spécifiq ue aux États-Unis).
USA_PASSPORT_NUMBER	Le numéro de passeport (pour les ressortis sants des États-Unis).
USA_PTIN	Le numéro d'identification fiscale du préparate ur américain délivré l'Internal Revenue Service.
USA_SSN	Le numéro de sécurité sociale (pour les ressortissants des États-Unis).

Types de données pour l'Argentine

Type de données	Description
ARGENTINA_TAX_IDENTIFICATIO N_NUMBER	Le numéro d'identification fiscale de l'Argenti ne. Également connu sous le nom de CUIT ou CUIL.

Types de données pour l'Australie

Type de données	Description
AUSTRALIA_BUSINESS_NUMBER	Le numéro d'entreprise australien (ABN). Un identifiant unique délivré par l'Australian Business Register (ABR) pour identifier les

Type de données	Description
	entreprises auprès du gouvernement et de la communauté.
AUSTRALIA_COMPANY_NUMBER	Le numéro de société australien (ACN). Un identifiant unique délivré par l'Australian Securities and Investments Commission.
AUSTRALIA_DRIVING_LICENSE	Un numéro de permis de conduire pour l'Australie.
AUSTRALIA_MEDICARE_NUMBER	Le numéro Medicare australien. Un identifiant personnel délivré par l'Australian Health Insurance Commission.
AUSTRALIA_PASSPORT_NUMBER	Le numéro de passeport australien.
AUSTRALIA_TAX_FILE_NUMBER	Le numéro de dossier fiscal australien (TFN). Délivré par l'Australian Taxation Office (ATO) aux contribuables (particuliers, entreprises, etc.) pour la gestion de leurs impôts.

Types de données pour l'Autriche

Type de données	Description
AUSTRIA_DRIVING_LICENSE	Le numéro de permis de conduire (spécifique à l'Autriche).
AUSTRIA_PASSPORT_NUMBER	Le numéro de passeport (spécifique à l'Autriche).
AUSTRIA_SSN	Le numéro de sécurité sociale (pour les ressortissants de l'Autriche).
AUSTRIA_TAX_IDENTIFICATION_NUMBER	Le numéro d'identification fiscale (spécifique à l'Autriche).

Type de données	Description
AUSTRIA_VALUE_ADDED_TAX	La taxe sur la valeur ajoutée (spécifique à l'Autriche).

Types de données pour les Balkans

Type de données	Description
BOSNIA_UNIQUE_MASTER_CITIZEN_NUMBER	Le numéro d'identification nationale (JMBG) pour les citoyens de Bosnie-Herzégovine.
KOSOVO_UNIQUE_MASTER_CITIZEN_NUMBER	Le numéro d'identification nationale (JMBG) pour le Kosovo.
MACEDONIA_UNIQUE_MASTER_CITIZEN_NUMBER	Le numéro d'identification nationale pour la Macédoine.
MONTENEGRO_UNIQUE_MASTER_CITIZEN_NUMBER	Le numéro d'identification nationale (JMBG) pour le Monténégro.
SERBIA_UNIQUE_MASTER_CITIZEN_NUMBER	Le numéro d'identification nationale (JMBG) pour la Serbie.
SERBIA_VALUE_ADDED_TAX	La taxe sur la valeur ajoutée (spécifique à la Serbie).
VOJVODINA_UNIQUE_MASTER_CITIZEN_NUMBER	Le numéro d'identification nationale (JMBG) pour Voïvodine.

Types de données pour la Belgique

Type de données	Description
BELGIUM_DRIVING_LICENSE	Le numéro de permis de conduire (spécifique à la Belgique).

Type de données	Description
BELGIUM_NATIONAL_IDENTIFICATION_NUMBER	Le numéro de registre national.
BELGIUM_PASSPORT_NUMBER	Le numéro de passeport (spécifique à la Belgique).
BELGIUM_TAX_IDENTIFICATION_NUMBER	Le numéro d'identification fiscale (spécifique à la Belgique).
BELGIUM_VALUE_ADDED_TAX	La taxe sur la valeur ajoutée (spécifique à la Belgique).

Types de données pour le Brésil

Type de données	Description
BRAZIL_BANK_ACCOUNT	Le numéro de compte bancaire (spécifique au Brésil).
BRAZIL_NATIONAL_IDENTIFICATION_NUMBER	L'identifiant national (spécifique au Brésil).
BRAZIL_NATIONAL_REGISTRY_OF_LEGAL_ENTITIES_NUMBER	Le numéro d'identification délivré aux entreprises (spécifique au Brésil), également connu sous le nom de CNPJ.
BRAZIL_NATURAL_PERSON_REGISTRY_NUMBER	Le cadastre de personne physique connu sous le nom de CPF.

Types de données pour la Bulgarie

Type de données	Description
BULGARIA_DRIVING_LICENSE	Le numéro de permis de conduire (spécifique à la Bulgarie).

Type de données	Description
BULGARIA_UNIFORM_CIVIL_NUMBER	Le numéro civil unifié (EGN) qui fait office de numéro d'identification national.
BULGARIA_VALUE_ADDED_TAX	La taxe sur la valeur ajoutée (spécifique à la Bulgarie).

Types de données pour le Canada

Type de données	Description
CANADA_DRIVING_LICENSE	Le numéro de permis de conduire (spécifique au Canada).
CANADA_GOVERNMENT_IDENTIFICATION_CARD_NUMBER	L'identifiant national (spécifique au Canada).
CANADA_PASSPORT_NUMBER	Le numéro de passeport (spécifique au Canada).
CANADA_PERMANENT_RESIDENCE_NUMBER	Certificat d'inscription au registre des étrangers (numéro de carte RP).
CANADA_PERSONAL_HEALTH_NUMBER	L'identifiant unique pour les soins de santé (numéro PHN).
CANADA_SOCIAL_INSURANCE_NUMBER	Le numéro d'assurance sociale (SIN) au Canada.

Types de données pour le Chili

Type de données	Description
CHILE_DRIVING_LICENSE	Le numéro de permis de conduire (spécifique au Chili).

Type de données	Description
CHILE_NATIONAL_IDENTIFICATION_NUMBER	L'identifiant national du Chili, également connu sous le nom de RUT ou RUN.

Types de données pour la Chine, Hong Kong, Macao et Taïwan

Type de données	Description
CHINA_IDENTIFICATION	L'identifiant de la Chine.
CHINA_LICENSE_PLATE_NUMBER	Le numéro de permis de conduire (spécifique à la Chine).
CHINA_MAINLAND_TRAVEL_PERMIT_ID_HONG_KONG_MACAU	Le permis de voyage continental pour les résidents de Hong Kong et de Macao.
CHINA_MAINLAND_TRAVEL_PERMIT_ID_TAIWAN	Le permis de voyage continental pour les résidents de Taïwan délivré par le gouvernement de la Chine.
CHINA_PASSPORT_NUMBER	Le numéro de passeport (spécifique à la Chine).
CHINA_PHONE_NUMBER	Le numéro de téléphone (spécifique à la Chine).
HONG_KONG_IDENTITY_CARD	Le document d'identité officiel délivré par le ministère de l'Immigration de Hong Kong.
MACAU_RESIDENT_IDENTITY_CARD	La carte d'identité de résident de Macao ou BIR est une carte d'identité officielle délivrée par le Bureau des services d'identification de Macao.
TAIWAN_NATIONAL_IDENTIFICATION_NUMBER	L'identifiant national (spécifique à Taïwan).
TAIWAN_PASSPORT_NUMBER	Le numéro de passeport (spécifique à Taïwan).

Types de données pour la Colombie

Type de données	Description
COLOMBIA_PERSONAL_IDENTIFICATION_NUMBER	Un identifiant unique attribué aux Colombiens à la naissance.
COLOMBIA_TAX_IDENTIFICATION_NUMBER	Le numéro d'identification fiscale (spécifique à la Colombie).

Types de données pour la Croatie

Type de données	Description
CROATIA_DRIVING_LICENSE	Le numéro de permis de conduire (spécifique à la Croatie).
CROATIA_IDENTITY_NUMBER	L'identifiant national (spécifique à la Croatie).
CROATIA_PASSPORT_NUMBER	Le numéro de passeport (spécifique à la Croatie).
CROATIA_PERSONAL_IDENTIFICATION_NUMBER	Le numéro d'identification personnel (OIB).

Types de données pour Chypre

Type de données	Description
CYPRUS_DRIVING_LICENSE	Le numéro de permis de conduire (spécifique à Chypre).
CYPRUS_NATIONAL_IDENTIFICATION_NUMBER	La carte d'identité chypriote.
CYPRUS_PASSPORT_NUMBER	Le numéro de passeport (spécifique à Chypre).

Type de données	Description
CYPRUS_TAX_IDENTIFICATION_NUMBER	Le numéro d'identification fiscale (spécifique à Chypre).
CYPRUS_VALUE_ADDED_TAX	La taxe sur la valeur ajoutée (spécifique à Chypre).

Types de données pour la Tchéquie

Type de données	Description
CZECHIA_DRIVING_LICENSE	Le numéro de permis de conduire (spécifique à la Tchéquie).
CZECHIA_PERSONAL_IDENTIFICATION_NUMBER	Le numéro d'identification personnel (spécifique à la Tchéquie).
CZECHIA_VALUE_ADDED_TAX	La taxe sur la valeur ajoutée (spécifique à la Tchéquie).

Types de données pour le Danemark

Type de données	Description
DENMARK_DRIVING_LICENSE	Le numéro de permis de conduire (spécifique Danemark).
DENMARK_PERSONAL_IDENTIFICATION_NUMBER	Le numéro d'identification personnel (spécifique au Danemark).
DENMARK_TAX_IDENTIFICATION_NUMBER	Le numéro d'identification fiscale (spécifique au Danemark).
DENMARK_VALUE_ADDED_TAX	La taxe sur la valeur ajoutée (spécifique au Danemark).

Types de données pour l'Estonie

Type de données	Description
ESTONIA_DRIVING_LICENSE	Le numéro de permis de conduire (spécifique à l'Estonie).
ESTONIA_PASSPORT_NUMBER	Le numéro de passeport (spécifique à l'Estonie).
ESTONIA_PERSONAL_IDENTIFICATION_CODE	Le numéro d'identification personnel (spécifique à l'Estonie).
ESTONIA_VALUE_ADDED_TAX	La taxe sur la valeur ajoutée (spécifique à l'Estonie).

Types de données pour la Finlande

Type de données	Description
FINLAND_DRIVING_LICENSE	Le numéro de permis de conduire (spécifique à la Finlande).
FINLAND_HEALTH_INSURANCE_NUMBER	Le numéro d'assurance maladie (spécifique à la Finlande).
FINLAND_NATIONAL_IDENTIFICATION_NUMBER	Le numéro d'identifiant national (spécifique à la Finlande).
FINLAND_PASSPORT_NUMBER	Le numéro de passeport (spécifique à la Finlande).
FINLAND_VALUE_ADDED_TAX	La taxe sur la valeur ajoutée (spécifique à la Finlande).

Types de données pour la France

Type de données	Description
FRANCE_BANK_ACCOUNT	Le numéro de compte bancaire (spécifique à la France).
FRANCE_DRIVING_LICENSE	Le numéro de permis de conduire (spécifique à la France).
FRANCE_HEALTH_INSURANCE_NUMBER	Le numéro d'assurance maladie de la France.
FRANCE_INSEE_CODE	Le numéro de sécurité sociale en France.
FRANCE_NATIONAL_IDENTIFICATION_NUMBER	Le numéro d'identification national (NNI) de la France.
FRANCE_PASSPORT_NUMBER	Le numéro de passeport (spécifique à la France).
FRANCE_TAX_IDENTIFICATION_NUMBER	Le numéro d'identification fiscale (spécifique à la France).
FRANCE_VALUE_ADDED_TAX	La taxe sur la valeur ajoutée (spécifique à la France).

Types de données pour l'Allemagne

Type de données	Description
GERMANY_BANK_ACCOUNT	Le numéro de compte bancaire (spécifique à l'Allemagne).
GERMANY_DRIVING_LICENSE	Le numéro de permis de conduire (spécifique à l'Allemagne).
GERMANY_PASSPORT_NUMBER	Le numéro de passeport (spécifique à l'Allemagne).

Type de données	Description
GERMANY_PERSONAL_IDENTIFICATION_NUMBER	Le numéro d'identification personnel (spécifique à l'Allemagne).
GERMANY_TAX_IDENTIFICATION_NUMBER	Le numéro d'identification fiscale (spécifique à l'Allemagne).
GERMANY_VALUE_ADDED_TAX	La taxe sur la valeur ajoutée (spécifique à l'Allemagne).

Types de données pour la Grèce

Type de données	Description
GREECE_DRIVING_LICENSE	Le numéro de permis de conduire (spécifique à la Grèce).
GREECE_PASSPORT_NUMBER	Le numéro de passeport (spécifique à la Grèce).
GREECE_SSN	Le numéro de sécurité sociale (pour les ressortissants de la Grèce).
GREECE_TAX_IDENTIFICATION_NUMBER	Le numéro d'identification fiscale (spécifique à la Grèce).
GREECE_VALUE_ADDED_TAX	La taxe sur la valeur ajoutée (spécifique à la Grèce).

Types de données pour la Hongrie

Type de données	Description
HUNGARY_DRIVING_LICENSE	Le numéro de permis de conduire (spécifique à la Hongrie).

Type de données	Description
HUNGARY_PASSPORT_NUMBER	Le numéro de passeport (spécifique à la Hongrie).
HUNGARY_SSN	Le numéro de sécurité sociale (pour les ressortissants de la Hongrie).
HUNGARY_TAX_IDENTIFICATION_NUMBER	Le numéro d'identification fiscale (spécifique à la Hongrie).
HUNGARY_VALUE_ADDED_TAX	La taxe sur la valeur ajoutée (spécifique à la Hongrie).

Types de données pour l'Islande

Type de données	Description
ICELAND_NATIONAL_IDENTIFICATION_NUMBER	L'identifiant national (spécifique à l'Islande).
ICELAND_PASSPORT_NUMBER	Le numéro de passeport (spécifique à l'Islande).
ICELAND_VALUE_ADDED_TAX	La taxe sur la valeur ajoutée (spécifique à l'Islande).

Types de données pour l'Inde

Type de données	Description
INDIA_AADHAAR_NUMBER	Le numéro d'identification Aadhaar délivré par l'Unique Identification Authority of India.
INDIA_PERMANENT_ACCOUNT_NUMBER	Le numéro PAN (Permanent Account Number, numéro de compte permanent).

Types de données pour l'Indonésie

Type de données	Description
INDONESIA_IDENTITY_CARD_NUMBER	L'identifiant national (spécifique à l'Indonésie).

Types de données pour l'Irlande

Type de données	Description
IRELAND_DRIVING_LICENSE	Le numéro de permis de conduire (spécifique à l'Irlande).
IRELAND_PASSPORT_NUMBER	Le numéro de passeport (spécifique à l'Irlande).
IRELAND_PERSONAL_PUBLIC_SERVICE_NUMBER	Le numéro personnel pour le service public (PPS) de l'Irlande.
IRELAND_TAX_IDENTIFICATION_NUMBER	Le numéro d'identification fiscale (spécifique à l'Irlande).
IRELAND_VALUE_ADDED_TAX	La taxe sur la valeur ajoutée (spécifique à l'Irlande).

Types de données pour Israël

Type de données	Description
ISRAEL_IDENTIFICATION_NUMBER	L'identifiant national (spécifique à Israël).

Types de données pour l'Italie

Type de données	Description
ITALY_BANK_ACCOUNT	Le numéro de compte bancaire (spécifique à l'Italie).

Type de données	Description
ITALY_DRIVING_LICENSE	Le numéro de permis de conduire (spécifique à l'Italie).
ITALY_FISCAL_CODE	Le numéro d'identification, également connu sous le nom de Codice Fiscale italien.
ITALY_PASSPORT_NUMBER	Le numéro de passeport (spécifique à l'Italie).
ITALY_VALUE_ADDED_TAX	La taxe sur la valeur ajoutée (spécifique à l'Italie).

Types de données pour le Japon

Type de données	Description
JAPAN_BANK_ACCOUNT	Compte en banque au Japon.
JAPAN_DRIVING_LICENSE	Un numéro de permis de conduire pour le Japon.
JAPAN_MY_NUMBER	Identifiant unique des citoyens ou des entrepris es du Japon utilisé pour l'administration fiscale, la sécurité sociale et les interventions en cas de catastrophe
JAPAN_PASSPORT_NUMBER	Numéro de passeport japonais.

Types de données pour la Corée

Type de données	Description
KOREA_PASSPORT_NUMBER	Le numéro de passeport (spécifique à la Corée).

Type de données	Description
KOREA_RESIDENCE_REGISTRATION_NUMBER_FOR_CITIZENS	Le numéro d'enregistrement de résident en Corée pour les résidents.
KOREA_RESIDENCE_REGISTRATION_NUMBER_FOR_FOREIGNERS	Le numéro d'enregistrement de résident en Corée pour les étrangers.

Types de données pour la Lettonie

Type de données	Description
LATVIA_DRIVING_LICENSE	Le numéro de permis de conduire (spécifique à la Lettonie).
LATVIA_PASSPORT_NUMBER	Le numéro de passeport (spécifique à la Lettonie).
LATVIA_PERSONAL_IDENTIFICATION_NUMBER	Le numéro d'identification personnel (spécifique à la Lettonie).
LATVIA_VALUE_ADDED_TAX	La taxe sur la valeur ajoutée (spécifique à la Lettonie).

Types de données pour le Liechtenstein

Type de données	Description
LIECHTENSTEIN_NATIONAL_IDENTIFICATION_NUMBER	L'identifiant national (spécifique au Liechtenstein).
LIECHTENSTEIN_PASSPORT_NUMBER	Le numéro de passeport (spécifique au Liechtenstein).
LIECHTENSTEIN_TAX_IDENTIFICATION_NUMBER	Le numéro d'identification fiscale (spécifique au Liechtenstein).

Types de données pour la Lituanie

Type de données	Description
LITHUANIA_DRIVING_LICENSE	Le numéro de permis de conduire (spécifique à la Lituanie).
LITHUANIA_PERSONAL_IDENTIFICATION_NUMBER	Le numéro d'identification personnel (spécifique à la Lituanie).
LITHUANIA_TAX_IDENTIFICATION_NUMBER	Le numéro d'identification fiscale (spécifique à la Lituanie).
LITHUANIA_VALUE_ADDED_TAX	La taxe sur la valeur ajoutée (spécifique à la Lituanie).

Types de données pour le Luxembourg

Type de données	Description
LUXEMBOURG_DRIVING_LICENSE	Le numéro de permis de conduire (spécifique au Luxembourg).
LUXEMBOURG_NATIONAL_INDIVIDUAL_NUMBER	L'identifiant national (spécifique au Luxembourg).
LUXEMBOURG_PASSPORT_NUMBER	Le numéro de passeport (spécifique au Luxembourg).
LUXEMBOURG_TAX_IDENTIFICATION_NUMBER	Le numéro d'identification fiscale (spécifique au Luxembourg).
LUXEMBOURG_VALUE_ADDED_TAX	La taxe sur la valeur ajoutée (spécifique au Luxembourg).

Types de données pour la Malaisie

Type de données	Description
MALAYSIA_MYKAD_NUMBER	L'identifiant national (spécifique à la Malaisie).
MALAYSIA_PASSPORT_NUMBER	Le numéro de passeport (spécifique à la Malaisie).

Types de données pour Malte

Type de données	Description
MALTA_DRIVING_LICENSE	Le numéro de permis de conduire (spécifique à Malte).
MALTA_NATIONAL_IDENTIFICATION_NUMBER	L'identifiant national (spécifique à Malte).
MALTA_TAX_IDENTIFICATION_NUMBER	Le numéro d'identification fiscale (spécifique à Malte).
MALTA_VALUE_ADDED_TAX	La taxe sur la valeur ajoutée (spécifique à Malte).

Types de données pour le Mexique

Type de données	Description
MEXICO_CLABE_NUMBER	Numéro CLABE du Mexique (Clave Bancaria Estandarizada), numéro bancaire.
MEXICO_DRIVING_LICENSE	Le numéro de permis de conduire (spécifique au Mexique).
MEXICO_PASSPORT_NUMBER	Le numéro de passeport (spécifique au Mexique).

Type de données	Description
MEXICO_TAX_IDENTIFICATION_NUMBER	Le numéro d'identification fiscale (spécifique au Mexique).
MEXICO_UNIQUE_POPULATION_REGISTRY_CODE	Le Clave Única de Registro de Población (CURP), le numéro d'identification national au Mexique.

Types de données pour les Pays-Bas

Type de données	Description
NETHERLANDS_CITIZEN_SERVICE_NUMBER	Le numéro de citoyen néerlandais (BSN, burgerservicenummer).
NETHERLANDS_DRIVING_LICENSE	Le numéro de permis de conduire (spécifique aux Pays-Bas).
NETHERLANDS_PASSPORT_NUMBER	Le numéro de passeport (spécifique aux Pays-Bas).
NETHERLANDS_TAX_IDENTIFICATION_NUMBER	Le numéro d'identification fiscale (spécifique aux Pays-Bas).
NETHERLANDS_VALUE_ADDED_TAX	La taxe sur la valeur ajoutée (spécifique aux Pays-Bas).
NETHERLANDS_BANK_ACCOUNT	Le numéro de compte bancaire (spécifique aux Pays-Bas).

Types de données pour la Nouvelle-Zélande

Type de données	Description
NEW_ZEALAND_DRIVING_LICENSE	Le numéro de permis de conduire (spécifique à la Nouvelle-Zélande).

Type de données	Description
NEW_ZEALAND_NATIONAL_HEALTH_INDEX_NUMBER	Le numéro de l'indice national de santé (NHI) de la Nouvelle-Zélande.
NEW_ZEALAND_TAX_IDENTIFICATION_NUMBER	Le numéro d'identification fiscale, également connu sous le nom de numéro de revenu intérieur (spécifique à la Nouvelle-Zélande).

Types de données pour la Norvège

Type de données	Description
NORWAY_BIRTH_NUMBER	Le numéro d'identité national norvégien.
NORWAY_DRIVING_LICENSE	Le numéro de permis de conduire (spécifique à la Norvège).
NORWAY_HEALTH_INSURANCE_NUMBER	Le numéro d'assurance maladie de la Norvège.
NORWAY_NATIONAL_IDENTIFICATION_NUMBER	Le numéro d'identifiant national (spécifique à la Norvège).
NORWAY_VALUE_ADDED_TAX	La taxe sur la valeur ajoutée (spécifique à la Norvège).

Types de données pour les Philippines

Type de données	Description
PHILIPPINES_DRIVING_LICENSE	Le numéro de permis de conduire (spécifique aux Philippines).
PHILIPPINES_PASSPORT_NUMBER	Le numéro de passeport (spécifique aux Philippines).

Types de données pour la Pologne

Type de données	Description
POLAND_DRIVING_LICENSE	Le numéro de permis de conduire (spécifique à la Pologne).
POLAND_IDENTIFICATION_NUMBER	Le numéro d'identification polonais.
POLAND_PASSPORT_NUMBER	Le numéro du passeport (spécifique à la Pologne).
POLAND_REGON_NUMBER	Le numéro d'identification REGON.
POLAND_SSN	Le numéro de sécurité sociale (pour les ressortissants de la Pologne).
POLAND_TAX_IDENTIFICATION_NUMBER	Le numéro d'identification fiscale (spécifique à la Pologne).
POLAND_VALUE_ADDED_TAX	La taxe sur la valeur ajoutée (spécifique à la Pologne).

Types de données pour le Portugal

Type de données	Description
PORTUGAL_DRIVING_LICENSE	Le numéro de permis de conduire (spécifique au Portugal).
PORTUGAL_NATIONAL_IDENTIFICATION_NUMBER	Le numéro d'identifiant national (spécifique au Portugal).
PORTUGAL_PASSPORT_NUMBER	Le numéro de passeport (spécifique au Portugal).
PORTUGAL_TAX_IDENTIFICATION_NUMBER	Le numéro d'identification fiscale (spécifique au Portugal).

Type de données	Description
PORTUGAL_VALUE_ADDED_TAX	La taxe sur la valeur ajoutée (spécifique au Portugal).

Types de données pour la Roumanie

Type de données	Description
ROMANIA_DRIVING_LICENSE	Le numéro de permis de conduire (spécifique à la Roumanie).
ROMANIA_NUMERICAL_PERSONAL_CODE	Le numéro d'identification personnel (spécifique à la Roumanie).
ROMANIA_PASSPORT_NUMBER	Le numéro de passeport (spécifique à la Roumanie).
ROMANIA_VALUE_ADDED_TAX	La taxe sur la valeur ajoutée (spécifique à la Roumanie).

Types de données pour Singapour

Type de données	Description
SINGAPORE_DRIVING_LICENSE	Le numéro de permis de conduire (spécifique à Singapour).
SINGAPORE_NATIONAL_REGISTRY _IDENTIFICATION_NUMBER	La carte d'identité nationale d'enregistrement pour Singapour.
SINGAPORE_PASSPORT_NUMBER	Le numéro de passeport (spécifique à Singapour).
SINGAPORE_UNIQUE_ENTITY_NUMBER	Le numéro d'entité unique pour Singapour.

Types de données pour la Slovaquie

Type de données	Description
SLOVAKIA_DRIVING_LICENSE	Le numéro de permis de conduire (spécifique à la Slovaquie).
SLOVAKIA_NATIONAL_IDENTIFICATION_NUMBER	Le numéro d'identifiant national (spécifique à la Slovaquie).
SLOVAKIA_PASSPORT_NUMBER	Le numéro de passeport (spécifique à la Slovaquie).
SLOVAKIA_VALUE_ADDED_TAX	La taxe sur la valeur ajoutée (spécifique à la Slovaquie).

Types de données pour la Slovénie

Type de données	Description
SLOVENIA_DRIVING_LICENSE	Le numéro de permis de conduire (spécifique à la Slovénie).
SLOVENIA_PASSPORT_NUMBER	Le numéro de passeport (spécifique à la Slovénie).
SLOVENIA_TAX_IDENTIFICATION_NUMBER	Le numéro d'identification fiscale (spécifique à la Slovénie).
SLOVENIA_UNIQUE_MASTER_CITIZEN_NUMBER	Le numéro d'identification nationale (JMBG) pour les citoyens slovènes.
SLOVENIA_VALUE_ADDED_TAX	La taxe sur la valeur ajoutée (spécifique à la Slovénie).

Types de données pour l'Afrique du Sud

Type de données	Description
SOUTH_AFRICA_PERSONAL_IDENTIFICATION_NUMBER	Le numéro d'identification personnel (spécifique à l'Afrique du Sud).

Types de données pour l'Espagne

Type de données	Description
SPAIN_BANK_ACCOUNT	Le numéro de compte bancaire (spécifique à l'Espagne).
SPAIN_DNI	La carte d'identité nationale (Documento Nacional de Identidad) de l'Espagne.
SPAIN_DRIVING_LICENSE	Le numéro de permis de conduire (spécifique à l'Espagne).
SPAIN_NIE	Le numéro d'identité d'étranger (spécifique à l'Espagne), également connu sous le nom de NIE.
SPAIN_NIF	Le numéro d'identification fiscale (spécifique à l'Espagne), également connu sous le nom de NIF.
SPAIN_PASSPORT_NUMBER	Le numéro de passeport (spécifique à l'Espagne).
SPAIN_SSN	Le numéro de sécurité sociale (pour les ressortissants de l'Espagne).
SPAIN_VALUE_ADDED_TAX	La taxe sur la valeur ajoutée (spécifique à l'Espagne).

Types de données pour le Sri Lanka

Type de données	Description
SRI_LANKA_NATIONAL_IDENTIFICATION_NUMBER	Le numéro d'identifiant national (spécifique au Sri Lanka).

Types de données pour la Suède

Type de données	Description
SWEDEN_DRIVING_LICENSE	Le numéro de permis de conduire (spécifique à la Suède).
SWEDEN_PASSPORT_NUMBER	Le numéro de passeport (spécifique à la Suède).
SWEDEN_PERSONAL_IDENTIFICATION_NUMBER	Le numéro d'identifiant national (spécifique à la Suède).
SWEDEN_TAX_IDENTIFICATION_NUMBER	Le numéro d'identification fiscale suédois (personnummer).
SWEDEN_VALUE_ADDED_TAX	La taxe sur la valeur ajoutée (spécifique à la Suède).

Types de données pour la Suisse

Type de données	Description
SWITZERLAND_AHV	Le numéro de sécurité sociale pour les ressortissants suisses (AHV).
SWITZERLAND_HEALTH_INSURANCE_NUMBER	Le numéro d'assurance maladie de la Suisse.
SWITZERLAND_PASSPORT_NUMBER	Le numéro de passeport (spécifique à la Suisse).

Type de données	Description
SWITZERLAND_VALUE_ADDED_TAX	La taxe sur la valeur ajoutée (spécifique à la Suisse).

Types de données pour la Thaïlande

Type de données	Description
THAILAND_PASSPORT_NUMBER	Le numéro de passeport (spécifique à la Thaïlande).
THAILAND_PERSONAL_IDENTIFICATION_NUMBER	Le numéro d'identification personnel (spécifique à la Thaïlande).

Types de données pour la Turquie

Type de données	Description
TURKEY_NATIONAL_IDENTIFICATION_NUMBER	Le numéro d'identifiant national (spécifique à la Turquie).
TURKEY_PASSPORT_NUMBER	Le numéro de passeport (spécifique à la Turquie).
TURKEY_VALUE_ADDED_TAX	La taxe sur la valeur ajoutée (spécifique à la Turquie).

Types de données pour l'Ukraine

Type de données	Description
UKRAINE_INDIVIDUAL_IDENTIFICATION_NUMBER	L'identifiant unique (spécifique à l'Ukraine).

Type de données	Description
UKRAINE_PASSPORT_NUMBER_DOMESTIC	Le numéro de passeport national (spécifique à l'Ukraine).
UKRAINE_PASSPORT_NUMBER_INTERNATIONAL	Le numéro de passeport international (spécifique à l'Ukraine).

Types de données pour les Émirats arabes unis (EAU)

Type de données	Description
UNITED_ARAB_EMIRATES_PERSONAL_NUMBER	Le numéro d'identification personnel (spécifique aux EAU).

Types de données pour le Royaume-Uni

Type de données	Description
UK_BANK_ACCOUNT	Le compte bancaire au Royaume-Uni (UK).
UK_BANK_SORT_CODE	Le Sort code du Royaume-Uni (UK). Les Sort Codes sont des codes bancaires utilisés pour acheminer les transferts d'argent entre les banques de leurs pays respectifs via leurs organismes de compensation respectifs.
UK_DRIVING_LICENSE	Numéro de permis de conduire pour le Royaume-Uni de Grande-Bretagne et d'Irlande du Nord (spécifique au Royaume-Uni)
UK_ELECTORAL_ROLL_NUMBER	L'Electoral Roll Number (ERN) est le numéro d'identification délivré aux citoyens pour s'inscrire aux élections du Royaume-Uni. Le format de ce numéro est spécifié par les

Type de données	Description
	normes du Bureau du Cabinet du gouvernement du Royaume-Uni.
UK_NATIONAL_HEALTH_SERVICE_NUMBER	Le National Health Service (NHS) est le numéro unique attribué aux utilisateurs enregistrés auprès des services de santé publique du Royaume-Uni.
UK_NATIONAL_INSURANCE_NUMBER	Le National Insurance Number (NINO) est un numéro utilisé au Royaume-Uni pour identifier une personne inscrite au régime national d'assurance ou au système de sécurité sociale. Ce numéro est parfois appelé NI No ou NINO.
UK_PASSPORT_NUMBER	Le numéro de passeport du Royaume-Uni (UK).
UK_UNIQUE_TAXPAYER_REFERENCE_NUMBER	Numéro de référence fiscale (UTR) du Royaume-Uni. Identifiant utilisé par le gouvernement du Royaume-Uni pour gérer le système fiscal.
UK_VALUE_ADDED_TAX	La TVA est une taxe à la consommation qui est prise en charge par le consommateur final. La TVA est payée pour chaque transaction du processus de fabrication et de distribution. Pour le Royaume-Uni, le numéro de TVA est délivré par le bureau de TVA de la région dans laquelle l'entreprise est établie.
UK_PHONE_NUMBER	Le numéro de téléphone du Royaume-Uni (UK).

Types de données pour le Venezuela

Type de données	Description
VENEZUELA_DRIVING_LICENSE	Le numéro de permis de conduire (spécifique au Venezuela).
VENEZUELA_NATIONAL_IDENTIFICATION_NUMBER	Le numéro d'identifiant national (spécifique au Venezuela).
VENEZUELA_VALUE_ADDED_TAX	La taxe sur la valeur ajoutée (spécifique au Venezuela).

Utilisation d'une détection détaillée des données sensibles

> **Note**
>
> Les actions détaillées ne sont disponibles que dans les versions AWS Glue 3.0 et 4.0. Cela inclut l'expérience AWS Glue Studio. Les modifications persistantes du journal d'audit ne sont pas non plus disponibles dans la version 2.0.
> Tous les travaux visuels des versions AWS Glue Studio 3.0 et 4.0 seront accompagnés d'un script créé qui utilise automatiquement des actions précises. APIs

La transformation Detect Sensitive Data permet de détecter, de masquer ou de supprimer des entités que vous définissez ou qui sont prédéfinies par AWS Glue. Les actions détaillées vous permettent en outre d'appliquer une action spécifique par entité. Les avantages supplémentaires incluent :

- Performances améliorées, car les actions sont appliquées dès que des données sont détectées.
- La possibilité d'inclure ou exclure des colonnes spécifiques.
- La possibilité d'utiliser un masquage partiel. Cela vous permet de masquer partiellement les entités de données sensibles détectées, plutôt que de masquer la chaîne entière. Les deux paramètres simples avec décalages et regex sont pris en charge.

Vous trouverez ci-dessous des extraits de code relatifs à la détection de données sensibles APIs et des actions détaillées utilisées dans les exemples de tâches mentionnés dans la section suivante.

API de détection : les actions détaillées utilisent le nouveau paramètre `detectionParameters` :

```
def detect(
    frame: DynamicFrame,
    detectionParameters: JsonOptions,
    outputColumnName: String = "DetectedEntities",
    detectionSensitivity: String = "LOW"
): DynamicFrame = {}
```

Utilisation de la détection des données sensibles APIs avec des actions précises

La détection des données sensibles APIs à l'aide de la fonction de détection analyse les données fournies, détermine si les lignes ou les colonnes sont des types d'entités de données sensibles et exécute les actions spécifiées par l'utilisateur pour chaque type d'entité.

Utilisation de l'API de détection avec des actions détaillées

Utilisez l'API de détection et spécifiez le `outputColumnName` et `detectionParameters`.

```
   object GlueApp {
     def main(sysArgs: Array[String]) {

       val spark: SparkContext = new SparkContext()
       val glueContext: GlueContext = new GlueContext(spark)

       // @params: [JOB_NAME]
       val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
       Job.init(args("JOB_NAME"), glueContext, args.asJava)

       // Script generated for node S3 bucket. Creates DataFrame from data stored in
S3.
       val S3bucket_node1 =
glueContext.getSourceWithFormat(formatOptions=JsonOptions("""{"quoteChar":
"\"", "withHeader": true, "separator": ",", "optimizePerformance": false}"""),
connectionType="s3", format="csv", options=JsonOptions("""{"paths":
["s3://189657479688-ddevansh-pii-test-bucket/tiny_pii.csv"], "recurse": true}"""),
transformationContext="S3bucket_node1").getDynamicFrame()

       // Script generated for node Detect Sensitive Data. Will run detect API for the
DataFrame
       // detectionParameter contains information on which EntityType are being
detected
       // and what actions are being applied to them when detected.
```

```
        val DetectSensitiveData_node2 = EntityDetector.detect(
            frame = S3bucket_node1,
            detectionParameters = JsonOptions(
             """
                {
                    "PHONE_NUMBER": [
                        {
                            "action": "PARTIAL_REDACT",
                            "actionOptions": {
                                "numLeftCharsToExclude": "3",
                                "numRightCharsToExclude": "4",
                                "redactChar": "#"
                            },
                            "sourceColumnsToExclude": [ "Passport No", "DL NO#" ]
                        }
                    ],
                    "USA_PASSPORT_NUMBER": [
                        {
                            "action": "SHA256_HASH",
                            "sourceColumns": [ "Passport No" ]
                        }
                    ],
                    "USA_DRIVING_LICENSE": [
                        {
                            "action": "REDACT",
                            "actionOptions": {
                                "redactText": "USA_DL"
                            },
                            "sourceColumns": [ "DL NO#" ]
                        }
                    ]

                }
            """
            ),
            outputColumnName = "DetectedEntities"
        )

        // Script generated for node S3 bucket. Store Results of detect to S3 location
        val S3bucket_node3 = glueContext.getSinkWithFormat(connectionType="s3",
options=JsonOptions("""{"path": "s3://amzn-s3-demo-bucket/test-output/",
"partitionKeys": []}"""), transformationContext="S3bucket_node3",
format="json").writeDynamicFrame(DetectSensitiveData_node2)
```

```
    Job.commit()
  }
```

Le script ci-dessus créera un DataFrame depuis un emplacement dans Amazon S3, puis exécutera l'`detect`API. Comme l'`detect`API exige que le champ `detectionParameters` (une carte du nom de l'entité avec une liste de tous les paramètres d'action à utiliser pour cette entité) soit représenté par l'`JsonOptions`objet de AWS Glue, cela nous permettra également d'étendre les fonctionnalités de l'API.

Pour chaque action spécifiée par entité, entrez une liste de tous les noms de colonnes auxquels appliquer la entity/action combinaison. Cela vous permet de personnaliser les entités à détecter pour chaque colonne de votre jeu de données et d'ignorer les entités dont vous savez qu'elles ne figurent pas dans une colonne spécifique. Cela permet également à vos tâches d'être plus performantes en n'effectuant pas d'appels de détection inutiles à ces entités et en vous permettant d'effectuer des actions uniques pour chaque combinaison de colonnes et d'entités.

En examinant de plus près les `detectionParameters`, il existe trois types d'entités dans l'exemple de tâche. Il s'agit des `Phone Number`, `USA_PASSPORT_NUMBER`, et `USA_DRIVING_LICENSE`. Pour chacun de ces types d'entités, AWS Glue exécutera différentes actions, à `PARTIAL_REDACT` savoir`SHA256_HASH`,`REDACT`, et`DETECT`. Chacun des types d'entités doit également s'`sourceColumns`appliquer and/or `sourceColumnsToExclude` s'il est détecté.

> **Note**
>
> Une seule edit-in-place action (`PARTIAL_REDACT`SHA256_HASH, ou`REDACT`) peut être utilisée par colonne, mais l'`DETECT`action peut être utilisée avec n'importe laquelle de ces actions.

La disposition du champ `detectionParameters` est la suivante :

```
ENTITY_NAME -> List[Actions]
{
 "ENTITY_NAME": [{
  Action, // required
  ColumnSpecs,
```

```
      ActionOptionsMap
        }],
        "ENTITY_NAME2": [{
      ...
        }]
    }
```

Les types d'`actions` et `actionOptions` sont répertoriés ci-dessous :

```
DETECT
{
    # Required
    "action": "DETECT",
    # Optional, depending on action chosen
    "actionOptions": {
        // There are no actionOptions for DETECT
    },
    # 1 of below required, both can also used
    "sourceColumns": [
        "COL_1", "COL_2", ..., "COL_N"
    ],
    "sourceColumnsToExclude": [
        "COL_5"
    ]
}

SHA256_HASH
{
    # Required
    "action": "SHA256_HASH",
    # Required or optional, depending on action chosen
    "actionOptions": {
        // There are no actionOptions for SHA256_HASH
    },

    # 1 of below required, both can also used
    "sourceColumns": [
        "COL_1", "COL_2", ..., "COL_N"
    ],
    "sourceColumnsToExclude": [
        "COL_5"
    ]
```

```
}

REDACT
{
    # Required
    "action": "REDACT",
    # Required or optional, depending on action chosen
    "actionOptions": {
        // The text that is being replaced
        "redactText": "USA_DL"
    },

    # 1 of below required, both can also used
    "sourceColumns": [
        "COL_1", "COL_2", ..., "COL_N"
    ],
    "sourceColumnsToExclude": [
        "COL_5"
    ]
}

PARTIAL_REDACT
{
    # Required
    "action": "PARTIAL_REDACT",
    # Required or optional, depending on action chosen
    "actionOptions": {
        // number of characters to not redact from the left side
        "numLeftCharsToExclude": "3",
        // number of characters to not redact from the right side
        "numRightCharsToExclude": "4",
        // the partial redact will be made with this redacted character
        "redactChar": "#",
        // regex pattern for partial redaction
        "matchPattern": "[0-9]"
    },

    # 1 of below required, both can also used
    "sourceColumns": [
        "COL_1", "COL_2", ..., "COL_N"
    ],
    "sourceColumnsToExclude": [
        "COL_5"
    ]
```

```
}
```

Une fois le script exécuté, les résultats sont envoyés à l'emplacement Amazon S3 indiqué. Vous pouvez afficher vos données dans Amazon S3, mais les types d'entités sélectionnés sont sensibilisés en fonction de l'action sélectionnée. Dans ce cas, nous aurions une ligne qui ressemblerait à ceci :

```
{
    "Name": "Colby Schuster",
    "Address": "39041 Antonietta Vista, South Rodgerside, Nebraska 24151",
    "Car Owned": "Fiat",
    "Email": "Kitty46@gmail.com",
    "Company": "O'Reilly Group",
    "Job Title": "Dynamic Functionality Facilitator",
    "ITIN": "991-22-2906",
    "Username": "Cassandre.Kub43",
    "SSN": "914-22-2906",
    "DOB": "2020-08-27",
    "Phone Number": "1-2#######1718",
    "Bank Account No": "69741187",
    "Credit Card Number": "6441-6289-6867-2162-2711",
    "Passport No": "94f311e93a623c72ccb6fc46cf5f5b0265ccb42c517498a0f27fd4c43b47111e",
    "DL NO#": "USA_DL"
}
```

Dans le script ci-dessus, le `Phone Number` a été partiellement expurgé avec #. `Passport No`Il a été transformé en SHA256 hachage. Le `DL NO#` a été détecté comme un numéro de permis de conduire américain et a été expurgé en « USA_DL », comme indiqué dans les `detectionParameters`.

> **Note**
>
> L'API classifyColumns n'est pas disponible pour une utilisation avec des actions détaillées en raison de la nature de l'API. Cette API effectue un échantillonnage de colonnes (ajustable par l'utilisateur mais possède des valeurs par défaut) pour effectuer la détection plus rapidement. Pour cette raison, les actions détaillées nécessitent une itération sur chaque valeur.

Journal d'audit permanent

Une nouvelle fonctionnalité introduite avec des actions précises (mais également disponible en cas d'utilisation normale APIs) est la présence d'un journal d'audit permanent. Actuellement, l'exécution de l'API de détection ajoute un paramètre de colonne supplémentaire (par défaut `DetectedEntities` mais personnalisable via le `outputColumnName`) avec les métadonnées de détection des informations personnelles identifiables. Cela possède désormais une clé de métadonnées « actionUsed », qui est l'une des valeurs `DETECT`, `PARTIAL_REDACT`, `SHA256_HASH`, `REDACT`.

```
"DetectedEntities": {
    "Credit Card Number": [
        {
            "entityType": "CREDIT_CARD",
            "actionUsed": "DETECT",
            "start": 0,
            "end": 19
        }
    ],
    "Phone Number": [
        {
            "entityType": "PHONE_NUMBER",
            "actionUsed": "REDACT",
            "start": 0,
            "end": 14
        }
    ]
}
```

Même les clients qui n'ont APIs pas recours à des actions précises, telles que celles-ci, `detect(entityTypesToDetect, outputColumnName)` verront ce journal d'audit permanent s'afficher dans la trame de données qui en résulte.

Les clients utilisant APIs des actions détaillées verront toutes les actions, qu'elles soient expurgées ou non. Exemple :

```
+--------------------+---------------
+-------------------------------------------------------------------------------------------------
+
```

```
| Credit Card Number  |  Phone Number  |
                                              DetectedEntities
                                                                  |
+--------------------+---------------
+-------------------------------------------------------------------------------------------------------
+
| 622126741306XXXX    | +12#####7890   | {"Credit Card Number":
[{"entityType":"CREDIT_CARD","actionUsed":"PARTIAL_REDACT","start":0,"end":16}],"Phone
 Number":
[{"entityType":"PHONE_NUMBER","actionUsed":"PARTIAL_REDACT","start":0,"end":12}]}} |
| 6221 2674 1306 XXXX | +12#######7890 | {"Credit Card Number":
[{"entityType":"CREDIT_CARD","actionUsed":"PARTIAL_REDACT","start":0,"end":19}],"Phone
 Number":
[{"entityType":"PHONE_NUMBER","actionUsed":"PARTIAL_REDACT","start":0,"end":14}]}} |
| 6221-2674-1306-XXXX | 22#######7890  | {"Credit Card Number":
[{"entityType":"CREDIT_CARD","actionUsed":"PARTIAL_REDACT","start":0,"end":19}],"Phone
 Number":
[{"entityType":"PHONE_NUMBER","actionUsed":"PARTIAL_REDACT","start":0,"end":14}]}} |
+--------------------+---------------
+-------------------------------------------------------------------------------------------------------
+
```

Si vous ne souhaitez pas voir la DetectedEntitiescolonne, vous pouvez simplement supprimer la colonne supplémentaire dans un script personnalisé.

API Visual Job AWS Glue

AWS Glue fournit une API qui vous autorise à créer des tâches d'intégration de données à l'aide de l'API AWS Glue à partir d'un objet JSON qui représente un flux de travail visuel par étapes. Les clients peuvent ensuite utiliser l'éditeur visuel dans AWS Glue Studio pour travailler avec ces tâches.

Pour plus d'informations sur le types de données de l'API Visual Job, voir API Visual Job.

Rubriques

- Conception d'API et CRUD APIs
- Prise en main
- Limitations de tâche visuelle

Conception d'API et CRUD APIs

Les CreateJob et prennent UpdateJob APIs désormais en charge un paramètre optionnel supplémentaire, codeGenConfiguration les nœuds. Si vous fournissez une structure JSON non vide pour ce champ, le DAG sera enregistré dans AWS Glue Studio pour la tâche créée et le code associé généré. Une valeur null ou une chaîne vide pour ce champ lors de la création de la tâche sera ignorée.

Les mises à jour du champ codeGenConfiguration Nodes seront effectuées via l' UpdateJob AWS GlueAPI de la même manière que CreateJob. Le champ entier doit être spécifié UpdateJob là où le DAG a été modifié comme vous le souhaitez. Une valeur null fournie sera ignorée et aucune mise à jour du DAG ne sera effectuée. Une structure ou une chaîne vide entraînera la définition codeGenConfiguration des nœuds comme vides et la suppression de tout DAG précédent. L' GetJob API renverra un DAG s'il en existe un. L' DeleteJob API supprimera également tout DAG associé.

Prise en main

Pour créer une tâche, utilisez l' CreateJob action. L'entrée de `CreateJob` demande comportera un champ supplémentaire « codeGenConfiguration Nodes » dans lequel vous pourrez spécifier l'objet DAG en JSON.

Points à garder à l'esprit :

- Le champ « codeGenConfiguration Nodes » est une carte entre NodeID et nœud.
- Chaque nœud commence par une clé identifiant de quel type de nœud il s'agit.
- Il ne peut y avoir qu'une seule clé spécifiée car un nœud ne peut être que d'un seul type.
- Le champ de saisie contient les nœuds parents du nœud actuel.

Ce qui suit est une représentation JSON d'une CreateJobentrée.

```
{
  "node-1": {
    "S3CatalogSource": {
      "Table": "csvFormattedTable",
      "PartitionPredicate": "",
      "Name": "S3 bucket",
      "AdditionalOptions": {},
      "Database": "myDatabase"
    }
  },
```

```
  "node-3": {
    "S3DirectTarget": {
      "Inputs": ["node-2"],
      "PartitionKeys": [],
      "Compression": "none",
      "Format": "json",
      "SchemaChangePolicy": { "EnableUpdateCatalog": false },
      "Path": "",
      "Name": "S3 bucket"
    }
  },
  "node-2": {
    "ApplyMapping": {
      "Inputs": ["node-1"],
      "Name": "ApplyMapping",
      "Mapping": [
        {
          "FromType": "long",
          "ToType": "long",
          "Dropped": false,
          "ToKey": "myheader1",
          "FromPath": ["myheader1"]
        },
        {
          "FromType": "long",
          "ToType": "long",
          "Dropped": false,
          "ToKey": "myheader2",
          "FromPath": ["myheader2"]
        },
        {
          "FromType": "long",
          "ToType": "long",
          "Dropped": false,
          "ToKey": "myheader3",
          "FromPath": ["myheader3"]
        }
      ]
    }
  }
}
```

Mise à jour et obtention de tâches

Comme il UpdateJoby aura également un champ « codeGenConfiguration Nœuds », le format de saisie sera le même. Voir UpdateJobAction.

L'GetJobaction renverra également un champ « codeGenConfiguration Nœuds » dans le même format. Voir GetJobAction.

Limitations de tâche visuelle

Étant donné que le paramètre codeGenConfiguration « Nodes » a été ajouté au paramètre existant APIs, toutes les limites de celui-ci APIs seront héritées. De plus, la taille des codeGenConfiguration nœuds et de certains nœuds sera limitée. Pour plus d'informations, voir Structure de tâche .

Programmation de scripts Ray

> **Important**
>
> AWS Glue for Ray ne sera plus ouvert aux nouveaux clients à compter du 30 avril 2026. Si vous souhaitez utiliser AWS Glue for Ray, inscrivez-vous avant cette date. Les clients existants peuvent continuer à utiliser le service normalement. Pour des fonctionnalités similaires à celles de AWS Glue for Ray, explorez Amazon EKS. Pour plus d'informations, consultez la section AWS Glue relative à la fin du support de Ray.

AWS Glue facilite l'écriture et l'exécution de scripts Ray. Cette section décrit les fonctionnalités Ray prises en charge et disponibles dans AWS Glue for Ray. Vous programmez des scripts Ray dans Python.

Votre script personnalisé doit être compatible avec la version de Ray définie par le champ `Runtime` de votre définition de tâche. Pour plus d'informations relatives aux `Runtime` dans l'API Tâche, consultez the section called “Tâches”. Pour en savoir plus chaque environnement d'exécution, consultez the section called “Environnements d'exécution Ray pris en charge”.

Rubriques

- Tutoriel : Écrire un script ETL AWS Glue pour Ray
- Utiliser Ray Core et Ray Data AWS Glue pour Ray

- Fournir des fichiers et des bibliothèques Python aux tâches Ray
- Connexion aux données dans les tâches Ray

Tutoriel : Écrire un script ETL AWS Glue pour Ray

> **Important**
>
> AWS Glue for Ray ne sera plus ouvert aux nouveaux clients à compter du 30 avril 2026. Si vous souhaitez utiliser AWS Glue for Ray, inscrivez-vous avant cette date. Les clients existants peuvent continuer à utiliser le service normalement. Pour des fonctionnalités similaires à celles de AWS Glue for Ray, explorez Amazon EKS. Pour plus d'informations, consultez la section AWS Glue relative à la fin du support de Ray.

Ray vous permet d'écrire et de dimensionner des tâches distribuées de manière native en Python. AWS Glue for Ray propose des environnements Ray sans serveur auxquels vous pouvez accéder à la fois à partir de tâches et de sessions interactives (les sessions interactives Ray sont en version préliminaire). Le système de AWS Glue tâches fournit un moyen cohérent de gérer et d'exécuter vos tâches, selon un calendrier, à partir d'un déclencheur ou depuis la console. AWS Glue

La combinaison de ces AWS Glue outils crée une puissante chaîne d'outils que vous pouvez utiliser pour les charges de travail d'extraction, de transformation et de chargement (ETL), un cas d'utilisation populaire. AWS Glue Dans ce didacticiel, vous apprendrez les bases de la mise en place de cette solution.

Nous prenons également en charge l'utilisation AWS Glue de Spark pour vos charges de travail ETL. Pour un didacticiel sur l'écriture d'un script AWS Glue pour Spark, consultezthe section called “Didacticiel : rédiger un script Spark”. Pour plus d'informations sur les moteurs disponibles, consultez the section called “AWS Glue pour Spark et AWS Glue pour Ray”. Ray est capable de traiter de nombreux types de tâches dans les domaines de l'analyse, du machine learning (ML) et du développement d'applications.

Dans ce didacticiel, vous allez extraire, transformer et charger un jeu de données CSV hébergé dans Amazon Simple Storage Service (Amazon S3). Vous allez commencer par le jeu de données d'enregistrement de voyages de l'agence New York City Taxi and Limousine Commission (TLC), qui est stocké dans un compartiment Amazon S3 public. Pour plus d'informations sur ce jeu de données, consultez le Registre des données ouvertes sur AWS.

Vous transformerez vos données à l'aide de transformations prédéfinies disponibles dans la bibliothèque Ray Data. Ray Data est une bibliothèque de préparation de jeux de données conçue par Ray et incluse par défaut dans AWS Glue les environnements for Ray. Pour plus d'informations sur les bibliothèques incluses par défaut, consultez the section called “Modules fournis avec les tâches Ray”. Vous écrirez ensuite vos données transformées dans un compartiment Amazon S3 que vous contrôlez.

Conditions préalables — Pour ce didacticiel, vous avez besoin d'un AWS compte avec accès à Amazon S3 AWS Glue et à Amazon S3.

Étape 1 : créer un compartiment dans Amazon S3 pour stocker vos données de sortie

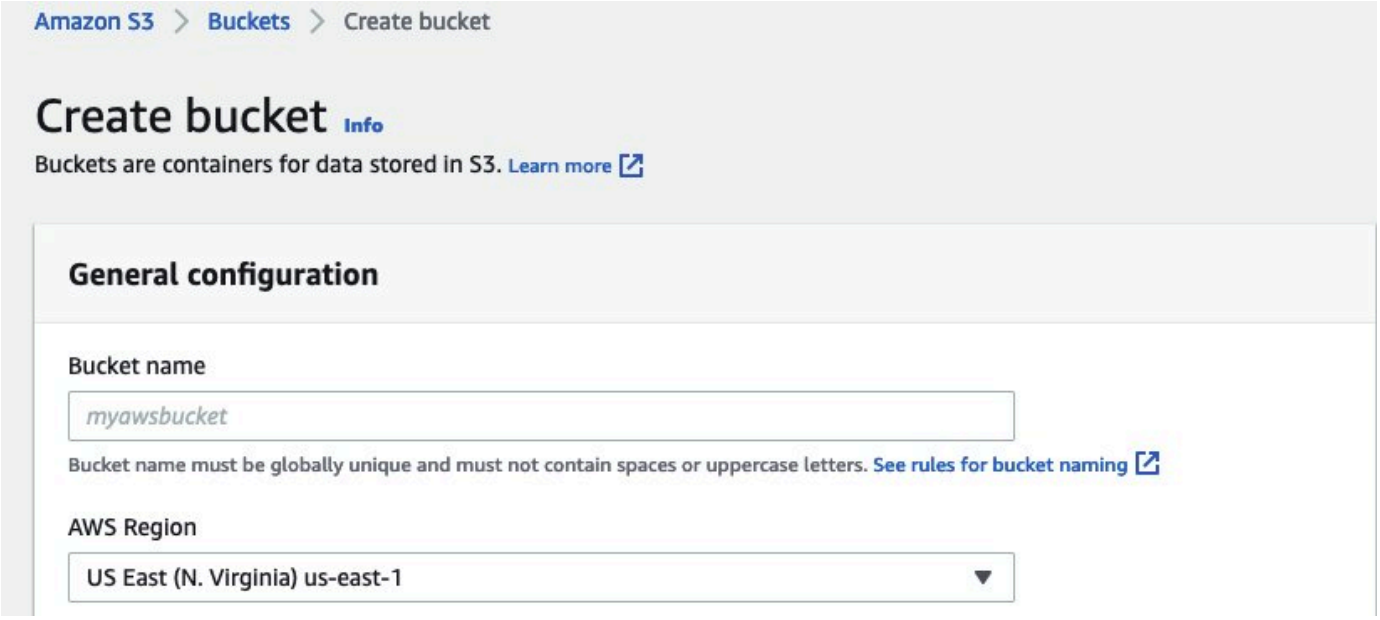
Vous aurez besoin d'un compartiment Amazon S3 que vous contrôlez pour servir de récepteur pour les données créées dans ce didacticiel. Vous pouvez créer ce compartiment en procédant comme suit.

> **Note**
>
> Si vous souhaitez écrire vos données dans un compartiment existant que vous contrôlez, vous pouvez ignorer cette étape. Prenez note du *yourBucketName* nom du compartiment existant, à utiliser ultérieurement.

Pour créer un compartiment pour la sortie de votre tâche Ray

- Créez un compartiment en suivant les étapes décrites dans la section Créer un compartiment du Guide de l'utilisateur Amazon S3.
  - Lorsque vous choisissez un nom de bucket, prenez-en note*yourBucketName*, auquel vous vous référerez dans les étapes suivantes.
  - Pour les autres configurations, les paramètres suggérés fournis dans la console Amazon S3 devraient fonctionner correctement dans ce didacticiel.

  Par exemple, la boîte de dialogue de création de compartiments peut ressembler à ceci dans la console Amazon S3.



Étape 2 : créer un rôle IAM pour votre politique pour votre tâche Ray

Votre travail nécessitera un rôle Gestion des identités et des accès AWS (IAM) présentant les caractéristiques suivantes :

- Autorisations accordées par la politique gérée `AWSGlueServiceRole`. Il s'agit des autorisations de base nécessaires pour exécuter une AWS Glue tâche.
- Autorisations du niveau d'accès `Read` pour la ressource Amazon S3 `nyc-tlc/*`.
- Autorisations du niveau d'accès `Write` pour la ressource Amazon S3 *`yourBucketName`*`/*`.
- Une relation de confiance qui permet au principal `glue.amazonaws.com` d'assumer le rôle.

Vous pouvez créer ce rôle en procédant comme suit.

Pour créer un rôle IAM pour votre job AWS Glue for Ray

> **Note**
>
> Vous pouvez créer un rôle IAM en suivant de nombreuses procédures différentes. Pour plus d'informations ou d'options sur le provisionnement des ressources IAM, consultez la documentation Gestion des identités et des accès AWS.

1. Créez une politique qui définit les autorisations Amazon S3 décrites précédemment en suivant les étapes décrites dans la section Création de politiques avec l'éditeur visuel du Guide de l'utilisateur IAM.

   - Lorsque vous sélectionnez un service, choisissez Amazon S3.
   - Lorsque vous sélectionnez des autorisations pour votre politique, associez les ensembles d'actions suivants pour les ressources suivantes (mentionnées précédemment) :
     - Autorisations du niveau d'accès de lecture pour la ressource Amazon S3 `nyc-tlc/*`.
     - Autorisations du niveau d'accès d'écriture pour la ressource Amazon S3 *`yourBucketName`*`/*`.
   - Lorsque vous sélectionnez le nom de la politique*`YourPolicyName`*, prenez-en note, auquel vous vous référerez ultérieurement.
2. Créez un rôle pour votre tâche AWS Glue for Ray en suivant les étapes décrites dans la section Création d'un rôle pour un AWS service (console) dans le guide de l'utilisateur IAM.

   - Lorsque vous sélectionnez une entité AWS de service fiable, choisissez`Glue`. Cela renseignera automatiquement la relation de confiance nécessaire pour votre tâche.
   - Lorsque vous sélectionnez des politiques pour la politique d'autorisations, joignez les politiques suivantes :
     - `AWSGlueServiceRole`
     - *`YourPolicyName`*
   - Lorsque vous sélectionnez le nom du rôle, prenez-en note*`YourRoleName`*, auquel vous vous référerez dans les étapes ultérieures.

Étape 3 : créer et exécuter une tâche AWS Glue pour Ray

Au cours de cette étape, vous créez une AWS Glue tâche à l'aide du AWS Management Console, vous lui fournissez un exemple de script et vous exécutez la tâche. Lorsque vous créez une tâche, elle crée un emplacement dans la console où vous pouvez stocker, configurer et modifier votre script Ray. Pour plus d'informations sur la création de tâches, consultez the section called “Se connecter à la console”.

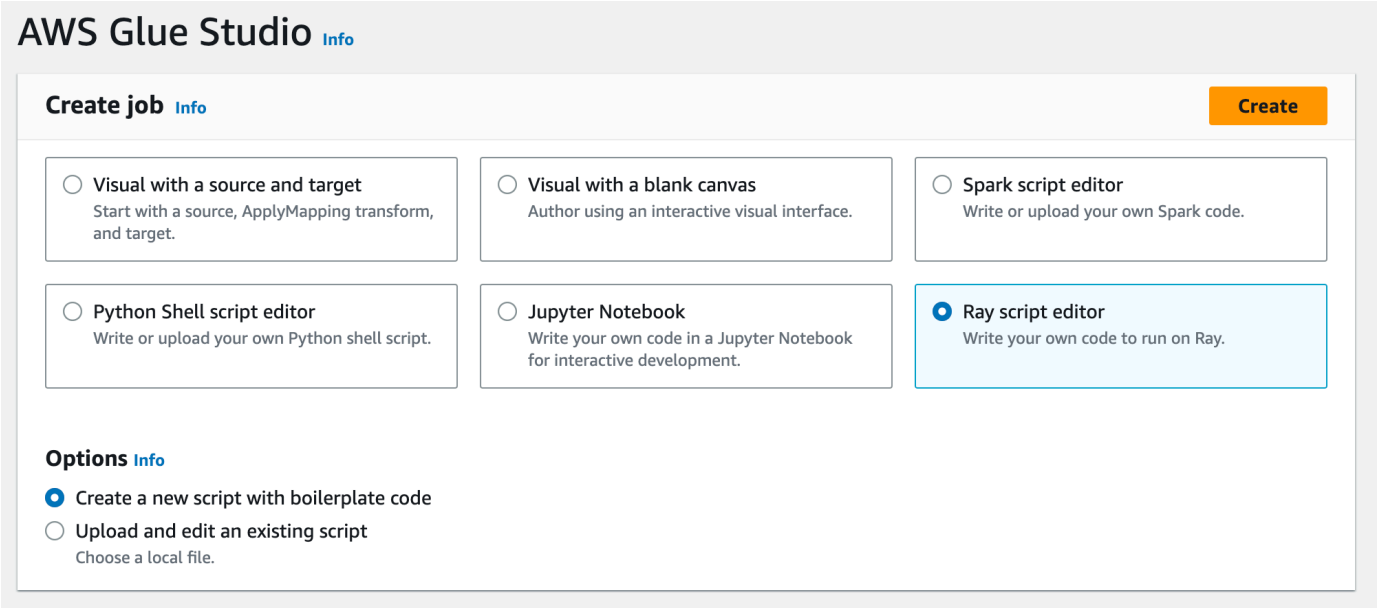
Dans ce didacticiel, nous abordons le scénario ETL suivant : vous souhaitez lire les enregistrements de janvier 2022 du jeu de données New York City TLC Trip Record, ajouter une nouvelle colonne (`tip_rate`) au jeu de données en combinant les données dans des colonnes existantes, puis supprimer un certain nombre de colonnes qui ne sont pas pertinentes pour votre analyse actuelle,

puis vous souhaitez y écrire les résultats. *yourBucketName* Le script Ray suivant exécute les étapes suivantes :

```
import ray
import pandas
from ray import data

ray.init('auto')

ds = ray.data.read_csv("s3://nyc-tlc/opendata_repo/opendata_webconvert/yellow/
yellow_tripdata_2022-01.csv")

# Add the given new column to the dataset and show the sample record after adding a new
 column
ds = ds.add_column( "tip_rate", lambda df: df["tip_amount"] / df["total_amount"])

# Dropping few columns from the underlying Dataset
ds = ds.drop_columns(["payment_type", "fare_amount", "extra", "tolls_amount",
 "improvement_surcharge"])

ds.write_parquet("s3://yourBucketName/ray/tutorial/output/")
```

Pour créer et exécuter une tâche AWS Glue pour Ray

1. Dans le AWS Management Console, accédez à la page de AWS Glue destination.
2. Dans le volet de navigation latéral, choisissez Tâches ETL.
3. Dans Créer une tâche, choisissez Éditeur de script Ray, puis choisissez Créer, comme dans l'illustration suivante.



4. Collez le texte complet du script dans le volet Script et remplacez le texte existant.
5. Accédez aux détails du Job et définissez la propriété IAM Role sur*YourRoleName*.
6. Choisissez Enregistrer, puis Exécuter.

Étape 4 : inspecter votre sortie

Après avoir exécuté votre AWS Glue tâche, vous devez vérifier que le résultat correspond aux attentes de ce scénario. Pour ce faire, suivez la procédure suivante.

Pour valider si votre tâche Ray a été exécutée correctement

1. Sur la page des détails de la tâche, accédez à Exécutions.
2. Au bout de quelques minutes, vous devriez voir une exécution dont le statut d'exécution est Réussi.
3. Accédez à la console Amazon S3 à l'adresse https://console.aws.amazon.com/s3/et inspectez-la*yourBucketName*. Vous devez voir des fichiers écrits dans votre compartiment de sortie.
4. Lisez les fichiers Parquet et vérifiez leur contenu. Vous pouvez le faire avec vos outils existants. Si vous n'avez pas de processus de validation des fichiers Parquet, vous pouvez le faire dans la AWS Glue console avec une session AWS Glue interactive, en utilisant Spark ou Ray (en version préliminaire).

   Dans une session interactive, vous avez accès aux bibliothèques Ray Data, Spark ou Pandas, qui sont fournies par défaut (en fonction du moteur que vous avez choisi). Pour

vérifier le contenu de votre fichier, vous pouvez utiliser les méthodes d'inspection courantes disponibles dans ces bibliothèques (des méthodes comme `count`, `schema` et `show`). Pour plus d'informations sur les sessions interactives dans la console, consultez la section Utilisation de blocs-notes avec AWS Glue Studio et AWS Glue.

Comme vous avez confirmé que les fichiers ont été écrits dans le compartiment, vous pouvez affirmer avec une relative certitude que si votre sortie présente des problèmes, ils ne sont pas liés à la configuration IAM. Configurez votre session *`yourRoleName`* pour avoir accès aux fichiers concernés.

Si vous n'obtenez pas les résultats escomptés, consultez le contenu de dépannage de ce guide pour identifier la source de l'erreur et y remédier. Vous trouverez le contenu de dépannage dans le chapitre Résolution des problèmes de AWS Glue. Pour les erreurs spécifiques liées aux tâches Ray, consultez the section called “Dépannage des erreurs Ray” au chapitre de dépannage.

Étapes suivantes

Vous avez maintenant vu et exécuté un processus ETL utilisant AWS Glue for Ray de bout en bout. Vous pouvez utiliser les ressources suivantes pour comprendre AWS Glue les outils fournis par Ray pour transformer et interpréter vos données à grande échelle.

- Pour plus d'informations sur le modèle de tâche de Ray, consultez the section called “Utiliser Ray Core et Ray Data AWS Glue pour Ray”. Pour plus d'expérience dans l'utilisation des tâches Ray, suivez les exemples de la documentation Ray Core. Consultez Ray Core: Ray Tutorials and Examples (2.4.0) dans la documentation Ray.
- Pour obtenir des conseils sur les bibliothèques de gestion de données disponibles dans AWS Glue for Ray, consultezthe section called “Connexion aux données”. Pour en savoir plus sur l'utilisation de Ray Data pour transformer et écrire des jeux de données, suivez les exemples de la documentation Ray Data. Consultez Ray Data: Examples (2.4.0).
- Pour plus d'informations sur la configuration AWS Glue des tâches Ray, consultezthe section called “Utilisation des tâches Ray”.
- Pour plus d'informations sur l'écriture de scripts AWS Glue pour Ray, continuez à lire la documentation de cette section.

Utiliser Ray Core et Ray Data AWS Glue pour Ray

> **Important**
>
> AWS Glue for Ray ne sera plus ouvert aux nouveaux clients à compter du 30 avril 2026. Si vous souhaitez utiliser AWS Glue for Ray, inscrivez-vous avant cette date. Les clients existants peuvent continuer à utiliser le service normalement. Pour des fonctionnalités similaires à celles de AWS Glue for Ray, explorez Amazon EKS. Pour plus d'informations, consultez la section AWS Glue relative à la fin du support de Ray.

Ray est un cadre permettant d'augmenter les scripts Python en répartissant le travail sur un cluster. Vous pouvez utiliser Ray comme solution à de nombreux types de problèmes. Ray fournit donc des bibliothèques pour optimiser certaines tâches. Dans AWS Glue, nous nous concentrons sur l'utilisation de Ray pour transformer de grands ensembles de données. AWS Glue propose un support pour Ray Data et certaines parties de Ray Core afin de faciliter cette tâche.

Qu'est-ce que Ray Core ?

La première étape de la création d'une application distribuée consiste à identifier et à définir le travail pouvant être effectué simultanément. Ray Core contient les parties de Ray que vous utilisez pour définir les tâches qui peuvent être effectuées simultanément. Ray fournit des informations de référence et de démarrage rapide que vous pouvez utiliser pour découvrir les outils qu'ils fournissent. Pour plus d'informations, consultez Qu'est-ce que Ray Core ? et Ray Core Quick Start. Pour plus d'informations sur la définition efficace des tâches simultanées dans Ray, consultez Tips for first-time users.

> **Tâches et acteurs Ray**
>
> Dans AWS Glue la documentation de Ray, nous pouvons faire référence aux tâches et aux acteurs, qui sont des concepts fondamentaux de Ray.
> Ray utilise les fonctions et les classes Python comme éléments de base d'un système informatique distribué. Tout comme lorsque les fonctions et les variables Python deviennent des « méthodes » et des « attributs » lorsqu'elles sont utilisées dans une classe, les fonctions deviennent des « tâches » et les classes deviennent des « acteurs » lorsqu'elles sont utilisées dans Ray pour envoyer du code aux travailleurs. Vous pouvez identifier les fonctions et les classes susceptibles d'être utilisées par Ray grâce à l'annotation `@ray.remote`.

Les tâches et les acteurs sont configurables, ont un cycle de vie et utilisent des ressources de calcul tout au long de leur cycle de vie. Le code qui génère des erreurs peut être retracé jusqu'à une tâche ou à un acteur lorsque vous trouvez la cause première des problèmes. Ces termes peuvent donc apparaître lorsque vous apprenez à configurer, surveiller ou déboguer AWS Glue des tâches Ray.
Pour commencer à apprendre à utiliser efficacement les tâches et les acteurs pour créer une application distribuée, consultez Key Concepts dans la documentation Ray.

Ray Core va AWS Glue remplacer Ray

AWS Glue les environnements for Ray gèrent la formation et le dimensionnement des clusters, ainsi que la collecte et la visualisation des journaux. Parce que nous gérons ces problèmes, nous limitons par conséquent l'accès et le support APIs au cœur intégré qui serait utilisé pour résoudre ces problèmes dans un cluster open source.

Dans l'environnement d'exécution `Ray2.4` géré, nous ne prenons pas en charge :

- CLI Ray Core
- CLI Ray State
- Les méthodes de l'utilitaire métrique `ray.util.metrics` Prometheus :
  - Compteur
  - Jauge
  - Histogramme
- Autres outils de débogage :
  - ray.util.pdb.set_trace
  - ray.util.inspect_serializability
  - ray.timeline

Qu'est-ce que Ray Data ?

Lorsque vous vous connectez à des sources de données et à des destinations, que vous manipulez des jeux de données et que vous initiez des transformations courantes, Ray Data est une méthodologie simple qui permet d'utiliser Ray pour résoudre les problèmes de transformation des jeux de données Ray. Pour plus d'informations sur l'utilisation de Ray Data, consultez Ray Datasets: Distributed Data Preprocessing.

Vous pouvez utiliser Ray Data ou d'autres outils pour accéder à vos données. Pour plus d'informations sur l'accès à vos données dans Ray, consultez the section called "Connexion aux données".

Ray Data à la AWS Glue place de Ray

Ray Data est pris en charge et fourni par défaut dans l'environnement d'exécution géré `Ray2.4`. Pour plus d'informations sur les modules fournis, consultez the section called "Modules fournis avec les tâches Ray".

Fournir des fichiers et des bibliothèques Python aux tâches Ray

> **Important**
>
> AWS Glue for Ray ne sera plus ouvert aux nouveaux clients à compter du 30 avril 2026. Si vous souhaitez utiliser AWS Glue for Ray, inscrivez-vous avant cette date. Les clients existants peuvent continuer à utiliser le service normalement. Pour des fonctionnalités similaires à celles de AWS Glue for Ray, explorez Amazon EKS. Pour plus d'informations, consultez la section AWS Glue relative à la fin du support de Ray.

Cette section fournit les informations dont vous avez besoin pour utiliser les bibliothèques Python avec les tâches AWS Glue Ray. Vous pouvez utiliser certaines bibliothèques communes incluses par défaut dans toutes les tâches Ray. Vous pouvez également fournir vos propres bibliothèques Python à votre tâche Ray.

Modules fournis avec les tâches Ray

Vous pouvez exécuter des flux de travail d'intégration de données dans une tâche Ray à l'aide des packages fournis suivants. Ces packages sont disponibles par défaut dans les tâches Ray.

AWS Glue version 4.0

Dans la AWS Glue version 4.0, l'environnement Ray (`Ray2.4`runtime) fournit les packages suivants :

- boto3 == 1.26.133
- ray == 2.4.0

- pyarrow == 11.0.0
- pandas == 1.5.3
- numpy == 1.24.3
- fsspec == 2023.4.0

Cette liste inclut tous les packages qui seront installés avec `ray[data] == 2.4.0`. Ray Data est prêt à l'emploi.

Fournir des fichiers à votre tâche Ray

Vous pouvez fournir des fichiers à votre tâche Ray avec le paramètre `--working-dir`. Fournissez à ce paramètre un chemin d'accès à un fichier .zip hébergé sur Amazon S3. Dans le fichier .zip, vos fichiers doivent être contenus dans un seul répertoire de premier niveau. Aucun autre fichier ne doit se trouver au niveau supérieur.

Vos fichiers sont distribués à chaque nœud Ray avant que votre script ne commence à s'exécuter. Réfléchissez à l'impact que cela peut avoir sur l'espace disque disponible pour chaque nœud Ray. L'espace disque disponible est déterminé par l' WorkerType ensemble défini dans la configuration de la tâche. Si vous souhaitez fournir vos données de tâche à grande échelle, ce mécanisme n'est pas la bonne solution. Pour plus d'informations sur la fourniture de données à votre tâche, consultez the section called “Connexion aux données”.

Vos fichiers seront accessibles comme si le répertoire avait été fourni à Ray via le paramètre `working_dir`. Par exemple, pour lire un fichier nommé `sample.txt` dans le répertoire de premier niveau de votre fichier .zip, vous pouvez appeler :

```
@ray.remote
def do_work():
    f = open("sample.txt", "r")
    print(f.read())
```

Pour plus d'informations sur `working_dir`, consultez Ray documentation. Cette fonctionnalité se comporte de la même manière que les fonctionnalités natives de Ray.

Modules Python supplémentaires pour les tâches Ray

Modules supplémentaires de PyPI

Les tâches Ray utilisent Python Package Installer (pip3) pour installer les modules supplémentaires qui seront utilisés par un script Ray. Vous pouvez utiliser le paramètre `--pip-install` avec une liste de modules Python séparés par des virgules pour ajouter un nouveau module ou modifier la version d'un module existant.

Par exemple, pour mettre à jour ou ajouter un nouveau module `scikit-learn`, utilisez la paire clé-valeur suivante :

```
"--pip-install", "scikit-learn==0.21.3"
```

Si vous avez des modules ou des correctifs personnalisés, vous pouvez distribuer vos propres bibliothèques depuis Amazon S3 avec le paramètre `--s3-py-modules`. Avant de charger votre distribution, il peut être nécessaire de la reconditionner et de la recréer. Suivez les lignes directrices indiquées dans the section called "Inclure du code Python dans les tâches Ray".

Distributions personnalisées depuis Amazon S3

Les distributions personnalisées doivent respecter les directives d'empaquetage de Ray en ce qui concerne les dépendances. La section suivante explique comment créer ces distributions. Pour plus d'informations sur la façon dont Ray configure les dépendances, consultez Environment Dependencies (Dépendances d'environnement) dans la documentation Ray.

Pour inclure un élément distribuable personnalisé après avoir évalué son contenu, chargez votre élément distribuable dans un compartiment disponible pour le rôle IAM de la tâche. Indiquez le chemin Amazon S3 vers une archive zip Python dans la configuration de vos paramètres. Si vous fournissez plusieurs éléments distribuables, séparez-les par des virgules. Par exemple :

```
"--s3-py-modules", "s3://s3bucket/pythonPackage.zip"
```

Limites

Les tâches Ray ne prennent pas en charge la compilation de code natif dans l'environnement de la tâche. Cela peut vous limiter si vos dépendances Python dépendent de manière transitive du code compilé natif. Les tâches Ray peuvent exécuter les fichiers binaires fournis, mais ils doivent être compilés pour Linux sur ARM64. Vous pourrez ainsi peut-être utiliser le contenu des roues `aarch64manylinux`. Vous pouvez fournir vos dépendances natives dans un format compilé en reconditionnant une roue selon les normes Ray. Cela implique généralement de supprimer des dossiers `dist-info` pour ne laisser qu'un seul dossier à la racine de l'archive.

Vous ne pouvez pas mettre à niveau la version de `ray` ou de `ray[data]` en utilisant ce paramètre. Pour utiliser une nouvelle version de Ray, vous devrez modifier le champ d'exécution de votre

tâche, une fois que nous aurons mis en place la prise en charge de cette version. Pour de plus amples informations sur les versions de Ray pris en charge, consultez the section called "AWS Glue versions".

Inclure du code Python dans les tâches Ray

La Python Software Foundation propose des comportements standardisés pour l'empaquetage de fichiers Python destinés à être utilisés avec différents environnements d'exécution. Ray introduit des limites que vous devez connaître en matière de normes d'empaquetage. AWS Glue ne spécifie pas de normes d'empaquetage autres que celles spécifiées à Ray. Les instructions suivantes fournissent des conseils standards sur l'empaquetage de packages Python simples.

Empaquetez vos fichiers dans une archive `.zip`. Un répertoire doit se trouver à la racine de l'archive. Il ne doit pas y avoir d'autres fichiers au niveau de la racine de l'archive, sous peine de provoquer un comportement inattendu. Le répertoire racine est le package, et son nom est utilisé pour faire référence à votre code Python lorsque vous l'importez.

Si vous fournissez une distribution sous cette forme à une tâche Ray avec `--s3-py-modules`, vous pourrez importer du code Python depuis votre package dans votre script Ray.

Votre package peut fournir un seul module Python avec des fichiers Python, ou vous pouvez regrouper plusieurs modules. Lorsque vous reconditionnez des dépendances, telles que les bibliothèques de PyPI, vérifiez la présence de fichiers cachés et de répertoires de métadonnées dans ces packages.

> **Warning**
>
> Certains comportements du système d'exploitation font qu'il est difficile de suivre correctement ces instructions d'empaquetage.
>
> - OSX peut ajouter des fichiers cachés tels que `__MACOSX` à votre fichier zip au niveau supérieur.
> - Windows peut ajouter automatiquement vos fichiers à un dossier à l'intérieur du fichier zip, créant ainsi involontairement un dossier imbriqué.

Les procédures suivantes supposent que vous interagissez avec vos fichiers sous Amazon Linux 2 ou un système d'exploitation similaire qui fournit une distribution des utilitaires Info-ZIP `zip` et `zipinfo`. Nous vous recommandons d'utiliser ces outils pour éviter les comportements inattendus.

Pour empaqueter des fichiers Python à utiliser dans Ray

1. Créez un répertoire temporaire avec le nom de votre package, puis confirmez que votre répertoire de travail est son répertoire parent. Vous pouvez le faire à l'aide des commandes suivantes :

```
cd parent_directory
mkdir temp_dir
```

2. Copiez vos fichiers dans le répertoire temporaire, puis confirmez la structure de votre répertoire. Le contenu de ce répertoire sera directement accessible en tant que module Python. Vous pouvez le faire à l'aide de la commande suivante :

```
ls -AR temp_dir
# my_file_1.py
# my_file_2.py
```

3. Compressez votre dossier temporaire en utilisant zip. Vous pouvez le faire à l'aide des commandes suivantes :

```
zip -r zip_file.zip temp_dir
```

4. Vérifiez que votre fichier est correctement empaqueté. *zip_file*`.zip` doit maintenant se trouver dans votre répertoire de travail. Vous pouvez l'inspecter avec la commande suivante :

```
zipinfo -1 zip_file.zip
# temp_dir/
# temp_dir/my_file_1.py
# temp_dir/my_file_2.py
```

Pour reconditionner un package Python à utiliser dans Ray.

1. Créez un répertoire temporaire avec le nom de votre package, puis confirmez que votre répertoire de travail est son répertoire parent. Vous pouvez le faire à l'aide des commandes suivantes :

```
cd parent_directory
mkdir temp_dir
```

2. Décompressez votre package et copiez-en le contenu dans votre répertoire temporaire. Supprimez les fichiers liés à votre ancienne norme d'emballage, en ne laissant que le contenu du module. Vérifiez que la structure de votre fichier semble correcte à l'aide de la commande suivante :

```
ls -AR temp_dir
# my_module
# my_module/__init__.py
# my_module/my_file_1.py
# my_module/my_submodule/__init__.py
# my_module/my_submodule/my_file_2.py
# my_module/my_submodule/my_file_3.py
```

3. Compressez votre dossier temporaire en utilisant zip. Vous pouvez le faire à l'aide des commandes suivantes :

```
zip -r zip_file.zip temp_dir
```

4. Vérifiez que votre fichier est correctement empaqueté. *zip_file*.zip doit maintenant se trouver dans votre répertoire de travail. Vous pouvez l'inspecter avec la commande suivante :

```
zipinfo -1 zip_file.zip
# temp_dir/my_module/
# temp_dir/my_module/__init__.py
# temp_dir/my_module/my_file_1.py
# temp_dir/my_module/my_submodule/
# temp_dir/my_module/my_submodule/__init__.py
# temp_dir/my_module/my_submodule/my_file_2.py
# temp_dir/my_module/my_submodule/my_file_3.py
```

Connexion aux données dans les tâches Ray

> **Important**
>
> AWS Glue for Ray ne sera plus ouvert aux nouveaux clients à compter du 30 avril 2026. Si vous souhaitez utiliser AWS Glue for Ray, inscrivez-vous avant cette date. Les clients existants peuvent continuer à utiliser le service normalement. Pour des fonctionnalités

similaires à celles de AWS Glue for Ray, explorez Amazon EKS. Pour plus d'informations, consultez la section AWS Glue relative à la fin du support de Ray.

AWS Glue Ray Jobs peut utiliser un large éventail de packages Python conçus pour vous permettre d'intégrer rapidement des données. Nous fournissons un ensemble minimal de dépendances afin de ne pas encombrer votre environnement. Pour plus d'informations sur ce qui est inclus par défaut, consultez the section called "Modules fournis avec les tâches Ray".

> **Note**
>
> AWS Glue extract, transform, and load (ETL) fournit l' DynamicFrame abstraction nécessaire pour rationaliser les flux de travail ETL dans lesquels vous résolvez les différences de schéma entre les lignes de votre ensemble de données. AWS Glue L'ETL fournit des fonctionnalités supplémentaires : signets de tâches et regroupement de fichiers d'entrée. Actuellement, nous ne proposons pas de fonctionnalités correspondantes dans les tâches Ray.
>
> AWS Glue for Spark fournit un support direct pour la connexion à certains formats de données, sources et récepteurs. Dans Ray, le kit AWS SDK pour Pandas et les bibliothèques tierces actuelles répondent en grande partie à ce besoin. Vous devrez consulter ces bibliothèques pour connaître les fonctionnalités disponibles.

AWS Glue pour Ray, l'intégration avec Amazon VPC n'est pas disponible actuellement. Les ressources d'un Amazon VPC ne seront pas accessibles sans routage public. Pour plus d'informations sur l'utilisation AWS Glue avec Amazon VPC, consultez. the section called "Configuration de points de terminaison d'un VPC d'interface (AWS PrivateLink) pour AWS Glue "

Bibliothèques communes pour travailler avec des données dans Ray

Ray Data : Ray Data fournit des méthodes pour gérer les formats de données, les sources et les récepteurs courants. Pour plus d'informations sur les formats et les sources pris en charge dans Ray Data, consultez Input/Output dans la documentation Ray Data. Ray Data est une bibliothèque d'opinion, plutôt qu'une bibliothèque générale, destinée à manipuler des jeux de données.

Ray fournit des conseils sur les cas d'utilisation où Ray Data pourrait être la meilleure solution pour votre tâche. Pour en savoir plus, consultez Ray use cases dans la documentation Ray.

AWS SDK pour pandas (awswrangler) — Le AWS SDK pour pandas est un AWS produit qui fournit des solutions propres et testées pour lire et écrire vers des AWS services lorsque vos transformations gèrent des données avec des pandas. DataFrames Pour plus d'informations sur les formats et les sources pris en charge dans le AWS SDK pour pandas, consultez la référence d'API dans la documentation du AWS SDK pour pandas.

Pour des exemples de lecture et d'écriture de données avec le AWS SDK pour pandas, voir Quick Start dans la documentation du AWS SDK pour pandas. Le AWS SDK pour pandas ne fournit pas de transformations pour vos données. Il fournit uniquement une prise en charge pour la lecture et l'écriture à partir de sources.

Modin : Modin est une bibliothèque Python qui implémente les opérations Pandas courantes de manière distribuable. Pour plus d'informations sur Modin, consultez Modin documentation. Modin lui-même ne fournit pas de prise en charge de la lecture et de l'écriture à partir de sources. Il fournit des implémentations distribuées de transformations courantes. Modin est supporté par le AWS SDK pour pandas.

Lorsque vous exécutez Modin et le AWS SDK pour pandas ensemble dans un environnement Ray, vous pouvez effectuer des tâches ETL courantes avec des résultats performants. Pour plus d'informations sur l'utilisation de Modin avec le AWS SDK pour pandas, voir At scale dans la documentation du AWS SDK pour pandas.

Autres cadres : pour plus d'informations sur les cadres pris en charge par Ray, consultez The Ray Ecosystem dans la documentation Ray. Nous ne prenons pas en charge les autres frameworks AWS Glue de Ray.

Connexion aux données via le catalogue de données

La gestion de vos données via le catalogue de données en conjonction avec les tâches Ray est prise en charge par le AWS SDK pour pandas. Pour plus d'informations, consultez Glue Catalog sur le site AWS Web du SDK pour pandas.

Utilisation de ce service avec un AWS SDK

AWS des kits de développement logiciel (SDKs) sont disponibles pour de nombreux langages de programmation courants. Chaque kit SDK fournit une API, des exemples de code et de la documentation qui facilitent la création d'applications par les développeurs dans leur langage préféré.

Documentation SDK	Exemples de code
AWS SDK pour C++	AWS SDK pour C++ exemples de code
AWS CLI	AWS CLI exemples de code
AWS SDK pour Go	AWS SDK pour Go exemples de code
AWS SDK pour Java	AWS SDK pour Java exemples de code
AWS SDK pour JavaScript	AWS SDK pour JavaScript exemples de code
AWS SDK pour Kotlin	AWS SDK pour Kotlin exemples de code
AWS SDK pour .NET	AWS SDK pour .NET exemples de code
AWS SDK pour PHP	AWS SDK pour PHP exemples de code
Outils AWS pour PowerShell	Outils AWS pour PowerShell exemples de code
AWS SDK pour Python (Boto3)	AWS SDK pour Python (Boto3) exemples de code
AWS SDK pour Ruby	AWS SDK pour Ruby exemples de code
AWS SDK pour Rust	AWS SDK pour Rust exemples de code
AWS SDK pour SAP ABAP	AWS SDK pour SAP ABAP exemples de code
AWS SDK pour Swift	AWS SDK pour Swift exemples de code

Pour voir des exemples spécifiques à ce service, consultez AWS Glue Exemples de code API utilisant AWS SDKs.

Exemple de disponibilité

Vous n'avez pas trouvé ce dont vous avez besoin ? Demandez un exemple de code en utilisant le lien Provide feedback (Fournir un commentaire) en bas de cette page.

AWS Glue API

Cette section décrit les types de données et les primitives utilisés par AWS Glue SDKs et Tools. Il existe trois méthodes générales d'interaction AWS Glue programmatique en dehors du AWS Management Console, chacune avec sa propre documentation :

- Les bibliothèques SDK linguistiques vous permettent d'accéder aux ressources AWS provenant de langages de programmation courants. Pour plus d'informations, consultez Outils pour créer sur AWS.
- Vous AWS CLI permet d'accéder aux AWS ressources depuis la ligne de commande. Pour plus d'informations, consultez Référence des commandes AWS CLI.
- CloudFormation vous permet de définir un ensemble de AWS ressources à provisionner ensemble de manière cohérente. Pour plus d'informations, consultez CloudFormation: référence AWS Glue des types de ressources.

Cette section documente les primitives partagées indépendamment de celles-ci SDKs et des outils. Les outils utilisent le AWS Glue Web API Reference pour communiquer avec AWS.

Table des matières

- Sécurité APIs dans AWS Glue
  - Types de données
  - DataCatalogEncryptionSettings structure
  - EncryptionAtRest structure
  - ConnectionPasswordEncryption structure
  - EncryptionConfiguration structure
  - Structure S3Encryption
  - CloudWatchEncryption structure
  - JobBookmarksEncryption structure
  - SecurityConfiguration structure
  - GluePolicy structure
  - DataQualityEncryption structure
  - Opérations
  - GetDataCatalogEncryptionSettings action (Python : get_data_catalog_encryption_settings)

- PutDataCatalogEncryptionSettings action (Python : put_data_catalog_encryption_settings)
- PutResourcePolicy action (Python : put_resource_policy)
- GetResourcePolicy action (Python : get_resource_policy)
- DeleteResourcePolicy action (Python : delete_resource_policy)
- CreateSecurityConfiguration action (Python : create_security_configuration)
- DeleteSecurityConfiguration action (Python : delete_security_configuration)
- GetSecurityConfiguration action (Python : get_security_configuration)
- GetSecurityConfigurations action (Python : get_security_configurations)
- GetResourcePolicies action (Python : get_resource_policies)
- API des objets du catalogue
  - API de catalogues
    - Types de données
    - Structure du catalogue
    - CatalogInput structure
    - TargetRedshiftCatalog structure
    - CatalogProperties structure
    - CatalogPropertiesOutput structure
    - DataLakeAccessProperties structure
    - IcebergOptimizationProperties structure
    - DataLakeAccessPropertiesOutput structure
    - IcebergOptimizationPropertiesOutput structure
    - FederatedCatalog structure
    - Opérations
    - CreateCatalog action (Python : create_catalog)
    - UpdateCatalog action (Python : update_catalog)
    - DeleteCatalog action (Python : supprimer_catalogue)
    - GetCatalog action (Python : get_catalog)
    - GetCatalogs action (Python : get_catalogs)
  - API d'une base de données
    - Types de données

- Structure Database
- DatabaseInput structure
- PrincipalPermissions structure
- DataLakePrincipal structure
- DatabaseIdentifier structure
- FederatedDatabase structure
- Opérations
- CreateDatabase action (Python : créer_base de données)
- UpdateDatabase action (Python : update_database)
- DeleteDatabase action (Python : supprimer_base de données)
- GetDatabase action (Python : get_database)
- GetDatabases action (Python : get_databases)

- API de table
  - Types de données
  - Structure de table
  - TableInput structure
  - FederatedTable structure
  - Structure de colonne
  - StorageDescriptor structure
  - SchemaReference structure
  - SerDeInfo structure
  - Structure de tri
  - SkewedInfo structure
  - TableVersion structure
  - TableError structure
  - TableVersionError structure
  - SortCriterion structure
  - TableIdentifier structure
  - KeySchemaElement structure
  - PartitionIndex structure

- PartitionIndexDescriptor structure
- BackfillError structure
- IcebergInput structure
- OpenTableFormatInput structure
- ViewDefinition structure
- ViewDefinitionInput structure
- ViewRepresentation structure
- ViewRepresentationInput structure
- UpdateOpenTableFormatInput structure
- UpdateIcebergInput structure
- CreateIcebergTableInput structure
- UpdateIcebergTableInput structure
- IcebergSortOrder structure
- IcebergSortField structure
- IcebergPartitionSpec structure
- IcebergPartitionField structure
- IcebergSchema structure
- IcebergStructField structure
- IcebergTableUpdate structure
- AuditContext structure
- Opérations
- CreateTable action (Python : créer_table)
- UpdateTable action (Python : update_table)
- DeleteTable action (Python : supprimer_table)
- BatchDeleteTable action (Python : batch_delete_table)
- GetTable action (Python : get_table)
- GetTables action (Python : get_tables)
- GetTableVersion action (Python : get_table_version)
- GetTableVersions action (Python : get_table_versions)
- DeleteTableVersion action (Python : supprimer_table_version)

- BatchDeleteTableVersion action (Python : batch_delete_table_version)
- SearchTables action (Python : tables de recherche)
- GetPartitionIndexes action (Python : get_partition_indexes)
- CreatePartitionIndex action (Python : create_partition_index)
- DeletePartitionIndex action (Python : delete_partition_index)
- GetColumnStatisticsForTable action (Python : get_column_statistics_for_table)
- UpdateColumnStatisticsForTable action (Python : update_column_statistics_for_table)
- DeleteColumnStatisticsForTable action (Python : delete_column_statistics_for_table)

- API de partition
  - Types de données
  - Structure de partition
  - PartitionInput structure
  - PartitionSpecWithSharedStorageDescriptor structure
  - PartitionListComposingSpec structure
  - PartitionSpecProxy structure
  - PartitionValueList structure
  - Structure d'un segment
  - PartitionError structure
  - BatchUpdatePartitionFailureEntry structure
  - BatchUpdatePartitionRequestEntry structure
  - StorageDescriptor structure
  - SchemaReference structure
  - SerDeInfo structure
  - SkewedInfo structure
  - Opérations
  - CreatePartition action (Python : créer_partition)
  - BatchCreatePartition action (Python : batch_create_partition)
  - UpdatePartition action (Python : update_partition)
  - DeletePartition action (Python : supprimer_partition)
  - BatchDeletePartition action (Python : batch_supprimer_partition)

- GetPartition action (Python : get_partition)
- GetPartitions action (Python : get_partitions)
- BatchGetPartition action (Python : batch_get_partition)
- BatchUpdatePartition action (Python : batch_update_partition)
- GetColumnStatisticsForPartition action (Python : get_column_statistics_for_partition)
- UpdateColumnStatisticsForPartition action (Python : update_column_statistics_for_partition)
- DeleteColumnStatisticsForPartition action (Python : supprimer_column_statistics_for_partition)

- API de connexions
  - API de connexion
    - Types de données
    - Structure de connexion
    - ConnectionInput structure
    - TestConnectionInput structure
    - PhysicalConnectionRequirements structure
    - GetConnectionsFilter structure
    - AuthenticationConfiguration structure
    - AuthenticationConfigurationInput structure
    - OAuth2Structure des propriétés
    - OAuth2PropertiesInput structure
    - OAuth2ClientApplication structure
    - AuthorizationCodeProperties structure
    - BasicAuthenticationCredentials structure
    - OAuth2Structure des informations d'identification
    - Opérations
    - CreateConnection action (Python : créer_connexion)
    - DeleteConnection action (Python : supprimer_connexion)
    - GetConnection action (Python : get_connection)
    - GetConnections action (Python : get_connections)
    - UpdateConnection action (Python : update_connection)

        - TestConnection action (Python : test_connection)
        - BatchDeleteConnection action (Python : batch_delete_connection)
    - API de types de connexion
        - Gestion des connexions APIs
        - DescribeConnectionType action (Python : describe_connection_type)
        - ListConnectionTypes action (Python : list_connection_types)
        - ConnectionTypeBrief structure
        - ConnectionTypeVariant structure
        - types de données
        - Structure de validation
        - AuthConfiguration structure
        - Structure des fonctionnalités
        - Structure de la propriété
        - AllowedValue structure
        - ComputeEnvironmentConfiguration structure
    - Métadonnées de connexion et API de prévisualisation
        - Types de données
        - Structure de l'entité
        - Structure de champ
        - Opérations
        - ListEntities action (Python : list_entities)
        - DescribeEntity action (Python : describe_entity)
        - GetEntityRecords action (Python : get_entity_records)
- API de fonction définie par l'utilisateur
    - Types de données
    - UserDefinedFunction structure
    - UserDefinedFunctionInput structure
    - Opérations
    - CreateUserDefinedFunction action (Python : create_user_defined_function)
    - UpdateUserDefinedFunction action (Python : update_user_defined_function)

        - DeleteUserDefinedFunction action (Python : delete_user_defined_function)
        - GetUserDefinedFunction action (Python : get_user_defined_function)
        - GetUserDefinedFunctions action (Python : get_user_defined_functions)
    - Importation d'un Athena catalogue dans AWS Glue
        - Types de données
        - CatalogImportStatus structure
        - Opérations
        - ImportCatalogToGlue action (Python : import_catalog_to_glue)
        - GetCatalogImportStatus action (Python : get_catalog_import_status)
- API d'optimiseur de table
    - Types de données
    - TableOptimizer structure
    - TableOptimizerConfiguration structure
    - TableOptimizerVpcConfiguration structure
    - CompactionConfiguration structure
    - IcebergCompactionConfiguration structure
    - TableOptimizerRun structure
    - BatchGetTableOptimizerEntry structure
    - BatchTableOptimizer structure
    - BatchGetTableOptimizerError structure
    - RetentionConfiguration structure
    - IcebergRetentionConfiguration structure
    - OrphanFileDeletionConfiguration structure
    - IcebergOrphanFileDeletionConfiguration structure
    - CompactionMetrics structure
    - RetentionMetrics structure
    - OrphanFileDeletionMetrics structure
    - IcebergCompactionMetrics structure
    - IcebergRetentionMetrics structure
    - IcebergOrphanFileDeletionMetrics structure

- RunMetrics structure
- Opérations
- GetTableOptimizer action (Python : get_table_optimizer)
- BatchGetTableOptimizer action (Python : batch_get_table_optimizer)
- ListTableOptimizerRuns action (Python : list_table_optimizer_runs)
- CreateTableOptimizer action (Python : create_table_optimizer)
- DeleteTableOptimizer action (Python : delete_table_optimizer)
- UpdateTableOptimizer action (Python : update_table_optimizer)

- API de classifieurs et d'crawlers
  - API du classifieur
    - Types de données
    - Structure du classifieur
    - GrokClassifier structure
    - XMLClassifier structure
    - JsonClassifier structure
    - CsvClassifier structure
    - CreateGrokClassifierRequest structure
    - UpdateGrokClassifierRequest structure
    - Créer une structure XMLClassifier de demande
    - Structure de XMLClassifier demande de mise à jour
    - CreateJsonClassifierRequest structure
    - UpdateJsonClassifierRequest structure
    - CreateCsvClassifierRequest structure
    - UpdateCsvClassifierRequest structure
    - Opérations
    - CreateClassifier action (Python : create_classifier)
    - DeleteClassifier action (Python : delete_classifier)
    - GetClassifier action (Python : get_classifier)
    - GetClassifiers action (Python : get_classifiers)
    - UpdateClassifier action (Python : update_classifier)

- API du crawler
  - Types de données
  - Structure du crawler
  - Structure du planificateur
  - CrawlerTargets structure
  - Structure de la S3Target
  - DeltaCatalogTarget Structure S3
  - DeltaDirectTarget Structure S3
  - JdbcTarget structure
  - Structure Mongo DBTarget
  - Structure de la dynamo DBTarget
  - DeltaTarget structure
  - IcebergTarget structure
  - HudiTarget structure
  - CatalogTarget structure
  - CrawlerMetrics structure
  - CrawlerHistory structure
  - CrawlsFilter structure
  - SchemaChangePolicy structure
  - LastCrawlInfo structure
  - RecrawlPolicy structure
  - LineageConfiguration structure
  - LakeFormationConfiguration structure
  - Opérations
  - CreateCrawler action (Python : create_crawler)
  - DeleteCrawler action (Python : delete_crawler)
  - GetCrawler action (Python : get_crawler)
  - GetCrawlers action (Python : get_crawlers)
  - GetCrawlerMetrics action (Python : get_crawler_metrics)
  - UpdateCrawler action (Python : update_crawler)

- StartCrawler action (Python : start_crawler)
- StopCrawler action (Python : stop_crawler)
- BatchGetCrawlers action (Python : batch_get_crawlers)
- ListCrawlers action (Python : list_crawlers)
- ListCrawls action (Python : list_crawls)

- API de statistiques de colonne
  - Types de données
  - ColumnStatisticsTaskRun structure
  - ColumnStatisticsTaskSettings structure
  - ExecutionAttempt structure
  - Opérations
  - StartColumnStatisticsTaskRun action (Python : start_column_statistics_task_run)
  - GetColumnStatisticsTaskRun action (Python : get_column_statistics_task_run)
  - GetColumnStatisticsTaskRuns action (Python : get_column_statistics_task_runs)
  - ListColumnStatisticsTaskRuns action (Python : list_column_statistics_task_runs)
  - StopColumnStatisticsTaskRun action (Python : stop_column_statistics_task_run)
  - CreateColumnStatisticsTaskSettings action (Python : create_column_statistics_task_settings)
  - UpdateColumnStatisticsTaskSettings action (Python : update_column_statistics_task_settings)
  - GetColumnStatisticsTaskSettings action (Python : get_column_statistics_task_settings)
  - DeleteColumnStatisticsTaskSettings action (Python : delete_column_statistics_task_settings)
  - StartColumnStatisticsTaskRunSchedule action (Python : start_column_statistics_task_run_schedule)
  - StopColumnStatisticsTaskRunSchedule action (Python : stop_column_statistics_task_run_schedule)
  - Exceptions
  - ColumnStatisticsTaskRunningException structure
  - ColumnStatisticsTaskNotRunningException structure
  - ColumnStatisticsTaskStoppingException structure
  - ColumnStatisticsTaskAutoConcurrencyLimitException structure

    - InvalidCatalogSettingException structure
  - API du planificateur du crawler
    - Types de données
    - Structure du planificateur
    - Opérations
    - UpdateCrawlerSchedule action (Python : update_crawler_schedule)
    - StartCrawlerSchedule action (Python : start_crawler_schedule)
    - StopCrawlerSchedule action (Python : stop_crawler_schedule)
- Génération automatique d'API de scripts ETL
  - Types de données
  - CodeGenNode structure
  - CodeGenNodeArg structure
  - CodeGenEdge structure
  - Structure de l'emplacement
  - CatalogEntry structure
  - MappingEntry structure
  - Opérations
  - CreateScript action (Python : créer_script)
  - GetDataflowGraph action (Python : get_dataflow_graph)
  - GetMapping action (Python : get_mapping)
  - GetPlan action (Python : get_plan)
- API Visual Job
  - Types de données
  - CodeGenConfigurationNode structure
  - JDBCConnectorStructure des options
  - StreamingDataPreviewOptions structure
  - AthenaConnectorSource structure
  - JDBCConnectorStructure de la source
  - SparkConnectorSource structure
  - CatalogSource structure

- Structure SQLCatalog de ma source
- Structure de la SQLCatalog source Postgre
- Structure d'Oracle SQLCatalog Source
- SQLServerCatalogSource Structure de Microsoft
- CatalogKinesisSource structure
- DirectKinesisSource structure
- KinesisStreamingSourceOptions structure
- CatalogKafkaSource structure
- DirectKafkaSource structure
- KafkaStreamingSourceOptions structure
- RedshiftSource structure
- AmazonRedshiftSource structure
- AmazonRedshiftNodeData structure
- AmazonRedshiftAdvancedOption structure
- Structure de l'option
- CatalogSource Structure S3
- SourceAdditionalOptions Structure S3
- CsvSource Structure S3
- JDBCSource Structure directe
- DirectSourceAdditionalOptions Structure S3
- JsonSource Structure S3
- ParquetSource Structure S3
- DeltaSource Structure S3
- CatalogDeltaSource Structure S3
- CatalogDeltaSource structure
- HudiSource Structure S3
- CatalogHudiSource Structure S3
- ExcelSource Structure S3
- CatalogHudiSource structure
- Structure de Dynamo DBCatalog Source

- RelationalCatalogSource structure
- JDBCConnectorStructure de la cible
- SparkConnectorTarget structure
- BasicCatalogTarget structure
- Structure de My SQLCatalog Target
- Structure de SQLCatalog PostgreTarget
- Structure d'Oracle SQLCatalog Target
- SQLServerCatalogTarget Structure de Microsoft
- RedshiftTarget structure
- AmazonRedshiftTarget structure
- UpsertRedshiftTargetOptions structure
- CatalogTarget Structure S3
- GlueParquetTarget Structure S3
- CatalogSchemaChangePolicy structure
- DirectTarget Structure S3
- HudiCatalogTarget Structure S3
- HudiDirectTarget Structure S3
- DeltaCatalogTarget Structure S3
- DeltaDirectTarget Structure S3
- HyperDirectTarget Structure S3
- IcebergDirectTarget Structure S3
- DirectSchemaChangePolicy structure
- ApplyMapping structure
- Structure de mappage
- SelectFields structure
- DropFields structure
- RenameField structure
- Structure Spigot
- Structure Join
- JoinColumn structure

- SplitFields structure
- SelectFromCollection structure
- FillMissingValues structure
- Structure Filtre
- FilterExpression structure
- FilterValue structure
- CustomCode structure
- Structure SparkSQL
- SqlAlias structure
- DropNullFields structure
- NullCheckBoxList structure
- NullValueField structure
- Structure Datatype
- Structure Fusion
- Structure Union
- PIIDetection structure
- Structure Aggregate
- DropDuplicates structure
- GovernedCatalogTarget structure
- GovernedCatalogSource structure
- AggregateOperation structure
- GlueSchema structure
- GlueStudioSchemaColumn structure
- GlueStudioColumn structure
- DynamicTransform structure
- TransformConfigParameter structure
- EvaluateDataQuality structure
- DQResultsPublishingOptions structure
- DQStopJobOnFailureOptions structure
- EvaluateDataQualityMultiFrame structure

- Structure de la recette
- RecipeReference structure
- SnowflakeNodeData structure
- SnowflakeSource structure
- SnowflakeTarget structure
- ConnectorDataSource structure
- ConnectorDataTarget structure
- RecipeStep structure
- RecipeAction structure
- ConditionExpression structure
- CatalogIcebergSource Structure S3
- CatalogIcebergSource structure
- IcebergCatalogTarget Structure S3
- Structure de Dynamo DBELTConnector Source
- DDBELTConnectionStructure des options
- DDBELTCatalogAdditionalOptions structure
- Structure de route
- GroupFilters structure
- AutoDataQuality structure

- API de tâches
  - Tâches
    - Types de données
    - Structure Job
    - ExecutionProperty structure
    - NotificationProperty structure
    - JobCommand structure
    - ConnectionsList structure
    - JobUpdate structure
    - SourceControlDetails structure
    - Opérations

    - CreateJob action (Python : create_job)
    - UpdateJob action (Python : update_job)
    - GetJob action (Python : get_job)
    - GetJobs action (Python : get_jobs)
    - DeleteJob action (Python : supprimer_tâche)
    - ListJobs action (Python : list_jobs)
    - BatchGetJobs action (Python : batch_get_jobs)
- Exécutions de tâches
    - Types de données
    - JobRun structure
    - Structure de Predecessor
    - JobBookmarkEntry structure
    - BatchStopJobRunSuccessfulSubmission structure
    - BatchStopJobRunError structure
    - NotificationProperty structure
    - Opérations
    - StartJobRun action (Python : start_job_run)
    - BatchStopJobRun action (Python : batch_stop_job_run)
    - GetJobRun action (Python : get_job_run)
    - GetJobRuns action (Python : get_job_runs)
    - GetJobBookmark action (Python : get_job_bookmark)
    - GetJobBookmarks action (Python : get_job_bookmarks)
    - ResetJobBookmark action (Python : reset_job_bookmark)
- Triggers
    - Types de données
    - Structure Trigger
    - TriggerUpdate structure
    - Structure Predicate
    - Structure Condition
    - Structure Action

    - EventBatchingCondition structure
    - Opérations
    - CreateTrigger action (Python : create_trigger)
    - StartTrigger action (Python : start_trigger)
    - GetTrigger action (Python : get_trigger)
    - GetTriggers action (Python : get_triggers)
    - UpdateTrigger action (Python : update_trigger)
    - StopTrigger action (Python : stop_trigger)
    - DeleteTrigger action (Python : delete_trigger)
    - ListTriggers action (Python : list_triggers)
    - BatchGetTriggers action (Python : batch_get_triggers)
- Intégration APIs dans AWS Glue
    - Types de données
    - Structure d'intégration
    - IntegrationConfig structure
    - IntegrationPartition structure
    - IntegrationError structure
    - IntegrationFilter structure
    - InboundIntegration structure
    - SourceProcessingProperties structure
    - TargetProcessingProperties structure
    - SourceTableConfig structure
    - TargetTableConfig structure
    - Opérations
    - CreateIntegration action (Python : create_integration)
    - ModifyIntegration action (Python : modifier_intégration)
    - DescribeIntegrations action (Python : describe_integrations)
    - DeleteIntegration action (Python : delete_integration)
    - DescribeInboundIntegrations action (Python : describe_inbound_integrations)
    - CreateIntegrationTableProperties action (Python : create_integration_table_properties)

- UpdateIntegrationTableProperties action (Python : update_integration_table_properties)
- GetIntegrationTableProperties action (Python : get_integration_table_properties)
- DeleteIntegrationTableProperties action (Python : delete_integration_table_properties)
- CreateIntegrationResourceProperty action (Python : create_integration_resource_property)
- UpdateIntegrationResourceProperty action (Python : update_integration_resource_property)
- GetIntegrationResourceProperty action (Python : get_integration_resource_property)
- UntagResource action (Python : untag_resource)
- ListTagsForResource action (Python : list_tags_for_resource)
- Exceptions
- ResourceNotFoundException structure
- InternalServerException structure
- IntegrationAlreadyExistsFault structure
- IntegrationConflictOperationFault structure
- IntegrationQuotaExceededFault structure
- KMSKeyNotAccessibleFault structure
- IntegrationNotFoundFault structure
- TargetResourceNotFound structure
- InvalidIntegrationStateFault structure
- Séances interactives API
  - Types de données
  - Structure de séance
  - SessionCommand structure
  - Structure de la déclaration
  - StatementOutput structure
  - StatementOutputData structure
  - ConnectionsList structure
  - Opérations
  - CreateSession action (Python : créer_session)
  - StopSession action (Python : stop_session)
  - DeleteSession action (Python : supprimer_session)

- GetSession action (Python : get_session)
- ListSessions action (Python : list_sessions)
- RunStatement action (Python : run_statement)
- CancelStatement action (Python : annuler_statement)
- GetStatement action (Python : get_statement)
- ListStatements action (Python : list_statements)
- GetGlueIdentityCenterConfiguration action (Python : get_glue_identity_center_configuration)
- UpdateGlueIdentityCenterConfiguration action (Python : update_glue_identity_center_configuration)
- CreateGlueIdentityCenterConfiguration action (Python : create_glue_identity_center_configuration)
- DeleteGlueIdentityCenterConfiguration action (Python : delete_glue_identity_center_configuration)

- API de points de terminaison de développement
  - Types de données
  - DevEndpoint structure
  - DevEndpointCustomLibraries structure
  - Opérations
  - CreateDevEndpoint action (Python : create_dev_endpoint)
  - UpdateDevEndpoint action (Python : update_dev_endpoint)
  - DeleteDevEndpoint action (Python : delete_dev_endpoint)
  - GetDevEndpoint action (Python : get_dev_endpoint)
  - GetDevEndpoints action (Python : get_dev_endpoints)
  - BatchGetDevEndpoints action (Python : batch_get_dev_endpoints)
  - ListDevEndpoints action (Python : list_dev_endpoints)

- Registre de schémas
  - Types de données
  - RegistryId structure
  - RegistryListItem structure
  - MetadataInfo structure
  - OtherMetadataValueListItem structure

- SchemaListItem structure
- SchemaVersionListItem structure
- MetadataKeyValuePair structure
- SchemaVersionErrorItem structure
- ErrorDetails structure
- SchemaVersionNumber structure
- SchemaId structure
- Opérations
- CreateRegistry action (Python : create_registry)
- CreateSchema action (Python : créer_schéma)
- GetSchema action (Python : get_schema)
- ListSchemaVersions action (Python : list_schema_versions)
- GetSchemaVersion action (Python : get_schema_version)
- GetSchemaVersionsDiff action (Python : get_schema_versions_diff)
- ListRegistries action (Python : list_registries)
- ListSchemas action (Python : list_schemas)
- RegisterSchemaVersion action (Python : register_schema_version)
- UpdateSchema action (Python : update_schema)
- CheckSchemaVersionValidity action (Python : check_schema_version_validity)
- UpdateRegistry action (Python : update_registry)
- GetSchemaByDefinition action (Python : get_schema_by_definition)
- GetRegistry action (Python : get_registry)
- PutSchemaVersionMetadata action (Python : put_schema_version_metadata)
- QuerySchemaVersionMetadata action (Python : query_schema_version_metadata)
- RemoveSchemaVersionMetadata action (Python : remove_schema_version_metadata)
- DeleteRegistry action (Python : supprimer_registre)
- DeleteSchema action (Python : supprimer_schéma)
- DeleteSchemaVersions action (Python : delete_schema_versions)

- Flux de travail
  - Types de données

- JobNodeDetails structure
- CrawlerNodeDetails structure
- TriggerNodeDetails structure
- Structure Crawl
- Structure de nœud
- Structure Edge
- Structure de flux de travail
- WorkflowGraph structure
- WorkflowRun structure
- WorkflowRunStatistics structure
- StartingEventBatchCondition structure
- Structure du plan
- BlueprintDetails structure
- LastActiveDefinition structure
- BlueprintRun structure
- Opérations
- CreateWorkflow action (Python : create_workflow)
- UpdateWorkflow action (Python : update_workflow)
- DeleteWorkflow action (Python : delete_workflow)
- GetWorkflow action (Python : get_workflow)
- ListWorkflows action (Python : list_workflows)
- BatchGetWorkflows action (Python : batch_get_workflows)
- GetWorkflowRun action (Python : get_workflow_run)
- GetWorkflowRuns action (Python : get_workflow_runs)
- GetWorkflowRunProperties action (Python : get_workflow_run_properties)
- PutWorkflowRunProperties action (Python : put_workflow_run_properties)
- CreateBlueprint action (Python : créer_blueprint)
- UpdateBlueprint action (Python : update_blueprint)
- DeleteBlueprint action (Python : delete_blueprint)
- ListBlueprints action (Python : list_blueprints)

- BatchGetBlueprints action (Python : batch_get_blueprints)
- StartBlueprintRun action (Python : start_blueprint_run)
- GetBlueprintRun action (Python : get_blueprint_run)
- GetBlueprintRuns action (Python : get_blueprint_runs)
- StartWorkflowRun action (Python : start_workflow_run)
- StopWorkflowRun action (Python : stop_workflow_run)
- ResumeWorkflowRun action (Python : resume_workflow_run)
- Profils d'utilisation
  - Types de données
  - ProfileConfiguration structure
  - ConfigurationObject structure
  - UsageProfileDefinition structure
  - Opérations
  - CreateUsageProfile action (Python : create_usage_profile)
  - GetUsageProfile action (Python : get_usage_profile)
  - UpdateUsageProfile action (Python : update_usage_profile)
  - DeleteUsageProfile action (Python : delete_usage_profile)
  - ListUsageProfiles action (Python : list_usage_profiles)
- API de Machine Learning
  - Types de données
  - TransformParameters structure
  - EvaluationMetrics structure
  - MLTransform structure
  - FindMatchesParameters structure
  - FindMatchesMetrics structure
  - ConfusionMatrix structure
  - GlueTable structure
  - TaskRun structure
  - TransformFilterCriteria structure
  - TransformSortCriteria structure

- TaskRunFilterCriteria structure
- TaskRunSortCriteria structure
- TaskRunProperties structure
- FindMatchesTaskRunProperties structure
- ImportLabelsTaskRunProperties structure
- ExportLabelsTaskRunProperties structure
- LabelingSetGenerationTaskRunProperties structure
- SchemaColumn structure
- TransformEncryption structure
- MLUserDataEncryption structure
- ColumnImportance structure
- Opérations
- Créer une MLTransform action (Python : create_ml_transform)
- MLTransform Action de mise à jour (Python : update_ml_transform)
- MLTransform Action de suppression (Python : delete_ml_transform)
- Obtenir une MLTransform action (Python : get_ml_transform)
- Obtenir une MLTransforms action (Python : get_ml_transform)
- MLTransforms Action de liste (Python : list_ml_transform)
- Lancer MLEvaluation TaskRun l'action (Python : start_ml_evaluation_task_run)
- Lancer MLLabeling SetGenerationTaskRun l'action (Python : start_ml_labeling_set_generation_task_run)
- Action Get MLTask Run (Python : get_ml_task_run)
- Action Get MLTask Runs (Python : get_ml_task_runs)
- Annuler l'action MLTask Exécuter (Python : cancel_ml_task_run)
- StartExportLabelsTaskRun action (Python : start_export_labels_task_run)
- StartImportLabelsTaskRun action (Python : start_import_labels_task_run)

- API Qualité des données
  - Types de données
  - DataSource structure
  - DataQualityRulesetListDetails structure

- DataQualityTargetTable structure
- DataQualityRulesetEvaluationRunDescription structure
- DataQualityRulesetEvaluationRunFilter structure
- DataQualityEvaluationRunAdditionalRunOptions structure
- DataQualityRuleRecommendationRunDescription structure
- DataQualityRuleRecommendationRunFilter structure
- DataQualityResult structure
- DataQualityAnalyzerResult structure
- DataQualityObservation structure
- MetricBasedObservation structure
- DataQualityMetricValues structure
- DataQualityRuleResult structure
- DataQualityResultDescription structure
- DataQualityResultFilterCriteria structure
- DataQualityRulesetFilterCriteria structure
- DataQualityAggregatedMetrics structure
- StatisticAnnotation structure
- TimestampedInclusionAnnotation structure
- AnnotationError structure
- DatapointInclusionAnnotation structure
- StatisticSummaryList liste
- StatisticSummary structure
- RunIdentifier structure
- StatisticModelResult structure
- DataQualityGlueTable structure
- Opérations
- StartDataQualityRulesetEvaluationRun action (Python : start_data_quality_ruleset_evaluation_run)
- CancelDataQualityRulesetEvaluationRun action (Python : cancel_data_quality_ruleset_evaluation_run)
- GetDataQualityRulesetEvaluationRun action (Python : get_data_quality_ruleset_evaluation_run)

- ListDataQualityRulesetEvaluationRuns action (Python : list_data_quality_ruleset_evaluation_runs)
- StartDataQualityRuleRecommendationRun action (Python : start_data_quality_rule_recommendation_run)
- CancelDataQualityRuleRecommendationRun action (Python : cancel_data_quality_rule_recommendation_run)
- GetDataQualityRuleRecommendationRun action (Python : get_data_quality_rule_recommendation_run)
- ListDataQualityRuleRecommendationRuns action (Python : list_data_quality_rule_recommendation_runs)
- GetDataQualityResult action (Python : get_data_quality_result)
- BatchGetDataQualityResult action (Python : batch_get_data_quality_result)
- ListDataQualityResults action (Python : list_data_quality_results)
- CreateDataQualityRuleset action (Python : create_data_quality_ruleset)
- DeleteDataQualityRuleset action (Python : delete_data_quality_ruleset)
- GetDataQualityRuleset action (Python : get_data_quality_ruleset)
- ListDataQualityRulesets action (Python : list_data_quality_rulesets)
- UpdateDataQualityRuleset action (Python : update_data_quality_ruleset)
- ListDataQualityStatistics action (Python : list_data_quality_statistics)
- TimestampFilter structure
- CreateDataQualityRulesetRequest structure
- GetDataQualityRulesetResponse structure
- GetDataQualityResultResponse structure
- StartDataQualityRuleRecommendationRunRequest structure
- GetDataQualityRuleRecommendationRunResponse structure
- BatchPutDataQualityStatisticAnnotation action (Python : batch_put_data_quality_statistic_annotation)
- GetDataQualityModel action (Python : get_data_quality_model)
- GetDataQualityModelResult action (Python : get_data_quality_model_result)
- ListDataQualityStatisticAnnotations action (Python : list_data_quality_statistic_annotations)
- PutDataQualityProfileAnnotation action (Python : put_data_quality_profile_annotation)

- API Détection des données sensibles
  - Types de données
  - CustomEntityType structure
  - Opérations
  - CreateCustomEntityType action (Python : create_custom_entity_type)
  - DeleteCustomEntityType action (Python : delete_custom_entity_type)
  - GetCustomEntityType action (Python : get_custom_entity_type)
  - BatchGetCustomEntityTypes action (Python : batch_get_custom_entity_types)
  - ListCustomEntityTypes action (Python : list_custom_entity_types)
- Marquage APIs AWS Glue
  - Types de données
  - Structure Tag
  - Opérations
  - TagResource action (Python : tag_resource)
  - UntagResource action (Python : untag_resource)
  - GetTags action (Python : get_tags)
- Types de données courants
  - Structure de balise
  - DecimalNumber structure
  - ErrorDetail structure
  - PropertyPredicate structure
  - ResourceUri structure
  - ColumnStatistics structure
  - ColumnStatisticsError structure
  - ColumnError structure
  - ColumnStatisticsData structure
  - BooleanColumnStatisticsData structure
  - DateColumnStatisticsData structure
  - DecimalColumnStatisticsData structure
  - DoubleColumnStatisticsData structure

- LongColumnStatisticsData structure
- StringColumnStatisticsData structure
- BinaryColumnStatisticsData structure
- Modèles de chaîne

- Exceptions
  - AccessDeniedException structure
  - AlreadyExistsException structure
  - ConcurrentModificationException structure
  - ConcurrentRunsExceededException structure
  - CrawlerNotRunningException structure
  - CrawlerRunningException structure
  - CrawlerStoppingException structure
  - EntityNotFoundException structure
  - FederationSourceException structure
  - FederationSourceRetryableException structure
  - GlueEncryptionException structure
  - IdempotentParameterMismatchException structure
  - IllegalWorkflowStateException structure
  - InternalServiceException structure
  - InvalidExecutionEngineException structure
  - InvalidInputException structure
  - InvalidStateException structure
  - InvalidTaskStatusTransitionException structure
  - JobDefinitionErrorException structure
  - JobRunInTerminalStateException structure
  - JobRunInvalidStateTransitionException structure
  - JobRunNotInTerminalStateException structure
  - LateRunnerException structure
  - NoScheduleException structure
  - OperationTimeoutException structure

- ResourceNotReadyException structure
- ResourceNumberLimitExceededException structure
- SchedulerNotRunningException structure
- SchedulerRunningException structure
- SchedulerTransitioningException structure
- UnrecognizedRunnerException structure
- ValidationException structure
- VersionMismatchException structure

Sécurité APIs dans AWS Glue

L'API de sécurité décrit les types de données de sécurité et l'API associée à la sécurité dans AWS Glue.

Types de données

- DataCatalogEncryptionSettings structure
- EncryptionAtRest structure
- ConnectionPasswordEncryption structure
- EncryptionConfiguration structure
- Structure S3Encryption
- CloudWatchEncryption structure
- JobBookmarksEncryption structure
- SecurityConfiguration structure
- GluePolicy structure
- DataQualityEncryption structure

DataCatalogEncryptionSettings structure

Contient des informations de configuration sur le maintien de la sécurité du catalogue de données.

Champs

- `EncryptionAtRest` : un objet EncryptionAtRest.

Spécifie la encryption-at-rest configuration du catalogue de données.

- `ConnectionPasswordEncryption` – Un objet ConnectionPasswordEncryption.

  Lorsque la protection par mot de passe de connexion est activée, le Catalogue de données utilise une clé fournie par le client pour chiffrer le mot de passe dans le cadre de `CreateConnection` ou `UpdateConnection` et le stocke dans le champ `ENCRYPTED_PASSWORD` des propriétés de connexion. Vous pouvez activer le chiffrage de catalogue ou uniquement le chiffrage de mot de passe.

EncryptionAtRest structure

Spécifie la encryption-at-rest configuration du catalogue de données.

Champs

- `CatalogEncryptionMode` – Obligatoire : Chaîne UTF-8 (valeurs valides : `DISABLED` | `SSE-KMS="SSEKMS"` | `SSE-KMS-WITH-SERVICE-ROLE="SSEKMSWITHSERVICEROLE"`).

  encryption-at-restMode de chiffrement des données du catalogue de données.

- `SseAwsKmsKeyId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID de la AWS KMS clé à utiliser pour le chiffrement au repos.

- `CatalogEncryptionServiceRole` – Chaîne UTF-8, correspondant au Custom string pattern #53.

  Rôle censé chiffrer et déchiffrer les objets du catalogue de données pour le compte de l'appelant. AWS Glue

ConnectionPasswordEncryption structure

La structure des données utilisée par le Catalogue de données pour chiffrer le mot de passe dans le cadre de `CreateConnection` ou `UpdateConnection` et le stocker dans le champ `ENCRYPTED_PASSWORD` des propriétés de connexion. Vous pouvez activer le chiffrage de catalogue ou uniquement le chiffrage de mot de passe.

Lorsqu'une `CreationConnection` demande contenant un mot de passe arrive, le catalogue de données chiffre d'abord le mot de passe à l'aide de votre AWS KMS clé. Il chiffre de nouveau l'objet de connexion entier si le chiffrage de catalogue est également activé.

Ce chiffrement nécessite que vous définissiez des autorisations AWS KMS clés pour activer ou restreindre l'accès à la clé de mot de passe conformément à vos exigences de sécurité. Par exemple, vous pourriez vouloir autoriser uniquement des administrateurs à disposer des autorisations de déchiffrement de la clé de mot de passe.

Champs

- `ReturnConnectionPasswordEncrypted` – obligatoire : booléen.

  Lorsque le signalement `ReturnConnectionPasswordEncrypted` est réglé sur « vrai », les mots de passe restent chiffrés dans les réponses de `GetConnection` et de `GetConnections`. Ce chiffrage prend effet indépendamment du chiffrage de catalogue.
- `AwsKmsKeyId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  AWS KMS Clé utilisée pour chiffrer le mot de passe de connexion.

  Si la protection par mot de passe de connexion est activée, l'`CreateConnection`appelant `UpdateConnection` doit au moins `kms:Encrypt` obtenir une autorisation sur la AWS KMS clé spécifiée pour chiffrer les mots de passe avant de les stocker dans le catalogue de données.

  Vous pouvez régler l'autorisation de déchiffrage pour activer ou restreindre l'accès à la clé de mot de passe selon vos exigences de sécurité.

EncryptionConfiguration structure

Spécifie une configuration de chiffrement.

Champs

- `S3Encryption` : un tableau d'objets S3Encryption.

  Configuration de chiffrement pour les données Amazon Simple Storage Service (Amazon S3).
- `CloudWatchEncryption` – Un objet CloudWatchEncryption.

  Configuration du chiffrement pour Amazon CloudWatch.

- `JobBookmarksEncryption` – Un objet JobBookmarksEncryption.

  Configuration de chiffrement pour les signets de tâche.
- `DataQualityEncryption` – Un objet DataQualityEncryption.

  Configuration du chiffrement pour les actifs de qualité des AWS Glue données.

Structure S3Encryption

Spécifie la façon dont les données Amazon Simple Storage Service (Amazon S3) doivent être chiffrées.

Champs

- `S3EncryptionMode` – Chaîne UTF-8 (valeurs valides : `DISABLED` | `SSE-KMS="SSEKMS"` | `SSE-S3="SSES3"`).

  Mode de chiffrement à utiliser pour les données Amazon S3.
- `KmsKeyArn` – Chaîne UTF-8, correspondant au Custom string pattern #42.

  L'Amazon Resource Name (ARN) de la clé KMS à utiliser pour chiffrer les données.

CloudWatchEncryption structure

Spécifie la manière dont CloudWatch les données Amazon doivent être cryptées.

Champs

- `CloudWatchEncryptionMode` – Chaîne UTF-8 (valeurs valides : `DISABLED` | `SSE-KMS="SSEKMS"`).

  Mode de chiffrement à utiliser pour les CloudWatch données.
- `KmsKeyArn` – Chaîne UTF-8, correspondant au Custom string pattern #42.

  L'Amazon Resource Name (ARN) de la clé KMS à utiliser pour chiffrer les données.

JobBookmarksEncryption structure

Spécifie la manière dont les données de signet de tâche doivent être chiffrées.

Champs

- `JobBookmarksEncryptionMode` – Chaîne UTF-8 (valeurs valides : `DISABLED` | `CSE-KMS="CSEKMS"`).

  Mode de chiffrement à utiliser pour les données de signet de tâche.
- `KmsKeyArn` – Chaîne UTF-8, correspondant au Custom string pattern #42.

  L'Amazon Resource Name (ARN) de la clé KMS à utiliser pour chiffrer les données.

SecurityConfiguration structure

Spécifie une configuration de sécurité.

Champs

- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la configuration de sécurité.
- `CreatedTimeStamp` – Horodatage.

  Horodatage de la création de la configuration de sécurité.
- `EncryptionConfiguration` : un objet EncryptionConfiguration.

  Configuration de chiffrement associée à cette configuration de sécurité.

GluePolicy structure

Structure pour le renvoi d'une politique de ressources.

Champs

- `PolicyInJson` — Chaîne UTF-8, d'une longueur minimale de 2 octets.

  Contient le document de politique demandé, au format JSON.
- `PolicyHash` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

Contient la valeur de hachage associée à cette politique.

- `CreateTime` – Horodatage.

  Date et heure de création de la politique.

- `UpdateTime` – Horodatage.

  Date et heure de dernière mise à jour de la politique.

DataQualityEncryption structure

Spécifie la manière dont les ressources de qualité des données de votre compte doivent être chiffrées.

Champs

- `DataQualityEncryptionMode` – Chaîne UTF-8 (valeurs valides : `DISABLED` | `SSE-KMS="SSEKMS"`).

  Mode de chiffrement à utiliser pour chiffrer les ressources de qualité des données. Ces ressources incluent des ensembles de règles de qualité des données, des résultats, des statistiques, des modèles de détection des anomalies et des observations.

  Les valeurs valides sont `SSEKMS` pour le chiffrement à l'aide d'une clé KMS gérée par le client, ou `DISABLED`.

- `KmsKeyArn` – Chaîne UTF-8, correspondant au Custom string pattern #42.

  L'Amazon Resource Name (ARN) de la clé KMS à utiliser pour chiffrer les données.

Opérations

- GetDataCatalogEncryptionSettings action (Python : get_data_catalog_encryption_settings)
- PutDataCatalogEncryptionSettings action (Python : put_data_catalog_encryption_settings)
- PutResourcePolicy action (Python : put_resource_policy)
- GetResourcePolicy action (Python : get_resource_policy)
- DeleteResourcePolicy action (Python : delete_resource_policy)
- CreateSecurityConfiguration action (Python : create_security_configuration)

- DeleteSecurityConfiguration action (Python : delete_security_configuration)
- GetSecurityConfiguration action (Python : get_security_configuration)
- GetSecurityConfigurations action (Python : get_security_configurations)
- GetResourcePolicies action (Python : get_resource_policies)

GetDataCatalogEncryptionSettings action (Python : get_data_catalog_encryption_settings)

Récupère la configuration de sécurité d'un catalogue spécifié.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données pour lequel extraire la configuration de sécurité. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.

Réponse

- `DataCatalogEncryptionSettings` – Un objet DataCatalogEncryptionSettings.

  Configuration de sécurité demandée.

Erreurs

- `InternalServiceException`
- `InvalidInputException`
- `OperationTimeoutException`

PutDataCatalogEncryptionSettings action (Python : put_data_catalog_encryption_settings)

Définit la configuration de sécurité d'un catalogue spécifié. Une fois la configuration définie, le chiffrement spécifié est appliqué à chaque écriture ultérieure dans le catalogue.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données pour lequel définir la configuration de sécurité. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.
- `DataCatalogEncryptionSettings` – Obligatoire : un objet DataCatalogEncryptionSettings.

  Configuration de sécurité à définir.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `InternalServiceException`
- `InvalidInputException`
- `OperationTimeoutException`

PutResourcePolicy action (Python : put_resource_policy)

Définit la politique de ressources du catalogue de données pour le contrôle d'accès.

Demande

- `PolicyInJson` – Obligatoire : Chaîne UTF-8, d'une longueur minimale de 2 octets.

  Contient le document de politique à définir, au format JSON.
- `ResourceArn` – chaîne UTF-8, d'une longueur comprise entre 1 et 10 240 octets, correspondant à Custom string pattern #50.

  Ne pas utiliser. Pour utilisation interne uniquement.
- `PolicyHashCondition` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

La valeur de hachage renvoyée lors de la définition de la politique précédente à l'aide de `PutResourcePolicy`. Son objectif est d'empêcher les modifications simultanées d'une politique. N'utilisez pas ce paramètre si aucune politique précédente n'a été définie.

- `PolicyExistsCondition` – Chaîne UTF-8 (valeurs valides : `MUST_EXIST` | `NOT_EXIST` | `NONE`).

  La valeur `MUST_EXIST` est utilisée pour mettre à jour une politique. La valeur `NOT_EXIST` est utilisée pour créer une politique. Si une valeur `NONE` ou null est utilisée, l'appel ne dépend pas de l'existence d'une politique.

- `EnableHybrid` – Chaîne UTF-8 (valeurs valides : `TRUE` | `FALSE`).

  Si la valeur est `'TRUE'`, cela indique que vous utilisez les deux méthodes pour accorder un accès entre comptes aux ressources Data Catalog :

  - En mettant directement à jour la politique de ressources avec `PutResourePolicy`
  - En utilisant la commande Grant persmissions (Accorder des autorisations) sur la AWS Management Console.

  Doit être défini sur `'TRUE'` si vous avez déjà utilisé la console de gestion pour accorder l'accès entre comptes, sinon l'appel échoue. La valeur par défaut est 'FALSE'.

Réponse

- `PolicyHash` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Un hachage de la politique vient d'être défini. Cela doit être inclus dans un appel ultérieur qui remplace ou met à jour cette politique.

Erreurs

- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`
- `ConditionCheckFailureException`

GetResourcePolicy action (Python : get_resource_policy)

Extrait une politique de ressource spécifiée.

Demande

- `ResourceArn` – chaîne UTF-8, d'une longueur comprise entre 1 et 10 240 octets, correspondant à Custom string pattern #50.

  L'ARN de la AWS Glue ressource pour laquelle vous souhaitez récupérer la politique de ressource. Si elle n'est pas fournie, la politique de ressources Data Catalog est renvoyée. Utilisez `GetResourcePolicies` pour afficher toutes les politiques de ressources existantes. Pour plus d'informations, voir Spécification de AWS Glue la ressource ARNs.

Réponse

- `PolicyInJson` — Chaîne UTF-8, d'une longueur minimale de 2 octets.

  Contient le document de politique demandé, au format JSON.
- `PolicyHash` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Contient la valeur de hachage associée à cette politique.
- `CreateTime` – Horodatage.

  Date et heure de création de la politique.
- `UpdateTime` – Horodatage.

  Date et heure de dernière mise à jour de la politique.

Erreurs

- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`

DeleteResourcePolicy action (Python : delete_resource_policy)

Supprime une politique spécifiée.

Demande

- `PolicyHashCondition` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Valeur de hachage renvoyée lorsque cette politique a été définie.
- `ResourceArn` – chaîne UTF-8, d'une longueur comprise entre 1 et 10 240 octets, correspondant à Custom string pattern #50.

  L'ARN de la AWS Glue ressource pour la politique de ressources à supprimer.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`
- `ConditionCheckFailureException`

CreateSecurityConfiguration action (Python : create_security_configuration)

Crée une nouvelle configuration de sécurité. Une configuration de sécurité est un ensemble de paramètres de sécurité pouvant être utilisées par AWS Glue. Vous pouvez utiliser une configuration de sécurité pour chiffrer les données au repos. Pour plus d'informations sur l'utilisation des configurations de sécurité dans AWS Glue, consultez la section Chiffrement des données écrites par les robots d'exploration, les tâches et les points de terminaison de développement.

Demande

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la nouvelle configuration de sécurité.
- `EncryptionConfiguration` – Obligatoire : un objet EncryptionConfiguration.

  Configuration de chiffrement associée à la nouvelle configuration de sécurité.

Réponse

- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom affecté à la nouvelle configuration de sécurité.
- `CreatedTimestamp` – Horodatage.

  Horodatage de la création de la nouvelle configuration de sécurité.

Erreurs

- `AlreadyExistsException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`

DeleteSecurityConfiguration action (Python : delete_security_configuration)

Supprime une configuration de sécurité spécifiée.

Demande

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la configuration de sécurité à supprimer.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

GetSecurityConfiguration action (Python : get_security_configuration)

Extrait une configuration de sécurité spécifiée.

Demande

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la configuration de sécurité à récupérer.

Réponse

- `SecurityConfiguration` – Un objet SecurityConfiguration.

  Configuration de sécurité demandée.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

GetSecurityConfigurations action (Python : get_security_configurations)

Extrait une liste de toutes les configurations de sécurité.

Demande

- `MaxResults` – Nombre (entier), compris entre 1 et 1 000.

  Nombre maximal de résultats à renvoyer.
- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, s'il s'agit d'un appel de continuation.

Réponse

- `SecurityConfigurations` : un tableau d'objets SecurityConfiguration.

  Liste de configurations de sécurité.
- `NextToken` – Chaîne UTF-8.

  Jeton de liaison, s'il y a des configurations de sécurité supplémentaires à renvoyer.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

GetResourcePolicies action (Python : get_resource_policies)

Récupère les politiques de ressources définies pour les ressources individuelles AWS Resource Access Manager lors de l'octroi d'autorisations entre comptes. Récupère également la politique de ressources Data Catalog.

Si vous avez activé le chiffrement des métadonnées dans les paramètres du catalogue de données et que vous n'êtes pas autorisé à utiliser la AWS KMS clé, l'opération ne peut pas renvoyer la politique de ressources du catalogue de données.

Demande

- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, s'il s'agit d'une requête de continuation.
- `MaxResults` – Nombre (entier), compris entre 1 et 1 000.

  La taille maximale d'une liste à renvoyer.

Réponse

- `GetResourcePoliciesResponseList` : un tableau d'objets GluePolicy.

  Liste répertoriant chacune des politiques de ressources et politique de ressources au niveau du compte.
- `NextToken` – Chaîne UTF-8.

  Un jeton de continuation, si la liste renvoyée ne contient pas la dernière politique de ressources disponible.

Erreurs

- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`
- `GlueEncryptionException`

API des objets du catalogue

L'API des objets du catalogue décrit les types de données et l'API liés à l'utilisation des catalogues dans AWS Glue.

Rubriques

- API de catalogues
- API d'une base de données
- API de table

- API de partition
- API de connexions
- API de fonction définie par l'utilisateur
- Importation d'un Athena catalogue dans AWS Glue

API de catalogues

L'API Catalogs décrit comment créer, supprimer, localiser, mettre à jour et répertorier les catalogues. APIs

Types de données

- Structure du catalogue
- CatalogInput structure
- TargetRedshiftCatalog structure
- CatalogProperties structure
- CatalogPropertiesOutput structure
- DataLakeAccessProperties structure
- IcebergOptimizationProperties structure
- DataLakeAccessPropertiesOutput structure
- IcebergOptimizationPropertiesOutput structure
- FederatedCatalog structure

Structure du catalogue

L'objet de catalogue représente un regroupement logique de bases de données dans le catalogue de AWS Glue données ou dans une source fédérée. Vous pouvez désormais créer un catalogue fédéré Redshift ou un catalogue contenant des liens de ressources vers des bases de données Redshift d'un autre compte ou d'une autre région.

Champs

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

L'ID du catalogue. Pour autoriser l'accès au catalogue par défaut, ce champ ne doit pas être fourni.

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 64 octets, correspondant au Custom string pattern #25.

  Le nom du catalogue. Il ne peut pas être identique à l'identifiant du compte.

- `ResourceArn` – Chaîne UTF-8.

  Amazon Resource Name (ARN) affecté à la ressource du catalogue.

- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au modèle de chaîne à plusieurs lignes d'adresse URI. Une description du catalogue de données.

- `Parameters` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne de clé, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 512 000 octets.

  Un tableau de mappage de paires clé-valeur qui définissent les paramètres et les propriétés du catalogue.

- `CreateTime` – Horodatage.

  Heure à laquelle le catalogue a été créé.

- `UpdateTime` – Horodatage.

  Heure de dernière mise à jour du catalogue.

- `TargetRedshiftCatalog` – Un objet TargetRedshiftCatalog.

  Objet `TargetRedshiftCatalog` qui décrit un catalogue cible pour la liaison de ressources de la base de données.

- `FederatedCatalog` – Un objet FederatedCatalog.

  Un `FederatedCatalog` objet qui pointe vers une entité en dehors du catalogue de AWS Glue données.

- `CatalogProperties` – Un objet CatalogPropertiesOutput.

Objet `CatalogProperties` qui spécifie les propriétés d'accès au lac de données et d'autres propriétés personnalisées.

- `CreateTableDefaultPermissions` : un tableau d'objets PrincipalPermissions.

  Tableau d'objets `PrincipalPermissions`. Crée un ensemble d'autorisations par défaut sur la/les table(s) pour les principaux. Utilisé par AWS Lake Formation. Non utilisé dans le cours normal des AWS Glue opérations.
- `CreateDatabaseDefaultPermissions` : un tableau d'objets PrincipalPermissions.

  Tableau d'objets `PrincipalPermissions`. Crée un ensemble d'autorisations par défaut sur la/les base(s) de données pour les principaux. Utilisé par AWS Lake Formation. Non utilisé dans le cours normal des AWS Glue opérations.
- `AllowFullTableExternalDataAccess` – Chaîne UTF-8 (valeurs valides : `True` | `False`).

  Permet à des moteurs tiers d'accéder aux données Amazon S3 situées dans des emplacements enregistrés auprès de Lake Formation.

CatalogInput structure

Structure qui décrit les propriétés du catalogue.

Champs

- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au modèle de chaîne à plusieurs lignes d'adresse URI. Une description du catalogue de données.
- `FederatedCatalog` – Un objet FederatedCatalog.

  Un objet `FederatedCatalog`. `FederatedCatalog`Structure qui fait référence à une entité extérieure au catalogue de AWS Glue données, par exemple une base de données Redshift.
- `Parameters` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne de clé, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 512 000 octets.

Un tableau de mappage de paires clé-valeur qui définissent les paramètres et les propriétés du catalogue.

- `TargetRedshiftCatalog` – Un objet TargetRedshiftCatalog.

  Objet `TargetRedshiftCatalog` qui décrit un catalogue cible pour la liaison de ressources.

- `CatalogProperties` – Un objet CatalogProperties.

  Objet `CatalogProperties` qui spécifie les propriétés d'accès au lac de données et d'autres propriétés personnalisées.

- `CreateTableDefaultPermissions` : un tableau d'objets PrincipalPermissions.

  Tableau d'objets `PrincipalPermissions`. Crée un ensemble d'autorisations par défaut sur la/les table(s) pour les principaux. Utilisé par AWS Lake Formation. Il doit généralement être défini explicitement sous forme de liste vide.

- `CreateDatabaseDefaultPermissions` : un tableau d'objets PrincipalPermissions.

  Tableau d'objets `PrincipalPermissions`. Crée un ensemble d'autorisations par défaut sur la/les base(s) de données pour les principaux. Utilisé par AWS Lake Formation. Il doit généralement être défini explicitement sous forme de liste vide.

- `AllowFullTableExternalDataAccess` – Chaîne UTF-8 (valeurs valides : `True` | `False`).

  Permet à des moteurs tiers d'accéder aux données Amazon S3 situées dans des emplacements enregistrés auprès de Lake Formation.

TargetRedshiftCatalog structure

Structure qui décrit un catalogue cible pour la liaison de ressources.

Champs

- `CatalogArn` – Obligatoire : chaîne UTF-8.

  Amazon Resource Name (ARN) de la ressource de catalogue.

CatalogProperties structure

Structure qui spécifie les propriétés d'accès au lac de données et d'autres propriétés personnalisées.

Champs

- `DataLakeAccessProperties` – Un objet DataLakeAccessProperties.

  `DataLakeAccessProperties`Objet qui spécifie les propriétés permettant de configurer l'accès au lac de données pour la ressource de votre catalogue dans le catalogue de AWS Glue données.

- `IcebergOptimizationProperties` : un objet IcebergOptimizationProperties.

  Structure qui spécifie les propriétés d'optimisation de table Iceberg pour le catalogue. Cela inclut la configuration des opérations de compactage, de conservation et de suppression de fichiers orphelins qui peuvent être appliquées aux tables Iceberg de ce catalogue.

- `CustomProperties` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne de clé, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 512 000 octets.

  Propriétés clé-valeur supplémentaires pour le catalogue, telles que les optimisations des statistiques de colonne.

CatalogPropertiesOutput structure

Attributs de propriété qui incluent les propriétés de configuration pour la ressource du catalogue.

Champs

- `DataLakeAccessProperties` – Un objet DataLakeAccessPropertiesOutput.

  `DataLakeAccessProperties`Objet doté de propriétés d'entrée permettant de configurer l'accès au lac de données pour les ressources de votre catalogue dans le catalogue de AWS Glue données.

- `IcebergOptimizationProperties` : un objet IcebergOptimizationPropertiesOutput.

  Objet `IcebergOptimizationPropertiesOutput` qui spécifie les paramètres d'optimisation des tables Iceberg pour le catalogue, notamment les configurations pour les opérations de compactage, de conservation et de suppression de fichiers orphelins.

- `CustomProperties` – Tableau de mappage de paires valeur-clé.

Chaque clé est une chaîne de clé, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 512 000 octets.

Propriétés clé-valeur supplémentaires pour le catalogue, telles que les optimisations des statistiques de colonne.

DataLakeAccessProperties structure

Entrez des propriétés pour configurer l'accès au lac de données pour la ressource de votre catalogue dans le catalogue de AWS Glue données.

Champs

- `DataLakeAccess` : booléen.

  Active ou désactive l'accès aux lacs de données pour les applications Apache Spark qui accèdent aux bases de données Amazon Redshift dans le catalogue de données à partir d'un moteur autre que Redshift, tel qu'Amazon Athena, Amazon EMR ou ETL. AWS Glue
- `DataTransferRole` – Chaîne UTF-8, correspondant au Custom string pattern #53.

  Rôle qui sera assumé par le transfert AWS Glue des données into/out du bucket intermédiaire lors d'une requête.
- `KmsKey` – Chaîne UTF-8.

  Clé de chiffrement qui sera utilisée pour le compartiment intermédiaire qui sera créé avec le catalogue.
- `CatalogType` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Spécifie un type de catalogue fédéré pour la ressource de catalogue native. Le type `aws:redshift` est actuellement pris en charge.

IcebergOptimizationProperties structure

Structure qui spécifie les propriétés d'optimisation des tables Iceberg pour le catalogue, y compris les configurations pour les opérations de compactage, de conservation et de suppression de fichiers orphelins.

Champs

- `RoleArn` – Chaîne UTF-8, correspondant au Custom string pattern #53.

  Amazon Resource Name (ARN) du rôle IAM qui sera endossé pour effectuer les opérations d'optimisation des tables Iceberg.

- `Compaction` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne de clé, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 512 000 octets.

  Carte de paires clé-valeur qui spécifient les paramètres de configuration pour les opérations de compactage des tables Iceberg, qui optimisent la disposition des fichiers de données afin d'améliorer les performances des requêtes.

- `Retention` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne de clé, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 512 000 octets.

  Carte de paires clé-valeur qui spécifient les paramètres de configuration pour les opérations de conservation des tables Iceberg, qui gèrent le cycle de vie des instantanés de tables afin de contrôler les coûts de stockage.

- `OrphanFileDeletion` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne de clé, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 512 000 octets.

Carte de paires clé-valeur qui spécifient les paramètres de configuration pour les opérations de suppression de fichiers orphelins Iceberg, qui identifient et suppriment les fichiers qui ne sont plus référencés par les métadonnées de table.

DataLakeAccessPropertiesOutput structure

Les propriétés de sortie de la configuration d'accès au lac de données pour votre ressource de catalogue dans le catalogue de AWS Glue données.

Champs

- `DataLakeAccess` : booléen.

  Active ou désactive l'accès au lac de données pour les applications Apache Spark qui accèdent aux bases de données Amazon Redshift dans le catalogue de données.
- `DataTransferRole` – Chaîne UTF-8, correspondant au Custom string pattern #53.

  Rôle qui sera assumé par le transfert AWS Glue des données into/out du bucket intermédiaire lors d'une requête.
- `KmsKey` – Chaîne UTF-8.

  Clé de chiffrement qui sera utilisée pour le compartiment intermédiaire qui sera créé avec le catalogue.
- `ManagedWorkgroupName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du calcul Redshift sans serveur géré qui est créé pour votre ressource de catalogue.
- `ManagedWorkgroupStatus` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  État du calcul Redshift sans serveur géré.
- `RedshiftDatabaseName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la ressource de base de données Redshift par défaut dans le calcul géré.
- `StatusMessage` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

Message qui fournit des informations plus détaillées sur l'état du groupe de travail géré.

- `CatalogType` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Spécifie un type de catalogue fédéré pour la ressource de catalogue native. Le type `aws:redshift` est actuellement pris en charge.

IcebergOptimizationPropertiesOutput structure

Structure qui contient les propriétés de sortie de la configuration d'optimisation des tables Iceberg pour votre ressource de catalogue dans le catalogue de AWS Glue données.

Champs

- `RoleArn` – Chaîne UTF-8, correspondant au Custom string pattern #53.

  Amazon Resource Name (ARN) du rôle IAM qui est utilisé pour effectuer les opérations d'optimisation des tables Iceberg.

- `Compaction` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne de clé, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 512 000 octets.

  Carte de paires clé-valeur qui spécifient les paramètres de configuration pour les opérations de compactage des tables Iceberg, qui optimisent la disposition des fichiers de données afin d'améliorer les performances des requêtes.

- `Retention` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne de clé, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 512 000 octets.

  Carte de paires clé-valeur qui spécifient les paramètres de configuration pour les opérations de conservation des tables Iceberg, qui gèrent le cycle de vie des instantanés de tables afin de contrôler les coûts de stockage.

- `OrphanFileDeletion` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne de clé, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 512 000 octets.

  Carte de paires clé-valeur qui spécifient les paramètres de configuration pour les opérations de suppression de fichiers orphelins Iceberg, qui identifient et suppriment les fichiers qui ne sont plus référencés par les métadonnées de table.
- `LastUpdatedTime` – Horodatage.

  Horodatage de la dernière mise à jour des propriétés d'optimisation Iceberg.

FederatedCatalog structure

Catalogue qui pointe vers une entité extérieure au catalogue AWS Glue de données.

Champs

- `Identifier` – Chaîne UTF-8, d'une longueur comprise entre 1 et 512 octets, correspondant au Single-line string pattern.

  Identifiant unique pour le catalogue fédéré.
- `ConnectionName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la connexion à une source de données externe, par exemple un catalogue fédéré par Redshift.
- `ConnectionType` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Type de connexion utilisé pour accéder au catalogue fédéré, spécifiant le protocole ou la méthode de connexion à la source de données externe.

Opérations

- CreateCatalog action (Python : create_catalog)
- UpdateCatalog action (Python : update_catalog)

- DeleteCatalog action (Python : supprimer_catalogue)
- GetCatalog action (Python : get_catalog)
- GetCatalogs action (Python : get_catalogs)

CreateCatalog action (Python : create_catalog)

Crée un nouveau catalogue dans le catalogue AWS Glue de données.

Demande

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 64 octets, correspondant au Custom string pattern #25.

  Nom du catalogue à créer.
- `CatalogInput` – Obligatoire : un objet CatalogInput.

  Objet `CatalogInput` qui définit les métadonnées du catalogue.
- `Tags` – Tableau de mappage de paires clé-valeur, avec 50 paires au maximum.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 256 octets.

  Tableau de mappage de paires valeur-clé, avec 50 paires au maximum. Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets. Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 256 octets. Les balises que vous attribuez au catalogue.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `InvalidInputException`
- `AlreadyExistsException`
- `ResourceNumberLimitExceededException`
- `InternalServiceException`

- `OperationTimeoutException`
- `GlueEncryptionException`
- `ConcurrentModificationException`
- `AccessDeniedException`
- `EntityNotFoundException`
- `FederatedResourceAlreadyExistsException`
- `FederationSourceException`

UpdateCatalog action (Python : update_catalog)

Met à jour les propriétés d'un catalogue existant dans le catalogue de AWS Glue données.

Demande

- `CatalogId` – Obligatoire : Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  L'ID du catalogue.
- `CatalogInput` – Obligatoire : un objet CatalogInput.

  Objet `CatalogInput` spécifiant les nouvelles propriétés d'un catalogue existant.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`
- `ConcurrentModificationException`
- `AccessDeniedException`

- `FederationSourceException`

DeleteCatalog action (Python : supprimer_catalogue)

Supprime le catalogue spécifié du catalogue de AWS Glue données.

Une fois cette opération terminée, vous n'avez plus accès aux bases de données, aux tables (ni à toutes les versions de tables et partitions susceptibles d'appartenir aux tables) ni aux fonctions définies par l'utilisateur dans le catalogue supprimé. AWS Glue supprime ces ressources « orphelines » de manière asynchrone en temps opportun, à la discrétion du service.

Pour garantir la suppression immédiate de toutes les ressources connexes, avant d'appeler l'opération `DeleteCatalog`, utilisez `DeleteTableVersion` (ou `BatchDeleteTableVersion`), `DeletePartition` (ou `BatchDeletePartition`), `DeleteTable` (ou `BatchDeleteTable`) `DeleteUserDefinedFunction` et `DeleteDatabase` pour supprimer les ressources appartenant au catalogue.

Demande

- `CatalogId` – Obligatoire : Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  L'ID du catalogue.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`
- `ConcurrentModificationException`
- `AccessDeniedException`

- `FederationSourceException`

GetCatalog action (Python : get_catalog)

Nom du catalogue à extraire. Il doit être entièrement en minuscules.

Demande

- `CatalogId` – Obligatoire : Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue parent dans lequel réside le catalogue. Si aucun numéro n'est fourni, le numéro de AWS compte est utilisé par défaut.

Réponse

- `Catalog` – Un objet Catalogue.

  Un objet `Catalog`. Définition du catalogue spécifié dans le catalogue de AWS Glue données.

Erreurs

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`
- `AccessDeniedException`
- `EntityNotFoundException`
- `FederationSourceException`
- `FederationSourceRetryableException`

GetCatalogs action (Python : get_catalogs)

Récupère tous les catalogues définis dans un catalogue dans le catalogue de données AWS Glue . Pour un cas d'utilisation de catalogue fédéré par Redshift, cette opération renvoie la liste des catalogues mappés aux bases de données Redshift dans le catalogue d'espace de noms Redshift.

Demande

- `ParentCatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue parent dans lequel réside le catalogue. Si aucun numéro n'est fourni, le numéro de AWS compte est utilisé par défaut.
- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, s'il s'agit d'un appel de continuation.
- `MaxResults` – Nombre (entier), compris entre 1 et 1 000.

  Nombre maximum de catalogues renvoyés par réponse.
- `Recursive` : booléen.

  Indique s'il faut répertorier tous les catalogues de la hiérarchie des catalogues, en commençant par la valeur `ParentCatalogId`. La valeur par défaut est `false` . Lorsque `true`, tous les objets du catalogue de la hiérarchie `ParentCatalogID` sont énumérés dans la réponse.
- `IncludeRoot` : booléen.

  Indique s'il faut répertorier le catalogue par défaut dans le compte et la région dans la réponse. La valeur par défaut est `false` . Quand `true` et `ParentCatalogId = NULL | AWS Account ID`, tous les catalogues et le catalogue par défaut sont énumérés dans la réponse.

  Lorsque la valeur `ParentCatalogId` n'est pas égal à null et que cet attribut est transmis en tant que `false` ou `true`, une `InvalidInputException` est émise.

Réponse

- `CatalogList` – Obligatoire : un tableau d'objets Catalogue.

  Tableau d'objets `Catalog`. Liste d'objets `Catalog` à partir du catalogue parent spécifié.
- `NextToken` – Chaîne UTF-8.

Jeton de continuation pour la pagination de la liste des jetons renvoyés, renvoyé si le segment actuel de la liste n'est pas le dernier.

Erreurs

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`
- `AccessDeniedException`
- `EntityNotFoundException`
- `FederationSourceException`
- `FederationSourceRetryableException`

API d'une base de données

L'API de base de données décrit les types de données d'une base de données et comprend l'API permettant de créer, supprimer, localiser, mettre à jour et répertorier des bases de données.

Types de données

- Structure Database
- DatabaseInput structure
- PrincipalPermissions structure
- DataLakePrincipal structure
- DatabaseIdentifier structure
- FederatedDatabase structure

Structure Database

L'objet `Database` représente un groupement logique de tables pouvant résider dans un metastore Hive ou un SGBDR.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la base de données. Pour des raisons de compatibilité avec Hive, ce nom est converti en minuscules lors de son stockage.

- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description de la base de données.

- `LocationUri` – Identificateur de ressource uniforme (URI), d'une longueur comprise entre 1 et 1024 octets, correspondant au URI address multi-line string pattern.

  Emplacement de la base de données (par exemple, un chemin HDFS).

- `Parameters` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne de clé, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 512 000 octets.

  Ces paires clé-valeur définissent les paramètres et les propriétés de la base de données.

- `CreateTime` – Horodatage.

  Heure à laquelle la base de données de métadonnées a été créée dans le catalogue.

- `CreateTableDefaultPermissions` : un tableau d'objets PrincipalPermissions.

  Crée un ensemble d'autorisations par défaut sur le tableau pour les principaux. Utilisé par AWS Lake Formation. Non utilisé dans le cours normal des AWS Glue opérations.

- `TargetDatabase` – Un objet DatabaseIdentifier.

  Structure `DatabaseIdentifier` qui décrit une base de données cible pour la liaison de ressources.

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données dans lequel réside la base de données.

- `FederatedDatabase` – Un objet FederatedDatabase.

  Une structure `FederatedDatabase` qui fait référence à une entité extérieure à AWS Glue Data Catalog.

DatabaseInput structure

Structure utilisée pour créer ou mettre à jour une base de données.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la base de données. Pour des raisons de compatibilité avec Hive, ce nom est converti en minuscules lors de son stockage.
- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description de la base de données.
- `LocationUri` – Identificateur de ressource uniforme (URI), d'une longueur comprise entre 1 et 1024 octets, correspondant au URI address multi-line string pattern.

  Emplacement de la base de données (par exemple, un chemin HDFS).
- `Parameters` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne de clé, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 512 000 octets.

  Ces paires clé-valeur définissent les paramètres et les propriétés de la base de données.

  Ces paires clé-valeur définissent les paramètres et les propriétés de la base de données.
- `CreateTableDefaultPermissions` : un tableau d'objets PrincipalPermissions.

  Crée un ensemble d'autorisations par défaut sur le tableau pour les principaux. Utilisé par AWS Lake Formation. Non utilisé dans le cours normal des AWS Glue opérations.
- `TargetDatabase` – Un objet DatabaseIdentifier.

Structure `DatabaseIdentifier` qui décrit une base de données cible pour la liaison de ressources.

- `FederatedDatabase` – Un objet FederatedDatabase.

  Une structure `FederatedDatabase` qui fait référence à une entité extérieure à AWS Glue Data Catalog.

PrincipalPermissions structure

Autorisations accordées à un principal.

Champs

- `Principal` – Un objet DataLakePrincipal.

  Principal à qui les autorisations sont accordées.
- `Permissions` – Tableau de chaînes UTF-8.

  Autorisations accordées au principal.

DataLakePrincipal structure

Le AWS Lake Formation directeur.

Champs

- `DataLakePrincipalIdentifier` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets.

  Un identifiant pour le AWS Lake Formation principal.

DatabaseIdentifier structure

Structure qui décrit une base de données cible pour la liaison de ressources.

Champs

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

ID du catalogue de données dans lequel réside la base de données.

- `DatabaseName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la base de données du catalogue.

- `Region` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Région de la base de données cible.

FederatedDatabase structure

Une base de données qui pointe vers une entité extérieure à AWS Glue Data Catalog.

Champs

- `Identifier` – Chaîne UTF-8, d'une longueur comprise entre 1 et 512 octets, correspondant au Single-line string pattern.

  Un identifiant unique pour la base de données fédérée.

- `ConnectionName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Le nom de la connexion au métastore externe.

- `ConnectionType` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Type de connexion utilisé pour accéder à la base de données fédérée, tel que JDBC, ODBC ou d'autres protocoles de connexion pris en charge.

Opérations

- CreateDatabase action (Python : créer_base de données)
- UpdateDatabase action (Python : update_database)
- DeleteDatabase action (Python : supprimer_base de données)
- GetDatabase action (Python : get_database)

- GetDatabases action (Python : get_databases)

CreateDatabase action (Python : créer_base de données)

Crée une nouvelle base de données dans un catalogue de données.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données dans lequel créer la base de données. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.
- `DatabaseInput` – Obligatoire : un objet DatabaseInput.

  Métadonnées pour la base de données.
- `Tags` – Tableau de mappage de paires clé-valeur, avec 50 paires au maximum.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 256 octets.

  Les balises que vous attribuez à la base de données.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `InvalidInputException`
- `AlreadyExistsException`
- `ResourceNumberLimitExceededException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`
- `ConcurrentModificationException`

- `FederatedResourceAlreadyExistsException`
- `FederationSourceException`
- `FederationSourceRetryableException`

UpdateDatabase action (Python : update_database)

Met à jour une définition de base de données existante dans un catalogue de données.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données dans lequel réside la base de données de métadonnées. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.
- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la base de données à mettre à jour dans le catalogue. Pour la compatibilité Hive, ce nom est converti en minuscules.
- `DatabaseInput` – Obligatoire : un objet DatabaseInput.

  Objet `DatabaseInput` spécifiant la nouvelle définition de la base de données de métadonnées dans le catalogue.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`

- `ConcurrentModificationException`
- `FederationSourceException`
- `FederationSourceRetryableException`
- `AlreadyExistsException`

DeleteDatabase action (Python : supprimer_base de données)

Supprime une base de données spécifiée d'un catalogue de données.

> **Note**
>
> Une fois cette opération terminée, vous n'avez plus accès aux tables (ni à toutes les versions de tables et partitions susceptibles d'appartenir aux tables) ni aux fonctions définies par l'utilisateur dans la base de données supprimée. AWS Glue supprime ces ressources « orphelines » de manière asynchrone en temps opportun, à la discrétion du service. Pour garantir la suppression immédiate de toutes les ressources connexes, avant d'appeler `DeleteDatabase`, utilisez `DeleteTableVersion` ou `BatchDeleteTableVersion`, `DeletePartition` ou `BatchDeletePartition`, `DeleteUserDefinedFunction`, et `DeleteTable` ou `BatchDeleteTable` pour supprimer les ressources appartenant à la base de données.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données dans lequel réside la base de données. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la base de données à supprimer. Pour la compatibilité Hive, ce nom doit être entièrement en minuscules.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ConcurrentModificationException`
- `FederationSourceException`
- `FederationSourceRetryableException`

GetDatabase action (Python : get_database)

Extrait la définition d'une base de données spécifiée.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données dans lequel réside la base de données. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom la base de données à extraire. Pour la compatibilité Hive, ce nom doit être entièrement en minuscules.

Réponse

- `Database` – Un objet Base de données.

  Définition de la base de données spécifiée dans le catalogue de données.

Erreurs

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`
- `FederationSourceException`
- `FederationSourceRetryableException`

GetDatabases action (Python : get_databases)

Extrait toutes les bases de données définies dans un catalogue de données donnée.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données à partir duquel extraire `Databases`. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.
- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, s'il s'agit d'un appel de continuation.
- `MaxResults` – Nombre (entier), compris entre 1 et 100.

  Nombre maximum de bases de données renvoyées par réponse.
- `ResourceShareType` – Chaîne UTF-8 (valeurs valides : `FOREIGN` | `ALL` | `FEDERATED`).

  Permet de spécifier que vous souhaitez répertorier les bases de données partagées avec votre compte. Les valeurs autorisées sont `FEDERATED`, `FOREIGN` ou `ALL`.
  - Si la valeur est définie sur `FEDERATED`, la liste des bases de données fédérées (référençant une entité externe) partagées avec votre compte sera affichée.
  - Si défini sur `FOREIGN`, répertorie les bases de données partagées avec votre compte.
  - Si défini sur `ALL`, répertorie les bases de données partagées avec votre compte, ainsi que les bases de données dans votre compte local.

- `AttributesToGet` – Tableau de chaînes UTF-8.

  Spécifie les champs de base de données renvoyés par l'appel `GetDatabases`. Ce paramètre n'accepte pas une liste vide. La demande doit inclure le `NAME`.

Réponse

- `DatabaseList` – Obligatoire : un tableau d'objets Base de données.

  Liste d'objets `Database` à partir du catalogue spécifié.

- `NextToken` – Chaîne UTF-8.

  Jeton de continuation pour la pagination de la liste des jetons renvoyés, renvoyé si le segment actuel de la liste n'est pas le dernier.

Erreurs

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`
- `EntityNotFoundException`
- `FederationSourceException`
- `FederationSourceRetryableException`

API de table

L'API de tableau décrit les types de données et les opérations associés aux tableaux.

Types de données

- Structure de table
- TableInput structure
- FederatedTable structure
- Structure de colonne

- StorageDescriptor structure
- SchemaReference structure
- SerDeInfo structure
- Structure de tri
- SkewedInfo structure
- TableVersion structure
- TableError structure
- TableVersionError structure
- SortCriterion structure
- TableIdentifier structure
- KeySchemaElement structure
- PartitionIndex structure
- PartitionIndexDescriptor structure
- BackfillError structure
- IcebergInput structure
- OpenTableFormatInput structure
- ViewDefinition structure
- ViewDefinitionInput structure
- ViewRepresentation structure
- ViewRepresentationInput structure
- UpdateOpenTableFormatInput structure
- UpdateIcebergInput structure
- CreateIcebergTableInput structure
- UpdateIcebergTableInput structure
- IcebergSortOrder structure
- IcebergSortField structure
- IcebergPartitionSpec structure
- IcebergPartitionField structure
- IcebergSchema structure

- IcebergStructField structure
- IcebergTableUpdate structure
- AuditContext structure

Structure de table

Représente un ensemble de données connexes organisées en colonnes et en lignes.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table. Pour la compatibilité Hive, ce nom doit être entièrement en minuscules.
- `DatabaseName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la base de données où résident les métadonnées de table. Pour la compatibilité Hive, ce nom doit être entièrement en minuscules.
- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description de la table.
- `Owner` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Propriétaire de la table.
- `CreateTime` – Horodatage.

  Date de création de la définition de table dans le catalogue de données.
- `UpdateTime` – Horodatage.

  Date de la dernière mise à jour de la table.
- `LastAccessTime` – Horodatage.

  Date du dernier accès à la table. Cette information est généralement extraite de HDFS et peut ne pas être fiable.
- `LastAnalyzedTime` – Horodatage.

Dernière date de calcul des statistiques de colonne pour cette table.

- `Retention` – Nombre (entier), pas plus qu'Aucun.

  Durée de conservation de la table.

- `StorageDescriptor` – Un Objet StorageDescriptor.

  Descripteur de stockage contenant des informations sur le stockage physique de la table.

- `PartitionKeys` : un tableau d'objets Colonne.

  Liste de colonnes par laquelle la table est partitionnée. Seuls les types primitifs sont pris en charge en tant que clés de partition.

  Lorsque vous créez un tableau utilisé par Amazon Athena et que vous ne spécifiez pas de `partitionKeys`, vous devez au moins régler la valeur de `partitionKeys` à une liste vide. Par exemple :

  `"PartitionKeys": []`

- `ViewOriginalText` – Chaîne UTF-8, d'une longueur maximale de 409 600 octets.

  Inclus pour la compatibilité avec Apache Hive. Non utilisé dans le cours normal des AWS Glue opérations. Si la table est une`VIRTUAL_VIEW`, une certaine Athena configuration codée en base64.

- `ViewExpandedText` – Chaîne UTF-8, d'une longueur maximale de 409 600 octets.

  Inclus pour la compatibilité avec Apache Hive. Non utilisé dans le cours normal des AWS Glue opérations.

- `TableType` – Chaîne UTF-8, d'une longueur maximale de 255 octets.

  Type de cette table. AWS Glue créera des tables avec le `EXTERNAL_TABLE` type. D'autres services, tels que Athena, peuvent créer des tables avec des types de tables supplémentaires.

  AWS Glue types de tables associés :

  EXTERNAL_TABLE

  Attribut compatible avec Hive : indique une table non gérée par Hive.

  GOVERNED

  Utilisé par AWS Lake Formation. Le catalogue AWS Glue de données comprend`GOVERNED`.

- `Parameters` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne de clé, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 512 000 octets.

  Ces paires clé-valeur définissent les propriétés associées à la table.

- `CreatedBy` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Personne ou entité qui a créé la table.

- `IsRegisteredWithLakeFormation` : booléen.

  Indique si la table a été enregistrée auprès de AWS Lake Formation.

- `TargetTable` – Un objet TableIdentifier.

  Structure `TableIdentifier` qui décrit une table cible pour la liaison de ressources.

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données dans lequel réside la table.

- `VersionId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID de la version de tableau.

- `FederatedTable` – Un objet FederatedTable.

  Une structure `FederatedTable` qui fait référence à une entité extérieure à AWS Glue Data Catalog.

- `ViewDefinition` – Un objet ViewDefinition.

  Structure contenant toutes les informations qui définissent la vue, y compris le ou les dialectes de la vue, et la requête.

- `IsMultiDialectView` : booléen.

  Spécifie si la vue prend en charge les dialectes SQL d'un ou de plusieurs différents moteurs de requête et peut donc être lue par ces moteurs.

TableInput structure

Structure utilisé pour définir une table.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table. Pour des raisons de compatibilité avec Hive, ce nom est converti en minuscules lors de son stockage.

- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description de la table.

- `Owner` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Propriétaire de la table. Inclus pour la compatibilité avec Apache Hive. Non utilisé dans le cours normal des AWS Glue opérations.

- `LastAccessTime` – Horodatage.

  Date du dernier accès à la table.

- `LastAnalyzedTime` – Horodatage.

  Dernière date de calcul des statistiques de colonne pour cette table.

- `Retention` – Nombre (entier), pas plus qu'Aucun.

  Durée de conservation de la table.

- `StorageDescriptor` – Un Objet StorageDescriptor.

  Descripteur de stockage contenant des informations sur le stockage physique de la table.

- `PartitionKeys` : un tableau d'objets Colonne.

  Liste de colonnes par laquelle la table est partitionnée. Seuls les types primitifs sont pris en charge en tant que clés de partition.

Lorsque vous créez un tableau utilisé par Amazon Athena et que vous ne spécifiez pas de `partitionKeys`, vous devez au moins régler la valeur de `partitionKeys` à une liste vide. Par exemple :

`"PartitionKeys": []`

- `ViewOriginalText` – Chaîne UTF-8, d'une longueur maximale de 409 600 octets.

  Inclus pour la compatibilité avec Apache Hive. Non utilisé dans le cours normal des AWS Glue opérations. Si la table est une`VIRTUAL_VIEW`, une certaine Athena configuration codée en base64.

- `ViewExpandedText` – Chaîne UTF-8, d'une longueur maximale de 409 600 octets.

  Inclus pour la compatibilité avec Apache Hive. Non utilisé dans le cours normal des AWS Glue opérations.

- `TableType` – Chaîne UTF-8, d'une longueur maximale de 255 octets.

  Type de cette table. AWS Glue créera des tables avec le `EXTERNAL_TABLE` type. D'autres services, tels que Athena, peuvent créer des tables avec des types de tables supplémentaires.

  AWS Glue types de tables associés :

  EXTERNAL_TABLE

  Attribut compatible avec Hive : indique une table non gérée par Hive.

  GOVERNED

  Utilisé par AWS Lake Formation. Le catalogue AWS Glue de données comprend`GOVERNED`.

- `Parameters` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne de clé, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 512 000 octets.

  Ces paires clé-valeur définissent les propriétés associées à la table.

- `TargetTable` – Un objet TableIdentifier.

  Structure `TableIdentifier` qui décrit une table cible pour la liaison de ressources.

- `ViewDefinition` – Un objet ViewDefinitionInput.

Structure contenant toutes les informations qui définissent la vue, y compris le ou les dialectes de la vue, et la requête.

FederatedTable structure

Une table qui pointe vers une entité extérieure à AWS Glue Data Catalog.

Champs

- `Identifier` – Chaîne UTF-8, d'une longueur comprise entre 1 et 512 octets, correspondant au Single-line string pattern.

  Un identifiant unique pour la table fédérée.
- `DatabaseIdentifier` – Chaîne UTF-8, d'une longueur comprise entre 1 et 512 octets, correspondant au Single-line string pattern.

  Un identifiant unique pour la base de données fédérée.
- `ConnectionName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Le nom de la connexion au métastore externe.
- `ConnectionType` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Type de connexion utilisé pour accéder à la table fédérée, spécifiant le protocole ou la méthode de connexion à la source de données externe.

Structure de colonne

Colonne d'une `Table`.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Le nom de l'`Column`.

- `Type` – Chaîne UTF-8, d'une longueur maximale de 131 072 octets, correspondant au Single-line string pattern.

  Le type de données de `Column`.

- `Comment` – Chaîne de commentaire, d'une longueur maximale de 255 octets, correspondant au Single-line string pattern.

  Commentaire au format de texte libre.

- `Parameters` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne de clé, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 512 000 octets.

  Ces paires clé-valeur définissent les propriétés associées à la colonne.

StorageDescriptor structure

Décrit le stockage physique des données de table.

Champs

- `Columns` : un tableau d'objets Colonne.

  Liste des `Columns` de la table.

- `Location` – Chaîne de localisation, d'une longueur maximale de 2 056 octets, correspondant au URI address multi-line string pattern.

  Emplacement physique de la table. Par défaut, il prend la forme de l'emplacement de entrepôt, suivie de l'emplacement de la base de données dans l'entrepôt, suivi du nom de la table.

- `AdditionalLocations` – Tableau de chaînes UTF-8.

  Liste des emplacements pointant vers le chemin d'accès où se trouve une table Delta.

- `InputFormat` – Chaîne de format, d'une longueur maximale de 128 octets, correspondant au Single-line string pattern.

  Format d'entrée : `SequenceFileInputFormat` (binaire) ou `TextInputFormat` ou format personnalisé.

- `OutputFormat` – Chaîne de format, d'une longueur maximale de 128 octets, correspondant au Single-line string pattern.

  Format de sortie : `SequenceFileOutputFormat` (binaire) ou `IgnoreKeyTextOutputFormat` ou format personnalisé.

- `Compressed` : booléen.

  `True` si les données de la table sont compressées ou `False` si elles ne le sont pas.

- `NumberOfBuckets` : nombre (entier).

  Doit être spécifié si la table contient des colonnes de dimension.

- `SerdeInfo` – Un objet SerDeInfo.

  Les serialization/deserialization (SerDe) informations.

- `BucketColumns` – Tableau de chaînes UTF-8.

  Liste de colonnes de regroupement de réducteur, de colonnes de clustérisation et de colonnes de mise en compartiment de la table.

- `SortColumns` : un tableau d'objets Ordre.

  Liste spécifiant l'ordre de tri de chaque compartiment de la table.

- `Parameters` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne de clé, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 512 000 octets.

  Propriétés fournies par l'utilisateur sous la forme clé-valeur.

- `SkewedInfo` – Un objet SkewedInfo.

  Informations concernant les valeurs qui apparaissent fréquemment dans une colonne (valeurs asymétriques).

- `StoredAsSubDirectories` : booléen.

  `True` si les données de la table sont stockées dans les sous-répertoires ou `False` dans le cas contraire.

- `SchemaReference` – Un objet SchemaReference.

Objet qui fait référence à un schéma stocké dans le registre des AWS Glue schémas.

Lors de la création d'une table, vous pouvez transmettre une liste vide de colonnes pour le schéma et plutôt utiliser une référence de schéma.

SchemaReference structure

Objet qui fait référence à un schéma stocké dans le registre des AWS Glue schémas.

Champs

- `SchemaId` – Un objet SchemaId.

  Structure qui contient des champs d'identité de schéma. Ceci ou `SchemaVersionId` doit être fourni.
- `SchemaVersionId` – Obligatoire : chaîne UTF-8, d'une longueur exacte de 36 octets, correspondant au Custom string pattern #45.

  ID unique affecté à une version du schéma. Ceci ou `SchemaId` doit être fourni.
- `SchemaVersionNumber` – Nombre (long), compris entre 1 et 100 000.

  Numéro de version du schéma.

SerDeInfo structure

Informations sur un serialization/deserialization programme (SerDe) qui sert d'extracteur et de chargeur.

Champs

- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du SerDe.
- `SerializationLibrary` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Généralement, la classe qui implémente le SerDe. Par exemple : `org.apache.hadoop.hive.serde2.columnar.ColumnarSerDe`.

- `Parameters` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne de clé, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 512 000 octets.

  Ces paires clé-valeur définissent les paramètres d'initialisation pour. SerDe

Structure de tri

Indique l'ordre de tri d'une colonne triée.

Champs

- `Column` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Le nom de la colonne.
- `SortOrder` – Obligatoire : Nombre (entier), inférieur ou égal à 1.

  Indique que la colonne est triée en ordre croissant (`==  1`) ou en ordre décroissant(`==0`).

SkewedInfo structure

Spécifie les valeurs biaisées d'une table. Les valeurs biaisées sont celles qui se produisent à très haute fréquence.

Champs

- `SkewedColumnNames` – Tableau de chaînes UTF-8.

  Liste des noms de colonnes qui contiennent des valeurs biaisées.
- `SkewedColumnValues` – Tableau de chaînes UTF-8.

  Liste des valeurs qui apparaissent si fréquemment qu'elles sont considérées comme biaisées.
- `SkewedColumnValueLocationMaps` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8.

Chaque valeur est une chaîne UTF-8.

Mappage de valeurs biaisées avec les colonnes qui les contiennent.

TableVersion structure

Spécifie une version d'une table.

Champs

- `Table` – Un objet Table.

  Table en question

- `VersionId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Valeur d'ID qui identifie cette version de table. Un `VersionId` est une représentation de chaîne d'un nombre entier. Chaque version est incrémentée par 1.

TableError structure

Enregistrement d'erreur pour les opérations de table.

Champs

- `TableName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table. Pour la compatibilité Hive, ce nom doit être entièrement en minuscules.

- `ErrorDetail` – Un objet ErrorDetail.

  Détails relatifs à l'erreur.

TableVersionError structure

Enregistrement d'erreur pour les opérations de version table.

Champs

- `TableName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table en question.

- `VersionId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Valeur d'ID de la version en question. Un `VersionID` est une représentation de chaîne d'un nombre entier. Chaque version est incrémentée par 1.

- `ErrorDetail` – Un objet ErrorDetail.

  Détails relatifs à l'erreur.

SortCriterion structure

Spécifie un champ selon lequel trier et un ordre de tri.

Champs

- `FieldName` : chaîne de valeurs, d'une longueur comprise entre 1 et 1 024 octets.

  Nom du champ sur lequel trier.

- `Sort` – Chaîne UTF-8 (valeurs valides : `ASC="ASCENDING"` | `DESC="DESCENDING"`).

  Tri croissant ou décroissant.

TableIdentifier structure

Structure qui décrit une table cible pour la liaison de ressources.

Champs

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données dans lequel réside la table.

- `DatabaseName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la base de données du catalogue qui contient la table cible.
- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table cible.
- `Region` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Région de la table cible.

KeySchemaElement structure

Une paire de clés de partition composée d'un nom et d'un type.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom d'une clé de partition.
- `Type` – Obligatoire : chaîne UTF-8, d'une longueur comprise entre 1 et 131 072 octets, correspondant au Single-line string pattern.

  Type d'une clé de partition.

PartitionIndex structure

Structure pour un index de partition.

Champs

- `Keys` – Obligatoire : Tableau de chaînes UTF-8, au moins 1 chaîne.

  Les clés de l'index de partition.
- `IndexName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

Nom de l'index de partition.

PartitionIndexDescriptor structure

Descripteur d'un index de partition dans une table.

Champs

- `IndexName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de l'index de partition.

- `Keys` – Obligatoire : Tableau d'objets KeySchemaElement, au moins 1 structure.

  Liste d'une ou plusieurs clés, sous forme de structures `KeySchemaElement`, pour l'index de partition.

- `IndexStatus` – Obligatoire :Chaîne UTF-8 (valeurs valides : `CREATING` | `ACTIVE` | `DELETING` | `FAILED`).

  Statut de l'index de partition.

  Les statuts possibles sont les suivants :

  - CRÉATION : l'index est en cours de création. Lorsqu'un index est dans un état CRÉATION, l'index ou sa table ne peut pas être supprimé.
  - ACTIVE : la création de l'index a réussi.
  - ÉCHEC : la création de l'index a échoué.
  - SUPPRESSION : l'index est supprimé de la liste des index.

- `BackfillErrors` : un tableau d'objets BackfillError.

  Liste d'erreurs qui peuvent se produire lors de l'enregistrement d'index de partition pour une table existante.

BackfillError structure

Liste d'erreurs qui peuvent se produire lors de l'enregistrement d'index de partition pour une table existante.

Ces erreurs donnent les détails sur la raison pour laquelle un enregistrement d'index a échoué et fournissent un nombre limité de partitions dans la réponse, de sorte que vous pouvez corriger les partitions défaillantes et essayer d'enregistrer à nouveau l'index. Les erreurs les plus courantes pouvant se produire sont classées comme suit :

- EncryptedPartitionError: Les partitions sont cryptées.
- InvalidPartitionTypeDataError: La valeur de la partition ne correspond pas au type de données de cette colonne de partition.
- MissingPartitionValueError: Les partitions sont cryptées.
- UnsupportedPartitionCharacterError: Les caractères contenus dans la valeur de partition ne sont pas pris en charge. Par exemple : U+0000, U+0001, U+0002.
- InternalError: Toute erreur qui n'appartient pas à d'autres codes d'erreur.

Champs

- `Code` – Chaîne UTF-8 (valeurs valides : `ENCRYPTED_PARTITION_ERROR` | `INTERNAL_ERROR` | `INVALID_PARTITION_TYPE_DATA_ERROR` | `MISSING_PARTITION_VALUE_ERROR` | `UNSUPPORTED_PARTITION_CHARACTER_ERROR`).

  Code d'erreur correspondant à une erreur survenue lors de l'enregistrement d'index de partition pour une table existante.
- `Partitions` : un tableau d'objets PartitionValueList.

  Liste d'un nombre limité de partitions dans la réponse.

IcebergInput structure

Une structure qui définit une table de métadonnées Apache Iceberg à créer dans le catalogue.

Champs

- `MetadataOperation` – Obligatoire : Chaîne UTF-8 (valeurs valides : `CREATE`).

  Une opération de métadonnées obligatoire. Ne peut être défini que sur `CREATE`.
- `Version` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  La version de table pour la table Iceberg. La valeur par défaut est 2.

- `CreateIcebergTableInput` – Un objet CreateIcebergTableInput.

  Les paramètres de configuration requis pour créer une nouvelle table Iceberg dans le catalogue de AWS Glue données, y compris les propriétés de la table et les spécifications des métadonnées.

OpenTableFormatInput structure

Une structure représentant un tableau au format ouvert.

Champs

- `IcebergInput` : un objet IcebergInput.

  Spécifie une structure `IcebergInput` qui définit une table de métadonnées Apache Iceberg.

ViewDefinition structure

Structure contenant les détails des représentations.

Champs

- `IsProtected` : booléen.

  Vous pouvez définir cet indicateur sur true pour indiquer au moteur de ne pas transférer les opérations fournies par l'utilisateur dans le plan logique de la vue lors de la planification des requêtes. Cependant, le réglage de cet indicateur ne garantit pas que le moteur sera conforme. Reportez-vous à la documentation du moteur pour comprendre les garanties fournies, le cas échéant.
- `Definer` – chaîne UTF-8, d'une longueur comprise entre 20 et 2048 octets, correspondant au Single-line string pattern.

  Définisseur d'une vue dans SQL.
- `SubObjects` : tableau de chaînes UTF-8, avec 10 chaînes maximum.

  Liste des tables Amazon Resource Names (ARNs).
- `Representations` – tableau d'objets ViewRepresentation, avec 1 structure minimum et 1 000 structures maximum.

  Liste de représentations.

ViewDefinitionInput structure

Structure contenant les détails relatifs à la création ou à la mise à jour d'une AWS Glue vue.

Champs

- `IsProtected` : booléen.

  Vous pouvez définir cet indicateur sur true pour indiquer au moteur de ne pas transférer les opérations fournies par l'utilisateur dans le plan logique de la vue lors de la planification des requêtes. Cependant, le réglage de cet indicateur ne garantit pas que le moteur sera conforme. Reportez-vous à la documentation du moteur pour comprendre les garanties fournies, le cas échéant.
- `Definer` – chaîne UTF-8, d'une longueur comprise entre 20 et 2048 octets, correspondant au Single-line string pattern.

  Définisseur d'une vue dans SQL.
- `Representations` – Tableau d'objets ViewRepresentationInput, avec 1 structure minimum et 10 structures maximum.

  Liste de structures contenant le dialecte de la vue et la requête qui définit la vue.
- `SubObjects` : tableau de chaînes UTF-8, avec 10 chaînes maximum.

  Liste des tables de base ARNs qui constituent la vue.

ViewRepresentation structure

Structure contenant le dialecte de la vue et la requête qui définit la vue.

Champs

- `Dialect` – Chaîne UTF-8 (valeurs valides : `REDSHIFT` | `ATHENA` | `SPARK`).

  Dialecte du moteur de requête.
- `DialectVersion` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets.

  Version du dialecte du moteur de requête. Par exemple, 3.0.0.
- `ViewOriginalText` – Chaîne UTF-8, d'une longueur maximale de 409 600 octets.

Requête SELECT fournie par le client pendant CREATE VIEW DDL. Ce code SQL n'est pas utilisé lors d'une requête sur une vue (ViewExpandedText est utilisé à la place). ViewOriginalText est utilisé dans les cas comme SHOW CREATE VIEW où les utilisateurs souhaitent voir la commande DDL d'origine qui a créé la vue.

- ViewExpandedText – Chaîne UTF-8, d'une longueur maximale de 409 600 octets.

  Code SQL étendu pour la vue. Ce code SQL est utilisé par les moteurs lors du traitement d'une requête sur une vue. Les moteurs peuvent effectuer des opérations lors de la création de la vue pour transformer ViewOriginalText en ViewExpandedText. Par exemple :

  - Identifiants entièrement qualifiés : SELECT * from table1 -> SELECT * from db1.table1

- ValidationConnection – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la connexion à utiliser pour valider la représentation spécifique de la vue.

- IsStale : booléen.

  Les dialectes marqués comme périmés ne sont plus valides et doivent être mis à jour avant de pouvoir être interrogés dans leurs moteurs de requêtes respectifs.

ViewRepresentationInput structure

Structure contenant les détails d'une représentation pour mettre à jour ou créer une vue Lake Formation.

Champs

- Dialect – Chaîne UTF-8 (valeurs valides : REDSHIFT | ATHENA | SPARK).

  Paramètre qui indique le type de moteur d'une représentation spécifique.

- DialectVersion – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets.

  Paramètre qui indique la version du moteur d'une représentation spécifique.

- ViewOriginalText – Chaîne UTF-8, d'une longueur maximale de 409 600 octets.

  Chaîne qui représente la requête SQL d'origine qui décrit la vue.

- `ValidationConnection` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la connexion à utiliser pour valider la représentation spécifique de la vue.
- `ViewExpandedText` – Chaîne UTF-8, d'une longueur maximale de 409 600 octets.

  Chaîne qui représente la requête SQL qui décrit la vue avec une ressource étendue ARNs

UpdateOpenTableFormatInput structure

Paramètres d'entrée pour la mise à jour des tables au format de table ouvertes dans AWS Glue Data Catalog, servant de wrapper pour les opérations de mise à jour spécifiques au format, telles qu'Apache Iceberg.

Champs

- `UpdateIcebergInput` : un objet UpdateIcebergInput.

  Paramètres de mise à jour propres à Apache Iceberg qui définissent les modifications de table à appliquer, notamment les modifications de schéma, les spécifications de partition et les propriétés de table.

UpdateIcebergInput structure

Paramètres d'entrée spécifiques à la mise à jour des tables Apache Iceberg dans le catalogue de AWS Glue données, contenant les opérations de mise à jour à appliquer à une table Iceberg existante.

Champs

- `UpdateIcebergTableInput` – Obligatoire : un objet UpdateIcebergTableInput.

  Opérations de mise à jour spécifiques à appliquer à la table Iceberg, contenant une liste de mises à jour qui définissent le nouvel état de la table, y compris le schéma, les partitions et les propriétés.

CreateIcebergTableInput structure

Les paramètres de configuration requis pour créer une nouvelle table Iceberg dans le catalogue de AWS Glue données, y compris les propriétés de la table et les spécifications des métadonnées.

Champs

- `Location` – Obligatoire : chaîne de localisation, d'une longueur maximale de 2 056 octets, correspondant au URI address multi-line string pattern.

  L'emplacement S3 dans lequel les données de la table Iceberg seront stockées.

- `Schema` – Obligatoire : un objet IcebergSchema.

  Définition du schéma qui spécifie la structure, les types de champs et les métadonnées de la table Iceberg.

- `PartitionSpec` : un objet IcebergPartitionSpec.

  Spécification de partitionnement qui définit la manière dont les données de la table Iceberg seront organisées et partitionnées pour des performances de requête optimales.

- `WriteOrder` : un objet IcebergSortOrder.

  Spécification de l'ordre de tri qui définit la manière dont les données doivent être ordonnées au sein de chaque partition afin d'optimiser les performances des requêtes.

- `Properties` –

  Paires clé-valeur contenant des propriétés de table supplémentaires et des paramètres de configuration pour la table Iceberg.

UpdateIcebergTableInput structure

Contient les opérations de mise à jour à appliquer à une table Iceberg existante dans le catalogue de AWS Glue données, définissant le nouvel état des métadonnées de la table.

Champs

- `Updates` – Obligatoire : un tableau d'objets IcebergTableUpdate.

  Liste des opérations de mise à jour de la table qui spécifient les modifications à apporter à la table Iceberg, notamment les modifications du schéma, les spécifications de partition et les propriétés de la table.

IcebergSortOrder structure

Définit la spécification de l'ordre de tri pour une table Iceberg, déterminant comment les données doivent être classées dans les partitions afin d'optimiser les performances des requêtes.

Champs

- `OrderId` – Obligatoire : nombre (entier).

  Identifiant unique pour cette spécification d'ordre de tri dans les métadonnées de la table Iceberg.
- `Fields` – Obligatoire : un tableau d'objets IcebergSortField.

  La liste des champs et leurs directions de tri qui définissent les critères d'ordre des données de la table Iceberg.

IcebergSortField structure

Définit un champ unique dans une spécification d'ordre de tri Iceberg, y compris le champ source, la transformation, la direction du tri et l'ordre des valeurs nulles.

Champs

- `SourceId` – Obligatoire : nombre (entier).

  Identifiant du champ source issu du schéma de table sur lequel ce champ de tri est basé.
- `Transform` – Obligatoire : chaîne UTF-8.

  Fonction de transformation appliquée au champ source avant le tri, telle que identity, bucket ou truncate.
- `Direction` – Obligatoire : Chaîne UTF-8 (valeurs valides : `asc="ASC"` | `desc="DESC"`).

  La direction du tri pour ce champ, soit croissant, soit décroissant.
- `NullOrder` – Obligatoire : Chaîne UTF-8 (valeurs valides : `nulls-first="NULLS_FIRST"` | `nulls-last="NULLS_LAST"`).

  Comportement d'ordre des valeurs nulles dans ce champ, spécifiant si les valeurs nulles doivent apparaître en premier ou en dernier dans l'ordre de tri.

IcebergPartitionSpec structure

Définit les spécifications de partitionnement d'une table Iceberg, déterminant comment les données de la table seront organisées et partitionnées pour des performances de requête optimales.

Champs

- `Fields` – Obligatoire : un tableau d'objets IcebergPartitionField.

  Liste des champs de partition qui définissent la manière dont les données de la table doivent être partitionnées, y compris les champs source et leurs transformations.
- `SpecId` : nombre (entier).

  Identifiant unique de cette spécification de partition dans l'historique des métadonnées de la table Iceberg.

IcebergPartitionField structure

Définit un champ de partition unique dans une spécification de partition Iceberg, y compris le champ source, la fonction de transformation, le nom de la partition et l'identifiant unique.

Champs

- `SourceId` – Obligatoire : nombre (entier).

  Identifiant du champ source issu du schéma de table sur lequel ce champ de partition est basé.
- `Transform` – Obligatoire : chaîne UTF-8.

  Fonction de transformation appliquée au champ source pour créer la partition, telle que identity, bucket, truncate, year, month, day ou hour.
- `Name` – Obligatoire : chaîne UTF-8, d'une longueur comprise entre 1 et 1024 octets, correspondant au Single-line string pattern.

  Nom du champ de partition tel qu'il apparaîtra dans la structure de la table partitionnée.
- `FieldId` : nombre (entier).

  Identifiant unique attribué à ce champ de partition dans la spécification de partition de la table Iceberg.

IcebergSchema structure

Définit la structure du schéma d'une table Iceberg, y compris les définitions de champs, les types de données et les métadonnées du schéma.

Champs

- `SchemaId` : nombre (entier).

  Identifiant unique de cette version de schéma dans l'historique d'évolution du schéma de la table Iceberg.
- `IdentifierFieldIds` : un tableau d'entiers 32 bits signés.

  Liste des identifiants de champs qui identifient de manière unique les enregistrements de la table, utilisés pour les opérations au niveau des lignes et la déduplication.
- `Type` – Chaîne UTF-8 (valeurs valides : `struct="STRUCT"`).

  Type de racine de la structure du schéma, généralement « struct » pour les schémas de table Iceberg.
- `Fields` – Obligatoire : un tableau d'objets IcebergStructField.

  Liste des définitions de champs qui constituent le schéma de table, y compris les noms, les types et les métadonnées des champs.

IcebergStructField structure

Définit un champ unique dans un schéma de table Iceberg, y compris son identifiant, son nom, son type de données, son caractère nul et sa documentation.

Champs

- `Id` – Obligatoire : nombre (entier).

  Identifiant unique attribué à ce champ dans le schéma de table Iceberg, utilisé pour l'évolution du schéma et le suivi des champs.
- `Name` – Obligatoire : chaîne UTF-8, d'une longueur comprise entre 1 et 1024 octets, correspondant au Single-line string pattern.

  Nom du champ tel qu'il apparaît dans le schéma de table et dans les opérations de requête.

- `Type`— Obligatoire : une structure vide nommée `IcebergDocument`.

  Définition du type de données pour ce champ, spécifiant la structure et le format des données qu'il contient.

- `Required` – obligatoire : booléen.

  Indique si ce champ est obligatoire (n'autorisant pas les valeurs nulles) ou facultatif (autorisant les valeurs nulles) dans le schéma de table.

- `Doc` – Chaîne de commentaire, d'une longueur maximale de 255 octets, correspondant au Single-line string pattern.

  Documentation facultative ou texte de description fournissant un contexte supplémentaire sur l'objectif et l'utilisation de ce champ.

IcebergTableUpdate structure

Définit un ensemble complet de mises à jour à appliquer à une table Iceberg, y compris les modifications de schéma, les modifications de partitionnement, les ajustements d'ordre de tri, les mises à jour de localisation et les modifications de propriétés.

Champs

- `Schema` – Obligatoire : un objet IcebergSchema.

  Définition de schéma mise à jour pour la table Iceberg, spécifiant les modifications apportées à la structure des champs, aux types de données ou aux métadonnées du schéma.

- `PartitionSpec` : un objet IcebergPartitionSpec.

  Spécification de partitionnement mise à jour qui définit la manière dont les données de la table doivent être réorganisées et partitionnées.

- `SortOrder` : un objet IcebergSortOrder.

  Spécification de l'ordre de tri mise à jour qui définit la manière dont les données doivent être ordonnées au sein des partitions afin d'optimiser les performances des requêtes.

- `Location` – Obligatoire : chaîne de localisation, d'une longueur maximale de 2 056 octets, correspondant au URI address multi-line string pattern.

  L'emplacement S3 mis à jour dans lequel les données de la table Iceberg seront stockées.

- `Properties` –

  Paires clé-valeur mises à jour des propriétés de table et des paramètres de configuration pour la table Iceberg.

AuditContext structure

Structure contenant le contexte de l' Lake Formation audit.

Champs

- `AdditionalAuditContext`— Chaîne UTF-8 d'une longueur maximale de 2 048 octets.

  Chaîne contenant les informations contextuelles d'audit supplémentaires.
- `RequestedColumns` – Tableau de chaînes UTF-8.

  Les colonnes demandées pour l'audit.
- `AllColumnsRequested` : booléen.

  Toutes les colonnes font l'objet d'une demande d'audit.

Opérations

- CreateTable action (Python : créer_table)
- UpdateTable action (Python : update_table)
- DeleteTable action (Python : supprimer_table)
- BatchDeleteTable action (Python : batch_delete_table)
- GetTable action (Python : get_table)
- GetTables action (Python : get_tables)
- GetTableVersion action (Python : get_table_version)
- GetTableVersions action (Python : get_table_versions)
- DeleteTableVersion action (Python : supprimer_table_version)
- BatchDeleteTableVersion action (Python : batch_delete_table_version)
- SearchTables action (Python : tables de recherche)
- GetPartitionIndexes action (Python : get_partition_indexes)

- CreatePartitionIndex action (Python : create_partition_index)
- DeletePartitionIndex action (Python : delete_partition_index)
- GetColumnStatisticsForTable action (Python : get_column_statistics_for_table)
- UpdateColumnStatisticsForTable action (Python : update_column_statistics_for_table)
- DeleteColumnStatisticsForTable action (Python : delete_column_statistics_for_table)

CreateTable action (Python : créer_table)

Crée une nouvelle définition de table dans le catalogue de données.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données dans lequel créer la `Table`. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.
- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Base de données du catalogue dans laquelle créer la table. Pour la compatibilité Hive, ce nom doit être entièrement en minuscules.
- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant unique de la table de la base de données spécifiée qui sera créée dans le catalogue de AWS Glue données.
- `TableInput` – Un objet TableInput.

  Objet `TableInput` qui définit la table de métadonnées à créer dans le catalogue.
- `PartitionIndexes` – Un tableau d'objets PartitionIndex, 3 structures au maximum.

  Liste d'index de partition, structures de type `PartitionIndex`, à créer dans la table.
- `TransactionId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #44.

  ID de la transaction.

- `OpenTableFormatInput` : un objet OpenTableFormatInput.

  Spécifie une structure `OpenTableFormatInput` lors de la création d'un tableau au format ouvert.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `AlreadyExistsException`
- `InvalidInputException`
- `EntityNotFoundException`
- `ResourceNumberLimitExceededException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`
- `ConcurrentModificationException`
- `ResourceNotReadyException`
- `FederationSourceException`
- `FederationSourceRetryableException`

UpdateTable action (Python : update_table)

Met à jour une table de métadonnées dans le catalogue de données.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données où réside la table. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.
- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

Nom de la base de données du catalogue où se trouve la table. Pour la compatibilité Hive, ce nom doit être entièrement en minuscules.

- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant unique de la table de la base de données spécifiée qui sera créée dans le catalogue de AWS Glue données.

- `TableInput` – Un objet TableInput.

  Objet `TableInput` mis à jour qui définit la table de métadonnées du catalogue.

- `SkipArchive` : booléen.

  Par défaut, `UpdateTable` crée toujours une version archivée de la table avant de la mettre à jour. Toutefois, si `skipArchive` a la valeur true, `UpdateTable` ne crée pas la version archivée.

- `TransactionId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #44.

  L'ID de transaction auquel le contenu de la table doit être mis à jour.

- `VersionId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  L'ID de la version selon laquelle le contenu du tableau doit être mis à jour.

- `ViewUpdateAction` – Chaîne UTF-8 (valeurs valides : `ADD` | `REPLACE` | `ADD_OR_REPLACE` | `DROP`).

  Opération à effectuer lors de la mise à jour de la vue.

- `Force` : booléen.

  Indicateur qui peut être défini sur true pour ignorer les exigences de correspondance entre le descripteur de stockage et le sous-objet.

- `UpdateOpenTableFormatInput` : un objet UpdateOpenTableFormatInput.

  Paramètres d'entrée pour la mise à jour des tables au format de table ouvertes dans AWS Glue Data Catalog, servant de wrapper pour les opérations de mise à jour spécifiques au format, telles qu'Apache Iceberg.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ConcurrentModificationException`
- `ResourceNumberLimitExceededException`
- `GlueEncryptionException`
- `ResourceNotReadyException`
- `FederationSourceException`
- `FederationSourceRetryableException`
- `AlreadyExistsException`

DeleteTable action (Python : supprimer_table)

Supprime une définition de table du catalogue de données.

> **Note**
>
> Après avoir effectué cette opération, vous n'avez plus accès aux versions et partitions de table appartenant à la table supprimée. AWS Glue supprime ces ressources « orphelines » de manière asynchrone en temps opportun, à la discrétion du service.
> Pour garantir la suppression immédiate de toutes les ressources connexes, avant d'appeler `DeleteTable`, utilisez `DeleteTableVersion` ou `BatchDeleteTableVersion`, et `DeletePartition` ou `BatchDeletePartition`, pour supprimer les ressources appartenant à la table.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données où réside la table. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.
- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la base de données du catalogue où se trouve la table. Pour la compatibilité Hive, ce nom doit être entièrement en minuscules.
- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table à supprimer. Pour la compatibilité Hive, ce nom doit être entièrement en minuscules.
- `TransactionId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #44.

  L'ID de transaction pour laquelle supprimer le contenu de la table.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ConcurrentModificationException`
- `ResourceNotReadyException`
- `FederationSourceException`
- `FederationSourceRetryableException`

BatchDeleteTable action (Python : batch_delete_table)

Supprime plusieurs tables à la fois.

> **Note**
>
> Après avoir effectué cette opération, vous n'avez plus accès aux versions et partitions de table appartenant à la table supprimée. AWS Glue supprime ces ressources « orphelines » de manière asynchrone en temps opportun, à la discrétion du service.
> Pour garantir la suppression immédiate de toutes les ressources connexes, avant d'appeler `BatchDeleteTable`, utilisez `DeleteTableVersion` ou `BatchDeleteTableVersion`, et `DeletePartition` ou `BatchDeletePartition`, pour supprimer les ressources appartenant à la table.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données où réside la table. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.
- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la base de données de catalogue où résident les tables à supprimer. Pour la compatibilité Hive, ce nom doit être entièrement en minuscules.
- `TablesToDelete` – Obligatoire : Tableau de chaînes UTF-8, avec 100 chaînes maximum.

  Liste des tables à supprimer.
- `TransactionId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #44.

  L'ID de transaction pour laquelle supprimer le contenu de la table.

Réponse

- `Errors` : un tableau d'objets TableError.

Liste des erreurs survenues dans la tentative de supprimer les tables spécifiées.

Erreurs

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`
- `ResourceNotReadyException`

GetTable action (Python : get_table)

Extrait la définition de `Table` d'un catalogue de données pour une table spécifiée.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données où réside la table. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.
- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la base de données du catalogue où se trouve la table. Pour la compatibilité Hive, ce nom doit être entièrement en minuscules.
- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table pour laquelle récupérer la définition. Pour la compatibilité Hive, ce nom doit être entièrement en minuscules.
- `TransactionId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #44.

  L'ID de transaction pour laquelle lire le contenu de la table.

- `QueryAsOfTime` – Horodatage.

  L'Heure à laquelle lire le contenu de la table. S'il n'est pas défini, l'heure de validation de transaction la plus récente sera utilisée. Ne peut pas être spécifié avec `TransactionId`.
- `AuditContext` : un objet AuditContext.

  Structure contenant le contexte de l' Lake Formation audit.
- `IncludeStatusDetails` : booléen.

  Spécifie s'il faut inclure les détails du statut relatifs à une demande de création ou de mise à jour d'une vue du catalogue de AWS Glue données.

Réponse

- `Table` – Un Objet Table.

  Objet `Table` qui définit la table spécifiée.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`
- `ResourceNotReadyException`
- `FederationSourceException`
- `FederationSourceRetryableException`

GetTables action (Python : get_tables)

Récupère les définitions de tout ou partie des tables dans une `Database` donnée.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

L'ID du catalogue de données où résident les tables. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.

- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Base de données du catalogue dont les tables doivent être affichées. Pour la compatibilité Hive, ce nom doit être entièrement en minuscules.

- `Expression` – Chaîne UTF-8, d'une longueur maximale de 2 048 octets, correspondant au Single-line string pattern.

  Modèle d'expression régulière. S'il est présent, seules les tables dont les noms correspondent au modèle sont renvoyées.

- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, inclus s'il s'agit d'un appel de continuation.

- `MaxResults` – Nombre (entier), compris entre 1 et 100.

  Nombre maximal de tables à renvoyer dans une seule réponse.

- `TransactionId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #44.

  L'ID de transaction pour laquelle lire le contenu de la table.

- `QueryAsOfTime` – Horodatage.

  L'Heure à laquelle lire le contenu de la table. S'il n'est pas défini, l'heure de validation de transaction la plus récente sera utilisée. Ne peut pas être spécifié avec `TransactionId`.

- `AuditContext` : un objet AuditContext.

  Structure contenant le contexte de l' Lake Formation audit.

- `IncludeStatusDetails` : booléen.

  Spécifie s'il faut inclure les détails du statut relatifs à une demande de création ou de mise à jour d'une vue du catalogue de AWS Glue données.

- `AttributesToGet` – Tableau de chaînes UTF-8.

  Indique les champs de table renvoyés par l'appel `GetTables`. Ce paramètre n'accepte pas une liste vide. La demande doit inclure `NAME`.

Les combinaisons de valeurs valides sont les suivantes :

- `NAME` : nom de toutes les tables de la base de données.
- `NAME, TABLE_TYPE` : noms de toutes les tables et des types de table.

Réponse

- `TableList` : un tableau d'objets Table.

  Liste des objets `Table` demandés.
- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, présent si le segment de liste actuel n'est pas le dernier.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`
- `GlueEncryptionException`
- `FederationSourceException`
- `FederationSourceRetryableException`

GetTableVersion action (Python : get_table_version)

Extrait une version spécifiée d'une table.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  L'ID du catalogue de données où résident les tables. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.

- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Base de données du catalogue où se trouve la table. Pour la compatibilité Hive, ce nom doit être entièrement en minuscules.
- `TableName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table. Pour la compatibilité Hive, ce nom doit être entièrement en minuscules.
- `VersionId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Valeur d'ID de la version de table à récupérer. Un `VersionID` est une représentation de chaîne d'un nombre entier. Chaque version est incrémentée par 1.

Réponse

- `TableVersion` – Un objet TableVersion.

  Version de table demandée.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`

GetTableVersions action (Python : get_table_versions)

Extrait une liste de chaînes qui identifient les versions disponibles d'une table spécifiée.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

L'ID du catalogue de données où résident les tables. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.

- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Base de données du catalogue où se trouve la table. Pour la compatibilité Hive, ce nom doit être entièrement en minuscules.

- `TableName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table. Pour la compatibilité Hive, ce nom doit être entièrement en minuscules.

- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, s'il ne s'agit pas du premier appel.

- `MaxResults` – Nombre (entier), compris entre 1 et 100.

  Nombre maximal de versions de table à renvoyer dans une réponse.

Réponse

- `TableVersions` : un tableau d'objets TableVersion.

  Liste de chaînes qui identifient les versions disponibles de la table spécifiée.

- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, si la liste des version disponibles n'inclut pas la dernière version.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`

DeleteTableVersion action (Python : supprimer_table_version)

Supprime une version spécifiée d'une table.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  L'ID du catalogue de données où résident les tables. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.
- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Base de données du catalogue où se trouve la table. Pour la compatibilité Hive, ce nom doit être entièrement en minuscules.
- `TableName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table. Pour la compatibilité Hive, ce nom doit être entièrement en minuscules.
- `VersionId` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID de la version de table à supprimer. Un `VersionID` est une représentation de chaîne d'un nombre entier. Chaque version est incrémentée par 1.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

BatchDeleteTableVersion action (Python : batch_delete_table_version)

Supprime un lot spécifié de versions d'une table.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  L'ID du catalogue de données où résident les tables. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.

- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Base de données du catalogue où se trouve la table. Pour la compatibilité Hive, ce nom doit être entièrement en minuscules.

- `TableName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table. Pour la compatibilité Hive, ce nom doit être entièrement en minuscules.

- `VersionIds` – Obligatoire : Tableau de chaînes UTF-8, avec 100 chaînes maximum.

  Liste IDs des versions à supprimer. Un `VersionId` est une représentation de chaîne d'un nombre entier. Chaque version est incrémentée par 1.

Réponse

- `Errors` : un tableau d'objets TableVersionError.

  Liste des erreurs survenues lors de la tentative de suppression des versions de table spécifiées.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

SearchTables action (Python : tables de recherche)

Recherche un ensemble de tables en fonction des propriétés figurant dans les métadonnées de table, ainsi que dans la base de données parent. Vous pouvez effectuer une recherche sur des conditions de texte ou de filtre.

Vous pouvez uniquement obtenir les tables auxquelles vous avez accès en fonction des stratégies de sécurité définies dans Lake Formation. Vous avez besoin d'au moins un accès en lecture seule à la table pour qu'elle soit renvoyée. Si vous n'avez pas accès à toutes les colonnes de la table, ces colonnes ne feront pas l'objet d'une recherche lorsque la liste des tables vous sera renvoyée. Si vous avez accès aux colonnes, mais pas aux données des colonnes, ces colonnes et les métadonnées associées pour ces colonnes seront incluses dans la recherche.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant unique, composé de *`account_id`*.

- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, inclus s'il s'agit d'un appel de continuation.

- `Filters` : un tableau d'objets PropertyPredicate.

  Liste de paires clé-valeur et comparateur utilisé pour filtrer les résultats de recherche. Renvoie toutes les entités correspondant au prédicat.

  Le membre `Comparator` du struct `PropertyPredicate` est utilisé uniquement pour les champs temporels et peut être omis pour d'autres types de champs. En outre, lors de la comparaison de valeurs de chaîne, par exemple lorsque `Key=Name`, un algorithme de correspondance floue (fuzzy match) est utilisé. Le champ `Key` (par exemple, la valeur du champ `Name`) est divisé en jetons sur certains caractères de ponctuation, par exemple -, :, #, etc.. Ensuite, chaque jeton est comparé de manière exacte à un membre `Value` de `PropertyPredicate`. Par exemple, si `Key=Name` et `Value=link`, les tables nommées `customer-link` et `xx-link-yy` sont retournés, mais `xxlinkyy` ne l'est pas.

- `SearchText` : chaîne de valeurs, d'une longueur comprise entre 1 et 1 024 octets.

  Chaîne utilisée pour une recherche de texte.

La spécification d'une valeur entre guillemets filtre en fonction d'une correspondance exacte avec cette valeur.

- `SortCriteria` – Un tableau d'objets SortCriterion, pas plus d'une structure.

  Liste de critères de tri des résultats par un nom de champ, dans un ordre croissant ou décroissant.

- `MaxResults` – Nombre (entier), compris entre 1 et 1 000.

  Nombre maximal de tables à renvoyer dans une seule réponse.

- `ResourceShareType` – Chaîne UTF-8 (valeurs valides : `FOREIGN` | `ALL` | `FEDERATED`).

  Permet de spécifier que vous souhaitez rechercher les tables partagées avec votre compte. Les valeurs autorisées sont `FOREIGN` ou `ALL`.

  - Si défini sur `FOREIGN`, effectuera une recherche dans les tables partagées avec votre compte.
  - Si défini sur `ALL`, effectuera une recherche dans les tables partagées avec votre compte, ainsi que les tables dans votre compte local.

- `IncludeStatusDetails` : booléen.

  Spécifie s'il faut inclure les détails du statut relatifs à une demande de création ou de mise à jour d'une vue du catalogue de AWS Glue données.

Réponse

- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, présent si le segment de liste actuel n'est pas le dernier.

- `TableList` : un tableau d'objets Table.

  Liste des objets `Table` demandés. La réponse `SearchTables` renvoie uniquement les tables auxquelles vous avez accès.

Erreurs

- `InternalServiceException`
- `InvalidInputException`
- `OperationTimeoutException`

GetPartitionIndexes action (Python : get_partition_indexes)

Récupère les index de partition associés à une table.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue où réside la table.
- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Spécifie le nom d'une base de données à partir de laquelle vous souhaitez extraire des index de partition.
- `TableName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Spécifie le nom d'une table à partir de laquelle vous souhaitez extraire les index de partition.
- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, inclus s'il s'agit d'un appel de continuation.

Réponse

- `PartitionIndexDescriptorList` : un tableau d'objets PartitionIndexDescriptor.

  Liste des descripteurs d'index.
- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, présent si le segment de liste actuel n'est pas le dernier.

Erreurs

- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`
- `EntityNotFoundException`

- `ConflictException`

CreatePartitionIndex action (Python : create_partition_index)

Crée un index de partition spécifié dans une table existante.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue où réside la table.
- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Spécifie le nom d'une base de données dans laquelle vous souhaitez créer un index de partition.
- `TableName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Spécifie le nom d'une table dans laquelle vous souhaitez créer un index de partition.
- `PartitionIndex` – Obligatoire : un objet PartitionIndex.

  Spécifie une structure `PartitionIndex` pour créer un index de partition dans une table existante.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `AlreadyExistsException`
- `InvalidInputException`
- `EntityNotFoundException`
- `ResourceNumberLimitExceededException`
- `InternalServiceException`
- `OperationTimeoutException`

- `GlueEncryptionException`

DeletePartitionIndex action (Python : delete_partition_index)

Supprime un index de partition spécifié d'une table existante.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue où réside la table.
- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Spécifie le nom d'une base de données dont vous souhaitez supprimer un index de partition.
- `TableName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Spécifie le nom d'une table dont vous souhaitez supprimer un index de partition.
- `IndexName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de l'index de partition à supprimer.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`
- `EntityNotFoundException`
- `ConflictException`
- `GlueEncryptionException`

GetColumnStatisticsForTable action (Python : get_column_statistics_for_table)

Récupère les statistiques de table des colonnes.

L'autorisation Identity and Access Management (IAM) requise pour cette opération est `GetTable`.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données où résident les partitions en question. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.
- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la base de données de catalogue où résident les partitions.
- `TableName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table des partitions.
- `ColumnNames` – Obligatoire : Tableau de chaînes UTF-8, avec 100 chaînes maximum.

  Une liste des noms de colonnes.

Réponse

- `ColumnStatisticsList` : un tableau d'objets ColumnStatistics.

  Liste des ColumnStatistics.
- `Errors` : un tableau d'objets ColumnError.

  La liste de ces ColumnStatistics informations n'a pas pu être récupérée.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`

- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`

UpdateColumnStatisticsForTable action (Python : update_column_statistics_for_table)

Crée ou met à jour les statistiques de table des colonnes.

L'autorisation Identity and Access Management (IAM) requise pour cette opération est `UpdateTable`.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données où résident les partitions en question. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.
- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la base de données de catalogue où résident les partitions.
- `TableName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table des partitions.
- `ColumnStatisticsList` – Obligatoire : Un tableau d'objets ColumnStatistics, 25 structures maximum.

  Liste des statistiques de la colonne.

Réponse

- `Errors` : un tableau d'objets ColumnStatisticsError.

  Liste des ColumnStatisticsErrors.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`

DeleteColumnStatisticsForTable action (Python : delete_column_statistics_for_table)

Récupère les statistiques de table des colonnes.

L'autorisation Identity and Access Management (IAM) requise pour cette opération est `DeleteTable`.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données où résident les partitions en question. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.
- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la base de données de catalogue où résident les partitions.
- `TableName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table des partitions.
- `ColumnName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Le nom de la colonne.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`

API de partition

L'API de partition décrit les types de données et les opérations qui permettent de travailler avec les partitions.

Types de données

- Structure de partition
- PartitionInput structure
- PartitionSpecWithSharedStorageDescriptor structure
- PartitionListComposingSpec structure
- PartitionSpecProxy structure
- PartitionValueList structure
- Structure d'un segment
- PartitionError structure
- BatchUpdatePartitionFailureEntry structure
- BatchUpdatePartitionRequestEntry structure
- StorageDescriptor structure
- SchemaReference structure
- SerDeInfo structure
- SkewedInfo structure

Structure de partition

Représente une tranche de données d'une table.

Champs

- `Values` – Tableau de chaînes UTF-8.

  Valeurs de la partition.

- `DatabaseName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la base de données du catalogue dans laquelle créer la partition.

- `TableName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table de base de données dans laquelle créer la partition.

- `CreationTime` – Horodatage.

  Heure à laquelle la partition a été créée.

- `LastAccessTime` – Horodatage.

  Dernière date d'accès à la partition.

- `StorageDescriptor` – Un objet StorageDescriptor.

  Fournit des informations sur l'emplacement physique où la partition est stockée.

- `Parameters` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne de clé, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 512 000 octets.

  Ces paires clé-valeur définissent les paramètre de partition.

- `LastAnalyzedTime` – Horodatage.

  Dernière heure à laquelle les statistiques de colonne ont été calculées pour cette partition.

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données dans lequel réside la partition.

PartitionInput structure

Structure utilisée pour créer et mettre à jour une partition.

Champs

- `Values` – Tableau de chaînes UTF-8.

  Valeurs de la partition. Bien que ce paramètre ne soit pas requis par le SDK, vous devez spécifier ce paramètre pour une entrée valide.

  Les valeurs des clés de la nouvelle partition doivent être transmises sous la forme d'un tableau d'objets String qui doivent être classés dans le même ordre que les clés de partition qui apparaissent dans le préfixe Amazon S3. Sinon, les valeurs AWS Glue seront ajoutées aux mauvaises clés.
- `LastAccessTime` – Horodatage.

  Dernière date d'accès à la partition.
- `StorageDescriptor` – Un objet StorageDescriptor.

  Fournit des informations sur l'emplacement physique où la partition est stockée.
- `Parameters` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne de clé, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 512 000 octets.

  Ces paires clé-valeur définissent les paramètre de partition.
- `LastAnalyzedTime` – Horodatage.

  Dernière heure à laquelle les statistiques de colonne ont été calculées pour cette partition.

PartitionSpecWithSharedStorageDescriptor structure

Spécification de partition pour les partitions qui partagent un emplacement physique.

Champs

- `StorageDescriptor` – Un objet StorageDescriptor.

Informations sur le partage du stockage physique.

- `Partitions` : un tableau d'objets Partition.

  Liste des partitions qui partagent cet emplacement physique.

PartitionListComposingSpec structure

Répertorie les partitions connexes.

Champs

- `Partitions` : un tableau d'objets Partition.

  Liste des partitions dans la spécification de composition.

PartitionSpecProxy structure

Fournit un chemin d'accès racine aux partitions spécifiées.

Champs

- `DatabaseName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Base de données du catalogue dans laquelle résident les partitions.
- `TableName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table contenant les partitions.
- `RootPath` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chemin d'accès racine du proxy pour l'adressage des partitions.
- `PartitionSpecWithSharedSD` – Un objet PartitionSpecWithSharedStorageDescriptor.

  Spécification des partitions qui partagent le même emplacement de stockage physique.
- `PartitionListComposingSpec` – Un objet PartitionListComposingSpec.

  Spécifie une liste de partitions.

PartitionValueList structure

Contient une liste de valeurs définissant les partitions.

Champs

- `Values` – Obligatoire : Tableau de chaînes UTF-8.

  Liste de valeurs.

Structure d'un segment

Définit une région sans chevauchement des partitions d'une table, ce qui permet l'exécution en parallèle de plusieurs requêtes.

Champs

- `SegmentNumber` – Obligatoire : Nombre (entier), pas plus qu'Aucun.

  Numéro d'index de base zéro du segment. Par exemple, si le nombre total de segments est 4, les valeurs `SegmentNumber` vont de 0 à 3.
- `TotalSegments` – Obligatoire : Nombre (entier), compris entre 1 et 10.

  Nombre total de segments.

PartitionError structure

Contient les informations sur une erreur de partition.

Champs

- `PartitionValues` – Tableau de chaînes UTF-8.

  Valeurs qui définissent la partition.
- `ErrorDetail` – Un objet ErrorDetail.

  Détails sur l'erreur de partition.

BatchUpdatePartitionFailureEntry structure

Contient des informations sur une erreur de partition de mise à jour par lots.

Champs

- `PartitionValueList` – Tableau de chaînes UTF-8, avec 100 chaînes maximum.

  Liste des valeurs définissant la partition.
- `ErrorDetail` – Un objet ErrorDetail.

  Détails sur l'erreur de partition de mise à jour par lots.

BatchUpdatePartitionRequestEntry structure

Structure contenant les valeurs et la structure utilisées pour mettre à jour une partition.

Champs

- `PartitionValueList` – Obligatoire : Tableau de chaînes UTF-8, avec 100 chaînes maximum.

  Liste des valeurs définissant la partition.
- `PartitionInput` – Obligatoire : un objet PartitionInput.

  Structure utilisée pour mettre à jour une partition.

StorageDescriptor structure

Décrit le stockage physique des données de table.

Champs

- `Columns` : un tableau d'objets Colonne.

  Liste des `Columns` de la table.
- `Location` – Chaîne de localisation, d'une longueur maximale de 2 056 octets, correspondant au URI address multi-line string pattern.

  Emplacement physique de la table. Par défaut, il prend la forme de l'emplacement de entrepôt, suivie de l'emplacement de la base de données dans l'entrepôt, suivi du nom de la table.

- `AdditionalLocations` – Tableau de chaînes UTF-8.

  Liste des emplacements pointant vers le chemin d'accès où se trouve une table Delta.

- `InputFormat` – Chaîne de format, d'une longueur maximale de 128 octets, correspondant au Single-line string pattern.

  Format d'entrée : `SequenceFileInputFormat` (binaire) ou `TextInputFormat` ou format personnalisé.

- `OutputFormat` – Chaîne de format, d'une longueur maximale de 128 octets, correspondant au Single-line string pattern.

  Format de sortie : `SequenceFileOutputFormat` (binaire) ou `IgnoreKeyTextOutputFormat` ou format personnalisé.

- `Compressed` : booléen.

  `True` si les données de la table sont compressées ou `False` si elles ne le sont pas.

- `NumberOfBuckets` : nombre (entier).

  Doit être spécifié si la table contient des colonnes de dimension.

- `SerdeInfo` – Un objet SerDeInfo.

  Les serialization/deserialization (SerDe) informations.

- `BucketColumns` – Tableau de chaînes UTF-8.

  Liste de colonnes de regroupement de réducteur, de colonnes de clustérisation et de colonnes de mise en compartiment de la table.

- `SortColumns` : un tableau d'objets Ordre.

  Liste spécifiant l'ordre de tri de chaque compartiment de la table.

- `Parameters` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne de clé, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 512 000 octets.

  Propriétés fournies par l'utilisateur sous la forme clé-valeur.

- `SkewedInfo` – Un objet SkewedInfo.

Informations concernant les valeurs qui apparaissent fréquemment dans une colonne (valeurs asymétriques).

- `StoredAsSubDirectories` : booléen.

  `True` si les données de la table sont stockées dans les sous-répertoires ou `False` dans le cas contraire.

- `SchemaReference` – Un objet SchemaReference.

  Objet qui fait référence à un schéma stocké dans le registre des AWS Glue schémas.

  Lors de la création d'une table, vous pouvez transmettre une liste vide de colonnes pour le schéma et plutôt utiliser une référence de schéma.

SchemaReference structure

Objet qui fait référence à un schéma stocké dans le registre des AWS Glue schémas.

Champs

- `SchemaId` – Un objet SchemaId.

  Structure qui contient des champs d'identité de schéma. Ceci ou `SchemaVersionId` doit être fourni.

- `SchemaVersionId` – Obligatoire : chaîne UTF-8, d'une longueur exacte de 36 octets, correspondant au Custom string pattern #45.

  ID unique affecté à une version du schéma. Ceci ou `SchemaId` doit être fourni.

- `SchemaVersionNumber` – Nombre (long), compris entre 1 et 100 000.

  Numéro de version du schéma.

SerDeInfo structure

Informations sur un serialization/deserialization programme (SerDe) qui sert d'extracteur et de chargeur.

Champs

- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du SerDe.
- `SerializationLibrary` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Généralement, la classe qui implémente le SerDe. Par exemple : `org.apache.hadoop.hive.serde2.columnar.ColumnarSerDe`.
- `Parameters` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne de clé, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 512 000 octets.

  Ces paires clé-valeur définissent les paramètres d'initialisation pour. SerDe

SkewedInfo structure

Spécifie les valeurs biaisées d'une table. Les valeurs biaisées sont celles qui se produisent à très haute fréquence.

Champs

- `SkewedColumnNames` – Tableau de chaînes UTF-8.

  Liste des noms de colonnes qui contiennent des valeurs biaisées.
- `SkewedColumnValues` – Tableau de chaînes UTF-8.

  Liste des valeurs qui apparaissent si fréquemment qu'elles sont considérées comme biaisées.
- `SkewedColumnValueLocationMaps` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8.

  Chaque valeur est une chaîne UTF-8.

  Mappage de valeurs biaisées avec les colonnes qui les contiennent.

Opérations

- CreatePartition action (Python : créer_partition)
- BatchCreatePartition action (Python : batch_create_partition)
- UpdatePartition action (Python : update_partition)
- DeletePartition action (Python : supprimer_partition)
- BatchDeletePartition action (Python : batch_supprimer_partition)
- GetPartition action (Python : get_partition)
- GetPartitions action (Python : get_partitions)
- BatchGetPartition action (Python : batch_get_partition)
- BatchUpdatePartition action (Python : batch_update_partition)
- GetColumnStatisticsForPartition action (Python : get_column_statistics_for_partition)
- UpdateColumnStatisticsForPartition action (Python : update_column_statistics_for_partition)
- DeleteColumnStatisticsForPartition action (Python : supprimer_column_statistics_for_partition)

CreatePartition action (Python : créer_partition)

Crée une partition.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID de AWS compte du catalogue dans lequel la partition doit être créée.
- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la base de données de métadonnées dans laquelle la partition doit être créée.
- `TableName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table de métadonnées dans laquelle la partition doit être créée.
- `PartitionInput` – Obligatoire : un objet PartitionInput.

Structure `PartitionInput`définissant la partition à créer.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `InvalidInputException`
- `AlreadyExistsException`
- `ResourceNumberLimitExceededException`
- `InternalServiceException`
- `EntityNotFoundException`
- `OperationTimeoutException`
- `GlueEncryptionException`

BatchCreatePartition action (Python : batch_create_partition)

Crée une ou plusieurs partitions dans une opération par lot.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue dans lequel la partition doit être créée. Actuellement, il devrait s'agir de l'identifiant du AWS compte.
- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la base de données de métadonnées dans laquelle la partition doit être créée.
- `TableName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table de métadonnées dans laquelle la partition doit être créée.
- `PartitionInputList` – Obligatoire : Un tableau d'objets PartitionInput, 100 structures maximum.

Liste de structures `PartitionInput` qui définissent les partitions à créer.

Réponse

- `Errors` : un tableau d'objets PartitionError.

  Erreurs survenues pendant la tentative de création des partitions demandées.

Erreurs

- `InvalidInputException`
- `AlreadyExistsException`
- `ResourceNumberLimitExceededException`
- `InternalServiceException`
- `EntityNotFoundException`
- `OperationTimeoutException`
- `GlueEncryptionException`

UpdatePartition action (Python : update_partition)

Met à jour une partition.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données où réside la partition à mettre à jour. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.
- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la base de données du catalogue où se trouve la table en question.
- `TableName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

Nom de la table où se trouve la partition à mettre à jour.

- `PartitionValueList` – Obligatoire : Tableau de chaînes UTF-8, avec 100 chaînes maximum.

  Liste des valeurs de clé de partition qui définissent la partition à mettre à jour.

- `PartitionInput` – Obligatoire : un objet PartitionInput.

  Nouvel objet de partition où mettre à jour la partition.

  Cette propriété `Values` ne peut pas être modifiée. Si vous souhaitez modifier les valeurs de clé de partition d'une partition, supprimez et recréez la partition.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`

DeletePartition action (Python : supprimer_partition)

Supprime une partition spécifiée.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données où réside la partition à supprimer. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.

- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

Nom de la base de données du catalogue où se trouve la table en question.

- `TableName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table qui contient la partition à supprimer.

- `PartitionValues` – Obligatoire : Tableau de chaînes UTF-8.

  Valeurs qui définissent la partition.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

BatchDeletePartition action (Python : batch_supprimer_partition)

Supprime une ou plusieurs partitions dans une opération par lot.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données où réside la partition à supprimer. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.

- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la base de données du catalogue où se trouve la table en question.

- `TableName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table contenant les partitions à supprimer.

- `PartitionsToDelete` – Obligatoire : Un tableau d'objets PartitionValueList, 25 structures maximum.

  Liste de structures `PartitionInput` qui définissent les partitions à supprimer.

Réponse

- `Errors` : un tableau d'objets PartitionError.

  Erreurs survenues pendant la tentative de suppression des partitions demandées.

Erreurs

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`

GetPartition action (Python : get_partition)

Extrait les informations sur une partition spécifiée.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données où réside la partition en question. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.

- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la base de données de catalogue où réside la partition.

- `TableName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table des partitions.
- `PartitionValues` – Obligatoire : Tableau de chaînes UTF-8.

  Valeurs qui définissent la partition.

Réponse

- `Partition` – Un objet Partition.

  Informations demandées, sous la forme d'un objet `Partition`.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`
- `FederationSourceException`
- `FederationSourceRetryableException`

GetPartitions action (Python : get_partitions)

Extrait les informations sur les partitions d'une table.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données où résident les partitions en question. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.
- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

Nom de la base de données de catalogue où résident les partitions.

- `TableName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table des partitions.

- `Expression` – Chaîne de prédicat, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Expression qui filtre les partitions à renvoyer.

  L'expression utilise une syntaxe SQL similaire à la clause de filtre `WHERE` SQL. L'analyseur d'instructions SQL JSQLParseranalyse l'expression.

  Opérateurs : voici les opérateurs que vous pouvez utiliser dans l'appel d'API `Expression` :

  =

  Vérifie si les valeurs des deux opérandes sont égales ; si c'est le cas, la condition est vraie.

  Exemple : supposons que la « variable a » et la « variable b » sont de 20.

  (a = b) n'est pas vrai.

  < >

  Vérifie si les valeurs des deux opérandes sont égales ; si ce n'est pas le cas, la condition est vraie.

  Exemple : (a < > b) est vrai.

  >

  Vérifie si la valeur de l'opérande gauche est supérieure à la valeur de l'opérande droite ; si c'est le cas, la condition est vraie.

  Exemple : (a > b) n'est pas vrai.

  <

  Vérifie si la valeur de l'opérande gauche est inférieure à la valeur de l'opérande droite ; si c'est le cas, la condition est vraie.

  Exemple : (a < b) est vrai.

>=

Vérifie si la valeur de l'opérande gauche est supérieure ou égale à la valeur de l'opérande droite ; si c'est le cas, la condition est vraie.

Exemple : (a **>=** b) n'est pas vrai.

<=

Vérifie si la valeur de l'opérande gauche est inférieure ou égale à la valeur de l'opérande droite ; si c'est le cas, la condition est vraie.

Exemple : (a **<=** b) est vrai.

AND, OR, IN, BETWEEN, LIKE, NOT, IS NULL

Opérateurs logiques.

Types de clés de partition prises en charge : les clés de partition prises en charge sont énoncées ci-dessous.

- `string`
- `date`
- `timestamp`
- `int`
- `bigint`
- `long`
- `tinyint`
- `smallint`
- `decimal`

Si un type non valide est rencontré, une exception est levée.

La liste suivante présente les opérateurs valides sur chaque type. Lorsque vous définissez un crawler, le type `partitionKey` est créé en tant que `STRING`, pour être compatible avec le catalogue de partitions.

Exemple d'appel d'API :

Example

La table `twitter_partition` comprend trois partitions :

```
year = 2015
        year = 2016
        year = 2017
```

Example

Obtenir la partition `year` équivalant à 2015

```
aws glue get-partitions --database-name dbname --table-name twitter_partition
        --expression "year*=*'2015'"
```

Example

Obtenir la partition `year` entre 2016 et 2018 (exclusif)

```
aws glue get-partitions --database-name dbname --table-name twitter_partition
        --expression "year>'2016' AND year<'2018'"
```

Example

Obtenir la partition `year` entre 2015 et 2018 (inclusif). Les appels d'API suivants sont équivalents :

```
aws glue get-partitions --database-name dbname --table-name twitter_partition
        --expression "year>='2015' AND year<='2018'"

        aws glue get-partitions --database-name dbname --table-name
 twitter_partition
        --expression "year BETWEEN 2015 AND 2018"

        aws glue get-partitions --database-name dbname --table-name
 twitter_partition
        --expression "year IN (2015,2016,2017,2018)"
```

Example

Une partition à caractère générique où la sortie d'appel suivante est la partition année = 2017. Une expression régulière n'est pas prise en charge dans `LIKE`.

```
aws glue get-partitions --database-name dbname --table-name twitter_partition
        --expression "year LIKE '%7'"
```

- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, si ce n'est pas le premier appel pour récupérer ces partitions.

- `Segment` – Un objet Segment.

  Segment des partitions de la table à analyser dans cette demande.

- `MaxResults` – Nombre (entier), compris entre 1 et 1 000.

  Nombre maximal de partitions à renvoyer dans une seule réponse.

- `ExcludeColumnSchema` : booléen.

  Lorsque la valeur est « true (VRAI) », indique de ne pas renvoyer le schéma de la colonne de partition. Utile lorsque vous êtes intéressé uniquement par d'autres attributs de partition tels que les valeurs de partition ou l'emplacement. En ne renvoyant pas de données en double, cette approche évite le problème d'une grande réponse.

- `TransactionId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #44.

  L'ID de transaction auquel lire le contenu de la partition.

- `QueryAsOfTime` – Horodatage.

  L'Heure à laquelle lire le contenu de la partition. S'il n'est pas défini, l'heure de validation de transaction la plus récente sera utilisée. Ne peut pas être spécifié avec `TransactionId`.

Réponse

- `Partitions` : un tableau d'objets Partition.

  Liste des partitions demandées.

- `NextToken` – Chaîne UTF-8.

Jeton de continuation, si la liste de partitions renvoyée n'inclut pas la dernière partition.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`
- `GlueEncryptionException`
- `InvalidStateException`
- `ResourceNotReadyException`
- `FederationSourceException`
- `FederationSourceRetryableException`

BatchGetPartition action (Python : batch_get_partition)

Extrait les partitions d'une demande par lot.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données où résident les partitions en question. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.
- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la base de données de catalogue où résident les partitions.
- `TableName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table des partitions.
- `PartitionsToGet` – Obligatoire : Un tableau d'objets PartitionValueList, 1000 structures maximum.

Liste des valeurs de partition qui identifie les partitions à récupérer.

Réponse

- `Partitions` : un tableau d'objets Partition.

  Liste des partitions demandées.
- `UnprocessedKeys` – Un tableau d'objets PartitionValueList, 1000 structures maximum.

  Liste des valeurs de partition de la demande pour laquelle les partitions n'ont pas été retournées.

Erreurs

- `InvalidInputException`
- `EntityNotFoundException`
- `OperationTimeoutException`
- `InternalServiceException`
- `GlueEncryptionException`
- `InvalidStateException`
- `FederationSourceException`
- `FederationSourceRetryableException`

BatchUpdatePartition action (Python : batch_update_partition)

Met à jour une ou plusieurs partitions dans une opération par lot.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue dans lequel la partition doit être mise à jour. Actuellement, il devrait s'agir de l'identifiant du AWS compte.
- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

Nom de la base de données de métadonnées dans laquelle la partition doit être mise à jour.

- `TableName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table de métadonnées dans laquelle la partition doit être mise à jour.

- `Entries` – Obligatoire : Tableau d'objets BatchUpdatePartitionRequestEntry, 1 structure minimum et 100 structures maximum.

  Une liste de jusqu'à 100 objets `BatchUpdatePartitionRequestEntry` à mettre à jour.

Réponse

- `Errors` : un tableau d'objets BatchUpdatePartitionFailureEntry.

  Erreurs survenues pendant la tentative de mise à jour des partitions demandées. Liste d'objets `BatchUpdatePartitionFailureEntry`.

Erreurs

- `InvalidInputException`
- `EntityNotFoundException`
- `OperationTimeoutException`
- `InternalServiceException`
- `GlueEncryptionException`

GetColumnStatisticsForPartition action (Python : get_column_statistics_for_partition)

Récupère les statistiques de partition des colonnes.

L'autorisation Identity and Access Management (IAM) requise pour cette opération est `GetPartition`.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

ID du catalogue de données où résident les partitions en question. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.

- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la base de données de catalogue où résident les partitions.

- `TableName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table des partitions.

- `PartitionValues` – Obligatoire : Tableau de chaînes UTF-8.

  Liste des valeurs de partition qui identifie la partition.

- `ColumnNames` – Obligatoire : Tableau de chaînes UTF-8, avec 100 chaînes maximum.

  Une liste des noms de colonnes.

Réponse

- `ColumnStatisticsList` : un tableau d'objets ColumnStatistics.

  La liste n' ColumnStatistics a pas pu être récupérée.

- `Errors` : un tableau d'objets ColumnError.

  Une erreur s'est produite lors de la récupération des données statistiques de colonne.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`

UpdateColumnStatisticsForPartition action (Python : update_column_statistics_for_partition)

Crée ou met à jour les statistiques de partition des colonnes.

L'autorisation Identity and Access Management (IAM) requise pour cette opération est `UpdatePartition`.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données où résident les partitions en question. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.

- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la base de données de catalogue où résident les partitions.

- `TableName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table des partitions.

- `PartitionValues` – Obligatoire : Tableau de chaînes UTF-8.

  Liste des valeurs de partition qui identifie la partition.

- `ColumnStatisticsList` – Obligatoire : Un tableau d'objets ColumnStatistics, 25 structures maximum.

  Liste des statistiques de la colonne.

Réponse

- `Errors` : un tableau d'objets ColumnStatisticsError.

  Une erreur s'est produite lors de la mise à jour des données statistiques.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`

DeleteColumnStatisticsForPartition action (Python : supprimer_column_statistics_for_partition)

Supprime les statistiques de colonne de partition d'une colonne.

L'autorisation Identity and Access Management (IAM) requise pour cette opération est `DeletePartition`.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données où résident les partitions en question. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.
- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la base de données de catalogue où résident les partitions.
- `TableName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table des partitions.
- `PartitionValues` – Obligatoire : Tableau de chaînes UTF-8.

  Liste des valeurs de partition qui identifie la partition.
- `ColumnName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la colonne.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`

API de connexions

L'API Connections décrit les types de données et l'API liés à l'utilisation des connexions dans AWS Glue.

Rubriques

- API de connexion
- API de types de connexion
- Métadonnées de connexion et API de prévisualisation

API de connexion

L'API de connexion décrit les types de données de AWS Glue connexion, ainsi que l'API permettant de créer, de supprimer, de mettre à jour et de répertorier les connexions.

Types de données

- Structure de connexion
- ConnectionInput structure
- TestConnectionInput structure
- PhysicalConnectionRequirements structure
- GetConnectionsFilter structure
- AuthenticationConfiguration structure

- AuthenticationConfigurationInput structure
- OAuth2Structure des propriétés
- OAuth2PropertiesInput structure
- OAuth2ClientApplication structure
- AuthorizationCodeProperties structure
- BasicAuthenticationCredentials structure
- OAuth2Structure des informations d'identification

Structure de connexion

Définit une connexion à une source de données.

Champs

- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la définition de connexion.
- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description de la connexion.
- `ConnectionType`— Chaîne UTF-8 (valeurs valides : `JDBC` | `SFTP` | `MONGODB` | `KAFKA` | `NETWORK` | `MARKETPLACE` | `CUSTOM` | `SALESFORCE` | `VIEW_VALIDATION_REDSHIFT` | `VIEW_VALIDATION_ATHENA` | `GOOGLEADS` | `GOOGLESHEETS` | | `GOOGLEANALYTICS4` | `SERVICENOW` | `MARKETO` | `SAPODATA` | `ZENDESK` | `JIRACLOUD` | `NETSUITEERP` | `HUBSPOT` | `FACEBOOKADS` | `INSTAGRAMADS` | `ZOHOCRM` | `SALESFORCEPARDOT` | `SALESFORCEMARKETINGCLOUD` | `ADOBEANALYTICS` | `SLACK` | `LINKEDIN` | `MIXPANEL` | `ASANA` | `STRIPE` | `SMARTSHEET` | `DATADOG` | `WOOCOMMERCE INTERCOM SNAPCHATADS`| `PAYPAL` | `QUICKBOOKS` | `FACEBOOKPAGEINSIGHTS` | `FRESHDESK` | `TWILIO` | `DOCUSIGNMONITOR` | `FRESHSALES` | `ZOOM GOOGLESEARCHCONSOLE` | `SALESFORCECOMMERCECLOUD` | `SAPCONCUR` | `DYNATRACE` | `MICROSOFTDYNAMIC365FINANCEANDOPS` | `MICROSOFTTEAMS` | `BLACKBAUDRAISEREDGENXT` | `MAILCHIMP` | `GITLAB` | `PENDO` | `PRODUCTBOARD` | `CIRCLECI` | `PIPEDIVE` | `SENDGRID` | `AZURECOSMOS` | `AZURESQL` | `BIGQUERY` | `BLACKBAUD` | `CLOUDERAHIVE` `CLOUDERAIMPALA` | `CLOUDWATCH` | `CLOUDWATCHMETRICS` | `CMDB` | `DATALAKEGEN2` | `DB2` | `DB2AS400` | `DOCUMENTDB` | `DOMO` | `DYNAMODB` | `GOOGLECLOUDSTORAGE`| `HBASE KUSTOMER` |

MICROSOFTDYNAMICS365CRM | MONDAY | MYSQL | OKTA OPENSEARCH | ORACLE | PIPEDRIVE | POSTGRESQL | SAPHANA SQLSERVER | SYNAPSE | TERADATA | TERADATANOS | TIMESTREAM TPCDS |VERTICA).

Type de connexion. Actuellement, SFTP n'est pas pris en charge.

- `MatchCriteria` : tableau de chaînes UTF-8, avec 10 chaînes maximum.

  Liste des critères qui peuvent être utilisés dans la sélection de cette connexion.

- `ConnectionProperties` – Tableau de mappage de paires clé-valeur, 100 paires au maximum.

  Chaque clé est une chaîne UTF-8 (valeurs valides : HOST | PORT | USERNAME="USER_NAME" | PASSWORD | ENCRYPTED_PASSWORD | JDBC_DRIVER_JAR_URI | JDBC_DRIVER_CLASS_NAME | JDBC_ENGINE | JDBC_ENGINE_VERSION | CONFIG_FILES | INSTANCE_ID | JDBC_CONNECTION_URL | JDBC_ENFORCE_SSL | CUSTOM_JDBC_CERT | SKIP_CUSTOM_JDBC_CERT_VALIDATION | CUSTOM_JDBC_CERT_STRING | CONNECTION_URL | KAFKA_BOOTSTRAP_SERVERS | KAFKA_SSL_ENABLED | KAFKA_CUSTOM_CERT | KAFKA_SKIP_CUSTOM_CERT_VALIDATION | KAFKA_CLIENT_KEYSTORE | KAFKA_CLIENT_KEYSTORE_PASSWORD | KAFKA_CLIENT_KEY_PASSWORD | ENCRYPTED_KAFKA_CLIENT_KEYSTORE_PASSWORD | ENCRYPTED_KAFKA_CLIENT_KEY_PASSWORD | KAFKA_SASL_MECHANISM | KAFKA_SASL_PLAIN_USERNAME | KAFKA_SASL_PLAIN_PASSWORD | ENCRYPTED_KAFKA_SASL_PLAIN_PASSWORD | KAFKA_SASL_SCRAM_USERNAME | KAFKA_SASL_SCRAM_PASSWORD | KAFKA_SASL_SCRAM_SECRETS_ARN | ENCRYPTED_KAFKA_SASL_SCRAM_PASSWORD | KAFKA_SASL_GSSAPI_KEYTAB | KAFKA_SASL_GSSAPI_KRB5_CONF | KAFKA_SASL_GSSAPI_SERVICE | KAFKA_SASL_GSSAPI_PRINCIPAL | SECRET_ID | CONNECTOR_URL | CONNECTOR_TYPE | CONNECTOR_CLASS_NAME | ENDPOINT | ENDPOINT_TYPE | ROLE_ARN | REGION | WORKGROUP_NAME | CLUSTER_IDENTIFIER | DATABASE).

  Chaque valeur est une chaîne Valeur, d'une longueur comprise entre 1 et 1 024 octets.

  Ces paires clé-valeur définissent les paramètres pour la connexion lors de l'utilisation du schéma de connexion de la version 1 :

  - HOST- L'URI de l'hôte : soit le nom de domaine complet (FQDN), soit l' IPv4 adresse de l'hôte de la base de données.
  - PORT - Numéro du port, compris entre 1 024 et 65 535, sur lequel l'hôte de base de données écoute les connexions de la base de données.

- USER_NAME - Nom sous lequel vous vous connectez à la base de données. La valeur de chaîne pour USER_NAME est « USERNAME ».
- PASSWORD - Mot de passe, le cas échéant, pour le nom d'utilisateur.
- ENCRYPTED_PASSWORD - Lorsque vous activez la protection de mot de passe en réglant ConnectionPasswordEncryption dans les paramètres de chiffrage de Catalogue de données, ce champ stocke le mot de passe chiffré.
- JDBC_DRIVER_JAR_URI - Chemin d'accès Amazon Simple Storage Service (Amazon S3) du fichier JAR qui contient le pilote JDBC à utiliser.
- JDBC_DRIVER_CLASS_NAME - Nom de classe du pilote JDBC à utiliser.
- JDBC_ENGINE – Nom du moteur JDBC à utiliser.
- JDBC_ENGINE_VERSION - Version du moteur JDBC à utiliser.
- CONFIG_FILES - (Réservé pour une utilisation ultérieure.)
- INSTANCE_ID - ID d'instance à utiliser.
- JDBC_CONNECTION_URL - URL de connexion à une source de données JDBC.
- JDBC_ENFORCE_SSL : chaîne booléenne insensible à la casse (true, false) spécifiant si Secure Sockets Layer (SSL) avec le nom d'hôte correspondant est appliqué pour la connexion JDBC sur le client. La valeur par défaut est false.
- CUSTOM_JDBC_CERT- Un emplacement Amazon S3 spécifiant le certificat racine du client. AWS Glue utilise ce certificat racine pour valider le certificat du client lors de la connexion à la base de données clients. AWS Glue gère uniquement les certificats X.509. Le certificat fourni doit être codé DER et fourni au format PEM d'encodage Base64.
- SKIP_CUSTOM_JDBC_CERT_VALIDATION- Par défaut, c'est le casfalse. AWS Glue valide l'algorithme de signature et l'algorithme de clé publique du sujet pour le certificat client. Les seuls algorithmes autorisés pour l'algorithme de signature sont SHA256with RSA, RSA ou SHA384with SHA512with RSA. Pour l'algorithme de clé publique d'objet, la longueur de clé doit être d'au moins 2048. Vous pouvez mettre la valeur de cette propriété à true pour ignorer la validation par AWS Glue du certificat du client.
- CUSTOM_JDBC_CERT_STRING- Une chaîne de certificat JDBC personnalisée qui est utilisée pour la correspondance de domaine ou de nom distinctif afin de prévenir une man-in-the-middle attaque. Dans Oracle Database, elle est utilisée comme SSL_SERVER_CERT_DN. Dans Microsoft SQL Server, elle est utilisée comme hostNameInCertificate.
- CONNECTION_URL - URL de connexion à une source de données générale (non JDBC).
- SECRET_ID - L'ID secret utilisé pour le gestionnaire secret des informations d'identification.

- `CONNECTOR_URL` - L'URL du connecteur pour une connexion MARKETPLACE ou CUSTOM.
- `CONNECTOR_TYPE` - Le type du connecteur pour une connexion MARKETPLACE ou CUSTOM.
- `CONNECTOR_CLASS_NAME` - Le nom de la classe de connecteur pour une connexion MARKETPLACE ou CUSTOM.
- `KAFKA_BOOTSTRAP_SERVERS` - Liste séparée par des virgules composée de paires hôte et port qui sont les adresses des courtiers Apache Kafka dans un cluster Kafka auquel un client Kafka se connecte pour démarrer lui-même.
- `KAFKA_SSL_ENABLED` - Indique s'il convient d'activer ou de désactiver SSL sur une connexion Apache Kafka. La valeur par défaut est « true » (vrai).
- `KAFKA_CUSTOM_CERT` - L'URL d'Amazon S3 pour le fichier de certificat de l'autorité de certification privée (format .pem). La valeur par défaut est une chaîne vide.
- `KAFKA_SKIP_CUSTOM_CERT_VALIDATION`- S'il faut ou non ignorer la validation du fichier de certification CA. AWS Glue valide pour trois algorithmes : SHA256with RSA, RSA et SHA384with RSA. SHA512with La valeur par défaut est « false » (faux).
- `KAFKA_CLIENT_KEYSTORE` - Emplacement Amazon S3 du fichier de magasin de clés client pour l'authentification côté client Kafka (facultatif).
- `KAFKA_CLIENT_KEYSTORE_PASSWORD` - Le mot de passe pour accéder au magasin de clés fourni (facultatif).
- `KAFKA_CLIENT_KEY_PASSWORD` - Un magasin de clés peut être composé de plusieurs clés, il s'agit donc du mot de passe pour accéder à la clé client à utiliser avec la clé côté serveur Kafka (facultatif).
- `ENCRYPTED_KAFKA_CLIENT_KEYSTORE_PASSWORD`- La version cryptée du mot de passe du keystore du client Kafka (si l'utilisateur a sélectionné le paramètre de AWS Glue chiffrement des mots de passe).
- `ENCRYPTED_KAFKA_CLIENT_KEY_PASSWORD`- La version cryptée du mot de passe clé du client Kafka (si l'utilisateur a sélectionné le paramètre de AWS Glue cryptage des mots de passe).
- `KAFKA_SASL_MECHANISM`- `"SCRAM-SHA-512"` `"GSSAPI"`,`"AWS_MSK_IAM"`, ou`"PLAIN"`. Ce sont les mécanismes SASL pris en charge.
- `KAFKA_SASL_PLAIN_USERNAME` : nom d'utilisateur en texte clair utilisé pour s'authentifier avec le mécanisme « PLAIN ».
- `KAFKA_SASL_PLAIN_PASSWORD` : mot de passe en texte clair utilisé pour s'authentifier avec le mécanisme « PLAIN ».

- `ENCRYPTED_KAFKA_SASL_PLAIN_PASSWORD`- La version cryptée du mot de passe Kafka SASL PLAIN (si l'utilisateur a sélectionné le paramètre de AWS Glue cryptage des mots de passe).
- `KAFKA_SASL_SCRAM_USERNAME` - Un nom d'utilisateur en texte clair utilisé pour s'authentifier avec le mécanisme « SCRAM-SHA-512 ».
- `KAFKA_SASL_SCRAM_PASSWORD` - Un mot de passe en texte clair utilisé pour s'authentifier avec le mécanisme « SCRAM-SHA-512 ».
- `ENCRYPTED_KAFKA_SASL_SCRAM_PASSWORD`- La version cryptée du mot de passe Kafka SASL SCRAM (si l'utilisateur a sélectionné le paramètre de AWS Glue cryptage des mots de passe).
- `KAFKA_SASL_SCRAM_SECRETS_ARN`- Le nom de ressource Amazon d'un secret dans AWS Secrets Manager.
- `KAFKA_SASL_GSSAPI_KEYTAB` - Emplacement S3 d'un fichier Kerberos `keytab`. Un keytab stocke les clés à long terme pour un ou plusieurs principaux. Pour en savoir plus, consultez Documentation du MIT Kerberos : Keytab.
- `KAFKA_SASL_GSSAPI_KRB5_CONF` - Emplacement S3 d'un fichier Kerberos `krb5.conf`. Un fichier krb5.conf stocke les informations de configuration Kerberos, telles que l'emplacement du serveur KDC. Pour en savoir plus, consultez Documentation MIT Kerberos : krb5.conf.
- `KAFKA_SASL_GSSAPI_SERVICE` - Le nom du service Kerberos, tel que défini avec `sasl.kerberos.service.name` dans votre Configuration Kafka.
- `KAFKA_SASL_GSSAPI_PRINCIPAL`- Le nom du principal Kerberos utilisé par. AWS Glue Pour plus d'informations, consultez Documentation Kafka : configuration des courtiers Kafka.
- `ROLE_ARN` : rôle à utiliser pour exécuter des requêtes.
- `REGION`- La AWS région dans laquelle les requêtes seront exécutées.
- `WORKGROUP_NAME` : nom d'un groupe de travail Amazon Redshift sans serveur ou d'un groupe de travail Amazon Athena dans lequel les requêtes seront exécutées.
- `CLUSTER_IDENTIFIER` : identifiant de cluster d'un cluster Amazon Redshift dans lequel les requêtes seront exécutées.
- `DATABASE` : base de données Amazon Redshift à laquelle vous vous connectez.

- `SparkProperties` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est une chaîne UTF-8, d'une longueur comprise entre 1 et 2 048 octets.

Propriétés de connexion propres à l'environnement de calcul Spark.

- `AthenaProperties` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est une chaîne UTF-8, d'une longueur comprise entre 1 et 2 048 octets.

  Propriétés de connexion propres à l'environnement de calcul Athena.

- `PythonProperties` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est une chaîne UTF-8, d'une longueur comprise entre 1 et 2 048 octets.

  Propriétés de connexion propres à l'environnement de calcul Python.

- `PhysicalConnectionRequirements` – Un objet PhysicalConnectionRequirements.

  Les exigences de connexion physique, comme un cloud privé virtuel (VPC) et `SecurityGroup`, qui sont nécessaires pour établir la connexion.

- `CreationTime` – Horodatage.

  Horodatage de l'heure à laquelle la définition de connexion a été créée.

- `LastUpdatedTime` – Horodatage.

  Horodatage de la dernière date à laquelle la définition de connexion a été mise à jour.

- `LastUpdatedBy` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  L'utilisateur, groupe ou rôle qui a mis à jour la définition de connexion pour la dernière fois.

- `Status` – Chaîne UTF-8 (valeurs valides : `READY` | `IN_PROGRESS` | `FAILED`).

  État de la connexion. Peut avoir l'une des valeurs suivantes : `READY`, `IN_PROGRESS` ou `FAILED`.

- `StatusReason` – Chaîne UTF-8, d'une longueur comprise entre 1 et 16 384 octets.

  Motif de l'état de la connexion.

- `LastConnectionValidationTime` – Horodatage.

  Horodatage de la dernière date de validation de cette connexion.

- `AuthenticationConfiguration` : un objet AuthenticationConfiguration.

  Les propriétés d'authentification de la connexion.

- `ConnectionSchemaVersion`— Nombre (entier), pas moins de 1 ou plus de 2.

  Version du schéma de connexion pour cette connexion. La version 2 prend en charge les propriétés pour des environnements de calcul spécifiques.

- `CompatibleComputeEnvironments` – Tableau de chaînes UTF-8.

  Liste des environnements de calcul compatibles avec la connexion.

ConnectionInput structure

Une structure utilisée pour spécifier la connexion à créer ou à mettre à jour.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la connexion.

- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description de la connexion.

- `ConnectionType`— Obligatoire : chaîne UTF-8 (valeurs valides : `JDBC SFTP MONGODB | KAFKA | NETWORK | MARKETPLACE | CUSTOM | SALESFORCE | VIEW_VALIDATION_REDSHIFT | VIEW_VALIDATION_ATHENA | | GOOGLEADS | GOOGLESHEETS | GOOGLEANALYTICS4 | SERVICENOW | MARKETO | SAPODATA | ZENDESK | JIRACLOUD | NETSUITEERP | HUBSPOT | FACEBOOKADS | INSTAGRAMADS | ZOHOCRM | SALESFORCEPARDOT | SALESFORCEMARKETINGCLOUD | ADOBEANALYTICS | SLACK | LINKEDIN | MIXPANEL | ASANA | STRIPE | SMARTSHEET DATADOG WOOCOMMERCE| INTERCOM | SNAPCHATADS | PAYPAL | QUICKBOOKS | FACEBOOKPAGEINSIGHTS | FRESHDESK | TWILIO | DOCUSIGNMONITOR FRESHSALES | ZOOM | GOOGLESEARCHCONSOLE | SALESFORCECOMMERCECLOUD | SAPCONCUR | DYNATRACE | MICROSOFTDYNAMIC365FINANCEANDOPS | MICROSOFTTEAMS | BLACKBAUDRAISEREDGENXT | MAILCHIMP | GITLAB | PENDO | PRODUCTBOARD | CIRCLECI | PIPEDIVE | SENDGRID | AZURECOSMOS | AZURESQL | BIGQUERY BLACKBAUD | CLOUDERAHIVE | CLOUDERAIMPALA | CLOUDWATCH | CLOUDWATCHMETRICS | CMDB | DATALAKEGEN2 | DB2 |`

DB2AS400 | DOCUMENTDB | DOMO| DYNAMODB | GOOGLECLOUDSTORAGE HBASE | KUSTOMER | MICROSOFTDYNAMICS365CRM | MONDAY | MYSQL | OKTA | OPENSEARCH | ORACLE PIPEDRIVE | POSTGRESQL | SAPHANA | SQLSERVER | SYNAPSE | TERADATA | TERADATANOS TIMESTREAM | TPCDS |VERTICA).

Type de connexion. Actuellement, ces types sont pris en charge :

- JDBC - Désigne une connexion à une base de données via Java Database Connectivity (JDBC).

  JDBCLes connexions utilisent les méthodes suivantes ConnectionParameters.

  - Obligatoire : toutes les valeurs (HOST, PORT, JDBC_ENGINE) ou JDBC_CONNECTION_URL.
  - Obligatoire : toutes les valeurs (USERNAME, PASSWORD) ou SECRET_ID.
  - Facultatif : JDBC_ENFORCE_SSL, CUSTOM_JDBC_CERT, CUSTOM_JDBC_CERT_STRING, SKIP_CUSTOM_JDBC_CERT_VALIDATION. Ces paramètres sont utilisés pour configurer le SSL avec JDBC.
- KAFKA - Désigne une connexion à une plateforme de streaming Apache Kafka.

  KAFKALes connexions utilisent les méthodes suivantes ConnectionParameters.

  - Obligatoire : KAFKA_BOOTSTRAP_SERVERS.
  - Facultatif : KAFKA_SSL_ENABLED, KAFKA_CUSTOM_CERT, KAFKA_SKIP_CUSTOM_CERT_VALIDATION. Ces paramètres sont utilisés pour configurer le SSL avec KAFKA.
  - Facultatif : KAFKA_CLIENT_KEYSTORE, KAFKA_CLIENT_KEYSTORE_PASSWORD, KAFKA_CLIENT_KEY_PASSWORD, ENCRYPTED_KAFKA_CLIENT_KEYSTORE_PASSWORD, ENCRYPTED_KAFKA_CLIENT_KEY_PASSWORD. Ces paramètres sont utilisés pour configurer la configuration client TLS avec SSL dans KAFKA.
  - Facultatif : KAFKA_SASL_MECHANISM. Peut être spécifié comme suit : SCRAM-SHA-512, GSSAPI ou AWS_MSK_IAM.
  - Facultatif : KAFKA_SASL_SCRAM_USERNAME, KAFKA_SASL_SCRAM_PASSWORD, ENCRYPTED_KAFKA_SASL_SCRAM_PASSWORD. Ces paramètres sont utilisés pour configurer l'authentification SASL/SCRAMSHA-512 avec KAFKA.
  - Facultatif : KAFKA_SASL_GSSAPI_KEYTAB, KAFKA_SASL_GSSAPI_KRB5_CONF, KAFKA_SASL_GSSAPI_SERVICE, KAFKA_SASL_GSSAPI_PRINCIPAL. Ces paramètres sont utilisés pour configurer SASL/GSSAPI l'authentification avecKAFKA.
- MONGODB - Désigne une connexion à une base de données de documents MongoDB.

`MONGODB`Les connexions utilisent les méthodes suivantes ConnectionParameters.

- Obligatoire : `CONNECTION_URL`.
- Obligatoire : toutes les valeurs (`USERNAME`, `PASSWORD`) ou `SECRET_ID`.

- `VIEW_VALIDATION_REDSHIFT` : désigne une connexion utilisée pour la validation des vues par Amazon Redshift.
- `VIEW_VALIDATION_ATHENA` : désigne une connexion utilisée pour la validation des vues par Amazon Athena.
- `NETWORK` - Désigne une connexion réseau à une source de données dans un environnement Amazon Virtual Private Cloud (Amazon VPC).

  `NETWORK`Les connexions ne sont pas nécessaires ConnectionParameters. Fournissez plutôt un PhysicalConnectionRequirements.
- `MARKETPLACE`- Utilise les paramètres de configuration contenus dans un connecteur acheté AWS Marketplace pour lire et écrire dans des magasins de données qui ne sont pas pris en charge de manière native par AWS Glue.

  `MARKETPLACE`Les connexions utilisent les méthodes suivantes ConnectionParameters.

  - Obligatoire : `CONNECTOR_TYPE`, `CONNECTOR_URL`, `CONNECTOR_CLASS_NAME`, `CONNECTION_URL`.
  - Obligatoire pour les connexions `JDBC` `CONNECTOR_TYPE` : toutes les valeurs (`USERNAME`, `PASSWORD`) ou `SECRET_ID`.
- `CUSTOM` - Utilise les paramètres de configuration contenus dans un connecteur personnalisé pour lire et écrire dans des magasins de données qui ne sont pas pris en charge en mode natif par AWS Glue.

Pour plus d'informations sur les paramètres de connexion nécessaires pour un connecteur particulier, consultez la documentation du connecteur dans la section Ajout d'une AWS Glue connexion dans le guide de AWS Glue l'utilisateur.

`SFTP` n'est pas pris en charge.

Pour plus d'informations sur la manière dont les fonctionnalités facultatives ConnectionProperties sont utilisées pour configurer les fonctionnalités dans AWS Glue, consultez les propriétés de AWS Glue connexion.

Pour plus d'informations sur la manière dont ConnectionProperties les options sont utilisées pour configurer les fonctionnalités dans AWS Glue Studio, consultez Utilisation de connecteurs et de connexions.

- `MatchCriteria` : tableau de chaînes UTF-8, avec 10 chaînes maximum.

  Liste des critères qui peuvent être utilisés dans la sélection de cette connexion.

- `ConnectionProperties` – Obligatoire : Tableau de mappage de paires clé-valeur, avec 100 paires au maximum.

  Chaque clé est une chaîne UTF-8 (valeurs valides : `HOST` | `PORT` | `USERNAME="USER_NAME"` | `PASSWORD` | `ENCRYPTED_PASSWORD` | `JDBC_DRIVER_JAR_URI` | `JDBC_DRIVER_CLASS_NAME` | `JDBC_ENGINE` | `JDBC_ENGINE_VERSION` | `CONFIG_FILES` | `INSTANCE_ID` | `JDBC_CONNECTION_URL` | `JDBC_ENFORCE_SSL` | `CUSTOM_JDBC_CERT` | `SKIP_CUSTOM_JDBC_CERT_VALIDATION` | `CUSTOM_JDBC_CERT_STRING` | `CONNECTION_URL` | `KAFKA_BOOTSTRAP_SERVERS` | `KAFKA_SSL_ENABLED` | `KAFKA_CUSTOM_CERT` | `KAFKA_SKIP_CUSTOM_CERT_VALIDATION` | `KAFKA_CLIENT_KEYSTORE` | `KAFKA_CLIENT_KEYSTORE_PASSWORD` | `KAFKA_CLIENT_KEY_PASSWORD` | `ENCRYPTED_KAFKA_CLIENT_KEYSTORE_PASSWORD` | `ENCRYPTED_KAFKA_CLIENT_KEY_PASSWORD` | `KAFKA_SASL_MECHANISM` | `KAFKA_SASL_PLAIN_USERNAME` | `KAFKA_SASL_PLAIN_PASSWORD` | `ENCRYPTED_KAFKA_SASL_PLAIN_PASSWORD` | `KAFKA_SASL_SCRAM_USERNAME` | `KAFKA_SASL_SCRAM_PASSWORD` | `KAFKA_SASL_SCRAM_SECRETS_ARN` | `ENCRYPTED_KAFKA_SASL_SCRAM_PASSWORD` | `KAFKA_SASL_GSSAPI_KEYTAB` | `KAFKA_SASL_GSSAPI_KRB5_CONF` | `KAFKA_SASL_GSSAPI_SERVICE` | `KAFKA_SASL_GSSAPI_PRINCIPAL` | `SECRET_ID` | `CONNECTOR_URL` | `CONNECTOR_TYPE` | `CONNECTOR_CLASS_NAME` | `ENDPOINT` | `ENDPOINT_TYPE` | `ROLE_ARN` | `REGION` | `WORKGROUP_NAME` | `CLUSTER_IDENTIFIER` | `DATABASE`).

  Chaque valeur est une chaîne Valeur, d'une longueur comprise entre 1 et 1 024 octets.

  Ces paires clé-valeur définissent les paramètres pour la connexion.

- `SparkProperties` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est une chaîne UTF-8, d'une longueur comprise entre 1 et 2 048 octets.

Propriétés de connexion propres à l'environnement de calcul Spark.

- `AthenaProperties` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est une chaîne UTF-8, d'une longueur comprise entre 1 et 2 048 octets.

  Propriétés de connexion propres à l'environnement de calcul Athena.
- `PythonProperties` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est une chaîne UTF-8, d'une longueur comprise entre 1 et 2 048 octets.

  Propriétés de connexion propres à l'environnement de calcul Python.
- `PhysicalConnectionRequirements` – Un objet PhysicalConnectionRequirements.

  Les exigences de connexion physique, comme un cloud privé virtuel (VPC) et `SecurityGroup`, qui sont nécessaires pour bien établir cette connexion.
- `AuthenticationConfiguration` : un objet AuthenticationConfigurationInput.

  Les propriétés d'authentification de la connexion.
- `ValidateCredentials` : booléen.

  Un indicateur pour valider les informations d'identification lors de la création de la connexion. La valeur par défaut est true.
- `ValidateForComputeEnvironments` – Tableau de chaînes UTF-8.

  Environnements de calcul par rapport auxquels les propriétés de connexion spécifiées sont validées.

TestConnectionInput structure

Structure utilisée pour spécifier le test d'une connexion à un service.

Champs

- `ConnectionType`— Obligatoire : chaîne UTF-8 (valeurs valides : `JDBC` `SFTP` `MONGODB` | `KAFKA` | `NETWORK` | `MARKETPLACE` | `CUSTOM` | `SALESFORCE` | `VIEW_VALIDATION_REDSHIFT`

| VIEW_VALIDATION_ATHENA | | GOOGLEADS | GOOGLESHEETS | GOOGLEANALYTICS4 | SERVICENOW | MARKETO | SAPODATA | ZENDESK | JIRACLOUD | NETSUITEERP | HUBSPOT | FACEBOOKADS | INSTAGRAMADS | ZOHOCRM | SALESFORCEPARDOT | SALESFORCEMARKETINGCLOUD | ADOBEANALYTICS | SLACK | LINKEDIN | MIXPANEL | ASANA | STRIPE | SMARTSHEET DATADOG WOOCOMMERCE| INTERCOM | SNAPCHATADS | PAYPAL | QUICKBOOKS | FACEBOOKPAGEINSIGHTS | FRESHDESK | TWILIO | DOCUSIGNMONITOR FRESHSALES | ZOOM | GOOGLESEARCHCONSOLE | SALESFORCECOMMERCECLOUD | SAPCONCUR | DYNATRACE | MICROSOFTDYNAMIC365FINANCEANDOPS | MICROSOFTTEAMS | BLACKBAUDRAISEREDGENXT | MAILCHIMP | GITLAB | PENDO | PRODUCTBOARD | CIRCLECI | PIPEDIVE | SENDGRID | AZURECOSMOS | AZURESQL | BIGQUERY BLACKBAUD | CLOUDERAHIVE | CLOUDERAIMPALA | CLOUDWATCH | CLOUDWATCHMETRICS | CMDB | DATALAKEGEN2 | DB2 | DB2AS400 | DOCUMENTDB | DOMO| DYNAMODB | GOOGLECLOUDSTORAGE HBASE | KUSTOMER | MICROSOFTDYNAMICS365CRM | MONDAY | MYSQL | OKTA | OPENSEARCH | ORACLE PIPEDRIVE | POSTGRESQL | SAPHANA | SQLSERVER | SYNAPSE | TERADATA | TERADATANOS TIMESTREAM | TPCDS |VERTICA).

Le type de connexion à tester. Cette opération n'est disponible que pour les types de connexion JDBC ou SALESFORCE.

- ConnectionProperties – Obligatoire : Tableau de mappage de paires clé-valeur, avec 100 paires au maximum.

  Chaque clé est une chaîne UTF-8 (valeurs valides : HOST | PORT | USERNAME="USER_NAME" | PASSWORD | ENCRYPTED_PASSWORD | JDBC_DRIVER_JAR_URI | JDBC_DRIVER_CLASS_NAME | JDBC_ENGINE | JDBC_ENGINE_VERSION | CONFIG_FILES | INSTANCE_ID | JDBC_CONNECTION_URL | JDBC_ENFORCE_SSL | CUSTOM_JDBC_CERT | SKIP_CUSTOM_JDBC_CERT_VALIDATION | CUSTOM_JDBC_CERT_STRING | CONNECTION_URL | KAFKA_BOOTSTRAP_SERVERS | KAFKA_SSL_ENABLED | KAFKA_CUSTOM_CERT | KAFKA_SKIP_CUSTOM_CERT_VALIDATION | KAFKA_CLIENT_KEYSTORE | KAFKA_CLIENT_KEYSTORE_PASSWORD | KAFKA_CLIENT_KEY_PASSWORD | ENCRYPTED_KAFKA_CLIENT_KEYSTORE_PASSWORD | ENCRYPTED_KAFKA_CLIENT_KEY_PASSWORD | KAFKA_SASL_MECHANISM | KAFKA_SASL_PLAIN_USERNAME | KAFKA_SASL_PLAIN_PASSWORD | ENCRYPTED_KAFKA_SASL_PLAIN_PASSWORD | KAFKA_SASL_SCRAM_USERNAME | KAFKA_SASL_SCRAM_PASSWORD | KAFKA_SASL_SCRAM_SECRETS_ARN | ENCRYPTED_KAFKA_SASL_SCRAM_PASSWORD | KAFKA_SASL_GSSAPI_KEYTAB | KAFKA_SASL_GSSAPI_KRB5_CONF | KAFKA_SASL_GSSAPI_SERVICE | KAFKA_SASL_GSSAPI_PRINCIPAL | SECRET_ID | CONNECTOR_URL | CONNECTOR_TYPE

| CONNECTOR_CLASS_NAME | ENDPOINT | ENDPOINT_TYPE | ROLE_ARN | REGION | WORKGROUP_NAME | CLUSTER_IDENTIFIER | DATABASE).

Chaque valeur est une chaîne Valeur, d'une longueur comprise entre 1 et 1 024 octets.

Les paires clé-valeur qui définissent les paramètres pour la connexion.

Les connexions JDBC utilisent les propriétés de connexion suivantes :

- Obligatoire : toutes les valeurs (HOST, PORT, JDBC_ENGINE) ou JDBC_CONNECTION_URL.
- Obligatoire : toutes les valeurs (USERNAME, PASSWORD) ou SECRET_ID.
- Facultatif : JDBC_ENFORCE_SSL, CUSTOM_JDBC_CERT, CUSTOM_JDBC_CERT_STRING, SKIP_CUSTOM_JDBC_CERT_VALIDATION. Ces paramètres sont utilisés pour configurer le SSL avec JDBC.

Les connexions SALESFORCE nécessitent que le membre AuthenticationConfiguration soit configuré.

- AuthenticationConfiguration : un objet AuthenticationConfigurationInput.

  Structure contenant la configuration d'authentification dans la TestConnection demande. Nécessaire pour une connexion à Salesforce à l'aide de OAuth l'authentification.

PhysicalConnectionRequirements structure

L'application OAuth cliente en GetConnection réponse.

Champs

- SubnetId – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID de sous-réseau utilisé par la connexion.
- SecurityGroupIdList – Tableau de chaînes UTF-8, avec 50 chaînes maximum.

  Liste des ID de groupe de sécurité utilisée par la connexion.
- AvailabilityZone – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Connexion de la zone de disponibilité.

GetConnectionsFilter structure

Filtre les définitions de connexion retournées par l'opération d'API `GetConnections`.

Champs

- `MatchCriteria` : tableau de chaînes UTF-8, avec 10 chaînes maximum.

  Chaîne des critères qui doivent répondre aux critères enregistrés dans la définition de connexion pour que cette définition soit renvoyée.

- `ConnectionType`— Chaîne UTF-8 (valeurs valides : `JDBC` | `SFTP` | `MONGODB` | `KAFKA` | `NETWORK` | `MARKETPLACE` | `CUSTOM` | `SALESFORCE` | `VIEW_VALIDATION_REDSHIFT` | `VIEW_VALIDATION_ATHENA` | `GOOGLEADS` | `GOOGLESHEETS` | | `GOOGLEANALYTICS4` | `SERVICENOW` | `MARKETO` | `SAPODATA` | `ZENDESK` | `JIRACLOUD` | `NETSUITEERP` | `HUBSPOT` | `FACEBOOKADS` | `INSTAGRAMADS` | `ZOHOCRM` | `SALESFORCEPARDOT` | `SALESFORCEMARKETINGCLOUD` | `ADOBEANALYTICS` | `SLACK` | `LINKEDIN` | `MIXPANEL` | `ASANA` | `STRIPE` | `SMARTSHEET` | `DATADOG` | `WOOCOMMERCE INTERCOM SNAPCHATADS`| `PAYPAL` | `QUICKBOOKS` | `FACEBOOKPAGEINSIGHTS` | `FRESHDESK` | `TWILIO` | `DOCUSIGNMONITOR` | `FRESHSALES` | `ZOOM GOOGLESEARCHCONSOLE` | `SALESFORCECOMMERCECLOUD` | `SAPCONCUR` | `DYNATRACE` | `MICROSOFTDYNAMIC365FINANCEANDOPS` | `MICROSOFTTEAMS` | `BLACKBAUDRAISEREDGENXT` | `MAILCHIMP` | `GITLAB` | `PENDO` | `PRODUCTBOARD` | `CIRCLECI` | `PIPEDIVE` | `SENDGRID` | `AZURECOSMOS` | `AZURESQL` | `BIGQUERY` | `BLACKBAUD` | `CLOUDERAHIVE` `CLOUDERAIMPALA` | `CLOUDWATCH` | `CLOUDWATCHMETRICS` | `CMDB` | `DATALAKEGEN2` | `DB2` | `DB2AS400` | `DOCUMENTDB` | `DOMO` | `DYNAMODB` | `GOOGLECLOUDSTORAGE`| `HBASE KUSTOMER` | `MICROSOFTDYNAMICS365CRM` | `MONDAY` | `MYSQL` | `OKTA OPENSEARCH` | `ORACLE` | `PIPEDRIVE` | `POSTGRESQL` | `SAPHANA SQLSERVER` | `SYNAPSE` | `TERADATA` | `TERADATANOS` | `TIMESTREAM` `TPCDS` |`VERTICA`).

  Type de connexions à renvoyer. Actuellement, SFTP n'est pas pris en charge.

- `ConnectionSchemaVersion`— Nombre (entier), pas moins de 1 ou plus de 2.

  Indique si la connexion a été créée avec la version 1 ou 2 du schéma.

AuthenticationConfiguration structure

Une structure contenant la configuration d'authentification.

Champs

- `AuthenticationType` – Chaîne UTF-8 (valeurs valides : `BASIC` | `OAUTH2` | `CUSTOM` | `IAM`).

  Une structure contenant la configuration d'authentification.

- `SecretArn` – Chaîne UTF-8, correspondant au Custom string pattern #36.

  ARN de Secrets Manager pour stocker les informations d'identification.

- `KmsKeyArn` – Chaîne UTF-8, correspondant au Custom string pattern #42.

  Amazon Resource Name (ARN) de la clé KMS à utiliser pour chiffrer les informations d'authentification sensibles. Cette clé est utilisée pour protéger les informations d'identification et autres données sensibles stockées dans la configuration d'authentification.

- `OAuth2Properties` : un objet OAuth2Propriétés.

  Les propriétés de l' OAuth2 authentification.

AuthenticationConfigurationInput structure

Structure contenant la configuration d'authentification dans la CreateConnection demande.

Champs

- `AuthenticationType` – Chaîne UTF-8 (valeurs valides : `BASIC` | `OAUTH2` | `CUSTOM` | `IAM`).

  Structure contenant la configuration d'authentification dans la CreateConnection demande.

- `OAuth2Properties` : un objet OAuth2PropertiesInput.

  Les propriétés d' OAuth2 authentification dans la CreateConnection demande.

- `SecretArn` – Chaîne UTF-8, correspondant au Custom string pattern #36.

  L'ARN du gestionnaire secret pour stocker les informations d'identification dans la CreateConnection demande.

- `KmsKeyArn` – Chaîne UTF-8, correspondant au Custom string pattern #42.

  ARN de la clé KMS utilisée pour chiffrer la connexion. Utilisé uniquement comme entrée dans la demande et stocké dans Secrets Manager.

- `BasicAuthenticationCredentials` – Un objet BasicAuthenticationCredentials.

Informations d'identification utilisées lorsque le type d'authentification est une authentification de base.

- `CustomAuthenticationCredentials` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est une chaîne UTF-8, d'une longueur comprise entre 1 et 2 048 octets.

  Informations d'identification utilisées lorsque le type d'authentification est une authentification personnalisée.

OAuth2Structure des propriétés

Structure contenant les propriétés d' OAuth2 authentification.

Champs

- `OAuth2GrantType` – Chaîne UTF-8 (valeurs valides : `AUTHORIZATION_CODE` | `CLIENT_CREDENTIALS` | `JWT_BEARER`).

  Le type de OAuth2 subvention. Par exemple, `AUTHORIZATION_CODE`, `JWT_BEARER` ou `CLIENT_CREDENTIALS`.

- `OAuth2ClientApplication` : un objet OAuth2ClientApplication.

  Type d'application client. Par exemple, AWS_MANAGED ou USER_MANAGED.

- `TokenUrl` – chaîne UTF-8, d'une longueur ne dépassant pas 256 octets, correspondant au Custom string pattern #40.

  URL du serveur d'authentification du fournisseur, pour échanger un code d'autorisation en vue d'obtenir un jeton d'accès.

- `TokenUrlParametersMap` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est une chaîne UTF-8, d'une longueur comprise entre 1 et 512 octets.

  Carte des paramètres ajoutés à la demande de jeton `GET`.

OAuth2PropertiesInput structure

Structure contenant les propriétés OAuth2 de la CreateConnection demande.

Champs

- `OAuth2GrantType` – Chaîne UTF-8 (valeurs valides : `AUTHORIZATION_CODE` | `CLIENT_CREDENTIALS` | `JWT_BEARER`).

  Type de OAuth2 subvention indiqué dans la CreateConnection demande. Par exemple, `AUTHORIZATION_CODE`, `JWT_BEARER` ou `CLIENT_CREDENTIALS`.

- `OAuth2ClientApplication` : un objet OAuth2ClientApplication.

  Type d'application client dans la CreateConnection demande. Par exemple, `AWS_MANAGED` ou `USER_MANAGED`.

- `TokenUrl` – chaîne UTF-8, d'une longueur ne dépassant pas 256 octets, correspondant au Custom string pattern #40.

  URL du serveur d'authentification du fournisseur, pour échanger un code d'autorisation en vue d'obtenir un jeton d'accès.

- `TokenUrlParametersMap` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est une chaîne UTF-8, d'une longueur comprise entre 1 et 512 octets.

  Carte des paramètres ajoutés à la demande de jeton `GET`.

- `AuthorizationCodeProperties` : un objet AuthorizationCodeProperties.

  Ensemble de propriétés requis pour le type de OAuth2 `AUTHORIZATION_CODE` subvention.

- `OAuth2Credentials` : un objet OAuth2Informations d'identification.

  Les informations d'identification utilisées lorsque le type d'authentification est OAuth2 l'authentification.

OAuth2ClientApplication structure

L'application OAuth2 cliente utilisée pour la connexion.

Champs

- `UserManagedClientApplicationClientId` – Chaîne UTF-8, d'une longueur maximale de 2 048 octets, correspondant au Custom string pattern #37.

  ClientiD de l'application clientId si c'est le cas. ClientAppType `USER_MANAGED`
- `AWSManagedClientApplicationReference` – Chaîne UTF-8, d'une longueur maximale de 2 048 octets, correspondant au Custom string pattern #37.

  La référence à l'application cliente côté SaaS qui est AWS gérée.

AuthorizationCodeProperties structure

Ensemble de propriétés requis pour le flux de travail du type de OAuth2 `AUTHORIZATION_CODE` subvention.

Champs

- `AuthorizationCode`— Chaîne UTF-8, d'une longueur d'au moins 1 ou supérieure à 4 096 octets, correspondant au. Custom string pattern #37

  Code d'autorisation à utiliser dans la troisième étape du flux de travail d'autorisation `AUTHORIZATION_CODE`. Il s'agit d'un code à usage unique qui devient invalide une fois échangé contre un jeton d'accès. Il est donc acceptable d'avoir cette valeur comme paramètre de demande.
- `RedirectUri`— Chaîne UTF-8, d'une longueur maximale de 512 octets, correspondant au. Custom string pattern #41

  URI de redirection vers lequel l'utilisateur est redirigé par le serveur d'autorisation lors de l'émission d'un code d'autorisation. L'URI est ensuite utilisé lorsque le code d'autorisation est échangé pour obtenir un jeton d'accès.

BasicAuthenticationCredentials structure

Pour fournir des informations d'authentification de base lorsque vous ne fournissez pas de valeur `SecretArn`.

Champs

- `Username`— Chaîne UTF-8, d'une longueur maximale de 512 octets, correspondant au. Custom string pattern #37

Nom d'utilisateur pour se connecter à la source de données.

- `Password`— Chaîne UTF-8, d'une longueur maximale de 512 octets, correspondant au. Custom string pattern #33

  Mot de passe pour se connecter à la source de données.

OAuth2Structure des informations d'identification

Les informations d'identification utilisées lorsque le type d'authentification est OAuth2 l'authentification.

Champs

- `UserManagedClientApplicationClientSecret`— Chaîne UTF-8, d'une longueur maximale de 512 octets, correspondant au. Custom string pattern #38

  Secret client de l'application client si l'application client est gérée par l'utilisateur.
- `AccessToken`— Chaîne UTF-8, d'une longueur maximale de 4096 octets, correspondant au. Custom string pattern #38

  Le jeton d'accès utilisé lorsque le type d'authentification est OAuth2.
- `RefreshToken`— Chaîne UTF-8, d'une longueur maximale de 4096 octets, correspondant au. Custom string pattern #38

  Le jeton d'actualisation utilisé lorsque le type d'authentification est OAuth2.
- `JwtToken`— Chaîne UTF-8, d'une longueur maximale de 8 000 octets, correspondant au. Custom string pattern #39

  Le jeton Web JSON (JWT) utilisé lorsque le type d'authentification est OAuth2.

Opérations

- CreateConnection action (Python : créer_connexion)
- DeleteConnection action (Python : supprimer_connexion)
- GetConnection action (Python : get_connection)
- GetConnections action (Python : get_connections)

- UpdateConnection action (Python : update_connection)
- TestConnection action (Python : test_connection)
- BatchDeleteConnection action (Python : batch_delete_connection)

CreateConnection action (Python : créer_connexion)

Crée une définition de connexion dans le catalogue de données.

Les connexions utilisées pour créer des ressources fédérées nécessitent l'autorisation IAM `glue:PassConnection`.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données dans lequel créer la connexion. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.
- `ConnectionInput` – Obligatoire : un objet ConnectionInput.

  Objet `ConnectionInput` définissant la connexion à créer.
- `Tags` – Tableau de mappage de paires clé-valeur, avec 50 paires au maximum.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 256 octets.

  Les identifications que vous attribuez à la connexion.

Réponse

- `CreateConnectionStatus` – Chaîne UTF-8 (valeurs valides : `READY` | `IN_PROGRESS` | `FAILED`).

  Statut de la demande de création de connexion. La demande peut prendre un certain temps pour certains types d'authentification, par exemple lors de la création d'une OAuth connexion avec échange de jetons via VPC.

Erreurs

- `AlreadyExistsException`
- `InvalidInputException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`
- `GlueEncryptionException`

DeleteConnection action (Python : supprimer_connexion)

Supprime une connexion du catalogue de données.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données dans lequel réside la connexion. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.
- `ConnectionName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la connexion à supprimer.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `EntityNotFoundException`
- `OperationTimeoutException`

GetConnection action (Python : get_connection)

Extrait une définition de connexion du catalogue de données.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données dans lequel réside la connexion. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la définition de connexion à extraire.

- `HidePassword` : booléen.

  Vous permet de récupérer les métadonnées de connexion sans renvoyer le mot de passe. Par exemple, la AWS Glue console utilise cet indicateur pour récupérer la connexion et n'affiche pas le mot de passe. Définissez ce paramètre lorsque l'appelant n'est peut-être pas autorisé à utiliser la AWS KMS clé pour déchiffrer le mot de passe, mais qu'il est autorisé à accéder au reste des propriétés de connexion.

- `ApplyOverrideForComputeEnvironment` – Chaîne UTF-8 (valeurs valides : `SPARK` | `ATHENA` | `PYTHON`).

  Pour les connexions qui peuvent être utilisées dans plusieurs services, spécifie les propriétés de renvoi pour l'environnement de calcul spécifié.

Réponse

- `Connection` – Un objet Connexion.

  Définition de connexion demandée.

Erreurs

- `EntityNotFoundException`
- `OperationTimeoutException`
- `InvalidInputException`
- `GlueEncryptionException`

GetConnections action (Python : get_connections)

Extrait une liste de définitions de connexion du catalogue de données.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données dans lequel résident les connexions. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.

- `Filter` – Un objet GetConnectionsFilter.

  Filtre qui contrôle les connexions à renvoyer.

- `HidePassword` : booléen.

  Vous permet de récupérer les métadonnées de connexion sans renvoyer le mot de passe. Par exemple, la AWS Glue console utilise cet indicateur pour récupérer la connexion et n'affiche pas le mot de passe. Définissez ce paramètre lorsque l'appelant n'est peut-être pas autorisé à utiliser la AWS KMS clé pour déchiffrer le mot de passe, mais qu'il est autorisé à accéder au reste des propriétés de connexion.

- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, s'il s'agit d'un appel de continuation.

- `MaxResults` – Nombre (entier), compris entre 1 et 1 000.

  Nombre maximal de connexions à renvoyer dans une réponse.

Réponse

- `ConnectionList` : un tableau d'objets Connexion.

  Liste des définitions de connexion demandées.

- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, si la liste des connexions renvoyées n'inclut pas la dernière connexion filtrée.

Erreurs

- `EntityNotFoundException`
- `OperationTimeoutException`
- `InvalidInputException`
- `GlueEncryptionException`

UpdateConnection action (Python : update_connection)

Met à jour une définition de connexion dans le catalogue de données.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données dans lequel réside la connexion. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.
- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la définition de connexion à mettre à jour.
- `ConnectionInput` – Obligatoire : un objet ConnectionInput.

  Objet `ConnectionInput` qui redéfinit la connexion en question.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `InvalidInputException`
- `EntityNotFoundException`
- `OperationTimeoutException`
- `InvalidInputException`
- `GlueEncryptionException`

TestConnection action (Python : test_connection)

Teste une connexion à un service pour valider les informations d'identification du service que vous indiquez.

Vous pouvez indiquer un nom de connexion existant ou une `TestConnectionInput` pour tester une entrée de connexion inexistante. Fournir les deux en même temps entraînera une erreur.

Si l'action aboutit, le service renvoie une réponse HTTP 200.

Demande

- `ConnectionName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Facultatif. Nom de la connexion à tester. Si seul le nom est fourni, l'opération obtiendra la connexion et l'utilisera pour les tests.
- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue où réside la connexion.
- `TestConnectionInput` – Un objet TestConnectionInput.

  Structure utilisée pour spécifier le test d'une connexion à un service.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `InvalidInputException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`
- `GlueEncryptionException`
- `FederationSourceException`
- `AccessDeniedException`

- `EntityNotFoundException`
- `ConflictException`
- `InternalServiceException`

BatchDeleteConnection action (Python : batch_delete_connection)

Supprime une liste de définitions de connexion du catalogue de données.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données dans lequel résident les connexions. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.
- `ConnectionNameList` – Obligatoire : Tableau de chaînes UTF-8, avec 25 chaînes maximum.

  Liste des noms des connexions à supprimer.

Réponse

- `Succeeded` – Tableau de chaînes UTF-8.

  Liste des noms des définitions de connexion qui ont été supprimées avec succès.
- `Errors` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaque valeur est un objet ErrorDetail.

  Carte des noms des connexions qui n'ont pas été supprimées avec succès pour les détails d'erreur.

Erreurs

- `InternalServiceException`
- `OperationTimeoutException`

API de types de connexion

L'API Connection Type décrit AWS Glue APIs les types de connexion.

Gestion des connexions APIs

- DescribeConnectionType action (Python : describe_connection_type)
- ListConnectionTypes action (Python : list_connection_types)
- ConnectionTypeBrief structure
- ConnectionTypeVariant structure

DescribeConnectionType action (Python : describe_connection_type)

L'API `DescribeConnectionType` fournit des informations complètes sur les options prises en charge pour un type de connexion donné dans AWS Glue.

Demande

- `ConnectionType` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du type de connexion à décrire.

Réponse

- `ConnectionType` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du type de connexion.
- `Description` – Chaîne UTF-8, d'une longueur maximale de 1 024 octets.

  Description du type de connexion.
- `Capabilities` – Un objet Fonctionnalités.

  Les types d'authentification, les types d'interface de données (environnements de calcul) et les opérations de données pris en charge par le connecteur.
- `ConnectionProperties` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

Chaque valeur est un objet Propriété.

Propriétés de connexion communes à tous les environnements de calcul.

- `ConnectionOptions` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est un objet Propriété.

  Renvoie les propriétés qui peuvent être définies lors de la création d'une connexion dans les `ConnectionInput.ConnectionProperties`. `ConnectionOptions` définit les paramètres qui peuvent être définis dans un script Spark ETL dans la carte des options de connexion transmise à un DataFrame.

- `AuthenticationConfiguration` : un objet AuthConfiguration.

  Le type d'autorisation utilisé pour la connexion.

- `ComputeEnvironmentConfigurations` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est un objet ComputeEnvironmentConfiguration.

  Les environnements de calcul pris en charge par la connexion.

- `PhysicalConnectionRequirements` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est un objet Propriété.

  Exigences physiques pour une connexion, telles que les spécifications VPC, Sous-réseau et Groupe de sécurité.

- `AthenaConnectionProperties` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est un objet Propriété.

  Propriétés de connexion propres à l'environnement de calcul Athena.

- `PythonConnectionProperties` – Tableau de mappage de paires valeur-clé.

Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

Chaque valeur est un objet Propriété.

Propriétés de connexion propres à l'environnement de calcul Python.

- `SparkConnectionProperties` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est un objet Propriété.

  Propriétés de connexion propres à l'environnement de calcul Spark.

Erreurs

- `ValidationException`
- `InvalidInputException`
- `InternalServiceException`

ListConnectionTypes action (Python : list_connection_types)

L'API `ListConnectionTypes` fournit un mécanisme de découverte permettant de connaître les types de connexion disponibles dans AWS Glue. La réponse contient une liste de types de connexion avec des informations détaillées sur ce qui est pris en charge pour chaque type de connexion.
Les types de connexion répertoriés sont l'ensemble des options prises en charge pour la valeur `ConnectionType` dans l'API `CreateConnection`.

Demande

- `MaxResults` – Nombre (entier), compris entre 1 et 1 000.

  Nombre maximal de résultats à renvoyer.

- `NextToken` – Chaîne UTF-8, d'une longueur comprise entre 1 et 2048 octets, correspondant au Custom string pattern #11.

  Jeton de continuation, s'il s'agit d'un appel de continuation.

Réponse

- `ConnectionTypes` : un tableau d'objets ConnectionTypeBrief.

  Une liste d'objets `ConnectionTypeBrief` contenant de brèves informations sur les types de connexion pris en charge.

- `NextToken` – Chaîne UTF-8, d'une longueur comprise entre 1 et 2048 octets, correspondant au Custom string pattern #11.

  Jeton de continuation, si le segment de liste actuel n'est pas le dernier.

Erreurs

- `InternalServiceException`

ConnectionTypeBrief structure

Brèves informations sur un type de connexion pris en charge renvoyées par l'API `ListConnectionTypes`.

Champs

- `ConnectionType`— Chaîne UTF-8 (valeurs valides : `JDBC` | `SFTP` | `MONGODB` | `KAFKA` | `NETWORK` | `MARKETPLACE` | `CUSTOM` | `SALESFORCE` | `VIEW_VALIDATION_REDSHIFT` | `VIEW_VALIDATION_ATHENA` | `GOOGLEADS` | `GOOGLESHEETS` | | `GOOGLEANALYTICS4` | `SERVICENOW` | `MARKETO` | `SAPODATA` | `ZENDESK` | `JIRACLOUD` | `NETSUITEERP` | `HUBSPOT` | `FACEBOOKADS` | `INSTAGRAMADS` | `ZOHOCRM` | `SALESFORCEPARDOT` | `SALESFORCEMARKETINGCLOUD` | `ADOBEANALYTICS` | `SLACK` | `LINKEDIN` | `MIXPANEL` | `ASANA` | `STRIPE` | `SMARTSHEET` | `DATADOG` | `WOOCOMMERCE INTERCOM SNAPCHATADS`| `PAYPAL` | `QUICKBOOKS` | `FACEBOOKPAGEINSIGHTS` | `FRESHDESK` | `TWILIO` | `DOCUSIGNMONITOR` | `FRESHSALES` | `ZOOM GOOGLESEARCHCONSOLE` | `SALESFORCECOMMERCECLOUD` | `SAPCONCUR` | `DYNATRACE` | `MICROSOFTDYNAMIC365FINANCEANDOPS` | `MICROSOFTTEAMS` | `BLACKBAUDRAISEREDGENXT` | `MAILCHIMP` | `GITLAB` | `PENDO` | `PRODUCTBOARD` | `CIRCLECI` | `PIPEDIVE` | `SENDGRID` | `AZURECOSMOS` | `AZURESQL` | `BIGQUERY` | `BLACKBAUD` | `CLOUDERAHIVE` `CLOUDERAIMPALA` | `CLOUDWATCH` | `CLOUDWATCHMETRICS` | `CMDB` | `DATALAKEGEN2` | `DB2` | `DB2AS400` | `DOCUMENTDB` | `DOMO` | `DYNAMODB` | `GOOGLECLOUDSTORAGE`| `HBASE` | `KUSTOMER` `MICROSOFTDYNAMICS365CRM` | `MONDAY` | `MYSQL` | `OKTA` | `OPENSEARCH` | `ORACLE PIPEDRIVE`

| POSTGRESQL | SAPHANA | SQLSERVER | SYNAPSE | TERADATA | TERADATANOS TIMESTREAM | TPCDS |VERTICA).

Nom du type de connexion.

- `DisplayName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Nom lisible par l'utilisateur du type de connexion affiché dans la console AWS Glue .

- `Vendor` – Chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Nom du fournisseur qui a créé ou qui gère ce type de connexion.

- `Description` – Chaîne UTF-8, d'une longueur maximale de 1 024 octets.

  Description du type de connexion.

- `Categories` – .

  Liste des catégories auxquelles appartient ce type de connexion. Les catégories aident les utilisateurs à filtrer et à trouver les types de connexion appropriés en fonction de leurs cas d'utilisation.

- `Capabilities` – Un objet Fonctionnalités.

  Les types d'authentification, les types d'interface de données (environnements de calcul) et les opérations de données pris en charge par le connecteur.

- `LogoUrl` – Chaîne UTF-8.

  URL du logo associé à un type de connexion.

- `ConnectionTypeVariants` : un tableau d'objets ConnectionTypeVariant.

  Liste des variantes disponibles pour ce type de connexion. Différentes variantes peuvent fournir des configurations spécialisées pour des cas d'utilisation ou des mises en œuvre spécifiques du même type de connexion général.

ConnectionTypeVariant structure

Représente une variante d'un type de connexion dans le catalogue de AWS Glue données. Les variantes de type de connexion fournissent des configurations et des comportements spécifiques pour différentes implémentations du même type de connexion général.

Champs

- `ConnectionTypeVariantName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Identifiant unique pour la variante du type de connexion. Ce nom est utilisé en interne pour identifier la variante spécifique d'un type de connexion.
- `DisplayName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Nom lisible par l'homme de la variante de type de connexion affichée dans la AWS Glue console.
- `Description` – Chaîne UTF-8, d'une longueur maximale de 1 024 octets.

  Description détaillée de la variante du type de connexion, y compris son objectif, ses cas d'utilisation et les exigences de configuration spécifiques.
- `LogoUrl` – Chaîne UTF-8.

  URL du logo associé à une variante du type de connexion.

types de données

- Structure de validation
- AuthConfiguration structure
- Structure des fonctionnalités
- Structure de la propriété
- AllowedValue structure
- ComputeEnvironmentConfiguration structure

Structure de validation

Définit la manière dont une validation est effectuée sur une propriété de connexion.

Champs

- `ValidationType` – Obligatoire : Chaîne UTF-8 (valeurs valides : `REGEX` | `RANGE`).

  Type de validation à effectuer, tel que `REGEX`.
- `Patterns` – .

  Liste des modèles qui s'appliquent à la validation.

- `Description` – Obligatoire : chaîne UTF-8, d'une longueur comprise entre 1 et 1 024 octets.

  Description de la validation.
- `MaxLength` : nombre (entier).

  Longueur maximale d'une propriété de connexion de chaîne.
- `Maximum` : nombre (entier).

  Valeur maximale lors de la spécification d'un type `RANGE` de validation.
- `Minimum` : nombre (entier).

  Valeur minimale lors de la spécification d'un type `RANGE` de validation.

AuthConfiguration structure

Configuration d'authentification pour une connexion renvoyée par l'API `DescribeConnectionType`.

Champs

- `AuthenticationType` – Obligatoire : un objet Propriété.

  Type d'autorisation pour une connexion.
- `SecretArn` – Un objet Propriété.

  Amazon Resource Name (ARN) pour Secrets Manager.
- `OAuth2Properties` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est un objet Propriété.

  Carte des paires clé-valeur pour les OAuth2 propriétés. Chaque valeur est un objet `Property`.
- `BasicAuthenticationProperties` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est un objet Propriété.

  Carte des paires clé-valeur pour les OAuth2 propriétés. Chaque valeur est un objet `Property`.
- `CustomAuthenticationProperties` – Tableau de mappage de paires valeur-clé.

Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

Chaque valeur est un objet Propriété.

Mappage de paires clé-valeur pour les propriétés d'authentification personnalisées. Chaque valeur est un objet `Property`.

Structure des fonctionnalités

Spécifie les types d'authentification pris en charge renvoyés par l'API `DescribeConnectionType`.

Champs

- `SupportedAuthenticationTypes` – Obligatoire : Tableau de chaînes UTF-8.

  Liste des types d'authentification pris en charge.
- `SupportedDataOperations` – Obligatoire : Tableau de chaînes UTF-8.

  Liste des opérations de données prises en charge.
- `SupportedComputeEnvironments` – Obligatoire : Tableau de chaînes UTF-8.

  Liste des environnements de calcul pris en charge.

Structure de la propriété

Objet qui définit un type de connexion pour un environnement de calcul.

Champs

- `Name`— Obligatoire : chaîne UTF-8 d'une longueur d'au moins 1 ou de plus de 128 octets.

  Le nom de la propriété.
- `Description` – Obligatoire : Chaîne UTF-8, d'une longueur ne dépassant pas 1 024 octets.

  Description de la propriété.
- `Required` – obligatoire : booléen.

  Indique si la propriété est obligatoire.
- `PropertyTypes` – Obligatoire : Tableau de chaînes UTF-8.

Décrit le type de propriété.

- `AllowedValues` : un tableau d'objets AllowedValue.

  Liste d'objets `AllowedValue` représentant les valeurs autorisées pour la propriété.

- `DataOperationScopes` – Tableau de chaînes UTF-8.

  Indique les opérations de données applicables à la propriété.

AllowedValue structure

Objet représentant une valeur autorisée pour une propriété.

Champs

- `Description` – Chaîne UTF-8, d'une longueur maximale de 1 024 octets.

  Description de la valeur autorisée.

- `Value`— Obligatoire : chaîne UTF-8 d'une longueur d'au moins 1 ou de plus de 128 octets.

  Valeur autorisée pour la propriété.

ComputeEnvironmentConfiguration structure

Objet contenant la configuration d'un environnement de calcul (tel que Spark, Python ou Athena) renvoyé par l'API `DescribeConnectionType`.

Champs

- `Name`— Obligatoire : chaîne UTF-8 d'une longueur d'au moins 1 ou de plus de 128 octets.

  Nom de la configuration de l'environnement de calcul.

- `Description` – Obligatoire : Chaîne UTF-8, d'une longueur ne dépassant pas 1 024 octets.

  Description de l'environnement de calcul.

- `ComputeEnvironment` – Obligatoire : Chaîne UTF-8 (valeurs valides : `SPARK` | `ATHENA` | `PYTHON`).

  Type d'environnement de calcul.

- `SupportedAuthenticationTypes` – Obligatoire : Tableau de chaînes UTF-8.

  Types d'authentification pris en charge pour l'environnement de calcul.
- `ConnectionOptions` – obligatoire : tableau de mappage de paires clé-valeur.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est un objet Propriété.

  Paramètres utilisés comme options de connexion pour l'environnement de calcul.
- `ConnectionPropertyNameOverrides` – obligatoire : tableau de mappage de paires clé-valeur.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Le nom de propriété de connexion est remplacé pour l'environnement de calcul.
- `ConnectionOptionNameOverrides` – obligatoire : tableau de mappage de paires clé-valeur.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Le nom d'option de connexion est remplacé pour l'environnement de calcul.
- `ConnectionPropertiesRequiredOverrides`— Obligatoire :.

  Les propriétés de connexion requises en tant que remplacements pour l'environnement de calcul.
- `PhysicalConnectionPropertiesRequired` : booléen.

  Indique si les `PhysicalConnectionProperties` sont requises pour l'environnement de calcul.

Métadonnées de connexion et API de prévisualisation

La connexion suivante APIs décrit les opérations permettant de décrire les métadonnées de connexion.

Types de données

- Structure de l'entité
- Structure de champ

Structure de l'entité

Entité prise en charge par un `ConnectionType` donné.

Champs

- `EntityName` – Chaîne UTF-8.

  Le nom de l'entité.
- `Label` – Chaîne UTF-8.

  Étiquette utilisée pour l'entité.
- `IsParentEntity` : booléen.

  Valeur booléenne qui permet de déterminer si des sous-objets peuvent être répertoriés.
- `Description` – Chaîne UTF-8.

  Une description de l'entité.
- `Category` – Chaîne UTF-8.

  Type d'entités présentes dans la réponse. Cette valeur dépend de la connexion source. Par exemple, il s'agit de `SObjects` pour Salesforce et de `databases`, `schemas` ou `tables` pour des sources telles qu'Amazon Redshift.
- `CustomProperties` –

  Carte facultative de clés qui peuvent être renvoyées pour une entité par un connecteur.

Structure de champ

L'objet `Field` contient des informations sur les différentes propriétés associées à un champ dans le connecteur.

Champs

- `FieldName` – Chaîne UTF-8.

  Un identifiant unique du champ.
- `Label` – Chaîne UTF-8.

  Une étiquette lisible pour le champ.

- `Description` – Chaîne UTF-8.

  Une description du champ.

- `FieldType`— Chaîne UTF-8 (valeurs valides : `INT` | | `SMALLINT` | `BIGINT` | | `FLOAT` | `LONG` | `DATE` | `BOOLEAN` | `MAP` | `ARRAY` | `STRING` | `TIMESTAMP` | | `DECIMAL` | `BYTE` `SHORT` `DOUBLE` |`STRUCT`).

  Type de données dans le champ.

- `IsPrimaryKey` : booléen.

  Indique si ce champ peut être utilisé comme clé primaire pour l'entité donnée.

- `IsNullable` : booléen.

  Indique si ce champ peut autoriser les valeurs nulles ou non.

- `IsRetrievable` : booléen.

  Indique si ce champ peut être ajouté dans la clause Select de la requête SQL ou s'il peut être extrait ou non.

- `IsFilterable` : booléen.

  Indique si ce champ peut être utilisé dans une clause de filtre (clause `WHERE`) d'une instruction SQL lors de l'interrogation de données.

- `IsPartitionable` : booléen.

  Indique si un champ donné peut être utilisé pour partitionner la requête envoyée à SaaS.

- `IsCreateable` : booléen.

  Indique si ce champ peut être créé dans le cadre d'une écriture de destination.

- `IsUpdateable` : booléen.

  Indique si ce champ peut être mis à jour dans le cadre d'une écriture de destination.

- `IsUpsertable` : booléen.

  Indique si ce champ peut faire l'objet d'un upsert dans le cadre d'une écriture de destination.

- `IsDefaultOnCreate` : booléen.

  Indique si ce champ est automatiquement renseigné lors de la création de l'objet, par exemple s'il est créé au niveau de l'horodatage.

- `SupportedValues` – .

  Liste des valeurs prises en charge pour le champ.

- `SupportedFilterOperators` – Tableau de chaînes UTF-8.

  Indique les opérateurs de filtre de prise en charge pour ce champ.

- `CustomProperties` –

  Carte facultative des clés qui peuvent être renvoyées.

Opérations

- ListEntities action (Python : list_entities)
- DescribeEntity action (Python : describe_entity)
- GetEntityRecords action (Python : get_entity_records)

ListEntities action (Python : list_entities)

Renvoie les entités disponibles prises en charge par le type de connexion.

Demande

- `ConnectionName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la connexion dont les informations d'identification sont requises pour interroger n'importe quel type de connexion.

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue contenant la connexion. Cela peut être nul. Par défaut, l'ID de AWS compte est l'ID du catalogue.

- `ParentEntityName` – Chaîne UTF-8.

  Nom de l'entité parent pour laquelle vous souhaitez répertorier les enfants. Ce paramètre utilise un chemin entièrement qualifié de l'entité afin de répertorier les entités enfant.

- `NextToken` – Chaîne UTF-8, d'une longueur comprise entre 1 et 2048 octets, correspondant au Custom string pattern #11.

Jeton de continuation, inclus s'il s'agit d'un appel de continuation.

- `DataStoreApiVersion` – Chaîne UTF-8, d'une longueur comprise entre 1 et 256 octets, correspondant au Custom string pattern #23.

  Version d'API du connecteur SaaS.

Réponse

- `Entities` : un tableau d'objets Entité.

  Liste d'objets `Entity`.
- `NextToken` – Chaîne UTF-8, d'une longueur comprise entre 1 et 2048 octets, correspondant au Custom string pattern #11.

  Un jeton de continuation, présent si le segment actuel n'est pas le dernier.

Erreurs

- `EntityNotFoundException`
- `OperationTimeoutException`
- `InvalidInputException`
- `GlueEncryptionException`
- `ValidationException`
- `FederationSourceException`
- `AccessDeniedException`

DescribeEntity action (Python : describe_entity)

Fournit des détails concernant l'entité utilisée avec le type de connexion, avec une description du modèle de données pour chaque champ de l'entité sélectionnée.

La réponse inclut tous les champs qui constituent l'entité.

Demande

- `ConnectionName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

Nom de la connexion qui contient les informations d'identification du type de connexion.

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue contenant la connexion. Cela peut être nul. Par défaut, l'ID de AWS compte est l'ID du catalogue.

- `EntityName` – Obligatoire : chaîne UTF-8.

  Nom de l'entité que vous souhaitez décrire à partir du type de connexion.

- `NextToken` – Chaîne UTF-8, d'une longueur comprise entre 1 et 2048 octets, correspondant au Custom string pattern #11.

  Jeton de continuation, inclus s'il s'agit d'un appel de continuation.

- `DataStoreApiVersion` – Chaîne UTF-8, d'une longueur comprise entre 1 et 256 octets, correspondant au Custom string pattern #23.

  Version de l'API utilisée pour l'entrepôt de données.

Réponse

- `Fields` : un tableau d'objets Champ.

  Décrit les champs de cette entité de connecteur. Il s'agit de la liste des objets `Field`. `Field` est très similaire à une colonne dans une base de données. L'objet `Field` contient des informations sur les différentes propriétés associées aux champs dans le connecteur.

- `NextToken` – Chaîne UTF-8, d'une longueur comprise entre 1 et 2048 octets, correspondant au Custom string pattern #11.

  Un jeton de continuation, présent si le segment actuel n'est pas le dernier.

Erreurs

- `EntityNotFoundException`
- `OperationTimeoutException`
- `InvalidInputException`
- `GlueEncryptionException`

- `ValidationException`
- `FederationSourceException`
- `AccessDeniedException`

GetEntityRecords action (Python : get_entity_records)

Cette API est utilisée pour interroger les données d'aperçu à partir d'un type de connexion donné ou d'un catalogue de AWS Glue données natif basé sur Amazon S3.

Renvoie les enregistrements sous forme de tableau de blobs JSON. Chaque enregistrement est formaté à l'aide de Jackson en JsonNode fonction du type de champ défini par l'`DescribeEntity`API.

Les connecteurs Spark génèrent des schémas selon le même mappage de type de données que dans l'API `DescribeEntity`. Les connecteurs Spark convertissent les données en types de données appropriés correspondant au schéma lors du renvoi de lignes.

Demande

- `ConnectionName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la connexion qui contient les informations d'identification du type de connexion.
- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue contenant la connexion. Cela peut être nul. Par défaut, l'ID de AWS compte est l'ID du catalogue.
- `EntityName` – Obligatoire : chaîne UTF-8.

  Nom de l'entité pour laquelle nous souhaitons interroger les données de prévisualisation à partir du type de connexion donné.
- `NextToken` – Chaîne UTF-8, d'une longueur comprise entre 1 et 2048 octets, correspondant au Custom string pattern #11.

  Jeton de continuation, inclus s'il s'agit d'un appel de continuation.
- `DataStoreApiVersion` – Chaîne UTF-8, d'une longueur comprise entre 1 et 256 octets, correspondant au Custom string pattern #23.

Version d'API du connecteur SaaS.

- `ConnectionOptions` – Tableau de mappage de paires clé-valeur, 100 paires au maximum.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 256 octets, correspondant au Custom string pattern #18.

  Chaque valeur est une chaîne UTF-8, d'une longueur comprise entre 1 et 256 octets, correspondant au Custom string pattern #17.

  Options de connecteur requises pour interroger les données.
- `FilterPredicate` – Chaîne UTF-8, d'une longueur comprise entre 1 et 100000 octets.

  Un prédicat de filtre que vous pouvez appliquer dans la demande de requête.
- `Limit` – Obligatoire : nombre (long), compris entre 1 et 1 000.

  Limite le nombre d'enregistrements récupérés avec la demande.
- `SelectedFields` – Tableau de chaînes UTF-8, avec 1 chaîne minimum et 1 000 chaînes maximum.

  Liste des champs que nous souhaitons récupérer dans le cadre des données de prévisualisation.

Réponse

- `Records` : tableau d'une structure.

  Liste des objets demandés.
- `NextToken` – Chaîne UTF-8, d'une longueur comprise entre 1 et 2048 octets, correspondant au Custom string pattern #11.

  Un jeton de continuation, présent si le segment actuel n'est pas le dernier.

Erreurs

- `EntityNotFoundException`
- `OperationTimeoutException`
- `InvalidInputException`
- `GlueEncryptionException`

- `ValidationException`
- `FederationSourceException`
- `AccessDeniedException`

API de fonction définie par l'utilisateur

L'API des fonctions définies par l'utilisateur décrit les types de AWS Glue données et les opérations utilisés lors de l'utilisation des fonctions.

Types de données

- UserDefinedFunction structure
- UserDefinedFunctionInput structure

UserDefinedFunction structure

Représente l'équivalent d'une définition de fonction Hive définie par l'utilisateur (`UDF`).

Champs

- `FunctionName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la fonction.
- `DatabaseName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la base de données du catalogue qui contient la fonction.
- `ClassName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Classe Java qui contient le code de fonction.
- `OwnerName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Propriétaire de la fonction.
- `OwnerType` – Chaîne UTF-8 (valeurs valides : `USER` | `ROLE` | `GROUP`).

Type de propriétaire.

- `CreateTime` – Horodatage.

  Heure à laquelle la fonction a été créée.

- `ResourceUris` – Un tableau d'objets ResourceUri, 1000 structures maximum.

  La ressource URIs associée à la fonction.

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données dans lequel réside la fonction.

UserDefinedFunctionInput structure

Structure utilisée pour créer ou mettre à jour une fonction définie par l'utilisateur.

Champs

- `FunctionName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la fonction.

- `ClassName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Classe Java qui contient le code de fonction.

- `OwnerName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Propriétaire de la fonction.

- `OwnerType` – Chaîne UTF-8 (valeurs valides : `USER` | `ROLE` | `GROUP`).

  Type de propriétaire.

- `ResourceUris` – Un tableau d'objets ResourceUri, 1000 structures maximum.

  La ressource URIs associée à la fonction.

Opérations

- CreateUserDefinedFunction action (Python : create_user_defined_function)
- UpdateUserDefinedFunction action (Python : update_user_defined_function)
- DeleteUserDefinedFunction action (Python : delete_user_defined_function)
- GetUserDefinedFunction action (Python : get_user_defined_function)
- GetUserDefinedFunctions action (Python : get_user_defined_functions)

CreateUserDefinedFunction action (Python : create_user_defined_function)

Crée une nouvelle définition de fonction dans le catalogue de données.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données dans lequel créer la fonction. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.
- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la base de données du catalogue dans laquelle créer la fonction.
- `FunctionInput` – Obligatoire : un objet UserDefinedFunctionInput.

  Objet `FunctionInput` qui définit la fonction à créer dans le catalogue de données.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `AlreadyExistsException`
- `InvalidInputException`
- `InternalServiceException`

- `EntityNotFoundException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`
- `GlueEncryptionException`

UpdateUserDefinedFunction action (Python : update_user_defined_function)

Met à jour une définition de fonction existante dans le catalogue de données.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données contenant la fonction à mettre à jour. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.
- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la base de données du catalogue contenant la fonction à mettre à jour.
- `FunctionName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la fonction.
- `FunctionInput` – Obligatoire : un objet UserDefinedFunctionInput.

  Objet `FunctionInput` qui redéfinit la fonction à créer dans le catalogue de données.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`

- `OperationTimeoutException`
- `GlueEncryptionException`

DeleteUserDefinedFunction action (Python : delete_user_defined_function)

Supprime une définition de fonction existante du catalogue de données.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données contenant la fonction à supprimer. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.
- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la base de données du catalogue contenant la fonction.
- `FunctionName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la définition de fonction à supprimer.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

GetUserDefinedFunction action (Python : get_user_defined_function)

Extrait une définition de fonction spécifiée du catalogue de données.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données contenant la fonction à extraire. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.
- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la base de données du catalogue contenant la fonction.
- `FunctionName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la fonction.

Réponse

- `UserDefinedFunction` – Un objet UserDefinedFunction.

  Définition de la fonction demandée.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`

GetUserDefinedFunctions action (Python : get_user_defined_functions)

Extrait plusieurs définitions de fonction du catalogue de données.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

ID du catalogue de données contenant les fonction à extraire. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.

- `DatabaseName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la base de données du catalogue contenant les fonctions. Si aucune n'est fournie, des fonctions de toutes les bases de données du catalogue seront renvoyées.

- `Pattern` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaîne de modèle nom-fonction facultative qui filtre les définitions de fonction renvoyées.

- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, s'il s'agit d'un appel de continuation.

- `MaxResults` – Nombre (entier), compris entre 1 et 100.

  Nombre maximum de fonctions renvoyées par réponse.

Réponse

- `UserDefinedFunctions` : un tableau d'objets UserDefinedFunction.

  Liste des définitions de fonction demandées.

- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, si la liste de fonctions renvoyée n'inclut pas la dernière fonction demandée.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`
- `GlueEncryptionException`

Importation d'un Athena catalogue dans AWS Glue

L'API de migration décrit AWS Glue les types de données et les opérations liées à la migration d'un catalogue de Athena données vers AWS Glue.

Types de données

- CatalogImportStatus structure

CatalogImportStatus structure

Structure contenant les informations sur le statut de la migration.

Champs

- `ImportCompleted` : booléen.

  `True` si la migration est terminée ou `False` dans le cas contraire.
- `ImportTime` – Horodatage.

  Heure à laquelle la migration a été démarrée.
- `ImportedBy` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la personne ayant lancé la migration.

Opérations

- ImportCatalogToGlue action (Python : import_catalog_to_glue)
- GetCatalogImportStatus action (Python : get_catalog_import_status)

ImportCatalogToGlue action (Python : import_catalog_to_glue)

Importe un catalogue de données Amazon Athena existant dans. AWS Glue

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

ID du catalogue à importer. Actuellement, il devrait s'agir de l'identifiant du AWS compte.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `InternalServiceException`
- `OperationTimeoutException`

GetCatalogImportStatus action (Python : get_catalog_import_status)

Extrait le statut d'une opération de migration.

Demande

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue à migrer. Actuellement, il devrait s'agir de l'identifiant du AWS compte.

Réponse

- `ImportStatus` – Un objet CatalogImportStatus.

  Statut de la migration du catalogue spécifié.

Erreurs

- `InternalServiceException`
- `OperationTimeoutException`

API d'optimiseur de table

L'API d'optimisation de table décrit l' AWS Glue API permettant d'activer le compactage afin d'améliorer les performances de lecture.

Types de données

- TableOptimizer structure
- TableOptimizerConfiguration structure
- TableOptimizerVpcConfiguration structure
- CompactionConfiguration structure
- IcebergCompactionConfiguration structure
- TableOptimizerRun structure
- BatchGetTableOptimizerEntry structure
- BatchTableOptimizer structure
- BatchGetTableOptimizerError structure
- RetentionConfiguration structure
- IcebergRetentionConfiguration structure
- OrphanFileDeletionConfiguration structure
- IcebergOrphanFileDeletionConfiguration structure
- CompactionMetrics structure
- RetentionMetrics structure
- OrphanFileDeletionMetrics structure
- IcebergCompactionMetrics structure
- IcebergRetentionMetrics structure
- IcebergOrphanFileDeletionMetrics structure
- RunMetrics structure

TableOptimizer structure

Contient des détails sur un optimiseur associé à une table.

Champs

- `type` – Chaîne UTF-8 (valeurs valides : `compaction="COMPACTION"` | `retention="RETENTION"` | `orphan_file_deletion="ORPHAN_FILE_DELETION"`).

  Le type d'optimiseur de table. Les valeurs valides sont :

  - `compaction` : pour gérer le compactage à l'aide d'un optimiseur de table.
  - `retention` : pour gérer la conservation des instantanés à l'aide d'un optimiseur de table.
  - `orphan_file_deletion` : pour gérer la suppression de fichiers orphelins à l'aide d'un optimiseur de table.
- `configuration` – Un objet TableOptimizerConfiguration.

  Un objet `TableOptimizerConfiguration` spécifié lors de la création ou de la mise à jour d'un optimiseur de table.
- `lastRun` – Un objet TableOptimizerRun.

  Un objet `TableOptimizerRun` représentant la dernière exécution de l'optimiseur de table.
- `configurationSource` – Chaîne UTF-8 (valeurs valides : `catalog="CATALOG"` | `table="TABLE"`).

  Spécifie la source de la configuration de l'optimiseur. Cela indique comment l'optimiseur de table a été configuré et quelle entité ou quel service a initié la configuration.

TableOptimizerConfiguration structure

Contient des détails sur la configuration d'un optimiseur de table. Vous transmettez cette configuration lors de la création ou de la mise à jour d'un optimiseur de table.

Champs

- `roleArn` – chaîne UTF-8, d'une longueur comprise entre 20 et 2048 octets, correspondant au Single-line string pattern.

  Un rôle transmis par l'appelant qui autorise le service à mettre à jour les ressources associées à l'optimiseur au nom de l'appelant.
- `enabled` : booléen.

  Si l'optimisation des tables est activée.

- `vpcConfiguration` – Un objet TableOptimizerVpcConfiguration.

  Un objet `TableOptimizerVpcConfiguration` représentant la configuration VPC pour un optimiseur de table.

  Cette configuration est nécessaire pour optimiser les tables qui se trouvent dans un VPC client.
- `compactionConfiguration` – Un objet CompactionConfiguration.

  Configuration d'un optimiseur de compactage. Cette configuration définit la manière dont les fichiers de données de votre table seront compactés afin d'améliorer les performances des requêtes et de réduire les coûts de stockage.
- `retentionConfiguration` – Un objet RetentionConfiguration.

  Configuration d'un optimiseur de conservation d'instantanés.
- `orphanFileDeletionConfiguration` : un objet OrphanFileDeletionConfiguration.

  Configuration d'un optimiseur de suppression de fichiers orphelins.

TableOptimizerVpcConfiguration structure

Un objet décrivant la configuration VPC pour un optimiseur de table.

Cette configuration est nécessaire pour optimiser les tables qui se trouvent dans un VPC client.

Champs

- `glueConnectionName` — Chaîne UTF-8, d'une longueur minimale de 1 octet.

  Nom de la AWS Glue connexion utilisée pour le VPC pour l'optimiseur de table.

CompactionConfiguration structure

Configuration d'un optimiseur de compactage. Cette configuration définit la manière dont les fichiers de données de votre table seront compactés afin d'améliorer les performances des requêtes et de réduire les coûts de stockage.

Champs

- `icebergConfiguration` : un objet IcebergCompactionConfiguration.

Configuration d'un optimiseur de compactage Iceberg.

IcebergCompactionConfiguration structure

Configuration d'un optimiseur de compactage Iceberg. Cette configuration définit les paramètres permettant d'optimiser la disposition des fichiers de données dans les tables Iceberg.

Champs

- `strategy` – Chaîne UTF-8 (valeurs valides : `binpack="BINPACK"` | `sort="SORT"` | `z-order="ZORDER"`).

  Stratégie à utiliser pour le compactage. Les valeurs valides sont :

  - `binpack` : combine de petits fichiers en fichiers plus volumineux, en ciblant généralement des tailles supérieures à 100 Mo, tout en appliquant les suppressions en attente. Il s'agit de la stratégie de compactage recommandée dans la plupart des cas d'utilisation.
  - `sort` : organise les données en fonction de colonnes spécifiées qui sont triées hiérarchiquement lors du compactage, améliorant ainsi les performances des requêtes pour les opérations filtrées. Cette stratégie est recommandée lorsque vos requêtes filtrent fréquemment sur des colonnes spécifiques. Pour utiliser cette stratégie, vous devez d'abord définir un ordre de tri dans les propriétés de votre table Iceberg à l'aide de la propriété de table `sort_order`.
  - `z-order` : optimise l'organisation des données en fusionnant plusieurs attributs en une seule valeur scalaire qui peut être utilisée pour le tri, ce qui permet d'effectuer des requêtes efficaces sur plusieurs dimensions. Cette stratégie est recommandée lorsque vous devez interroger des données simultanément sur plusieurs dimensions. Pour utiliser cette stratégie, vous devez d'abord définir un ordre de tri dans les propriétés de votre table Iceberg à l'aide de la propriété de table `sort_order`.

  Si aucune entrée n'est fournie, la valeur par défaut « binpack » sera utilisée.

- `minInputFiles` : nombre (entier).

  Le nombre minimum de fichiers de données qui doivent être présents dans une partition avant le compactage permet en réalité de compacter les fichiers. Ce paramètre permet de contrôler le moment où le compactage est déclenché, empêchant ainsi les opérations de compactage inutiles sur les partitions contenant peu de fichiers. Si aucune entrée n'est fournie, la valeur par défaut 100 sera utilisée.

- `deleteFileThreshold` : nombre (entier).

  Nombre minimal de suppressions qui doivent être présentes dans un fichier de données pour qu'il puisse être compacté. Ce paramètre permet d'optimiser le compactage en se concentrant sur les fichiers contenant un nombre important d'opérations de suppression, ce qui peut améliorer les performances des requêtes en supprimant les enregistrements supprimés. Si aucune entrée n'est fournie, la valeur par défaut 1 sera utilisée.

TableOptimizerRun structure

Contient les détails relatifs à l'exécution d'un optimiseur de table.

Champs

- `eventType` – Chaîne UTF-8 (valeurs valides : `starting="STARTING"` | `completed="COMPLETED"` | `failed="FAILED"` | `in_progress="IN_PROGRESS"`).

  Un type d'événement représentant l'état de l'exécution de l'optimiseur de table.

- `startTimestamp` – Horodatage.

  Représente l'horodatage de l'époque à laquelle la tâche de compactage a commencé dans Lake Formation.

- `endTimestamp` – Horodatage.

  Représente l'horodatage de l'époque à laquelle la tâche de compactage s'est terminée.

- `metrics` – Un objet RunMetrics.

  Un objet `RunMetrics` contenant des métriques pour l'exécution de l'optimiseur.

  Ce membre est obsolète. Consultez les membres individuels de la métrique pour le compactage, la conservation et la suppression des fichiers orphelins.

- `error` – Chaîne UTF-8.

  Une erreur survenue lors de l'exécution de l'optimiseur.

- `compactionMetrics` – Un objet CompactionMetrics.

  Un objet `CompactionMetrics` contenant des métriques pour l'exécution de l'optimiseur.

- `compactionStrategy` – Chaîne UTF-8 (valeurs valides : `binpack="BINPACK"` | `sort="SORT"` | `z-order="ZORDER"`).

  Stratégie utilisée pour l'exécution du compactage. Indique quel algorithme a été appliqué pour déterminer comment les fichiers ont été sélectionnés et combinés pendant le processus de compactage. Les valeurs valides sont :

  - `binpack` : combine de petits fichiers en fichiers plus volumineux, en ciblant généralement des tailles supérieures à 100 Mo, tout en appliquant les suppressions en attente. Il s'agit de la stratégie de compactage recommandée dans la plupart des cas d'utilisation.
  - `sort` : organise les données en fonction de colonnes spécifiées qui sont triées hiérarchiquement lors du compactage, améliorant ainsi les performances des requêtes pour les opérations filtrées. Cette stratégie est recommandée lorsque vos requêtes filtrent fréquemment sur des colonnes spécifiques. Pour utiliser cette stratégie, vous devez d'abord définir un ordre de tri dans les propriétés de votre table Iceberg à l'aide de la propriété de table `sort_order`.
  - `z-order` : optimise l'organisation des données en fusionnant plusieurs attributs en une seule valeur scalaire qui peut être utilisée pour le tri, ce qui permet d'effectuer des requêtes efficaces sur plusieurs dimensions. Cette stratégie est recommandée lorsque vous devez interroger des données simultanément sur plusieurs dimensions. Pour utiliser cette stratégie, vous devez d'abord définir un ordre de tri dans les propriétés de votre table Iceberg à l'aide de la propriété de table `sort_order`.
- `retentionMetrics` – Un objet RetentionMetrics.

  Un objet `RetentionMetrics` contenant des métriques pour l'exécution de l'optimiseur.
- `orphanFileDeletionMetrics` : un objet OrphanFileDeletionMetrics.

  Objet `OrphanFileDeletionMetrics` contenant des métriques pour l'exécution de l'optimiseur.

BatchGetTableOptimizerEntry structure

Représente un optimiseur de table à récupérer lors de l'opération `BatchGetTableOptimizer`.

Champs

- `catalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  L'ID de catalogue de la table.

- `databaseName` — Chaîne UTF-8, d'une longueur minimale de 1 octet.

  Nom de la base de données du catalogue où se trouve la table.

- `tableName` — Chaîne UTF-8, d'une longueur minimale de 1 octet.

  Nom de la table.

- `type` – Chaîne UTF-8 (valeurs valides : `compaction="COMPACTION"` | `retention="RETENTION"` | `orphan_file_deletion="ORPHAN_FILE_DELETION"`).

  Le type d'optimiseur de table.

BatchTableOptimizer structure

Contient les détails de l'un des optimiseurs de table renvoyés par l'opération `BatchGetTableOptimizer`.

Champs

- `catalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  L'ID de catalogue de la table.

- `databaseName` — Chaîne UTF-8, d'une longueur minimale de 1 octet.

  Nom de la base de données du catalogue où se trouve la table.

- `tableName` — Chaîne UTF-8, d'une longueur minimale de 1 octet.

  Nom de la table.

- `tableOptimizer` – Un objet TableOptimizer.

  Un objet `TableOptimizer` qui contient des détails sur la configuration et la dernière exécution d'un optimiseur de table.

BatchGetTableOptimizerError structure

Contient des détails sur l'une des erreurs de la liste d'erreurs renvoyée par l'opération `BatchGetTableOptimizer`.

Champs

- `error` : un objet ErrorDetail.

  Un objet `ErrorDetail` contenant le code et les détails du message sur l'erreur.

- `catalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  L'ID de catalogue de la table.

- `databaseName` — Chaîne UTF-8, d'une longueur minimale de 1 octet.

  Nom de la base de données du catalogue où se trouve la table.

- `tableName` — Chaîne UTF-8, d'une longueur minimale de 1 octet.

  Nom de la table.

- `type` – Chaîne UTF-8 (valeurs valides : `compaction="COMPACTION"` | `retention="RETENTION"` | `orphan_file_deletion="ORPHAN_FILE_DELETION"`).

  Le type d'optimiseur de table.

RetentionConfiguration structure

Configuration d'un optimiseur de conservation d'instantanés.

Champs

- `icebergConfiguration` : un objet IcebergRetentionConfiguration.

  Configuration d'un optimiseur de conservation des instantanés Iceberg.

IcebergRetentionConfiguration structure

Configuration d'un optimiseur de conservation des instantanés Iceberg.

Champs

- `snapshotRetentionPeriodInDays` : nombre (entier).

Nombre de jours pendant lesquels les instantanés Iceberg doivent être conservés. Si aucune entrée n'est fournie, le champ de configuration de table Iceberg correspondant sera utilisé ou, s'il n'est pas présent, la valeur par défaut 5 sera utilisée.

- `numberOfSnapshotsToRetain` : nombre (entier).

  Nombre d'instantanés Iceberg à conserver pendant la période de conservation. Si aucune entrée n'est fournie, le champ de configuration de table Iceberg correspondant sera utilisé ou, s'il n'est pas présent, la valeur par défaut 1 sera utilisée.

- `cleanExpiredFiles` : booléen.

  Si cette valeur est définie sur false, les instantanés sont uniquement supprimés des métadonnées de la table, et les fichiers de données et de métadonnées sous-jacents ne sont pas supprimés.

- `runRateInHours` : nombre (entier).

  Intervalle en heures entre les exécutions des tâches de conservation. Ce paramètre contrôle la fréquence d'exécution de l'optimiseur de conservation pour nettoyer les instantanés expirés. La valeur doit être comprise entre 3 et 168 heures (7 jours). Si aucune entrée n'est fournie, la valeur par défaut 24 sera utilisée.

OrphanFileDeletionConfiguration structure

Configuration d'un optimiseur de suppression de fichiers orphelins.

Champs

- `icebergConfiguration` : un objet IcebergOrphanFileDeletionConfiguration.

  Configuration d'un optimiseur de suppression de fichiers orphelins Iceberg.

IcebergOrphanFileDeletionConfiguration structure

Configuration d'un optimiseur de suppression de fichiers orphelins Iceberg.

Champs

- `orphanFileRetentionPeriodInDays` : nombre (entier).

Nombre de jours pendant lesquels les fichiers orphelins doivent être conservés avant leur suppression. Si aucune entrée n'est fournie, la valeur par défaut 3 sera utilisée.

- `location` – Chaîne UTF-8.

  Spécifie un répertoire dans lequel les fichiers seront recherchés (par défaut, l'emplacement de la table). Vous pouvez choisir un sous-répertoire plutôt que l'emplacement de la table de niveau supérieur.

- `runRateInHours` : nombre (entier).

  L'intervalle en heures entre les exécutions des tâches de suppression de fichiers orphelins. Ce paramètre contrôle la fréquence à laquelle l'optimiseur de suppression des fichiers orphelins sera exécuté pour nettoyer les fichiers orphelins. La valeur doit être comprise entre 3 et 168 heures (7 jours). Si aucune entrée n'est fournie, la valeur par défaut 24 sera utilisée.

CompactionMetrics structure

Structure contenant les métriques de compactage pour l'exécution de l'optimiseur.

Champs

- `IcebergMetrics` : un objet IcebergCompactionMetrics.

  Structure contenant les métriques de compactage Iceberg pour l'exécution de l'optimiseur.

RetentionMetrics structure

Structure contenant les métriques de conservation pour l'exécution de l'optimiseur.

Champs

- `IcebergMetrics` : un objet IcebergRetentionMetrics.

  Structure contenant les métriques de conservation Iceberg pour l'exécution de l'optimiseur.

OrphanFileDeletionMetrics structure

Structure contenant les métriques de suppression de fichiers orphelins pour l'exécution de l'optimiseur.

Champs

- `IcebergMetrics` : un objet IcebergOrphanFileDeletionMetrics.

  Structure contenant les métriques de suppression des fichiers orphelins Iceberg pour l'exécution de l'optimiseur.

IcebergCompactionMetrics structure

Métriques de compactage d'Iceberg pour l'exécution de l'optimiseur.

Champs

- `DpuHours` – Nombre (double).

  Nombre d'heures DPU consommées par la tâche.
- `NumberOfDpus` : nombre (entier).

  Le nombre de personnes DPUs consommées par la tâche, arrondi au nombre entier le plus proche.
- `JobDurationInHour` – Nombre (double).

  Durée de la tâche en heures.

IcebergRetentionMetrics structure

Métriques de conservation des instantanés d'Iceberg pour l'exécution de l'optimiseur.

Champs

- `DpuHours` – Nombre (double).

  Nombre d'heures DPU consommées par la tâche.
- `NumberOfDpus` : nombre (entier).

  Le nombre de personnes DPUs consommées par la tâche, arrondi au nombre entier le plus proche.
- `JobDurationInHour` – Nombre (double).

  Durée de la tâche en heures.

IcebergOrphanFileDeletionMetrics structure

Métriques de suppression de fichiers orphelins d'Iceberg pour l'exécution de l'optimiseur.

Champs

- `DpuHours` – Nombre (double).

  Nombre d'heures DPU consommées par la tâche.

- `NumberOfDpus` : nombre (entier).

  Le nombre de personnes DPUs consommées par la tâche, arrondi au nombre entier le plus proche.

- `JobDurationInHour` – Nombre (double).

  Durée de la tâche en heures.

RunMetrics structure

Métriques relatives à l'exécution de l'optimiseur.

Cette structure est obsolète. Consultez les membres individuels de la métrique pour le compactage, la conservation et la suppression des fichiers orphelins.

Champs

- `NumberOfBytesCompacted` – Chaîne UTF-8.

  Nombre d'octets supprimés lors de l'exécution de la tâche de compactage.

- `NumberOfFilesCompacted` – Chaîne UTF-8.

  Nombre de fichiers supprimés lors de l'exécution de la tâche de compactage.

- `NumberOfDpus` – Chaîne UTF-8.

  Le nombre de personnes DPUs consommées par la tâche, arrondi au nombre entier le plus proche.

- `JobDurationInHour` – Chaîne UTF-8.

  Durée de la tâche en heures.

Opérations

- GetTableOptimizer action (Python : get_table_optimizer)
- BatchGetTableOptimizer action (Python : batch_get_table_optimizer)
- ListTableOptimizerRuns action (Python : list_table_optimizer_runs)
- CreateTableOptimizer action (Python : create_table_optimizer)
- DeleteTableOptimizer action (Python : delete_table_optimizer)
- UpdateTableOptimizer action (Python : update_table_optimizer)

GetTableOptimizer action (Python : get_table_optimizer)

Renvoie la configuration de tous les optimiseurs associés à une table spécifiée.

Demande

- `CatalogId` – Obligatoire : Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  L'ID de catalogue de la table.
- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la base de données du catalogue où se trouve la table.
- `TableName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table.
- `Type` – Obligatoire : Chaîne UTF-8 (valeurs valides : `compaction="COMPACTION"` | `retention="RETENTION"` | `orphan_file_deletion="ORPHAN_FILE_DELETION"`).

  Le type d'optimiseur de table.

Réponse

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

L'ID de catalogue de la table.

- `DatabaseName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la base de données du catalogue où se trouve la table.

- `TableName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table.

- `TableOptimizer` – Un objet TableOptimizer.

  L'optimiseur associé à la table spécifiée.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `AccessDeniedException`
- `InternalServiceException`
- `ThrottlingException`

BatchGetTableOptimizer action (Python : batch_get_table_optimizer)

Renvoie la configuration des optimiseurs de table spécifiés.

Demande

- `Entries` – Obligatoire : un tableau d'objets BatchGetTableOptimizerEntry.

  Liste d'objets `BatchGetTableOptimizerEntry` spécifiant les optimiseurs de table à récupérer.

Réponse

- `TableOptimizers` : un tableau d'objets BatchTableOptimizer.

  Liste d'objets `BatchTableOptimizer`.

- `Failures` : un tableau d'objets BatchGetTableOptimizerError.

  Liste des erreurs liées à l'opération.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `AccessDeniedException`
- `InternalServiceException`
- `ThrottlingException`

ListTableOptimizerRuns action (Python : list_table_optimizer_runs)

Répertorie l'historique des exécutions précédentes de l'optimiseur pour une table spécifique.

Demande

- `CatalogId` – Obligatoire : Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  L'ID de catalogue de la table.
- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la base de données du catalogue où se trouve la table.
- `TableName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table.
- `Type` – Obligatoire : Chaîne UTF-8 (valeurs valides : `compaction="COMPACTION"` | `retention="RETENTION"` | `orphan_file_deletion="ORPHAN_FILE_DELETION"`).

  Le type d'optimiseur de table.
- `MaxResults` : nombre (entier).

  Nombre maximal d'exécutions d'optimisation à renvoyer à chaque appel.

- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, s'il s'agit d'un appel de continuation.

Réponse

- `CatalogId` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  L'ID de catalogue de la table.

- `DatabaseName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la base de données du catalogue où se trouve la table.

- `TableName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table.

- `NextToken` – Chaîne UTF-8.

  Jeton de continuation pour la pagination de la liste renvoyée des exécutions de l'optimiseur, renvoyé si le segment actuel de la liste n'est pas le dernier.

- `TableOptimizerRuns` : un tableau d'objets TableOptimizerRun.

  Une liste des exécutions d'optimiseur associées à une table.

Erreurs

- `EntityNotFoundException`
- `AccessDeniedException`
- `InvalidInputException`
- `ValidationException`
- `InternalServiceException`
- `ThrottlingException`

CreateTableOptimizer action (Python : create_table_optimizer)

Créer un optimiseur de table pour une fonction spécifique.

Demande

- `CatalogId` – Obligatoire : Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  L'ID de catalogue de la table.
- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la base de données du catalogue où se trouve la table.
- `TableName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table.
- `Type` – Obligatoire : Chaîne UTF-8 (valeurs valides : `compaction="COMPACTION"` | `retention="RETENTION"` | `orphan_file_deletion="ORPHAN_FILE_DELETION"`).

  Le type d'optimiseur de table.
- `TableOptimizerConfiguration` – Obligatoire : un objet TableOptimizerConfiguration.

  Un objet `TableOptimizerConfiguration` représentant la configuration d'un optimiseur de table.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `EntityNotFoundException`
- `ValidationException`
- `InvalidInputException`
- `AccessDeniedException`

- `AlreadyExistsException`
- `InternalServiceException`
- `ThrottlingException`

DeleteTableOptimizer action (Python : delete_table_optimizer)

Supprime un optimiseur et toutes les métadonnées associées à une table. L'optimisation ne sera plus effectuée sur la table.

Demande

- `CatalogId` – Obligatoire : Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  L'ID de catalogue de la table.
- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la base de données du catalogue où se trouve la table.
- `TableName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table.
- `Type` – Obligatoire : Chaîne UTF-8 (valeurs valides : `compaction="COMPACTION"` | `retention="RETENTION"` | `orphan_file_deletion="ORPHAN_FILE_DELETION"`).

  Le type d'optimiseur de table.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `AccessDeniedException`

- `InternalServiceException`
- `ThrottlingException`

UpdateTableOptimizer action (Python : update_table_optimizer)

Met à jour la configuration d'un optimiseur de table existant.

Demande

- `CatalogId` – Obligatoire : Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  L'ID de catalogue de la table.

- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la base de données du catalogue où se trouve la table.

- `TableName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table.

- `Type` – Obligatoire : Chaîne UTF-8 (valeurs valides : `compaction="COMPACTION"` | `retention="RETENTION"` | `orphan_file_deletion="ORPHAN_FILE_DELETION"`).

  Le type d'optimiseur de table.

- `TableOptimizerConfiguration` – Obligatoire : un objet TableOptimizerConfiguration.

  Un objet `TableOptimizerConfiguration` représentant la configuration d'un optimiseur de table.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `EntityNotFoundException`

- `InvalidInputException`
- `AccessDeniedException`
- `ValidationException`
- `InternalServiceException`
- `ThrottlingException`
- `ConcurrentModificationException`

API de classifieurs et d'crawlers

L'API Crawler and classifiers décrit les types de données du AWS Glue crawler et du classificateur, et inclut l'API permettant de créer, supprimer, mettre à jour et répertorier les crawlers ou les classificateurs.

Rubriques

- API du classifieur
- API du crawler
- API de statistiques de colonne
- API du planificateur du crawler

API du classifieur

L'API Classifier décrit les types de données du AWS Glue classificateur et inclut l'API permettant de créer, de supprimer, de mettre à jour et de répertorier les classificateurs.

Types de données

- Structure du classifieur
- GrokClassifier structure
- XMLClassifier structure
- JsonClassifier structure
- CsvClassifier structure
- CreateGrokClassifierRequest structure
- UpdateGrokClassifierRequest structure

- Créer une structure XMLClassifier de demande
- Structure de XMLClassifier demande de mise à jour
- CreateJsonClassifierRequest structure
- UpdateJsonClassifierRequest structure
- CreateCsvClassifierRequest structure
- UpdateCsvClassifierRequest structure

Structure du classifieur

Les classifieurs sont déclenchés durant une tâche d'analyse. Un classifieur vérifie si un fichier donné est dans un format qu'il peut gérer. Si c'est le cas, le classifieur crée un schéma sous la forme d'un objet `StructType` correspondant à ce format de données.

Vous pouvez utiliser les classificateurs standard AWS Glue fournis, ou vous pouvez écrire vos propres classificateurs pour classer au mieux vos sources de données et spécifier les schémas appropriés à utiliser pour celles-ci. Un classifieur peut être un classifieur `grok`, un classifieur `XML`, un classifieur `JSON` ou un classifieur `CSV` personnalisé, selon ce qui est spécifié dans l'un des champs de l'objet `Classifier`.

Champs

- `GrokClassifier` – Un objet GrokClassifier.

  Classificateur qui utilise `grok`.
- `XMLClassifier` – Un objet XMLClassifier.

  Classificateur de contenu XML.
- `JsonClassifier` – Un objet JsonClassifier.

  Classificateur de contenu JSON.
- `CsvClassifier` – Un objet CsvClassifier.

  Classificateur pour les valeurs séparées par des virgules (CSV).

GrokClassifier structure

Classifieur qui utilise des modèles `grok`.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du classifieur.
- `Classification` – Obligatoire : chaîne UTF-8.

  Identifiant du format des données auquel le classifieur correspond, comme les journaux Twitter, JSON, Omniture, etc.
- `CreationTime` – Horodatage.

  Heure à laquelle ce classificateur a été enregistré.
- `LastUpdated` – Horodatage.

  Heure de la dernière mise à jour de ce classifieur.
- `Version` – Nombre (long).

  Version de ce classifieur.
- `GrokPattern` – Obligatoire : chaîne UTF-8, d'une longueur comprise entre 1 et 2048 octets, correspondant au A Logstash Grok string pattern.

  Modèle grok appliqué à un magasin de données par ce classifieur. Pour plus d'informations, consultez les modèles intégrés dans Writing Custom Classifiers (Écriture de classifieurs personnalisés).
- `CustomPatterns` – Chaîne UTF-8, d'une longueur maximale de 16 000 octets, correspondant au URI address multi-line string pattern.

  Modèles grok personnalisés facultatifs définis par ce classifieur. Pour plus d'informations, consultez les modèles personnalisés dans Writing Custom Classifiers (Écriture de classifieurs personnalisés).

XMLClassifier structure

Classifieur de contenu `XML`.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

Nom du classifieur.

- `Classification` – Obligatoire : chaîne UTF-8.

  Identifiant du format des données que le classifieur fait correspondre.

- `CreationTime` – Horodatage.

  Heure à laquelle ce classificateur a été enregistré.

- `LastUpdated` – Horodatage.

  Heure de la dernière mise à jour de ce classifieur.

- `Version` – Nombre (long).

  Version de ce classifieur.

- `RowTag` – Chaîne UTF-8.

  Balise XML désignant l'élément contenant chaque enregistrement d'un document XML en cours d'analyse. Cela ne permet pas d'identifier un élément à fermeture automatique (fermé par `/>`). Un élément de ligne vide contenant uniquement des attributs peut être analysé tant qu'il se termine par une balise de fermeture (par exemple, `<row item_a="A" item_b="B"></row>` est correct, mais `<row item_a="A" item_b="B" />` ne l'est pas).

JsonClassifier structure

Classifieur de contenu `JSON`.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du classifieur.

- `CreationTime` – Horodatage.

  Heure à laquelle ce classificateur a été enregistré.

- `LastUpdated` – Horodatage.

  Heure de la dernière mise à jour de ce classifieur.

- `Version` – Nombre (long).

Version de ce classifieur.

- `JsonPath` – Obligatoire : chaîne UTF-8.

  `JsonPath`Chaîne définissant les données JSON à classer par le classificateur. AWS Glue prend en charge un sous-ensemble de JsonPath, comme décrit dans la section Écrire des JsonPath classificateurs personnalisés.

CsvClassifier structure

Classifieur de contenu `CSV` personnalisé.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du classifieur.
- `CreationTime` – Horodatage.

  Heure à laquelle ce classificateur a été enregistré.
- `LastUpdated` – Horodatage.

  Heure de la dernière mise à jour de ce classifieur.
- `Version` – Nombre (long).

  Version de ce classifieur.
- `Delimiter` – Chaîne UTF-8, d'une longueur d'au moins 1 ou de plus de 1 octet, correspondant au Custom string pattern #26.

  Symbole personnalisé pour indiquer ce qui sépare chaque entrée de colonne dans la ligne.
- `QuoteSymbol` – Chaîne UTF-8, d'une longueur d'au moins 1 ou de plus de 1 octet, correspondant au Custom string pattern #26.

  Symbole personnalisé pour indiquer ce qui combine le contenu en une seule valeur de colonne. Doit être différent du délimiteur de colonne.
- `ContainsHeader` – Chaîne UTF-8 (valeurs valides : `UNKNOWN` | `PRESENT` | `ABSENT`).

  Indique si le fichier CSV contient un en-tête.

- `Header` – Tableau de chaînes UTF-8.

  Liste des chaînes représentant les noms des colonnes.

- `DisableValueTrimming` : booléen.

  Spécifie de ne pas couper les valeurs avant d'identifier le type des valeurs de colonne. La valeur par défaut est `true`.

- `AllowSingleColumn` : booléen.

  Active le traitement des fichiers qui ne contiennent qu'une seule colonne.

- `CustomDatatypeConfigured` : booléen.

  Permet de configurer le type de données personnalisé.

- `CustomDatatypes` – Tableau de chaînes UTF-8.

  Liste de types de données personnalisés, notamment « BINARY », « BOOLEAN », « DATE », « DECIMAL », « DOUBLE », « FLOAT », « INT », « LONG », « SHORT », « STRING », « TIMESTAMP ».

- `Serde` – Chaîne UTF-8 (valeurs valides : `OpenCSVSerDe` | `LazySimpleSerDe` | `None`).

  Définit le CSV SerDe pour le traitement dans le classificateur, qui sera appliqué dans le catalogue de données. Les valeurs valides sont `OpenCSVSerDe`, `LazySimpleSerDe` et `None`. Vous pouvez spécifier la valeur `None` lorsque vous souhaitez que le Crawler effectue la détection.

CreateGrokClassifierRequest structure

Spécifie un classifieur `grok` que `CreateClassifier` doit créer.

Champs

- `Classification` – Obligatoire : chaîne UTF-8.

  Identifiant du format de données auquel le classificateur correspond, tel que Twitter, JSON, Omniture CloudWatch logs, Amazon Logs, etc.

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du nouveau classifieur.

- `GrokPattern` – Obligatoire : chaîne UTF-8, d'une longueur comprise entre 1 et 2048 octets, correspondant au A Logstash Grok string pattern.

  Modèle grok utilisé par ce classifieur.
- `CustomPatterns` – Chaîne UTF-8, d'une longueur maximale de 16 000 octets, correspondant au URI address multi-line string pattern.

  Modèles grok personnalisés facultatifs utilisés par ce classifieur.

UpdateGrokClassifierRequest structure

Spécifie un classifieur grok à mettre à jour lorsqu'il est transmis à `UpdateClassifier`.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Le nom de l'`GrokClassifier`.
- `Classification` – Chaîne UTF-8.

  Identifiant du format de données auquel le classificateur correspond, tel que Twitter, JSON, Omniture CloudWatch logs, Amazon Logs, etc.
- `GrokPattern` – Chaîne UTF-8, d'une longueur comprise entre 1 et 2048 octets, correspondant au A Logstash Grok string pattern.

  Modèle grok utilisé par ce classifieur.
- `CustomPatterns` – Chaîne UTF-8, d'une longueur maximale de 16 000 octets, correspondant au URI address multi-line string pattern.

  Modèles grok personnalisés facultatifs utilisés par ce classifieur.

Créer une structure XMLClassifier de demande

Spécifie un classifieur XML que `CreateClassifier` doit créer.

Champs

- `Classification` – Obligatoire : chaîne UTF-8.

Identifiant du format des données que le classifieur fait correspondre.

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du classifieur.

- `RowTag` – Chaîne UTF-8.

  Balise XML désignant l'élément contenant chaque enregistrement d'un document XML en cours d'analyse. Cela ne permet pas d'identifier un élément à fermeture automatique (fermé par `/>`). Un élément de ligne vide contenant uniquement des attributs peut être analysé tant qu'il se termine par une balise de fermeture (par exemple, `<row item_a="A" item_b="B"></row>` est correct, mais `<row item_a="A" item_b="B" />` ne l'est pas).

Structure de XMLClassifier demande de mise à jour

Spécifie un classifieur XML à mettre à jour.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du classifieur.

- `Classification` – Chaîne UTF-8.

  Identifiant du format des données que le classifieur fait correspondre.

- `RowTag` – Chaîne UTF-8.

  Balise XML désignant l'élément contenant chaque enregistrement d'un document XML en cours d'analyse. Cela ne permet pas d'identifier un élément à fermeture automatique (fermé par `/>`). Un élément de ligne vide contenant uniquement des attributs peut être analysé tant qu'il se termine par une balise de fermeture (par exemple, `<row item_a="A" item_b="B"></row>` est correct, mais `<row item_a="A" item_b="B" />` ne l'est pas).

CreateJsonClassifierRequest structure

Spécifie un classifieur JSON que `CreateClassifier` doit créer.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du classifieur.

- `JsonPath` – Obligatoire : chaîne UTF-8.

  `JsonPath`Chaîne définissant les données JSON à classer par le classificateur. AWS Glue prend en charge un sous-ensemble de JsonPath, comme décrit dans la section Écrire des JsonPath classificateurs personnalisés.

UpdateJsonClassifierRequest structure

Spécifie un classifieur JSON à mettre à jour.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du classifieur.

- `JsonPath` – Chaîne UTF-8.

  `JsonPath`Chaîne définissant les données JSON à classer par le classificateur. AWS Glue prend en charge un sous-ensemble de JsonPath, comme décrit dans la section Écrire des JsonPath classificateurs personnalisés.

CreateCsvClassifierRequest structure

Spécifie un classifieur CSV personnalisé que `CreateClassifier` doit créer.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du classifieur.

- `Delimiter` – Chaîne UTF-8, d'une longueur d'au moins 1 ou de plus de 1 octet, correspondant au Custom string pattern #26.

  Symbole personnalisé pour indiquer ce qui sépare chaque entrée de colonne dans la ligne.

- `QuoteSymbol` – Chaîne UTF-8, d'une longueur d'au moins 1 ou de plus de 1 octet, correspondant au Custom string pattern #26.

  Symbole personnalisé pour indiquer ce qui combine le contenu en une seule valeur de colonne. Doit être différent du délimiteur de colonne.

- `ContainsHeader` – Chaîne UTF-8 (valeurs valides : `UNKNOWN` | `PRESENT` | `ABSENT`).

  Indique si le fichier CSV contient un en-tête.

- `Header` – Tableau de chaînes UTF-8.

  Liste des chaînes représentant les noms des colonnes.

- `DisableValueTrimming` : booléen.

  Spécifie de ne pas couper les valeurs avant d'identifier le type des valeurs de colonne. La valeur par défaut est True.

- `AllowSingleColumn` : booléen.

  Active le traitement des fichiers qui ne contiennent qu'une seule colonne.

- `CustomDatatypeConfigured` : booléen.

  Permet la configuration du type de données personnalisé.

- `CustomDatatypes` – Tableau de chaînes UTF-8.

  Crée une liste des types de données personnalisés pris en charge.

- `Serde` – Chaîne UTF-8 (valeurs valides : `OpenCSVSerDe` | `LazySimpleSerDe` | `None`).

  Définit le CSV SerDe pour le traitement dans le classificateur, qui sera appliqué dans le catalogue de données. Les valeurs valides sont `OpenCSVSerDe`, `LazySimpleSerDe` et `None`. Vous pouvez spécifier la valeur `None` lorsque vous souhaitez que le Crawler effectue la détection.

UpdateCsvClassifierRequest structure

Spécifie un classifieur CSV personnalisé à mettre à jour.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du classifieur.

- `Delimiter` – Chaîne UTF-8, d'une longueur d'au moins 1 ou de plus de 1 octet, correspondant au Custom string pattern #26.

  Symbole personnalisé pour indiquer ce qui sépare chaque entrée de colonne dans la ligne.

- `QuoteSymbol` – Chaîne UTF-8, d'une longueur d'au moins 1 ou de plus de 1 octet, correspondant au Custom string pattern #26.

  Symbole personnalisé pour indiquer ce qui combine le contenu en une seule valeur de colonne. Doit être différent du délimiteur de colonne.

- `ContainsHeader` – Chaîne UTF-8 (valeurs valides : `UNKNOWN` | `PRESENT` | `ABSENT`).

  Indique si le fichier CSV contient un en-tête.

- `Header` – Tableau de chaînes UTF-8.

  Liste des chaînes représentant les noms des colonnes.

- `DisableValueTrimming` : booléen.

  Spécifie de ne pas couper les valeurs avant d'identifier le type des valeurs de colonne. La valeur par défaut est True.

- `AllowSingleColumn` : booléen.

  Active le traitement des fichiers qui ne contiennent qu'une seule colonne.

- `CustomDatatypeConfigured` : booléen.

  Spécifie la configuration du type de données personnalisé.

- `CustomDatatypes` – Tableau de chaînes UTF-8.

  Spécifie une liste des types de données personnalisés pris en charge.

- `Serde` – Chaîne UTF-8 (valeurs valides : `OpenCSVSerDe` | `LazySimpleSerDe` | `None`).

Définit le CSV SerDe pour le traitement dans le classificateur, qui sera appliqué dans le catalogue de données. Les valeurs valides sont `OpenCSVSerDe`, `LazySimpleSerDe` et `None`. Vous pouvez spécifier la valeur `None` lorsque vous souhaitez que le Crawler effectue la détection.

Opérations

- CreateClassifier action (Python : create_classifier)
- DeleteClassifier action (Python : delete_classifier)
- GetClassifier action (Python : get_classifier)
- GetClassifiers action (Python : get_classifiers)
- UpdateClassifier action (Python : update_classifier)

CreateClassifier action (Python : create_classifier)

Crée un classifieur dans le compte de l'utilisateur. Cela peut être un `GrokClassifier`, un `XMLClassifier`, un `JsonClassifier` ou un `CsvClassifier`, selon le champ de la demande qui est présent.

Demande

- `GrokClassifier` – Un objet CreateGrokClassifierRequest.

  Un `GrokClassifier` objet spécifiant le classifieur à créer.
- `XMLClassifier` – Un objet Créer une XMLClassifier demande.

  Un `XMLClassifier` objet spécifiant le classifieur à créer.
- `JsonClassifier` – Un objet CreateJsonClassifierRequest.

  Un `JsonClassifier` objet spécifiant le classifieur à créer.
- `CsvClassifier` – Un objet CreateCsvClassifierRequest.

  Un `CsvClassifier` objet spécifiant le classifieur à créer.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `AlreadyExistsException`
- `InvalidInputException`
- `OperationTimeoutException`

DeleteClassifier action (Python : delete_classifier)

Supprime un classifieur du catalogue de données.

Demande

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du classifieur à supprimer.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `EntityNotFoundException`
- `OperationTimeoutException`

GetClassifier action (Python : get_classifier)

Récupérer un classifieur par son nom.

Demande

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du classifieur à récupérer.

Réponse

- `Classifier` – Un objet Classifieur.

  Classifieur demandé.

Erreurs

- `EntityNotFoundException`
- `OperationTimeoutException`

GetClassifiers action (Python : get_classifiers)

Répertorie toutes les objets Classifieur du catalogue de données.

Demande

- `MaxResults` – Nombre (entier), compris entre 1 et 1 000.

  Taille de la liste à renvoyer (facultatif).
- `NextToken` – Chaîne UTF-8.

  Jeton de continuation facultatif.

Réponse

- `Classifiers` : un tableau d'objets Classifieur.

  Liste des objets Classifier demandée.
- `NextToken` – Chaîne UTF-8.

  Jeton de continuation.

Erreurs

- `OperationTimeoutException`

UpdateClassifier action (Python : update_classifier)

Modifie un classifieur existant (`GrokClassifier`, `XMLClassifier`, `JsonClassifier` ou `CsvClassifier`, selon le champ qui est présent).

Demande

- `GrokClassifier` – Un objet UpdateGrokClassifierRequest.

  Un objet `GrokClassifier` avec des champs mis à jour.
- `XMLClassifier` – Un objet XMLClassifierDemande de mise à jour.

  Un objet `XMLClassifier` avec des champs mis à jour.
- `JsonClassifier` – Un objet UpdateJsonClassifierRequest.

  Un objet `JsonClassifier` avec des champs mis à jour.
- `CsvClassifier` – Un objet UpdateCsvClassifierRequest.

  Un objet `CsvClassifier` avec des champs mis à jour.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `InvalidInputException`
- `VersionMismatchException`
- `EntityNotFoundException`
- `OperationTimeoutException`

API du crawler

L'API Crawler décrit les types de données des AWS Glue robots d'exploration, ainsi que l'API permettant de créer, de supprimer, de mettre à jour et de répertorier les robots d'exploration.

Types de données

- Structure du crawler
- Structure du planificateur
- CrawlerTargets structure
- Structure de la S3Target
- DeltaCatalogTarget Structure S3
- DeltaDirectTarget Structure S3
- JdbcTarget structure
- Structure Mongo DBTarget
- Structure de la dynamo DBTarget
- DeltaTarget structure
- IcebergTarget structure
- HudiTarget structure
- CatalogTarget structure
- CrawlerMetrics structure
- CrawlerHistory structure
- CrawlsFilter structure
- SchemaChangePolicy structure
- LastCrawlInfo structure
- RecrawlPolicy structure
- LineageConfiguration structure
- LakeFormationConfiguration structure

Structure du crawler

Spécifie un crawler qui examine une source de données et utilise des classifieurs pour tenter de déterminer son schéma. Si l'action aboutit, l'crawler enregistre les métadonnées relatives à la source de données dans le AWS Glue Data Catalog.

Champs

- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de l'crawler.

- `Role` – Chaîne UTF-8.

  L'Amazon Resource Name (ARN) d'un rôle IAM utilisé pour accéder aux ressources client, par exemple des données Amazon Simple Storage Service (Amazon S3).

- `Targets` – Un objet CrawlerTargets.

  Ensemble de cibles à analyser.

- `DatabaseName` – Chaîne UTF-8.

  Nom de la base de données dans laquelle le résultat de l'crawler est stockée.

- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description du crawler.

- `Classifiers` – Tableau de chaînes UTF-8.

  Liste de chaînes UTF-8 qui spécifient les classificateurs personnalisés et associés à l'crawler.

- `RecrawlPolicy` – Un objet RecrawlPolicy.

  Stratégie qui spécifie s'il faut analyser à nouveau le jeu de données entier ou analyser uniquement les dossiers ajoutés depuis la dernière exécution du crawler.

- `SchemaChangePolicy` – Un objet SchemaChangePolicy.

  Stratégie qui spécifie la mise à jour et la suppression des comportements pour l'crawler.

- `LineageConfiguration` – Un objet LineageConfiguration.

  Configuration qui spécifie si la lignée de données est activée pour le crawler.

- `State` – Chaîne UTF-8 (valeurs valides : `READY` | `RUNNING` | `STOPPING`).

  Indique si le crawler est en cours d'exécution, ou si une exécution est en attente.

- `TablePrefix` – Chaîne UTF-8, d'une longueur maximale de 128 octets.

Préfixe ajouté aux noms des tables créées.

- `Schedule` – Un objet Planificateur.

  Pour les crawlers planifiés, planification de l'exécution du crawler.

- `CrawlElapsedTime` – Nombre (long).

  Si le crawler est en cours d'exécution, contient le temps écoulé total depuis le début de la dernière analyse.

- `CreationTime` – Horodatage.

  Heure de création du crawler.

- `LastUpdated` – Horodatage.

  Heure de la dernière mise à jour du crawler.

- `LastCrawl` – Un objet LastCrawlInfo.

  État de la dernière analyse, et éventuellement informations d'erreur si une erreur s'est produite.

- `Version` – Nombre (long).

  Version de l'crawler.

- `Configuration` – Chaîne UTF-8.

  Informations sur la configuration du crawler. Cette chaîne JSON avec gestion des versions permet aux utilisateurs de spécifier des aspects du comportement d'un crawler. Pour plus d'informations, consultez Setting Crawler configuration options (Définition d'options de configuration du crawler).

- `CrawlerSecurityConfiguration` – Chaîne UTF-8, d'une longueur maximale de 128 octets.

  Nom de la structure `SecurityConfiguration` qui sera utilisée par ce crawler.

- `LakeFormationConfiguration` – Un objet LakeFormationConfiguration.

  Spécifie si le robot d'exploration doit utiliser les AWS Lake Formation informations d'identification du robot au lieu des informations d'identification du rôle IAM.

Structure du planificateur

Objet de planification utilisant une instruction `cron` pour planifier un événement.

Champs

- `ScheduleExpression` – Chaîne UTF-8.

  Une expression `cron` utilisée pour spécifier la planification (consultez Time-Based Schedules for Jobs and Crawlers (Planifications temporelles pour les tâches et les crawlers)). Par exemple, pour exécuter un élément tous les jours à 12h15 UTC, vous devez spécifier : `cron(15 12 * * ? *)`.
- `State` – Chaîne UTF-8 (valeurs valides : `SCHEDULED` | `NOT_SCHEDULED` | `TRANSITIONING`).

  État de la planification.

CrawlerTargets structure

Indique les magasins de données à analyser.

Champs

- `S3Targets` : un tableau d'objets S3Target.

  Spécifie des cibles Amazon Simple Storage Service (Amazon S3).
- `JdbcTargets` : un tableau d'objets JdbcTarget.

  Spécifie les cibles JDBC.
- `MongoDBTargets` – Un tableau d'objets Mongo DBTarget.

  Spécifie les cibles Amazon DocumentDB ou MongoDB.
- `DynamoDBTargets` – Un tableau d'objets Dynamo DBTarget.

  Spécifie des cibles Amazon DynamoDB.
- `CatalogTargets` – Un tableau d'objets CatalogTarget.

  Spécifie AWS Glue Data Catalog les cibles.
- `DeltaTargets` – Un tableau d'objets DeltaTarget.

  Spécifie les cibles du stockage de données Delta.
- `IcebergTargets` : un tableau d'objets IcebergTarget.

  Spécifie les cibles du magasin de données Apache Iceberg.
- `HudiTargets` : un tableau d'objets HudiTarget.

Spécifie les cibles du magasin de données Hudi Iceberg.

Structure de la S3Target

Spécifie un magasin de données dans Amazon Simple Storage Service (Amazon S3).

Champs

- `Path` – Chaîne UTF-8.

  Chemin vers la cible Amazon S3.

- `Exclusions` – Tableau de chaînes UTF-8.

  Liste de modèles glob utilisés à exclure de l'analyse. Pour en savoir plus, consultez Catalog Tables with a Crawler (Tables de catalogues avec un crawler).

- `ConnectionName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 2 048 octets.

  Nom d'une connexion qui permet à une tâche ou à un crawler d'accéder aux données dans Amazon S3 au sein d'un environnement Amazon Virtual Private Cloud (Amazon VPC).

- `SampleSize` : nombre (entier).

  Définit le nombre de fichiers dans chaque dossier feuille à analyser lors de l'analyse d'échantillons de fichiers dans un jeu de données. Si ce paramètre n'est pas défini, tous les fichiers sont analysés. Une valeur valide est un entier compris entre 1 et 249.

- `EventQueueArn` – Chaîne UTF-8.

  Un ARN Amazon SQS valide. Par exemple, `arn:aws:sqs:region:account:sqs`.

- `DlqEventQueueArn` – Chaîne UTF-8.

  Un ARN SQS de lettres mortes Amazon valide. Par exemple, `arn:aws:sqs:region:account:deadLetterQueue`.

DeltaCatalogTarget Structure S3

Spécifie une cible qui écrit dans une source de données Delta Lake dans le catalogue de AWS Glue données.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Le nom de la cible de données.
- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 1 chaîne maximum.

  Les nœuds qui constituent des entrées pour la cible de données.
- `PartitionKeys` – Tableau de chaînes UTF-8.

  Indique le partitionnement natif à l'aide d'une séquence de clés.
- `Table` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la table de la base de données dans laquelle écrire les données.
- `Database` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la base de données dans laquelle écrire les données.
- `AdditionalOptions` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, correspondant au Custom string pattern #61.

  Chaque valeur est une chaîne UTF-8, correspondant au Custom string pattern #61.

  Indique des options de connexion supplémentaires pour le connecteur.
- `SchemaChangePolicy` – Un objet CatalogSchemaChangePolicy.

  Une politique qui indique les évolutions de mise à jour pour le crawler.
- `AutoDataQuality` : un objet AutoDataQuality.

  Indique s'il faut activer automatiquement l'évaluation de la qualité des données pour la cible du catalogue S3 Delta. Lorsque ce paramètre est défini sur `true`, les contrôles de qualité des données sont effectués automatiquement pendant l'opération d'écriture.
- `OutputSchemas` : un tableau d'objets GlueSchema.

  Indique le schéma de données de la cible du catalogue S3 Delta.

DeltaDirectTarget Structure S3

Spécifie une cible qui écrit dans une source de données de Delta Lake dans Amazon S3.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Le nom de la cible de données.
- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 1 chaîne maximum.

  Les nœuds qui constituent des entrées pour la cible de données.
- `PartitionKeys` – Tableau de chaînes UTF-8.

  Indique le partitionnement natif à l'aide d'une séquence de clés.
- `Path` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le chemin d'accès Amazon S3 de votre source de données Delta Lake sur laquelle écrire.
- `Compression` – Obligatoire : Chaîne UTF-8 (valeurs valides : `uncompressed="UNCOMPRESSED"` | `snappy="SNAPPY"`).

  Indique la manière dont les données sont comprimées. Ce n'est généralement pas nécessaire si le fichier de données a une extension standard. Les valeurs possibles sont `"gzip"` et `"bzip"`).
- `NumberTargetPartitions` – Chaîne UTF-8.

  Indique le nombre de partitions cible pour distribuer les fichiers de jeux de données Delta Lake sur Amazon S3.
- `Format` – Obligatoire : chaîne UTF-8 (valeurs valides : `json="JSON"` | `csv="CSV"` | `avro="AVRO"` | `orc="ORC"` | `parquet="PARQUET"` | `hudi="HUDI"` | `delta="DELTA"` | `iceberg="ICEBERG"` | `hyper="HYPER"` | `xml="XML"`).

  Définit le format de sortie des données pour la cible.
- `AdditionalOptions` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, correspondant au Custom string pattern #61.

  Chaque valeur est une chaîne UTF-8, correspondant au Custom string pattern #61.

  Indique des options de connexion supplémentaires pour le connecteur.
- `SchemaChangePolicy` – Un objet DirectSchemaChangePolicy.

  Une politique qui indique les évolutions de mise à jour pour le crawler.
- `AutoDataQuality` : un objet AutoDataQuality.

Indique s'il faut activer automatiquement l'évaluation de la qualité des données pour la cible directe S3 Delta. Lorsque ce paramètre est défini sur `true`, les contrôles de qualité des données sont effectués automatiquement pendant l'opération d'écriture.

JdbcTarget structure

Indique les magasins de données JDBC à analyser.

Champs

- `ConnectionName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 2 048 octets.

  Nom de la connexion à utiliser pour se connecter à la cible JDBC.
- `Path` – Chaîne UTF-8.

  Le chemin de la cible JDBC.
- `Exclusions` – Tableau de chaînes UTF-8.

  Liste de modèles glob utilisés à exclure de l'analyse. Pour en savoir plus, consultez Catalog Tables with a Crawler (Tables de catalogues avec un crawler).
- `EnableAdditionalMetadata` – Tableau de chaînes UTF-8.

  Spécifiez une valeur de `RAWTYPES` ou `COMMENTS` pour activer des métadonnées supplémentaires dans les réponses des tables. `RAWTYPES` fournit le type de données de niveau natif. `COMMENTS` fournit des commentaires associés à une colonne ou à une table de la base de données.

  Si vous n'avez pas besoin d'autres métadonnées, laissez le champ vide.

Structure Mongo DBTarget

Indique les magasins de données Amazon DocumentDB ou MongoDB à analyser.

Champs

- `ConnectionName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 2 048 octets.

  Nom de la connexion à utiliser pour se connecter à la cible Amazon DocumentDB ou MongoDB.
- `Path` – Chaîne UTF-8.

Chemin d'accès de la cible Amazon DocumentDB ou MongoDB (base de données/collection).

- `ScanAll` : booléen.

  Indique s'il faut analyser tous les enregistrements ou échantillonner les lignes de la table. L'analyse de tous les enregistrements peut prendre beaucoup de temps lorsque la table n'est pas à haut débit.

  La valeur `true` implique l'analyse de tous les enregistrements, tandis que la valeur `false` implique l'échantillonnage des enregistrements. Si aucune valeur n'est spécifiée, la valeur par défaut est `true`.

Structure de la dynamo DBTarget

Spécifie une table Amazon DynamoDB à analyser.

Champs

- `Path` – Chaîne UTF-8.

  Nom de la table DynamoDB à analyser.

- `scanAll` : booléen.

  Indique s'il faut analyser tous les enregistrements ou échantillonner les lignes de la table. L'analyse de tous les enregistrements peut prendre beaucoup de temps lorsque la table n'est pas à haut débit.

  La valeur `true` implique l'analyse de tous les enregistrements, tandis que la valeur `false` implique l'échantillonnage des enregistrements. Si aucune valeur n'est spécifiée, la valeur par défaut est `true`.

- `scanRate` – Nombre (double).

  Pourcentage d'unités de capacité de lecture configurées à utiliser par le AWS Glue robot d'exploration. Unités de capacité de lecture est un terme défini par DynamoDB et est une valeur numérique qui sert de limiteur de vitesse pour le nombre de lectures pouvant être effectuées sur cette table par seconde.

  Les valeurs valides sont nulles ou une valeur comprise entre 0,1 et 1,5. Une valeur NULL est utilisée lorsque l'utilisateur ne fournit pas de valeur et que la valeur par défaut est 0,5 de l'unité de

capacité de lecture configurée (pour les tables provisionnées) ou 0,25 de l'unité de capacité de lecture maximale configurée (pour les tables utilisant le mode à la demande).

DeltaTarget structure

Spécifie un stockage de données Delta pour analyser un ou plusieurs tableaux Delta.

Champs

- `DeltaTables` – Tableau de chaînes UTF-8.

  Une liste de chemins Amazon S3 vers les tableaux Delta.
- `ConnectionName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 2 048 octets.

  Nom de la connexion à utiliser pour se connecter à la cible Delta.
- `WriteManifest` : booléen.

  Spécifie s'il faut écrire les fichiers manifestes dans le chemin d'accès au tableau Delta.
- `CreateNativeDeltaTable` : booléen.

  Spécifie si le crawler va créer des tables natives pour permettre l'intégration avec les moteurs de requêtes qui prennent directement en charge l'interrogation du journal de transactions Delta.

IcebergTarget structure

Spécifie une source de données Apache Iceberg où les tables Iceberg sont stockées dans Amazon S3.

Champs

- `Paths` – Tableau de chaînes UTF-8.

  Un ou plusieurs Amazon S3 chemins contenant les dossiers de métadonnées Iceberg en tant que`s3://bucket/prefix`.
- `ConnectionName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 2 048 octets.

  Nom de la connexion à utiliser pour se connecter à la cible Iceberg.
- `Exclusions` – Tableau de chaînes UTF-8.

Liste de modèles glob utilisés à exclure de l'analyse. Pour en savoir plus, consultez Catalog Tables with a Crawler (Tables de catalogues avec un crawler).

- `MaximumTraversalDepth` : nombre (entier).

  Profondeur maximale des Amazon S3 chemins que le robot d'exploration peut parcourir pour découvrir le dossier de métadonnées Iceberg dans votre Amazon S3 chemin. Utilisé pour limiter le temps d'exécution du Crawler.

HudiTarget structure

Spécifie une source de données Apache Hudi.

Champs

- `Paths` – Tableau de chaînes UTF-8.

  Tableau de chaînes de Amazon S3 localisation pour Hudi, chacune indiquant le dossier racine dans lequel se trouvent les fichiers de métadonnées d'une table Hudi. Le dossier Hudi peut se trouver dans un dossier enfant du dossier racine.

  Le Crawler examine tous les dossiers situés sous un chemin à la recherche d'un dossier Hudi.

- `ConnectionName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 2 048 octets.

  Nom de la connexion à utiliser pour se connecter à la cible Hudi. Si vos fichiers Hudi sont stockés dans des compartiments nécessitant une autorisation VPC, vous pouvez définir leurs propriétés de connexion ici.

- `Exclusions` – Tableau de chaînes UTF-8.

  Liste de modèles glob utilisés à exclure de l'analyse. Pour en savoir plus, consultez Catalog Tables with a Crawler (Tables de catalogues avec un crawler).

- `MaximumTraversalDepth` : nombre (entier).

  Profondeur maximale des Amazon S3 chemins que le robot d'exploration peut parcourir pour découvrir le dossier de métadonnées Hudi dans votre Amazon S3 chemin. Utilisé pour limiter le temps d'exécution du Crawler.

CatalogTarget structure

Spécifie une AWS Glue Data Catalog cible.

Champs

- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Le nom de la base de données à synchroniser.
- `Tables` – Obligatoire : Tableau de chaînes UTF-8, au moins 1 chaîne.

  Une liste des tables à synchroniser.
- `ConnectionName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 2 048 octets.

  Le nom de la connexion d'une table de catalogue de données basées sur Amazon S3 qui doit être la cible de l'analyse lors de l'utilisation d'un type de connexion `Catalog` associé à un type de connexion `NETWORK`.
- `EventQueueArn` – Chaîne UTF-8.

  Un ARN Amazon SQS valide. Par exemple, `arn:aws:sqs:region:account:sqs`.
- `DlqEventQueueArn` – Chaîne UTF-8.

  Un ARN SQS de lettres mortes Amazon valide. Par exemple, `arn:aws:sqs:region:account:deadLetterQueue`.

CrawlerMetrics structure

Métriques d'un crawler spécifié.

Champs

- `CrawlerName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de l'crawler.
- `TimeLeftSeconds` – Nombre (double), pas plus qu'Aucun.

  Estimation du temps restant pour terminer une analyse en cours d'exécution.

- `StillEstimating` : booléen.

  True si le crawler est toujours en cours d'estimation du temps nécessaire pour terminer cette exécution.

- `LastRuntimeSeconds` – Nombre (double), pas plus qu'Aucun.

  Durée de l'exécution la plus récente de l'crawler, en secondes.

- `MedianRuntimeSeconds` – Nombre (double), pas plus qu'Aucun.

  Durée médiane des exécutions de cet crawler, en secondes.

- `TablesCreated` – Nombre (entier), pas plus qu'Aucun.

  Nombre de tables créées par cet crawler.

- `TablesUpdated` – Nombre (entier), pas plus qu'Aucun.

  Nombre de tables mises à jour par cet crawler.

- `TablesDeleted` – Nombre (entier), pas plus qu'Aucun.

  Nombre de tables supprimées par cet crawler.

CrawlerHistory structure

Contient les informations pour une exécution d'un crawler.

Champs

- `CrawlId` – Chaîne UTF-8.

  Un identifiant UUID pour chaque analyse.

- `State` – Chaîne UTF-8 (valeurs valides : `RUNNING` | `COMPLETED` | `FAILED` | `STOPPED`).

  État de l'analyse.

- `StartTime` – Horodatage.

  Date et heure auxquelles le crawler a démarré.

- `EndTime` – Horodatage.

  Date et heure auxquelles l'analyse s'est achevée.

- `Summary` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Un résumé d'exécution pour l'analyse spécifique dans JSON. Contient les tables de catalogue et les partitions qui ont été ajoutées, mises à jour ou supprimées.

- `ErrorMessage` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Si une erreur s'est produite, le message d'erreur associé à l'analyse.

- `LogGroup` – Chaîne UTF-8, d'une longueur comprise entre 1 et 512 octets, correspondant au Log group string pattern.

  Groupe de journaux associés au crawler.

- `LogStream` – Chaîne UTF-8, d'une longueur comprise entre 1 et 512 octets, correspondant au Log-stream string pattern.

  Flux de journaux associé au crawler.

- `MessagePrefix` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Le préfixe d'un CloudWatch message concernant ce crawl.

- `DPUHour` – Nombre (double), pas plus qu'Aucun.

  Nombre d'unités de traitement de données (DPU) utilisées en heures pour l'analyse.

CrawlsFilter structure

Une liste de champs, de comparateurs et de valeurs que vous pouvez utiliser pour filtrer les exécutions de crawler pour un crawler spécifié.

Champs

- `FieldName` – Chaîne UTF-8 (valeurs valides : `CRAWL_ID` | `STATE` | `START_TIME` | `END_TIME` | `DPU_HOUR`).

  Une clé utilisée pour filtrer les exécutions de Crawler pour un Crawler spécifié. Les valeurs valides pour chacun des noms de champs sont les suivantes :

  - `CRAWL_ID` : une chaîne représentant l'identifiant UUID d'une analyse.

  - `STATE` : une chaîne représentant l'état de l'analyse.
  - `START_TIME` et `END_TIME` : l'horodatage de l'époque en millisecondes.
  - `DPU_HOUR` : le nombre d'heures d'unité de traitement de données (DPU) utilisées pour l'analyse.
- `FilterOperator` – Chaîne UTF-8 (valeurs valides : `GT` | `GE` | `LT` | `LE` | `EQ` | `NE`).

  Un comparateur défini qui agit sur la valeur. Les opérateurs disponibles sont les suivants :
  - `GT` : Supérieur à.
  - `GE` : Supérieur ou égal à.
  - `LT` : Inférieur à.
  - `LE` : Inférieur ou égal à.
  - `EQ` : Égal à.
  - `NE` : Pas égal à.
- `FieldValue` – Chaîne UTF-8.

  La valeur fournie pour la comparaison dans le champ d'analyse.

SchemaChangePolicy structure

Stratégie qui spécifie des comportements de mise à jour et de suppression pour l'crawler.

Champs

- `UpdateBehavior` – Chaîne UTF-8 (valeurs valides : `LOG` | `UPDATE_IN_DATABASE`).

  Comportement de mise à jour lorsque le crawler détecte un schéma modifié.
- `DeleteBehavior` – Chaîne UTF-8 (valeurs valides : `LOG` | `DELETE_FROM_DATABASE` | `DEPRECATE_IN_DATABASE`).

  Comportement de suppression lorsque le crawler détecte un objet supprimé.

LastCrawlInfo structure

Informations d'état et d'erreur sur l'analyse la plus récente.

Champs

- `Status` – Chaîne UTF-8 (valeurs valides : `SUCCEEDED` | `CANCELLED` | `FAILED`).

État de la dernière analyse.

- `ErrorMessage` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Si une erreur s'est produite, informations d'erreur sur la dernière analyse.

- `LogGroup` – Chaîne UTF-8, d'une longueur comprise entre 1 et 512 octets, correspondant au Log group string pattern.

  Groupe de journaux de la dernière analyse.

- `LogStream` – Chaîne UTF-8, d'une longueur comprise entre 1 et 512 octets, correspondant au Log-stream string pattern.

  Flux de journal de la dernière analyse.

- `MessagePrefix` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Préfixe d'un message sur cette analyse.

- `StartTime` – Horodatage.

  Heure à laquelle l'analyse a commencé.

RecrawlPolicy structure

Lorsque vous indexez une source de données Amazon S3 après la première indexation, spécifie s'il faut indexer à nouveau l'ensemble du jeu de données ou uniquement les dossiers ajoutés depuis la dernière exécution du crawler. Pour de plus amples informations, veuillez consulter la rubrique Analyses incrémentielles dans AWS Glue dans le guide du développeur.

Champs

- `RecrawlBehavior` – Chaîne UTF-8 (valeurs valides : `CRAWL_EVERYTHING` | `CRAWL_NEW_FOLDERS_ONLY` | `CRAWL_EVENT_MODE`).

  Spécifie s'il faut analyser à nouveau le jeu de données ou uniquement les dossiers ajoutés depuis la dernière exécution de l'crawler.

  Une valeur de `CRAWL_EVERYTHING` indique que l'ensemble du jeu de données doit être analysé à nouveau.

Une valeur de `CRAWL_NEW_FOLDERS_ONLY` indique que seuls les dossiers ajoutés depuis la dernière exécution du crawler doivent être indexés.

Une valeur de `CRAWL_EVENT_MODE` spécifie uniquement l'analyse des modifications identifiées par les événements Amazon S3.

LineageConfiguration structure

Spécifie les paramètres de configuration de la lignée de données pour l'crawler.

Champs

- `CrawlerLineageSettings` – Chaîne UTF-8 (valeurs valides : `ENABLE` | `DISABLE`).

  Indique si la lignée de données est activée pour le crawler. Les valeurs valides sont :

  - ENABLE : active la lignée des données pour le crawler
  - DISABLE : désactive la lignée de données pour le crawler

LakeFormationConfiguration structure

Spécifie les paramètres de AWS Lake Formation configuration du robot d'exploration.

Champs

- `UseLakeFormationCredentials` : booléen.

  Spécifie s'il faut utiliser les AWS Lake Formation informations d'identification du robot d'exploration au lieu des informations d'identification du rôle IAM.
- `AccountId` – Chaîne UTF-8, d'une longueur maximale de 12 octets.

  Obligatoire pour les analyses de compte croisées. Pour les mêmes analyses de compte que les données cibles, cela peut être laissé nul.

Opérations

- CreateCrawler action (Python : create_crawler)
- DeleteCrawler action (Python : delete_crawler)

- GetCrawler action (Python : get_crawler)
- GetCrawlers action (Python : get_crawlers)
- GetCrawlerMetrics action (Python : get_crawler_metrics)
- UpdateCrawler action (Python : update_crawler)
- StartCrawler action (Python : start_crawler)
- StopCrawler action (Python : stop_crawler)
- BatchGetCrawlers action (Python : batch_get_crawlers)
- ListCrawlers action (Python : list_crawlers)
- ListCrawls action (Python : list_crawls)

CreateCrawler action (Python : create_crawler)

Crée un crawler avec des cibles, un rôle, une configuration, et une éventuelle planification spécifiés. Au moins une cible d'analyse doit être spécifiée dans le champ `s3Targets`, le champ `jdbcTargets` ou le champ `DynamoDBTargets`.

Demande

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du nouveau crawler.

- `Role` – Obligatoire : chaîne UTF-8.

  Rôle IAM ou Amazon Resource Name (ARN) d'un rôle IAM utilisé par le nouveau crawler pour accéder aux ressources client.

- `DatabaseName` – Chaîne UTF-8.

  La AWS Glue base de données dans laquelle les résultats sont écrits, par exemple :`arn:aws:daylight:us-east-1::database/sometable/*`.

- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description du nouvel crawler.

- `Targets` – Obligatoire : un objet CrawlerTargets.

Liste de l'ensemble de cibles à analyser.

- `Schedule` – Chaîne UTF-8.

  Une expression `cron` utilisée pour spécifier la planification (consultez Time-Based Schedules for Jobs and Crawlers (Planifications temporelles pour les tâches et les crawlers)). Par exemple, pour exécuter un élément tous les jours à 12h15 UTC, vous devez spécifier : `cron(15 12 * * ? *)`.

- `Classifiers` – Tableau de chaînes UTF-8.

  Liste des classifieurs personnalisés que l'utilisateur a enregistrés. Par défaut, tous les classificateurs intégrés sont inclus dans une analyse, mais ces classificateurs personnalisés se substituent toujours aux classificateurs par défaut pour une classification donnée.

- `TablePrefix` – Chaîne UTF-8, d'une longueur maximale de 128 octets.

  Préfixe de table utilisé pour les tables catalogue créées.

- `SchemaChangePolicy` – Un objet SchemaChangePolicy.

  Stratégie du comportement de mise à jour et de suppression de l'crawler.

- `RecrawlPolicy` – Un objet RecrawlPolicy.

  Stratégie qui spécifie s'il faut analyser à nouveau le jeu de données entier ou analyser uniquement les dossiers ajoutés depuis la dernière exécution du crawler.

- `LineageConfiguration` – Un objet LineageConfiguration.

  Spécifie les paramètres de configuration de la lignée de données pour le crawler.

- `LakeFormationConfiguration` – Un objet LakeFormationConfiguration.

  Spécifie les paramètres de AWS Lake Formation configuration du robot d'exploration.

- `Configuration` – Chaîne UTF-8.

  Informations sur la configuration du crawler. Cette chaîne JSON avec gestion des versions permet aux utilisateurs de spécifier des aspects du comportement d'un crawler. Pour plus d'informations, consultez Setting Crawler configuration options (Définition d'options de configuration du crawler).

- `CrawlerSecurityConfiguration` – Chaîne UTF-8, d'une longueur maximale de 128 octets.

  Nom de la structure `SecurityConfiguration` qui sera utilisée par ce crawler.

- `Tags` – Tableau de mappage de paires clé-valeur, avec 50 paires au maximum.

Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 256 octets.

Balises à utiliser avec cette demande d'crawler. Vous pouvez utiliser des balises pour limiter l'accès à l'crawler. Pour plus d'informations sur les tags in AWS Glue, voir AWS Tags in AWS Glue dans le guide du développeur.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `InvalidInputException`
- `AlreadyExistsException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`

DeleteCrawler action (Python : delete_crawler)

Supprime un robot d'exploration spécifié du AWS Glue Data Catalog, sauf si l'état du robot est. `RUNNING`

Demande

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de l'crawler à supprimer.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `EntityNotFoundException`
- `CrawlerRunningException`
- `SchedulerTransitioningException`
- `OperationTimeoutException`

GetCrawler action (Python : get_crawler)

Récupère des métadonnées pour un crawler spécifié.

Demande

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de l'crawler pour lequel récupérer les métadonnées.

Réponse

- `Crawler` – Un objet crawler.

  Métadonnées pour l'crawler spécifié.

Erreurs

- `EntityNotFoundException`
- `OperationTimeoutException`

GetCrawlers action (Python : get_crawlers)

Récupère les métadonnées pour tous les crawlers définis dans le compte client.

Demande

- `MaxResults` – Nombre (entier), compris entre 1 et 1 000.

  Nombre d'crawlers à renvoyer à chaque appel.

- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, s'il s'agit d'une requête de continuation.

Réponse

- `Crawlers` : un tableau d'objets crawler.

  Liste des métadonnées de l'crawler.
- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, si la liste renvoyée n'a pas atteint la fin de ceux définis dans ce compte client.

Erreurs

- `OperationTimeoutException`

GetCrawlerMetrics action (Python : get_crawler_metrics)

Récupère les métriques sur les crawlers spécifiés.

Demande

- `CrawlerNameList` – Tableau de chaînes UTF-8, avec 100 chaînes maximum.

  Liste des noms des crawlers sur lesquels récupérer les métriques.
- `MaxResults` – Nombre (entier), compris entre 1 et 1 000.

  La taille maximale d'une liste à renvoyer.
- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, s'il s'agit d'un appel de continuation.

Réponse

- `CrawlerMetricsList` : un tableau d'objets CrawlerMetrics.

  Liste des métriques pour l'crawler spécifié.

- `NextToken` – Chaîne UTF-8.

  Jeton continuation, si la liste renvoyée ne contient pas la dernière métrique disponible.

Erreurs

- `OperationTimeoutException`

UpdateCrawler action (Python : update_crawler)

Met à jour un crawler. Si un crawler est en cours d'exécution, vous devez l'arrêter à l'aide de `StopCrawler` avant de le mettre à jour.

Demande

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du nouvel crawler.

- `Role` – Chaîne UTF-8.

  Rôle IAM ou Amazon Resource Name (ARN) d'un rôle IAM qui est utilisé par le nouvel crawler pour accéder aux ressources client.

- `DatabaseName` – Chaîne UTF-8.

  La AWS Glue base de données dans laquelle les résultats sont stockés, par exemple :`arn:aws:daylight:us-east-1::database/sometable/*`.

- `Description` – Chaîne UTF-8, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description du nouvel crawler.

- `Targets` – Un objet CrawlerTargets.

  Liste des cibles à analyser.

- `Schedule` – Chaîne UTF-8.

Une expression `cron` utilisée pour spécifier la planification (consultez Time-Based Schedules for Jobs and Crawlers (Planifications temporelles pour les tâches et les crawlers)). Par exemple, pour exécuter un élément tous les jours à 12h15 UTC, vous devez spécifier : `cron(15 12 * * ? *)`.

- `Classifiers` – Tableau de chaînes UTF-8.

  Liste des classifieurs personnalisés que l'utilisateur a enregistrés. Par défaut, tous les classificateurs intégrés sont inclus dans une analyse, mais ces classificateurs personnalisés se substituent toujours aux classificateurs par défaut pour une classification donnée.

- `TablePrefix` – Chaîne UTF-8, d'une longueur maximale de 128 octets.

  Préfixe de table utilisé pour les tables catalogue créées.

- `SchemaChangePolicy` – Un objet SchemaChangePolicy.

  Stratégie du comportement de mise à jour et de suppression de l'crawler.

- `RecrawlPolicy` – Un objet RecrawlPolicy.

  Stratégie qui spécifie s'il faut analyser à nouveau le jeu de données entier ou analyser uniquement les dossiers ajoutés depuis la dernière exécution du crawler.

- `LineageConfiguration` – Un objet LineageConfiguration.

  Spécifie les paramètres de configuration de la lignée de données pour le crawler.

- `LakeFormationConfiguration` – Un objet LakeFormationConfiguration.

  Spécifie les paramètres de AWS Lake Formation configuration du robot d'exploration.

- `Configuration` – Chaîne UTF-8.

  Informations sur la configuration du crawler. Cette chaîne JSON avec gestion des versions permet aux utilisateurs de spécifier des aspects du comportement d'un crawler. Pour plus d'informations, consultez Setting Crawler configuration options (Définition d'options de configuration du crawler).

- `CrawlerSecurityConfiguration` – Chaîne UTF-8, d'une longueur maximale de 128 octets.

  Nom de la structure `SecurityConfiguration` qui sera utilisée par cet crawler.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `InvalidInputException`
- `VersionMismatchException`
- `EntityNotFoundException`
- `CrawlerRunningException`
- `OperationTimeoutException`

StartCrawler action (Python : start_crawler)

Démarre une analyse à l'aide de l'crawler, indépendamment de ce qui est prévu. Si le robot d'exploration est déjà en cours d'exécution, renvoie un CrawlerRunningException.

Demande

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de l'crawler à démarrer.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `EntityNotFoundException`
- `CrawlerRunningException`
- `OperationTimeoutException`

StopCrawler action (Python : stop_crawler)

Si l'crawler spécifié est en cours d'exécution, arrête l'analyse.

Demande

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de l'crawler à arrêter.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `EntityNotFoundException`
- `CrawlerNotRunningException`
- `CrawlerStoppingException`
- `OperationTimeoutException`

BatchGetCrawlers action (Python : batch_get_crawlers)

Renvoie la liste des métadonnées de ressource pour une liste donnée de noms d'crawler. Après avoir appelé l'opération `ListCrawlers`, vous pouvez appeler cette opération pour accéder aux données sur lesquelles des autorisations vous ont été octroyées. Cette opération prend en charge toutes les autorisations IAM, y compris les conditions d'autorisation qui utilisent des balises.

Demande

- `CrawlerNames` – Obligatoire : Tableau de chaînes UTF-8, avec 100 chaînes maximum.

  Liste des noms d'crawler, qui peuvent être les noms renvoyés à partir de l'opération `ListCrawlers`.

Réponse

- `Crawlers` : un tableau d'objets crawler.

  Liste des définitions d'crawler.
- `CrawlersNotFound` – Tableau de chaînes UTF-8, avec 100 chaînes maximum.

Liste de noms d'crawler qui n'ont pas été trouvés.

Erreurs

- `InvalidInputException`
- `OperationTimeoutException`

ListCrawlers action (Python : list_crawlers)

Récupère les noms de toutes les ressources du robot d'exploration de ce AWS compte, ou des ressources portant le tag spécifié. Cette opération vous permet de voir quelles ressources sont disponibles dans votre compte, et leurs noms.

Cette opération accepte le champ `Tags` facultatif que vous pouvez utiliser comme filtre sur la réponse, afin que les ressources balisées puissent être récupérées en tant que groupe. Si vous choisissez d'utiliser le filtrage des balises, seules les ressources avec la balise sont récupérées.

Demande

- `MaxResults` – Nombre (entier), compris entre 1 et 1 000.

  La taille maximale d'une liste à renvoyer.
- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, s'il s'agit d'une requête de continuation.
- `Tags` – Tableau de mappage de paires clé-valeur, avec 50 paires au maximum.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 256 octets.

  Spécifie de renvoyer uniquement les ressources balisées.

Réponse

- `CrawlerNames` – Tableau de chaînes UTF-8, avec 100 chaînes maximum.

  Noms de tous les crawlers dans le compte ou des crawlers avec les balises spécifiées.

- `NextToken` – Chaîne UTF-8.

  Jeton continuation, si la liste renvoyée ne contient pas la dernière métrique disponible.

Erreurs

- `OperationTimeoutException`

ListCrawls action (Python : list_crawls)

Renvoie toutes les analyse d'un Crawler spécifié. Renvoie uniquement les analyses qui ont eu lieu depuis la date de lancement de la fonction d'historique du Crawler, et ne retient que jusqu'à 12 mois d'analyse. Les anciennes analyses ne seront pas renvoyées.

Vous pouvez utiliser cette API pour :

- Récupère toutes les analyse d'un Crawler spécifié.
- Récupère toutes les analyse d'un Crawler spécifié dans un nombre limité.
- Récupère toutes les analyse d'un Crawler spécifié dans une plage de temps spécifique.
- Récupère toutes les analyse d'un Crawler spécifié avec un état, un ID d'analyse ou une valeur d'heure DPU particuliers.

Demande

- `CrawlerName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du Crawler dont vous voulez récupérer les exécutions.
- `MaxResults` – Nombre (entier), compris entre 1 et 1 000.

  Nombre maximal de résultats à renvoyer. La valeur par défaut est 20 et la valeur maximale est 100.
- `Filters` : un tableau d'objets CrawlsFilter.

  Filtre les analyse en fonction de critères que vous spécifiez dans une liste objets `CrawlsFilter`.
- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, s'il s'agit d'un appel de continuation.

Réponse

- `Crawls` : un tableau d'objets CrawlerHistory.

  Une liste d'objets `CrawlerHistory` représentant les cycles d'analyse qui répondent à vos critères.

- `NextToken` – Chaîne UTF-8.

  Jeton de continuation pour la pagination de la liste des jetons renvoyés, renvoyé si le segment actuel de la liste n'est pas le dernier.

Erreurs

- `EntityNotFoundException`
- `OperationTimeoutException`
- `InvalidInputException`

API de statistiques de colonne

L'API de statistiques de colonnes décrit comment renvoyer AWS Glue APIs des statistiques sur les colonnes d'une table.

Types de données

- ColumnStatisticsTaskRun structure
- ColumnStatisticsTaskSettings structure
- ExecutionAttempt structure

ColumnStatisticsTaskRun structure

L'objet qui affiche les détails de l'exécution des statistiques de colonne.

Champs

- `CustomerId` – Chaîne UTF-8, d'une longueur maximale de 12 octets.

  L'identifiant AWS du compte.

- `ColumnStatisticsTaskRunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  L'identifiant d'exécution de la tâche de statistiques de colonne particulière.

- `DatabaseName` – Chaîne UTF-8.

  La base de données où réside la table.

- `TableName` – Chaîne UTF-8.

  Le nom de la table pour laquelle les statistiques de colonne sont générées.

- `ColumnNameList` – Tableau de chaînes UTF-8.

  Une liste des noms de colonnes. Si aucun nom n'est fourni, tous les noms de colonnes de la table seront utilisés par défaut.

- `CatalogID` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données où réside la table. Si aucun identifiant n'est fourni, l'identifiant du AWS compte est utilisé par défaut.

- `Role` – Chaîne UTF-8.

  Le rôle IAM que le service assume pour générer des statistiques.

- `SampleSize` – Nombre (double), 100 au maximum.

  Pourcentage de lignes utilisées pour générer des statistiques. Si aucun nom n'est fourni, la table entière sera utilisée pour générer des statistiques.

- `SecurityConfiguration` – Chaîne UTF-8, d'une longueur maximale de 128 octets.

  Nom de la configuration de sécurité utilisée pour chiffrer les CloudWatch journaux pour la tâche de statistiques de colonne exécutée.

- `NumberOfWorkers` – Nombre (entier), au moins égal à 1.

  Le nombre d'employés utilisés pour générer les statistiques de colonne. La tâche est préconfigurée pour effectuer une mise à l'échelle automatique jusqu'à 25 instances.

- `WorkerType` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Type de travailleurs utilisés pour générer des statistiques. La valeur par défaut est `g.1x`.

- `ComputationType` – Chaîne UTF-8 (valeurs valides : `FULL` | `INCREMENTAL`).

  Type de calcul de statistiques de colonne.
- `Status` – Chaîne UTF-8 (valeurs valides : `STARTING` | `RUNNING` | `SUCCEEDED` | `FAILED` | `STOPPED`).

  L'état d'exécution de la tâche.
- `CreationTime` – Horodatage.

  Heure à laquelle cette tâche a été créée.
- `LastUpdated` – Horodatage.

  Dernier moment où cette tâche a été modifiée.
- `StartTime` – Horodatage.

  L'heure de début de la tâche.
- `EndTime` – Horodatage.

  L'heure de fin de la tâche.
- `ErrorMessage` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Le message d'erreur pour la tâche.
- `DPUSeconds` – Nombre (double), pas plus qu'Aucun.

  L'utilisation de la DPU calculée en secondes pour tous les travailleurs mis à l'échelle automatique.

ColumnStatisticsTaskSettings structure

Paramètres d'une tâche de statistiques de colonne.

Champs

- `DatabaseName` – Chaîne UTF-8.

  Le nom de la base de données où réside la table.
- `TableName` – Chaîne UTF-8.

  Nom de la table pour laquelle les statistiques de colonne sont générées.

- `Schedule` – Un objet Planificateur.

  Calendrier d'exécution des statistiques de colonne, spécifié dans la syntaxe CRON.

- `ColumnNameList` – Tableau de chaînes UTF-8.

  Liste des noms de colonnes pour lesquelles les statistiques doivent être exécutées.

- `CatalogID` – Chaîne d'ID de catalogue, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données dans lequel réside la base de données.

- `Role` – Chaîne UTF-8.

  Rôle utilisé pour l'exécution des statistiques de colonne.

- `SampleSize` – Nombre (double), 100 au maximum.

  Pourcentage de données à échantillonner.

- `SecurityConfiguration` – Chaîne UTF-8, d'une longueur maximale de 128 octets.

  Nom de la configuration de sécurité utilisée pour chiffrer les CloudWatch journaux.

- `ScheduleType` – Chaîne UTF-8 (valeurs valides : `CRON` | `AUTO`).

  Type de planification pour une tâche de statistiques de colonne. Les valeurs possibles peuvent être `CRON` ou `AUTO`.

- `SettingSource` – Chaîne UTF-8 (valeurs valides : `CATALOG` | `TABLE`).

  Source de définition de la tâche de statistiques de colonne. Les valeurs possibles peuvent être `CATALOG` ou `TABLE`.

- `LastExecutionAttempt` : un objet ExecutionAttempt.

  Dernière `ExecutionAttempt` pour l'exécution de la tâche de statistiques de colonne.

ExecutionAttempt structure

Tentative d'exécution d'une tâche de statistiques de colonne.

Champs

- `Status` – Chaîne UTF-8 (valeurs valides : `FAILED` | `STARTED`).

Statut d'exécution de la dernière tâche de statistiques de colonne.

- `ColumnStatisticsTaskRunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant d'exécution de tâche de la dernière exécution de tâche de statistiques de colonne.

- `ExecutionTimestamp` – Horodatage.

  Horodatage de la dernière exécution de tâche de statistiques de colonne.

- `ErrorMessage` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Message d'erreur associé à la dernière exécution de tâche de statistiques de colonne.

Opérations

- StartColumnStatisticsTaskRun action (Python : start_column_statistics_task_run)
- GetColumnStatisticsTaskRun action (Python : get_column_statistics_task_run)
- GetColumnStatisticsTaskRuns action (Python : get_column_statistics_task_runs)
- ListColumnStatisticsTaskRuns action (Python : list_column_statistics_task_runs)
- StopColumnStatisticsTaskRun action (Python : stop_column_statistics_task_run)
- CreateColumnStatisticsTaskSettings action (Python : create_column_statistics_task_settings)
- UpdateColumnStatisticsTaskSettings action (Python : update_column_statistics_task_settings)
- GetColumnStatisticsTaskSettings action (Python : get_column_statistics_task_settings)
- DeleteColumnStatisticsTaskSettings action (Python : delete_column_statistics_task_settings)
- StartColumnStatisticsTaskRunSchedule action (Python : start_column_statistics_task_run_schedule)
- StopColumnStatisticsTaskRunSchedule action (Python : stop_column_statistics_task_run_schedule)

StartColumnStatisticsTaskRun action (Python : start_column_statistics_task_run)

Démarre une exécution de tâche de statistiques de colonne pour une table et des colonnes spécifiées.

Demande

- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Le nom de la base de données où réside la table.

- `TableName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Le nom de la table pour générer des statistiques.

- `ColumnNameList` – Tableau de chaînes UTF-8.

  Une liste des noms de colonnes pour générer des statistiques. Si aucun nom n'est fourni, tous les noms de colonnes de la table seront utilisés par défaut.

- `Role` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Le rôle IAM que le service assume pour générer des statistiques.

- `SampleSize` – Nombre (double), 100 au maximum.

  Pourcentage de lignes utilisées pour générer des statistiques. Si aucun nom n'est fourni, la table entière sera utilisée pour générer des statistiques.

- `CatalogID` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  L'ID du catalogue de données où réside la table. Si aucun nom n'est fourni, l'ID de compte AWS est utilisé par défaut.

- `SecurityConfiguration` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la configuration de sécurité utilisée pour chiffrer les CloudWatch journaux pour la tâche de statistiques de colonne exécutée.

Réponse

- `ColumnStatisticsTaskRunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

L'identifiant d'exécution de la tâche de statistiques de colonne.

Erreurs

- `AccessDeniedException`
- `EntityNotFoundException`
- `ColumnStatisticsTaskRunningException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`
- `InvalidInputException`

GetColumnStatisticsTaskRun action (Python : get_column_statistics_task_run)

Obtenez le code associé metadata/information à une exécution de tâche, en fonction d'un ID d'exécution de tâche.

Demande

- `ColumnStatisticsTaskRunId` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  L'identifiant d'exécution de la tâche de statistiques de colonne particulière.

Réponse

- `ColumnStatisticsTaskRun` – Un objet ColumnStatisticsTaskRun.

  Un objet `ColumnStatisticsTaskRun` représentant les détails de l'exécution des statistiques de colonne.

Erreurs

- `EntityNotFoundException`
- `OperationTimeoutException`
- `InvalidInputException`

GetColumnStatisticsTaskRuns action (Python : get_column_statistics_task_runs)

Récupère des informations sur toutes les exécutions associées à la table spécifiée.

Demande

- `DatabaseName` – Obligatoire : chaîne UTF-8.

  Le nom de la base de données où réside la table.

- `TableName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table.

- `MaxResults` – Nombre (entier), compris entre 1 et 1 000.

  Taille maximale de la réponse.

- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, s'il s'agit d'un appel de continuation.

Réponse

- `ColumnStatisticsTaskRuns` : un tableau d'objets ColumnStatisticsTaskRun.

  Une liste des exécutions de tâches de statistiques de colonne.

- `NextToken` – Chaîne UTF-8.

  Un jeton de continuation, si toutes les exécutions de tâches n'ont pas encore été renvoyées.

Erreurs

- `OperationTimeoutException`

ListColumnStatisticsTaskRuns action (Python : list_column_statistics_task_runs)

Répertoriez toutes les exécutions de tâches pour un compte spécifique.

Demande

- `MaxResults` – Nombre (entier), compris entre 1 et 1 000.

  Taille maximale de la réponse.
- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, s'il s'agit d'un appel de continuation.

Réponse

- `ColumnStatisticsTaskRunIds` – Tableau de chaînes UTF-8, avec 100 chaînes maximum.

  Liste des tâches de statistiques de colonnes exécutées IDs.
- `NextToken` – Chaîne UTF-8.

  Un jeton de continuation, si toutes les tâches exécutées n' IDs ont pas encore été renvoyées.

Erreurs

- `OperationTimeoutException`

StopColumnStatisticsTaskRun action (Python : stop_column_statistics_task_run)

Arrête l'exécution d'une tâche pour la table spécifiée.

Demande

- `DatabaseName` – Obligatoire : chaîne UTF-8.

  Le nom de la base de données où réside la table.
- `TableName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `EntityNotFoundException`
- `ColumnStatisticsTaskNotRunningException`
- `ColumnStatisticsTaskStoppingException`
- `OperationTimeoutException`

CreateColumnStatisticsTaskSettings action (Python : create_column_statistics_task_settings)

Crée les paramètres d'une tâche de statistiques de colonne.

Demande

- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Le nom de la base de données où réside la table.
- `TableName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table pour laquelle les statistiques de colonne sont générées.
- `Role` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Rôle utilisé pour l'exécution des statistiques de colonne.
- `Schedule` – Chaîne UTF-8.

  Calendrier d'exécution des statistiques de colonne, spécifié dans la syntaxe CRON.
- `ColumnNameList` – Tableau de chaînes UTF-8.

  Liste des noms de colonnes pour lesquelles les statistiques doivent être exécutées.
- `SampleSize` – Nombre (double), 100 au maximum.

  Pourcentage de données à échantillonner.
- `CatalogID` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

ID du catalogue de données dans lequel réside la base de données.

- `SecurityConfiguration` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la configuration de sécurité utilisée pour chiffrer les CloudWatch journaux.

- `Tags` – Tableau de mappage de paires clé-valeur, avec 50 paires au maximum.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 256 octets.

  Carte de balises.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `AlreadyExistsException`
- `AccessDeniedException`
- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`
- `ColumnStatisticsTaskRunningException`

UpdateColumnStatisticsTaskSettings action (Python : update_column_statistics_task_settings)

Met à jour les paramètres d'une tâche de statistiques de colonne.

Demande

- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

Le nom de la base de données où réside la table.

- `TableName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table pour laquelle les statistiques de colonne sont générées.

- `Role` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Rôle utilisé pour l'exécution des statistiques de colonne.

- `Schedule` – Chaîne UTF-8.

  Calendrier d'exécution des statistiques de colonne, spécifié dans la syntaxe CRON.

- `ColumnNameList` – Tableau de chaînes UTF-8.

  Liste des noms de colonnes pour lesquelles les statistiques doivent être exécutées.

- `SampleSize` – Nombre (double), 100 au maximum.

  Pourcentage de données à échantillonner.

- `CatalogID` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du catalogue de données dans lequel réside la base de données.

- `SecurityConfiguration` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la configuration de sécurité utilisée pour chiffrer les CloudWatch journaux.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `AccessDeniedException`
- `EntityNotFoundException`
- `InvalidInputException`

- `VersionMismatchException`
- `OperationTimeoutException`

GetColumnStatisticsTaskSettings action (Python : get_column_statistics_task_settings)

Obtient les paramètres d'une tâche de statistiques de colonne.

Demande

- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Le nom de la base de données où réside la table.
- `TableName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table pour laquelle les statistiques de colonne sont extraites.

Réponse

- `ColumnStatisticsTaskSettings` – Un objet ColumnStatisticsTaskSettings.

  Objet `ColumnStatisticsTaskSettings` représentant les paramètres de la tâche de statistiques de colonne.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`

DeleteColumnStatisticsTaskSettings action (Python : delete_column_statistics_task_settings)

Supprime les paramètres d'une tâche de statistiques de colonne.

Demande

- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Le nom de la base de données où réside la table.
- `TableName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table pour laquelle les statistiques de colonne sont supprimées.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`

StartColumnStatisticsTaskRunSchedule action (Python : start_column_statistics_task_run_schedule)

Démarre la planification d'exécution d'une tâche de statistiques de colonne.

Demande

- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Le nom de la base de données où réside la table.
- `TableName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table pour laquelle un calendrier d'exécution des tâches de statistiques de colonne doit être lancé.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `AccessDeniedException`
- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`

StopColumnStatisticsTaskRunSchedule action (Python : stop_column_statistics_task_run_schedule)

Arrête un calendrier d'exécution d'une tâche de statistiques de colonne.

Demande

- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Le nom de la base de données où réside la table.
- `TableName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table pour laquelle un calendrier d'exécution des tâches de statistiques de colonne doit être arrêté.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`

Exceptions

- ColumnStatisticsTaskRunningException structure
- ColumnStatisticsTaskNotRunningException structure
- ColumnStatisticsTaskStoppingException structure
- ColumnStatisticsTaskAutoConcurrencyLimitException structure
- InvalidCatalogSettingException structure

ColumnStatisticsTaskRunningException structure

Exception renvoyée lorsque vous essayez de démarrer une autre tâche lors de l'exécution d'une tâche de génération de statistiques de colonne.

Champs

- `Message` – Chaîne UTF-8.

  Un message décrivant le problème.

ColumnStatisticsTaskNotRunningException structure

Exception renvoyée lorsque vous essayez d'arrêter l'exécution d'une tâche alors qu'aucune tâche n'est en cours d'exécution.

Champs

- `Message` – Chaîne UTF-8.

  Un message décrivant le problème.

ColumnStatisticsTaskStoppingException structure

Exception renvoyée lorsque vous essayez d'arrêter l'exécution d'une tâche.

Champs

- `Message` – Chaîne UTF-8.

  Un message décrivant le problème.

ColumnStatisticsTaskAutoConcurrencyLimitException structure

Une exception est déclenchée lorsque vous avez déjà atteint la limite des tâches de statistiques automatiques simultanées.

Champs

- `Message` – Chaîne UTF-8.

  Un message décrivant le problème.

InvalidCatalogSettingException structure

Une exception déclenchée en cas de problème avec les paramètres du catalogue.

Champs

- `Message` – Chaîne UTF-8.

  Un message décrivant le problème.

API du planificateur du crawler

L'API du planificateur de robots décrit les types de données des robots d' AWS Glue exploration, ainsi que l'API permettant de créer, de supprimer, de mettre à jour et de répertorier les robots d'exploration.

Types de données

- Structure du planificateur

Structure du planificateur

Objet de planification utilisant une instruction `cron` pour planifier un événement.

Champs

- `ScheduleExpression` – Chaîne UTF-8.

Une expression `cron` utilisée pour spécifier la planification (consultez Time-Based Schedules for Jobs and Crawlers (Planifications temporelles pour les tâches et les crawlers)). Par exemple, pour exécuter un élément tous les jours à 12h15 UTC, vous devez spécifier : `cron(15 12 * * ? *)`.

- `State` – Chaîne UTF-8 (valeurs valides : `SCHEDULED` | `NOT_SCHEDULED` | `TRANSITIONING`).

  État de la planification.

Opérations

- UpdateCrawlerSchedule action (Python : update_crawler_schedule)
- StartCrawlerSchedule action (Python : start_crawler_schedule)
- StopCrawlerSchedule action (Python : stop_crawler_schedule)

UpdateCrawlerSchedule action (Python : update_crawler_schedule)

Met à jour la planification d'un crawler à l'aide d'une expression `cron`.

Demande

- `CrawlerName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de l'crawler dont la planification doit être mise à jour.

- `Schedule` – Chaîne UTF-8.

  L'expression `cron` mise à jour utilisée pour spécifier la planification (voir Planifications temporelles pour les tâches et les crawlers). Par exemple, pour exécuter un élément tous les jours à 12h15 UTC, vous devez spécifier : `cron(15 12 * * ? *)`.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `EntityNotFoundException`

- `InvalidInputException`
- `VersionMismatchException`
- `SchedulerTransitioningException`
- `OperationTimeoutException`

StartCrawlerSchedule action (Python : start_crawler_schedule)

Modifie l'état de la planification pour l'crawler spécifié par `SCHEDULED`, sauf si l'crawler est déjà en cours d'exécution ou si l'état de la planification est déjà `SCHEDULED`.

Demande

- `CrawlerName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de l'crawler à planifier.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `EntityNotFoundException`
- `SchedulerRunningException`
- `SchedulerTransitioningException`
- `NoScheduleException`
- `OperationTimeoutException`

StopCrawlerSchedule action (Python : stop_crawler_schedule)

Définit l'état de la planification de l'crawler spécifié sur `NOT_SCHEDULED`, mais n'arrête pas l'crawler s'il est déjà en cours d'exécution.

Demande

- `CrawlerName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de l'crawler dont l'état de la planification doit être défini.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `EntityNotFoundException`
- `SchedulerNotRunningException`
- `SchedulerTransitioningException`
- `OperationTimeoutException`

Génération automatique d'API de scripts ETL

L'API de génération de script ETL décrit les types de données et l'API pour générer les scripts ETL dans AWS Glue.

Types de données

- CodeGenNode structure
- CodeGenNodeArg structure
- CodeGenEdge structure
- Structure de l'emplacement
- CatalogEntry structure
- MappingEntry structure

CodeGenNode structure

Représente un nœud dans un graphe orienté acyclique (DAG)

Champs

- `Id` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Identifier string pattern.

  Identifiant de nœud qui est unique dans le graphe du nœud.

- `NodeType` – Obligatoire : chaîne UTF-8.

  Type de nœud.

- `Args` – Obligatoire : Un tableau d'objets CodeGenNodeArg, 50 structures maximum.

  Propriétés du nœud, sous forme de paires nom-valeur.

- `LineNumber` : nombre (entier).

  Numéro de ligne du nœud.

CodeGenNodeArg structure

Argument ou propriété d'un nœud.

Champs

- `Name` – Obligatoire : chaîne UTF-8.

  Nom de l'argument ou de la propriété.

- `Value` – Obligatoire : chaîne UTF-8.

  Valeur de l'argument ou de la propriété.

- `Param` : booléen.

  True si la valeur est utilisée en tant que paramètre.

CodeGenEdge structure

Représente un bord directionnel dans un graphe orienté acyclique (DAG).

Champs

- `Source` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Identifier string pattern.

  ID du nœud où commence le bord.

- `Target` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Identifier string pattern.

  ID du nœud où finit le bord.

- `TargetParameter` – Chaîne UTF-8.

  Cible du bord.

Structure de l'emplacement

Emplacement des ressources.

Champs

- `Jdbc` – Un tableau d'objets CodeGenNodeArg, 50 structures maximum.

  Emplacement JDBC.

- `S3` – Un tableau d'objets CodeGenNodeArg, 50 structures maximum.

  Emplacement Amazon Simple Storage Service (Amazon S3).

- `DynamoDB` – Un tableau d'objets CodeGenNodeArg, 50 structures maximum.

  Emplacement de table Amazon DynamoDB.

CatalogEntry structure

Spécifie une définition de table dans AWS Glue Data Catalog.

Champs

- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

Base de données dans laquelle réside les métadonnées de la table.

- `TableName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table en question.

MappingEntry structure

Définit un mappage.

Champs

- `SourceTable` – Chaîne UTF-8.

  Nom de la table source.
- `SourcePath` – Chaîne UTF-8.

  Le chemin d'accès source .
- `SourceType` – Chaîne UTF-8.

  Type source.
- `TargetTable` – Chaîne UTF-8.

  Table cible.
- `TargetPath` – Chaîne UTF-8.

  Chemin cible.
- `TargetType` – Chaîne UTF-8.

  Type de cible.

Opérations

- CreateScript action (Python : créer_script)
- GetDataflowGraph action (Python : get_dataflow_graph)
- GetMapping action (Python : get_mapping)
- GetPlan action (Python : get_plan)

CreateScript action (Python : créer_script)

Transforme un graphe orienté acyclique (DAG) en code.

Demande

- `DagNodes` : un tableau d'objets CodeGenNode.

  Liste de nœuds dans le DAG.
- `DagEdges` : un tableau d'objets CodeGenEdge.

  Liste de bords dans le DAG.
- `Language` – Chaîne UTF-8 (valeurs valides : `PYTHON` | `SCALA`).

  Langage de programmation du code obtenu à partir du DAG.

Réponse

- `PythonScript` – Chaîne UTF-8.

  Script Python généré à partir du DAG.
- `ScalaCode` – Chaîne UTF-8.

  Code Scala généré à partir du DAG.

Erreurs

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

GetDataflowGraph action (Python : get_dataflow_graph)

Transforme un script Python en graphe orienté acyclique (DAG).

Demande

- `PythonScript` – Chaîne UTF-8.

Script Python à transformer.

Réponse

- `DagNodes` : un tableau d'objets CodeGenNode.

  Liste de nœuds dans le DAG obtenu.
- `DagEdges` : un tableau d'objets CodeGenEdge.

  Liste de bords dans le DAG obtenu.

Erreurs

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

GetMapping action (Python : get_mapping)

Crée des mappages.

Demande

- `Source` – Obligatoire : un objet CatalogEntry.

  Spécifie la table source.
- `Sinks` : un tableau d'objets CatalogEntry.

  Liste des tables cible.
- `Location` – Un objet Location.

  Paramètres pour le mappage.

Réponse

- `Mapping` – Obligatoire : un tableau d'objets MappingEntry.

Liste des mappages pour les cibles spécifiées.

Erreurs

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `EntityNotFoundException`

GetPlan action (Python : get_plan)

Obtient le code pour effectuer un mappage spécifié.

Demande

- `Mapping` – Obligatoire : un tableau d'objets MappingEntry.

  Liste des mappages d'une table source vers des tables cibles.
- `Source` – Obligatoire : un objet CatalogEntry.

  Table source.
- `Sinks` – Un tableau CatalogEntry d'objets.

  Tables cibles.
- `Location` – Un objet Location.

  Paramètres pour le mappage.
- `Language` – Chaîne UTF-8 (valeurs valides : `PYTHON` | `SCALA`).

  Langage de programmation du code pour effectuer le mappage.
- `AdditionalPlanOptionsMap` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8.

  Chaque valeur est une chaîne UTF-8.

  Carte pour contenir les paramètres clé-valeur facultatifs supplémentaires.

Actuellement, ces paires clé-valeur sont prises en charge :

- `inferSchema`— Spécifie si le script par défaut généré par une AWS Glue tâche doit `inferSchema` être défini sur true ou false. Par exemple, pour définir `inferSchema` sur true, transmettez la paire de clé-valeur suivante :

  ```
  --additional-plan-options-map '{"inferSchema":"true"}'
  ```

Réponse

- `PythonScript` – Chaîne UTF-8.

  Script Python pour effectuer le mappage.
- `ScalaCode` – Chaîne UTF-8.

  Code Scala pour effectuer le mappage.

Erreurs

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

API Visual Job

L'API Visual job vous permet de créer des jobs d'intégration de données en utilisant l' AWS Glue API d'un objet JSON qui représente la configuration visuelle d'un AWS Glue job.

Une liste est fournie à une API de `CodeGenConfigurationNodes` création ou de mise à jour de tâche afin d'enregistrer un DAG dans AWS Glue Studio pour la tâche créée et de générer le code associé.

Types de données

- CodeGenConfigurationNode structure
- JDBCConnectorStructure des options
- StreamingDataPreviewOptions structure

- AthenaConnectorSource structure
- JDBCConnectorStructure de la source
- SparkConnectorSource structure
- CatalogSource structure
- Structure SQLCatalog de ma source
- Structure de la SQLCatalog source Postgre
- Structure d'Oracle SQLCatalog Source
- SQLServerCatalogSource Structure de Microsoft
- CatalogKinesisSource structure
- DirectKinesisSource structure
- KinesisStreamingSourceOptions structure
- CatalogKafkaSource structure
- DirectKafkaSource structure
- KafkaStreamingSourceOptions structure
- RedshiftSource structure
- AmazonRedshiftSource structure
- AmazonRedshiftNodeData structure
- AmazonRedshiftAdvancedOption structure
- Structure de l'option
- CatalogSource Structure S3
- SourceAdditionalOptions Structure S3
- CsvSource Structure S3
- JDBCSource Structure directe
- DirectSourceAdditionalOptions Structure S3
- JsonSource Structure S3
- ParquetSource Structure S3
- DeltaSource Structure S3
- CatalogDeltaSource Structure S3
- CatalogDeltaSource structure
- HudiSource Structure S3

- CatalogHudiSource Structure S3
- ExcelSource Structure S3
- CatalogHudiSource structure
- Structure de Dynamo DBCatalog Source
- RelationalCatalogSource structure
- JDBCConnectorStructure de la cible
- SparkConnectorTarget structure
- BasicCatalogTarget structure
- Structure de My SQLCatalog Target
- Structure de SQLCatalog PostgreTarget
- Structure d'Oracle SQLCatalog Target
- SQLServerCatalogTarget Structure de Microsoft
- RedshiftTarget structure
- AmazonRedshiftTarget structure
- UpsertRedshiftTargetOptions structure
- CatalogTarget Structure S3
- GlueParquetTarget Structure S3
- CatalogSchemaChangePolicy structure
- DirectTarget Structure S3
- HudiCatalogTarget Structure S3
- HudiDirectTarget Structure S3
- DeltaCatalogTarget Structure S3
- DeltaDirectTarget Structure S3
- HyperDirectTarget Structure S3
- IcebergDirectTarget Structure S3
- DirectSchemaChangePolicy structure
- ApplyMapping structure
- Structure de mappage
- SelectFields structure
- DropFields structure

- RenameField structure
- Structure Spigot
- Structure Join
- JoinColumn structure
- SplitFields structure
- SelectFromCollection structure
- FillMissingValues structure
- Structure Filtre
- FilterExpression structure
- FilterValue structure
- CustomCode structure
- Structure SparkSQL
- SqlAlias structure
- DropNullFields structure
- NullCheckBoxList structure
- NullValueField structure
- Structure Datatype
- Structure Fusion
- Structure Union
- PIIDetection structure
- Structure Aggregate
- DropDuplicates structure
- GovernedCatalogTarget structure
- GovernedCatalogSource structure
- AggregateOperation structure
- GlueSchema structure
- GlueStudioSchemaColumn structure
- GlueStudioColumn structure
- DynamicTransform structure
- TransformConfigParameter structure

- EvaluateDataQuality structure
- DQResultsPublishingOptions structure
- DQStopJobOnFailureOptions structure
- EvaluateDataQualityMultiFrame structure
- Structure de la recette
- RecipeReference structure
- SnowflakeNodeData structure
- SnowflakeSource structure
- SnowflakeTarget structure
- ConnectorDataSource structure
- ConnectorDataTarget structure
- RecipeStep structure
- RecipeAction structure
- ConditionExpression structure
- CatalogIcebergSource Structure S3
- CatalogIcebergSource structure
- IcebergCatalogTarget Structure S3
- Structure de Dynamo DBELTConnector Source
- DDBELTConnectionStructure des options
- DDBELTCatalogAdditionalOptions structure
- Structure de route
- GroupFilters structure
- AutoDataQuality structure

CodeGenConfigurationNode structure

`CodeGenConfigurationNode` Il énumère les différents types de nœuds valides. Une seule et unique de ses variables membres peut être renseignée.

Champs

- `AthenaConnectorSource` : un objet AthenaConnectorSource.

Indique un connecteur à une source de données Amazon Athena.

- `JDBCConnectorSource` – Un objet JDBCConnectorLa source.

  Indique un connecteur à une source de données JDBC.

- `SparkConnectorSource` – Un objet SparkConnectorSource.

  Indique un connecteur à une source de données Apache Spark.

- `CatalogSource` – Un objet CatalogSource.

  Spécifie un magasin de données dans le catalogue de AWS Glue données.

- `RedshiftSource` – Un objet RedshiftSource.

  Indique un stocker de données Amazon Redshift.

- `S3CatalogSource` – Un objet S3 CatalogSource.

  Spécifie un magasin de données Amazon S3 dans le catalogue de AWS Glue données.

- `S3CsvSource` – Objet S3 CsvSource.

  Indique un stocker de données CSV (valeurs séparées par commande) stocké dans Amazon S3.

- `S3JsonSource` – Un objet S3 JsonSource.

  Indique un stocker de données JSON stocké dans Amazon S3.

- `S3ParquetSource` – Un objet S3 ParquetSource.

  Indique un stocker de données Apache Parquet stocké dans Amazon S3.

- `RelationalCatalogSource` – Un objet RelationalCatalogSource.

  Spécifie un magasin de données de catalogue relationnel dans le catalogue de AWS Glue données.

- `DynamoDBCatalogSource` – Un objet Source de dynamo DBCatalog.

  Spécifie un magasin de données du catalogue DynamoDBC dans le catalogue de AWS Glue données.

- `JDBCConnectorTarget` – Un objet JDBCConnectorCible.

  Indique une cible de données qui écrit sur Amazon S3 dans un stockage en colonnes Apache Parquet.

- `SparkConnectorTarget` – Un objet SparkConnectorTarget.

  Indique une cible qui utilise un connecteur Apache Spark.

- `CatalogTarget` – Un objet BasicCatalogTarget.

  Spécifie une cible qui utilise une table AWS Glue de catalogue de données.

- `RedshiftTarget` – Un objet RedshiftTarget.

  Indique une cible qui utilise Amazon Redshift.

- `S3CatalogTarget` – Un objet S3 CatalogTarget.

  Spécifie une cible de données qui écrit sur Amazon S3 à l'aide du catalogue de AWS Glue données.

- `S3GlueParquetTarget` – Un objet S3 GlueParquetTarget.

  Indique une cible de données qui écrit sur Amazon S3 dans un stockage en colonnes Apache Parquet.

- `S3DirectTarget` – Un objet S3 DirectTarget.

  Indique une cible de données qui écrit dans Amazon S3.

- `ApplyMapping` : un objet ApplyMapping.

  Indique une transformation qui mappe les clés de propriétés de données de la source de données aux clés de propriété de données de la cible de données. Vous pouvez renommer les clés, modifier leur type de données et choisir les clés à supprimer du jeu de données.

- `SelectFields` – Un objet SelectFields.

  Indique une transformation qui choisit les clés de propriété de données que vous souhaitez conserver.

- `DropFields` – Un objet DropFields.

  Indique une transformation qui choisit les clés de propriété de données que vous souhaitez supprimer.

- `RenameField` – Un objet RenameField.

  Indique une transformation qui renomme une clé de propriété de données unique.

- `Spigot` – Un objet Spigot.

Indique une transformation qui écrit des échantillons de données dans un compartiment Amazon S3.

- `Join` – Un objet Join.

  Indique une transformation qui joint deux jeux de données en un jeu de données à l'aide d'une phrase de comparaison sur les clés de propriété de données spécifiées. Vous pouvez utiliser des jointures internes (ou intérieures), externes (ou extérieures), gauche, droite, semi gauche et anti gauche.

- `SplitFields` – Un objet SplitFields.

  Indique une transformation qui divise les clés de propriété de données en deux `DynamicFrames`. Le résultat est une collection de `DynamicFrames` : une avec les clés de propriété de données sélectionnées, et une autre avec les clés de propriété de données restantes.

- `SelectFromCollection` – Un objet SelectFromCollection.

  Indique une transformation qui en choisit une `DynamicFrame` provenant d'une collection de `DynamicFrames`. Le résultat est le `DynamicFrame` sélectionné

- `FillMissingValues` – Un objet FillMissingValues.

  Indique une transformation qui localise les registres dans le jeu de données dont les valeurs sont manquantes et ajoute un nouveau champ avec une valeur déterminée par imputation. Le jeu de données source est utilisé pour entraîner le modèle de machine learning (ML) qui détermine la valeur manquante.

- `Filter` – Un objet Filtre.

  Indique une transformation qui divise un jeu de données en deux, en fonction d'une condition de filtre.

- `CustomCode` – Un objet CustomCode.

  Indique une transformation qui utilise le code personnalisé que vous fournissez pour effectuer la transformation des données. La sortie est une collection de DynamicFrames.

- `SparkSQL` – Un objet SparkSQL.

  Indique une transformation dans laquelle vous entrez une requête SQL à l'aide de la syntaxe Spark SQL pour transformer les données. Le résultat est un `DynamicFrame` unique.

- `DirectKinesisSource` – Un objet DirectKinesisSource.

Indique une source de données Amazon Kinesis directe.

- `DirectKafkaSource` – Un objet DirectKafkaSource.

  Indique un stocker de données Apache Kafka.

- `CatalogKinesisSource` – Un objet CatalogKinesisSource.

  Spécifie une source de données Kinesis dans le catalogue de AWS Glue données.

- `CatalogKafkaSource` – Un objet CatalogKafkaSource.

  Indique un stocker de données Apache Kafka dans le catalogue de données.

- `DropNullFields` – Un objet DropNullFields.

  Indique une transformation qui supprime les colonnes du jeu de données si toutes les valeurs de la colonne sont « nulles ». Par défaut, AWS Glue Studio reconnaît les objets nuls, mais certaines valeurs telles que les chaînes vides, les chaînes « nulles », les entiers -1 ou d'autres espaces réservés tels que les zéros ne sont pas automatiquement reconnues comme nulles.

- `Merge` – Un objet Fusionner.

  Indique une transformation qui fusionne une `DynamicFrame` avec une `DynamicFrame` intermédiaire basée sur les clés primaires spécifiées pour identifier les registres. Les registres en double (registres avec les mêmes clés primaires) ne sont pas dédupliqués.

- `Union` : un objet Union.

  Indique une transformation qui combine les lignes de deux jeux de données ou plus en un seul résultat.

- `PIIDetection` – Un objet PIIDetection.

  Indique une transformation qui identifie, supprime ou masque les données d'informations personnelles identifiables (PII).

- `Aggregate` : un objet Regrouper.

  Indique une transformation qui regroupe les lignes par champs choisis et calcule la valeur agrégée par fonction spécifiée.

- `DropDuplicates` – Un objet DropDuplicates.

  Indique une transformation qui supprime des lignes de données répétitives d'un jeu de données.

- `GovernedCatalogTarget` – Un objet GovernedCatalogTarget.

Indique une cible de données qui écrit dans un catalogue gouverné.

- `GovernedCatalogSource` – Un objet GovernedCatalogSource.

Indique une source de données dans un catalogue de données gouverné.

- `MicrosoftSQLServerCatalogSource` – Un objet Microsoft SQLServer CatalogSource.

Indique une source de données Microsoft SQL Server dans le catalogue de données AWS Glue .

- `MySQLCatalogSource` – Un objet Ma SQLCatalog source.

Spécifie une source de données MySQL dans le catalogue de AWS Glue données.

- `OracleSQLCatalogSource` : un objet SQLCatalogSource d'Oracle.

Spécifie une source de données Oracle dans le catalogue de AWS Glue données.

- `PostgreSQLCatalogSource` – Un objet Source de Postgrer SQLCatalog.

Spécifie une source de données PostgressQL dans le catalogue de données. AWS Glue

- `MicrosoftSQLServerCatalogTarget` – Un objet Microsoft SQLServer CatalogTarget.

Indique une cible qui utilise Microsoft SQL.

- `MySQLCatalogTarget` – Un objet Ma SQLCatalog cible.

Indique une cible qui utilise MySQL.

- `OracleSQLCatalogTarget` : un objet SQLCatalogCible Oracle.

Indique une cible qui utilise Oracle SQL.

- `PostgreSQLCatalogTarget` – Un objet Cible Postgrer SQLCatalog.

Indique une cible qui utilise Postgre SQL.

- `Route` – Un objet Acheminement.

Spécifie un nœud de route qui dirige les données vers différents chemins de sortie en fonction de conditions de filtrage définies.

- `DynamicTransform` – Un objet DynamicTransform.

Spécifie une transformation visuelle personnalisée créée par un utilisateur.

- `EvaluateDataQuality` : un objet EvaluateDataQuality.

Spécifie vos critères d'évaluation de la qualité des données.

- `S3CatalogHudiSource` – Un objet S3 CatalogHudiSource.

  Spécifie une source de données Hudi enregistrée dans le catalogue de AWS Glue données. La source de données doit être stockée dans Amazon S3.

- `CatalogHudiSource` – Un objet CatalogHudiSource.

  Spécifie une source de données Hudi enregistrée dans le catalogue de AWS Glue données.

- `S3HudiSource` – Un objet S3 HudiSource.

  Spécifie une source de données Hudi stockée dans. Amazon S3

- `S3HudiCatalogTarget` – Un objet S3 HudiCatalogTarget.

  Spécifie une cible qui écrit dans une source de données Hudi du catalogue de AWS Glue données.

- `S3HudiDirectTarget` – Un objet S3 HudiDirectTarget.

  Spécifie une cible qui écrit dans une source de données Hudi en Amazon S3.

- `S3CatalogDeltaSource` – Un objet S3 CatalogDeltaSource.

  Spécifie une source de données Delta Lake enregistrée dans le catalogue de AWS Glue données. La source de données doit être stockée dans Amazon S3.

- `CatalogDeltaSource` – Un objet CatalogDeltaSource.

  Spécifie une source de données Delta Lake enregistrée dans le catalogue de AWS Glue données.

- `S3DeltaSource` – Un objet S3 DeltaSource.

  Spécifie une source de données Delta Lake stockée dans Amazon S3.

- `S3DeltaCatalogTarget` – Un objet S3 DeltaCatalogTarget.

  Spécifie une cible qui écrit dans une source de données Delta Lake dans le catalogue de AWS Glue données.

- `S3DeltaDirectTarget` – Un objet S3 DeltaDirectTarget.

  Spécifie une cible qui écrit dans une source de données de Delta Lake en Amazon S3.

- `AmazonRedshiftSource` : un objet AmazonRedshiftSource.

  Indique une cible qui écrit dans une source de données dans Amazon Redshift.

- `AmazonRedshiftTarget` : un objet AmazonRedshiftTarget.

  Indique une cible qui écrit dans une cible de données dans Amazon Redshift.

- `EvaluateDataQualityMultiFrame` : un objet EvaluateDataQualityMultiFrame.

  Spécifie vos critères d'évaluation de la qualité des données. Autorise plusieurs données d'entrée et renvoie une collection de cadres dynamiques.

- `Recipe` – Un objet Formule.

  Spécifie un nœud de AWS Glue DataBrew recette.

- `SnowflakeSource` – Un objet SnowflakeSource.

  Indique une source de données Snowflake.

- `SnowflakeTarget` – Un objet SnowflakeTarget.

  Indique une cible qui écrit dans une source de données Snowflake.

- `ConnectorDataSource` – Un objet ConnectorDataSource.

  Spécifie une source générée avec des options de connexion standard.

- `ConnectorDataTarget` – Un objet ConnectorDataTarget.

  Spécifie une cible générée avec des options de connexion standard.

- `S3CatalogIcebergSource` – Un objet S3 CatalogIcebergSource.

  Spécifie une source de données Apache Iceberg enregistrée dans le catalogue de AWS Glue données. La source de données Iceberg doit être stockée dans Amazon S3.

- `CatalogIcebergSource` – Un objet CatalogIcebergSource.

  Spécifie une source de données Apache Iceberg enregistrée dans le catalogue de AWS Glue données.

- `S3IcebergCatalogTarget` – Un objet S3 IcebergCatalogTarget.

  Spécifie une cible de catalogue Apache Iceberg qui écrit des données Amazon S3 et enregistre la table dans le catalogue de AWS Glue données.

- `S3IcebergDirectTarget` – Un objet S3 IcebergDirectTarget.

  Définit les paramètres de configuration pour écrire des données dans Amazon S3 sous forme de table Apache Iceberg.

- `S3ExcelSource` – Un objet S3 ExcelSource.

  Définit les paramètres de configuration pour lire des fichiers Excel depuis Amazon S3.
- `S3HyperDirectTarget` – Un objet S3 HyperDirectTarget.

  Définit les paramètres de configuration pour écrire des données sur Amazon S3 à l'aide de l' HyperDirect optimisation.
- `DynamoDBELTConnectorSource` – Un objet Source de dynamo DBELTConnector.

  Spécifie une source de connecteur DynamoDB ELT pour extraire les données des tables DynamoDB.

JDBCConnectorStructure des options

Options de connexion supplémentaires pour le connecteur.

Champs

- `FilterPredicate` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Clause de condition supplémentaire pour filtrer les données à partir de la source. Par exemple :

  ```
  BillingCity='Mountain View'
  ```

  Lorsque vous utilisez une requête au lieu d'un nom de tableau, vous devez vérifier que la requête fonctionne avec le `filterPredicate` spécifié.
- `PartitionColumn` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom d'une colonne entière utilisée pour le partitionnement. Cette option fonctionne uniquement lorsqu'elle est incluse dans `lowerBound`, `upperBound` et `numPartitions`. Cette option fonctionne de la même manière que dans le lecteur JDBC SQL Spark.
- `LowerBound` – Nombre (long), pas plus qu'Aucun.

  La valeur minimale de `partitionColumn` qui est utilisée pour décider de la progression de la partition.
- `UpperBound` – Nombre (long), pas plus qu'Aucun.

  La valeur maximale de `partitionColumn` qui est utilisée pour décider de la progression de la partition.

- `NumPartitions` – Nombre (long), pas plus qu'Aucun.

  Nombre de partitions. Cette valeur, ainsi que `lowerBound` (inclusive) et `upperBound` (exclusive) forment les progressions de partition pour les expressions de clause `WHERE` générées qui sont utilisées pour diviser le fichier `partitionColumn`.

- `JobBookmarkKeys` – Tableau de chaînes UTF-8.

  Le nom des clés de marque-page de tâches sur lesquelles effectuer le tri.

- `JobBookmarkKeysSortOrder` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Indique un ordre de tri croissant ou décroissant.

- `DataTypeMapping` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8 (valeurs valides :`ARRAY` | `BIGINT` | `BINARY` | `BIT` | `BLOB` | `BOOLEAN` | `CHAR` | `CLOB` | `DATALINK` | `DATE` | `DECIMAL` | `DISTINCT` | `DOUBLE` | `FLOAT` | `INTEGER` | `JAVA_OBJECT` | `LONGNVARCHAR` | `LONGVARBINARY` | `LONGVARCHAR` | `NCHAR` | `NCLOB` | `NULL` | `NUMERIC` | `NVARCHAR` | `OTHER` | `REAL` | `REF` | `REF_CURSOR` | `ROWID` | `SMALLINT` | `SQLXML` | `STRUCT` | `TIME` | `TIME_WITH_TIMEZONE` | `TIMESTAMP` | `TIMESTAMP_WITH_TIMEZONE` | `TINYINT` | `VARBINARY` | `VARCHAR`).

  Chaque valeur est une chaîne UTF-8 (valeurs valides : `DATE` | `STRING` | `TIMESTAMP` | `INT` | `FLOAT` | `LONG` | `BIGDECIMAL` | `BYTE` | `SHORT` | `DOUBLE`).

  Mappage de type de données personnalisé qui crée un mappage d'un type de données JDBC à un type de données AWS Glue . Par exemple, l'option `"dataTypeMapping":{"FLOAT":"STRING"}` mappe les champs de données de type JDBC `FLOAT` dans le `String` type Java en appelant la `ResultSet.getString()` méthode du pilote et l'utilise pour créer l' AWS Glue enregistrement. L'objet est `ResultSet` implémenté par chaque pilote, donc le comportement est spécifique au pilote que vous utilisez. Reportez-vous à la documentation de votre pilote JDBC pour comprendre comment le pilote effectue les conversions.

StreamingDataPreviewOptions structure

Indique les options liées à la prévisualisation des données pour visualiser un échantillon de vos données.

Champs

- `PollingTime` : nombre (long), au moins égal à 10.

  Temps d'interrogation en millisecondes.
- `RecordPollingLimit` : nombre (long), au moins égal à 1.

  Limite du nombre de registres interrogés.

AthenaConnectorSource structure

Indique un connecteur à une source de données Amazon Athena.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom de la source de données.
- `ConnectionName` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la connexion associée au connecteur.
- `ConnectorName` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom d'un connecteur qui facilite l'accès au magasin de données dans AWS Glue Studio.
- `ConnectionType` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Type de connexion, tel que marketplace.athena ou custom.athena, désignant une connexion à un stocker de données Amazon Athena.
- `ConnectionTable` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la table dans la source de données.
- `SchemaName` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom du groupe de journaux CloudWatch à partir duquel lire les données. Par exemple, `/aws-glue/jobs/output`.
- `OutputSchemas` : un tableau d'objets GlueSchema.

  Indique le schéma de données de la source Athena personnalisée.

JDBCConnectorStructure de la source

Indique un connecteur à une source de données JDBC.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom de la source de données.

- `ConnectionName` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la connexion associée au connecteur.

- `ConnectorName` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom d'un connecteur qui facilite l'accès au magasin de données dans AWS Glue Studio.

- `ConnectionType` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Type de connexion, tel que marketplace.jdbc ou custom.jdbc, désignant une connexion à un stocker de données JDBC.

- `AdditionalOptions` – Un objet JDBCConnectorOptions.

  Options de connexion supplémentaires pour le connecteur.

- `ConnectionTable` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la table dans la source de données.

- `Query` – Chaîne UTF-8, correspondant au Custom string pattern #62.

  La table ou la requête SQL à partir de laquelle obtenir les données. Vous pouvez préciser `ConnectionTable` ou `query`, mais pas les deux.

- `OutputSchemas` : un tableau d'objets GlueSchema.

  Indique le schéma de données de la source JDBC personnalisée.

SparkConnectorSource structure

Indique un connecteur à une source de données Apache Spark.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom de la source de données.
- `ConnectionName` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la connexion associée au connecteur.
- `ConnectorName` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom d'un connecteur qui facilite l'accès au magasin de données dans AWS Glue Studio.
- `ConnectionType` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Type de connexion, tel que marketplace.spark ou custom.spark, désignant une connexion à un stocker de données Apache Spark.
- `AdditionalOptions` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, correspondant au Custom string pattern #61.

  Chaque valeur est une chaîne UTF-8, correspondant au Custom string pattern #61.

  Options de connexion supplémentaires pour le connecteur.
- `OutputSchemas` : un tableau d'objets GlueSchema.

  Indique le schéma de données de la source Spark personnalisée.

CatalogSource structure

Spécifie un magasin de données dans le catalogue de AWS Glue données.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom du stocker de données.
- `Database` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la base de données à partir de laquelle lire les données.
- `Table` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

Le nom de la table dans la base de données à partir de laquelle lire les données.

- `PartitionPredicate` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Les partitions satisfaisant à ce prédicat sont supprimées. Les fichiers qui se situent dans la période de conservation pour ces partitions ne sont pas supprimés.
- `OutputSchemas` : un tableau d'objets GlueSchema.

  Indique le schéma de données de la source de catalogue.

Structure SQLCatalog de ma source

Spécifie une source de données MySQL dans le catalogue de AWS Glue données.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom de la source de données.
- `Database` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la base de données à partir de laquelle lire les données.
- `Table` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la table dans la base de données à partir de laquelle lire les données.

Structure de la SQLCatalog source Postgre

Spécifie une source de données PostgressQL dans le catalogue de données. AWS Glue

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom de la source de données.
- `Database` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la base de données à partir de laquelle lire les données.
- `Table` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

Le nom de la table dans la base de données à partir de laquelle lire les données.

Structure d'Oracle SQLCatalog Source

Spécifie une source de données Oracle dans le catalogue de AWS Glue données.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom de la source de données.
- `Database` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la base de données à partir de laquelle lire les données.
- `Table` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la table dans la base de données à partir de laquelle lire les données.

SQLServerCatalogSource Structure de Microsoft

Indique une source de données Microsoft SQL Server dans le catalogue de données AWS Glue .

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom de la source de données.
- `Database` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la base de données à partir de laquelle lire les données.
- `Table` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la table dans la base de données à partir de laquelle lire les données.

CatalogKinesisSource structure

Spécifie une source de données Kinesis dans le catalogue de AWS Glue données.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom de la source de données.
- `WindowSize` – Nombre (entier), pas plus qu'Aucun.

  Durée de traitement de chaque micro lot.
- `DetectSchema` : booléen.

  Indique s'il faut déterminer automatiquement le schéma à partir des données entrantes.
- `Table` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la table dans la base de données à partir de laquelle lire les données.
- `Database` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la base de données à partir de laquelle lire les données.
- `StreamingOptions` – Un objet KinesisStreamingSourceOptions.

  Options supplémentaires pour la source de données en streaming Kinesis.
- `DataPreviewOptions` – Un objet StreamingDataPreviewOptions.

  Options supplémentaires pour la prévisualisation des données.

DirectKinesisSource structure

Indique une source de données Amazon Kinesis directe.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom de la source de données.
- `WindowSize` – Nombre (entier), pas plus qu'Aucun.

  Durée de traitement de chaque micro lot.
- `DetectSchema` : booléen.

  Indique s'il faut déterminer automatiquement le schéma à partir des données entrantes.

- `StreamingOptions` – Un objet KinesisStreamingSourceOptions.

  Options supplémentaires pour la source de données en streaming Kinesis.

- `DataPreviewOptions` – Un objet StreamingDataPreviewOptions.

  Options supplémentaires pour la prévisualisation des données.

KinesisStreamingSourceOptions structure

Options supplémentaires pour la source de données Amazon Kinesis streaming.

Champs

- `EndpointUrl` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  L'URL du point de terminaison de Kinesis.

- `StreamName` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom du flux de données Kinesis.

- `Classification` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Une classification facultative.

- `Delimiter` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Indique le caractère délimiteur.

- `StartingPosition` – Chaîne UTF-8 (valeurs valides : `latest="LATEST"` | `trim_horizon="TRIM_HORIZON"` | `earliest="EARLIEST"` | `timestamp="TIMESTAMP"`).

  La position de départ dans le flux de données Kinesis à partir duquel lire les données. Les valeurs possibles sont `"latest"`, `"trim_horizon"`, `"earliest"`, ou une chaîne d'horodatage au format UTC dans le modèle `yyyy-mm-ddTHH:MM:SSZ` (où `Z` représente un décalage de fuseau horaire UTC avec un +/-. Par exemple : « 2023-04-04T08:00:00-04:00 »). La valeur par défaut est `"latest"`.

  Remarque : L'utilisation d'une valeur qui est une chaîne d'horodatage au format UTC pour « StartingPosition » n'est prise en charge que pour la AWS Glue version 4.0 ou ultérieure.

- `MaxFetchTimeInMs` – Nombre (long), pas plus qu'Aucun.

Le temps maximal passé dans l'exécuteur de tâches pour lire des enregistrements du lot actuel à partir du flux de données Kinesis, exprimé en millisecondes (ms). Plusieurs appels d'API `GetRecords` peuvent être effectués pendant cette période. La valeur par défaut est `1000`.

- `MaxFetchRecordsPerShard` – Nombre (long), pas plus qu'Aucun.

  Nombre maximal d'enregistrements à récupérer par partition dans le flux de données Kinesis par microlot. Remarque : Le client peut dépasser cette limite si la tâche de streaming a déjà lu des enregistrements supplémentaires provenant de Kinesis (dans le même appel get-records). Si `MaxFetchRecordsPerShard` doit être strict, il doit être un multiple de `MaxRecordPerRead`. La valeur par défaut est `100000`.

- `MaxRecordPerRead` – Nombre (long), pas plus qu'Aucun.

  Nombre maximal de registres à extraire du flux de données Kinesis dans chaque opération . La valeur par défaut est `10000`.

- `AddIdleTimeBetweenReads` : booléen.

  Ajoute un délai entre deux opérations getRecords consécutives. La valeur par défaut est `"False"`. Cette option n'est configurable que pour AWS Glue versions 2.0 et ultérieures.

- `IdleTimeBetweenReadsInMs` – Nombre (long), pas plus qu'Aucun.

  Le délai minimum entre deux opérations getRecords consécutives, spécifié en ms. La valeur par défaut est `1000`. Cette option n'est configurable que pour les AWS Glue versions 2.0 et supérieures.

- `DescribeShardInterval` – Nombre (long), pas plus qu'Aucun.

  Intervalle de temps minimum entre deux appels ListShards d'API avant que votre script envisage le repartage. La valeur par défaut est `1s`.

- `NumRetries` – Nombre (entier), pas plus qu'Aucun.

  Le nombre maximal de nouvelles tentatives pour les demandes d'API Kinesis Data Streams. La valeur par défaut est `3`.

- `RetryIntervalMs` – Nombre (long), pas plus qu'Aucun.

  Le délai de réflexion (spécifié en ms) avant de réessayer l'appel d'API Kinesis Data Streams. La valeur par défaut est `1000`.

- `MaxRetryIntervalMs` – Nombre (long), pas plus qu'Aucun.

Le délai d'attente maximal (spécifié en ms) entre deux tentatives d'appel d'API Kinesis Data Streams. La valeur par défaut est `10000`.

- `AvoidEmptyBatches` : booléen.

  Évite de créer une tâche de micro-lot vide en vérifiant les données non lues dans le flux de données Kinesis avant le démarrage du lot. La valeur par défaut est `"False"`.

- `StreamArn` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom Amazon Resource Name (ARN) du flux de données Kinesis.

- `RoleArn` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom Amazon Resource Name (ARN) du rôle à endosser à l'aide d'AWS Security Token Service (AWS STS). Ce rôle doit disposer des autorisations nécessaires pour décrire ou lire des registres pour le flux de données Kinesis. Vous devez utiliser ce paramètre lorsque vous accédez à un flux de données dans un autre compte. Utilisez conjointement avec `"awsSTSSessionName"`.

- `RoleSessionName` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Un identifiant de la session endossant le rôle à l'aide d'AWS STS. Vous devez utiliser ce paramètre lorsque vous accédez à un flux de données dans un autre compte. Utilisez conjointement avec `"awsSTSRoleARN"`.

- `AddRecordTimestamp` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Lorsque cette option est définie sur « true », la sortie de données contient une colonne supplémentaire nommée « __src_timestamp » qui indique l'heure à laquelle l'enregistrement correspondant est reçu par le flux. La valeur par défaut est « false ». Cette option est prise en charge dans AWS Glue la version 4.0 ou ultérieure.

- `EmitConsumerLagMetrics` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Lorsque cette option est définie sur « true », pour chaque lot, elle émet les métriques correspondant à la durée comprise entre le plus ancien enregistrement reçu par le flux et l'heure AWS Glue à laquelle il arrive CloudWatch. Le nom de la métrique est « glue.driver.streaming ». maxConsumerLagInMs». La valeur par défaut est « false ». Cette option est prise en charge dans AWS Glue version 4.0 ou ultérieure.

- `StartingTimestamp` – Chaîne UTF-8.

  L'horodatage de l'enregistrement dans le flux de données Kinesis à partir duquel les données doivent être lues. Les valeurs possibles sont une chaîne d'horodatage au format UTC du modèle

`yyyy-mm-ddTHH:MM:SSZ` (où Z représente un décalage de fuseau horaire UTC avec un +/-. Par exemple : « 2023-04-04T08:00:00+08:00 »).

- `FanoutConsumerARN` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Amazon Resource Name (ARN) du consommateur à ventilation améliorée Kinesis Data Streams. Lorsqu'il est spécifié, il permet une ventilation améliorée pour un débit dédié et une consommation de données à latence réduite.

CatalogKafkaSource structure

Indique un stocker de données Apache Kafka dans le catalogue de données.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom du stocker de données.

- `WindowSize` – Nombre (entier), pas plus qu'Aucun.

  Durée de traitement de chaque micro lot.

- `DetectSchema` : booléen.

  Indique s'il faut déterminer automatiquement le schéma à partir des données entrantes.

- `Table` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la table dans la base de données à partir de laquelle lire les données.

- `Database` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la base de données à partir de laquelle lire les données.

- `StreamingOptions` – Un objet KafkaStreamingSourceOptions.

  Indique les options de streaming.

- `DataPreviewOptions` – Un objet StreamingDataPreviewOptions.

  Indique les options liées à la prévisualisation des données pour visualiser un échantillon de vos données.

DirectKafkaSource structure

Indique un stocker de données Apache Kafka.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom du stocker de données.
- `StreamingOptions` – Un objet KafkaStreamingSourceOptions.

  Indique les options de streaming.
- `WindowSize` – Nombre (entier), pas plus qu'Aucun.

  Durée de traitement de chaque micro lot.
- `DetectSchema` : booléen.

  Indique s'il faut déterminer automatiquement le schéma à partir des données entrantes.
- `DataPreviewOptions` – Un objet StreamingDataPreviewOptions.

  Indique les options liées à la prévisualisation des données pour visualiser un échantillon de vos données.

KafkaStreamingSourceOptions structure

Options supplémentaires pour streaming.

Champs

- `BootstrapServers` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Une liste de serveurs bootstrap URLs, par exemple, commeb-1.vpc-test-2.o4q88o.c6.kafka.us-east-1.amazonaws.com:9094. Cette option doit être spécifiée dans l'appel d'API ou définie dans les métadonnées de la table dans le catalogue de données.
- `SecurityProtocol` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le protocole utilisé pour communiquer avec les agents. Les valeurs possibles sont `"SSL"` ou `"PLAINTEXT"`.

- `ConnectionName` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Nom de la connexion.

- `TopicName` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de rubrique tel que spécifié dans Apache Kafka. Vous devez Indiquer au moins un des éléments suivants : `"topicName"`, `"assign"` ou `"subscribePattern"`.

- `Assign` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Les `TopicPartitions` spécifiques à consommer. Vous devez Indiquer au moins un des éléments suivants : `"topicName"`, `"assign"` ou `"subscribePattern"`.

- `SubscribePattern` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Une chaîne d'expression rationnelle Java qui identifie la liste de rubriques à laquelle vous souhaitez vous abonner. Vous devez Indiquer au moins un des éléments suivants : `"topicName"`, `"assign"` ou `"subscribePattern"`.

- `Classification` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Une classification facultative.

- `Delimiter` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Indique le caractère délimiteur.

- `StartingOffsets` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  La position de départ dans la rubrique Kafka à partir de laquelle lire les données. Les valeurs possibles sont `"earliest"` ou `"latest"`. La valeur par défaut est `"latest"`.

- `EndingOffsets` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le point de fin lorsqu'une requête par lots est terminée. Les valeurs possibles sont `"latest"` ou une chaîne JSON qui Indique un décalage de fin pour chaque `TopicPartition`.

- `PollTimeoutMs` – Nombre (long), pas plus qu'Aucun.

  Le délai d'attente en millisecondes pour interroger les données de Kafka dans les exécuteurs de tâches Spark. La valeur par défaut est `512`.

- `NumRetries` – Nombre (entier), pas plus qu'Aucun.

  Le nombre de nouvelles tentatives avant de ne pas récupérer les décalages Kafka. La valeur par défaut est `3`.

- `RetryIntervalMs` – Nombre (long), pas plus qu'Aucun.

  Temps d'attente en millisecondes avant d'essayer de récupérer les décalages Kafka. La valeur par défaut est `10`.

- `MaxOffsetsPerTrigger` – Nombre (long), pas plus qu'Aucun.

  La limite de taux sur le nombre maximal de décalages qui sont traités par intervalle de déclenchement. Le nombre total spécifié de décalages est réparti proportionnellement entre les `topicPartitions` des différents volumes. La valeur par défaut est null, ce qui signifie que le consommateur lit tous les décalages jusqu'au dernier décalage connu.

- `MinPartitions` – Nombre (entier), pas plus qu'Aucun.

  Le nombre minimum de partitions à lire à partir de Kafka. La valeur par défaut est nulle, ce qui signifie que le nombre de partitions Spark est égal au nombre de partitions Kafka.

- `IncludeHeaders` : booléen.

  Indique s'il faut inclure les en-têtes Kafka. Lorsque l'option est définie sur « true » (vrai), la sortie de données contiendra une colonne supplémentaire nommée « glue_streaming_kafka_headers » avec le type `Array[Struct(key: String, value: String)]`. La valeur définie par défaut est « false ». Cette option n'est disponible que dans AWS Glue la version 3.0 ou ultérieure.

- `AddRecordTimestamp` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Lorsque cette option est définie sur « true », la sortie de données contient une colonne supplémentaire nommée « __src_timestamp » qui indique l'heure à laquelle l'enregistrement correspondant est reçu par la rubrique. La valeur par défaut est « false ». Cette option est prise en charge dans AWS Glue la version 4.0 ou ultérieure.

- `EmitConsumerLagMetrics` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Lorsque cette option est définie sur « vrai », pour chaque lot, elle émet les métriques correspondant à la durée comprise entre le plus ancien enregistrement reçu par le sujet et le moment où il arrive CloudWatch. AWS Glue Le nom de la métrique est « glue.driver.streaming ». maxConsumerLagInMs». La valeur par défaut est « false ». Cette option est prise en charge dans AWS Glue version 4.0 ou ultérieure.

- `StartingTimestamp` – Chaîne UTF-8.

  L'horodatage de l'enregistrement dans la rubrique Kafka à partir duquel les données doivent être lues. Les valeurs possibles sont une chaîne d'horodatage au format UTC du modèle `yyyy-mm-`

ddTHH:MM:SSZ (où Z représente un décalage de fuseau horaire UTC avec un +/-. Par exemple : « 2023-04-04T08:00:00+08:00 »).

Seul StartingTimestamp ou StartingOffsets doit être défini.

RedshiftSource structure

Indique un stocker de données Amazon Redshift.

Champs

- Name – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Le nom du stocker de données Amazon Redshift.
- Database – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  La base de données à partir de laquelle lire les données.
- Table – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Table de base de données à lire.
- RedshiftTmpDir – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Chemin Amazon S3 où les données temporaires peuvent être stockées lors de la copie à partir de la base de données.
- TmpDirIAMRole – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le rôle IAM avec les autorisations.

AmazonRedshiftSource structure

Indique une source Amazon Redshift.

Champs

- Name – Chaîne UTF-8, correspondant au Custom string pattern #63.

  Le nom de la source Amazon Redshift.
- Data : un objet AmazonRedshiftNodeData.

Indique les données du nœud source Amazon Reshift.

AmazonRedshiftNodeData structure

Indique un nœud Amazon Redshift.

Champs

- `AccessType` – Chaîne UTF-8, correspondant au Custom string pattern #60.

  Le type d'accès pour la connexion Redshift. Il peut s'agir d'une connexion directe ou de connexions au catalogue.
- `SourceType` – Chaîne UTF-8, correspondant au Custom string pattern #60.

  Le type de source pour spécifier si une table spécifique est la source ou une requête personnalisée.
- `Connection` : un objet Option.

  La AWS Glue connexion au cluster Redshift.
- `Schema` : un objet Option.

  Le nom du schéma Redshift lorsque vous travaillez avec une connexion directe.
- `Table` : un objet Option.

  Le nom de la table Redshift lorsque vous travaillez avec une connexion directe.
- `CatalogDatabase` : un objet Option.

  Nom de la base de AWS Glue données du catalogue de données lorsque vous travaillez avec un catalogue de données.
- `CatalogTable` : un objet Option.

  Le nom de la table du catalogue de AWS Glue données lorsque vous travaillez avec un catalogue de données.
- `CatalogRedshiftSchema` – Chaîne UTF-8.

  Le nom du schéma Redshift lorsque vous travaillez avec un catalogue de données.
- `CatalogRedshiftTable` – Chaîne UTF-8.

Table de base de données à lire.

- `TempDir` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Chemin Amazon S3 où les données temporaires peuvent être stockées lors de la copie à partir de la base de données.

- `IamRole` : un objet Option.

  Facultatif. Le nom de rôle utilisé lors de la connexion à S3. Le rôle IAM sera remplacé par défaut par le rôle sur la tâche lorsque ce champ est laissé vide.

- `AdvancedOptions` : un tableau d'objets AmazonRedshiftAdvancedOption.

  Les valeurs facultatives lors de la connexion au cluster Redshift.

- `SampleQuery` – Chaîne UTF-8.

  Le code SQL utilisé pour récupérer les données d'une source Redshift lorsqu'il SourceType s'agit d'une « requête ».

- `PreAction` – Chaîne UTF-8.

  Le code SQL utilisé avant l'exécution d'une opération MERGE ou APPEND avec insertion.

- `PostAction` – Chaîne UTF-8.

  Le code SQL utilisé avant l'exécution d'une opération MERGE ou APPEND avec insertion.

- `Action` – Chaîne UTF-8.

  Indique comment l'écriture dans un cluster Redshift se fera.

- `TablePrefix` – Chaîne UTF-8, correspondant au Custom string pattern #60.

  Indique le préfixe d'une table.

- `Upsert` : booléen.

  L'action utilisée sur les récepteurs Redshift lorsque vous effectuez une opération APPEND.

- `MergeAction` – Chaîne UTF-8, correspondant au Custom string pattern #60.

  L'action utilisée pour déterminer comment une opération MERGE dans un récepteur Redshift sera traitée.

- `MergeWhenMatched` – Chaîne UTF-8, correspondant au Custom string pattern #60.

L'action utilisée pour déterminer comment une opération MERGE dans un récepteur Redshift sera traitée lorsqu'un enregistrement existant correspond à un nouvel enregistrement.

- `MergeWhenNotMatched` – Chaîne UTF-8, correspondant au Custom string pattern #60.

  L'action utilisée pour déterminer comment une opération MERGE dans un récepteur Redshift sera traitée lorsqu'un enregistrement existant ne correspond pas à un nouvel enregistrement.

- `MergeClause` – Chaîne UTF-8.

  Le code SQL utilisé dans une fusion personnalisée pour traiter les enregistrements correspondants.

- `CrawlerConnection` – Chaîne UTF-8.

  Indique le nom de la connexion associée à la table de catalogue utilisée.

- `TableSchema` : un tableau d'objets Option.

  Le tableau de sortie du schéma pour un nœud donné.

- `StagingTable` – Chaîne UTF-8.

  Le nom de la table intermédiaire temporaire utilisée lors d'une opération MERGE ou APPEND avec insertion.

- `SelectedColumns` : un tableau d'objets Option.

  La liste des noms de colonnes utilisée pour déterminer un enregistrement correspondant lors d'une opération MERGE ou APPEND avec insertion.

AmazonRedshiftAdvancedOption structure

Indique une valeur facultative lors de la connexion au cluster Redshift.

Champs

- `Key` – Chaîne UTF-8.

  La clé de l'option de connexion supplémentaire.

- `Value` – Chaîne UTF-8.

  La valeur de l'option de connexion supplémentaire.

Structure de l'option

Indique une valeur d'option.

Champs

- `Value` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Indique la valeur de l'option.
- `Label` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Indique l'étiquette de l'option.
- `Description` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Indique la description de l'option.

CatalogSource Structure S3

Spécifie un magasin de données Amazon S3 dans le catalogue de AWS Glue données.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom du stocker de données.
- `Database` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  La base de données à partir de laquelle lire les données.
- `Table` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Table de base de données à lire.
- `PartitionPredicate` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Les partitions satisfaisant à ce prédicat sont supprimées. Les fichiers qui se situent dans la période de conservation pour ces partitions ne sont pas supprimés. Valeur définie sur "" – vide par défaut.
- `AdditionalOptions` – Un objet S3 SourceAdditionalOptions.

  Indique des options de connexion supplémentaires.

SourceAdditionalOptions Structure S3

Indique des options de connexion supplémentaires pour le stocker de données Amazon S3.

Champs

- `BoundedSize` – Nombre (long).

  Définit la limite supérieure de la dimension cible du jeu de données en octets à traiter.
- `BoundedFiles` – Nombre (long).

  Définit la limite supérieure du nombre cible de fichiers à traiter.

CsvSource Structure S3

Indique un stocker de données CSV (valeurs séparées par commande) stocké dans Amazon S3.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom du stocker de données.
- `Paths` – Obligatoire : Tableau de chaînes UTF-8.

  Une liste de chemins Amazon S3 à lire.
- `CompressionType` – Chaîne UTF-8 (valeurs valides : `gzip="GZIP"` | `bzip2="BZIP2"`).

  Indique la manière dont les données sont comprimées. Ce n'est généralement pas nécessaire si le fichier de données a une extension standard. Les valeurs possibles sont `"gzip"` et `"bzip"`).
- `Exclusions` – Tableau de chaînes UTF-8.

  Une chaîne contenant une liste JSON des modèles glob de style Unix à exclure. Par exemple, [\"**.pdf\"] permet d'exclure tous les fichiers PDF.
- `GroupSize` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  La dimension du groupe cible, en octets. La valeur par défaut est calculée en fonction de la dimension des données en entrée et de la dimension de votre cluster. Lorsqu'il y a moins de 50 000 fichiers en entrée, `"groupFiles"` doit être défini sur `"inPartition"` pour que cela prenne effet.

- `GroupFiles` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le groupement de fichiers est activé par défaut lorsque l'entrée contient plus de 50 000 fichiers. Pour activer le groupement lorsqu'il y a moins de 50 000 fichiers, définissez ce paramètre sur « inPartition ». Pour désactiver le groupement lorsqu'il y a plus de 50 000 fichiers, définissez ce paramètre sur `"none"`.

- `Recurse` : booléen.

  Si ce paramètre est défini sur « VRAI », les fichiers sont lus de manière récursive dans tous les sous-répertoires des chemins spécifiés.

- `MaxBand` – Nombre (entier), pas plus qu'Aucun.

  Cette option permet de contrôler la durée, en millisecondes, au delà de laquelle la liste S3 est susceptible d'être cohérente. Les fichiers dont l'horodatage des modifications se situe dans les dernières millisecondes de MaxBand sont suivis, en particulier lors de leur utilisation, afin de tenir compte de la cohérence éventuelle JobBookmarks d'Amazon S3. La plupart des utilisateurs n'ont pas besoin de définir cette option. La valeur par défaut est 900 000 millisecondes, soit 15 minutes.

- `MaxFilesInBand` – Nombre (entier), pas plus qu'Aucun.

  Cette option Indique le nombre maximal de fichiers à enregistrer à partir des dernières secondes maxBand. Si ce nombre est dépassé, les fichiers supplémentaires sont ignorés et traités dans l'exécution de tâche suivante.

- `AdditionalOptions` – Un objet S3 DirectSourceAdditionalOptions.

  Indique des options de connexion supplémentaires.

- `Separator` – Obligatoire : Chaîne UTF-8 (valeurs valides : `comma="COMMA"` | `ctrla="CTRLA"` | `pipe="PIPE"` | `semicolon="SEMICOLON"` | `tab="TAB"`).

  Indique le caractère délimiteur. La valeur par défaut est une virgule : « , », mais tout autre caractère peut être spécifié.

- `Escaper` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Indique le caractère à utiliser pour l'échappement. Cette option est utilisée uniquement lors de la lecture de fichiers CSV. La valeur par défaut est `none`. Si cette option est activée, le caractère suivant est immédiatement utilisé tel quel, sauf pour un petit ensemble d'échappements connus (`\n`, `\r`, `\t` et `\0`).

- `QuoteChar` – Obligatoire :Chaîne UTF-8 (valeurs valides : `quote="QUOTE"` | `quillemet="QUILLEMET"` | `single_quote="SINGLE_QUOTE"` | `disabled="DISABLED"`).

  Indique le caractère à utiliser pour les guillemets. La valeur par défaut est les guillemets doubles : `'"'`. Définissez ce champ sur `-1` pour désactiver entièrement les guillemets.

- `Multiline` : booléen.

  Une valeur booléenne qui indique si un même registre peut couvrir plusieurs lignes. Cela peut se produire lorsqu'un champ contient un caractère de nouvelle ligne. Vous devez définir cette option sur « VRAI » si aucun registre ne s'étend sur plusieurs lignes. La valeur par défaut est `False`, qui permet un fractionnement en fichiers plus intense pendant l'analyse.

- `WithHeader` : booléen.

  Une valeur booléenne qui indique s'il convient de traiter la première ligne comme un en-tête. La valeur par défaut est `False`.

- `WriteHeader` : booléen.

  Une valeur booléenne qui indique s'il faut écrire l'en-tête dans la sortie. La valeur par défaut est `True`.

- `SkipFirst` : booléen.

  Une valeur booléenne qui indique s'il faut ignorer la première ligne de données. La valeur par défaut est `False`.

- `OptimizePerformance` : booléen.

  Une valeur booléenne qui indique s'il faut utiliser le lecteur CSV SIMD avancé avec les formats de mémoire en colonnes basés sur Apache Arrow. Disponible uniquement dans AWS Glue la version 3.0.

- `OutputSchemas` : un tableau d'objets GlueSchema.

  Indique le schéma de données de la source S3 CSV.

JDBCSource Structure directe

Indique la connexion directe à la source JDBC.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom de la connexion à la source JDBC.
- `Database` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Base de données de la connexion à la source JDBC.
- `Table` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Table de la connexion à la source JDBC.
- `ConnectionName` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Nom de connexion de la source JDBC.
- `ConnectionType` – Obligatoire : Chaîne UTF-8 (valeurs valides : `sqlserver` | `mysql` | `oracle` | `postgresql` | `redshift`).

  Type de connexion de la source JDBC.
- `RedshiftTmpDir` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Répertoire temporaire de la source JDBC Redshift.
- `OutputSchemas` : un tableau d'objets GlueSchema.

  Indique le schéma de données de la source JDBC directe.

DirectSourceAdditionalOptions Structure S3

Indique des options de connexion supplémentaires pour le stocker de données Amazon S3.

Champs

- `BoundedSize` – Nombre (long).

  Définit la limite supérieure de la dimension cible du jeu de données en octets à traiter.
- `BoundedFiles` – Nombre (long).

  Définit la limite supérieure du nombre cible de fichiers à traiter.
- `EnableSamplePath` : booléen.

Définit l'option d'activation d'un exemple de chemin.

- `SamplePath` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Si cette option est activée, elle indique l'exemple de chemin.

JsonSource Structure S3

Indique un stocker de données JSON stocké dans Amazon S3.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom du stocker de données.

- `Paths` – Obligatoire : Tableau de chaînes UTF-8.

  Une liste de chemins Amazon S3 à lire.

- `CompressionType` – Chaîne UTF-8 (valeurs valides : `gzip="GZIP"` | `bzip2="BZIP2"`).

  Indique la manière dont les données sont comprimées. Ce n'est généralement pas nécessaire si le fichier de données a une extension standard. Les valeurs possibles sont `"gzip"` et `"bzip"`).

- `Exclusions` – Tableau de chaînes UTF-8.

  Une chaîne contenant une liste JSON des modèles glob de style Unix à exclure. Par exemple, [\"**.pdf\"] permet d'exclure tous les fichiers PDF.

- `GroupSize` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  La dimension du groupe cible, en octets. La valeur par défaut est calculée en fonction de la dimension des données en entrée et de la dimension de votre cluster. Lorsqu'il y a moins de 50 000 fichiers en entrée, `"groupFiles"` doit être défini sur `"inPartition"` pour que cela prenne effet.

- `GroupFiles` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le groupement de fichiers est activé par défaut lorsque l'entrée contient plus de 50 000 fichiers. Pour activer le groupement lorsqu'il y a moins de 50 000 fichiers, définissez ce paramètre sur « inPartition ». Pour désactiver le groupement lorsqu'il y a plus de 50 000 fichiers, définissez ce paramètre sur `"none"`.

- `Recurse` : booléen.

  Si ce paramètre est défini sur « VRAI », les fichiers sont lus de manière récursive dans tous les sous-répertoires des chemins spécifiés.

- `MaxBand` – Nombre (entier), pas plus qu'Aucun.

  Cette option permet de contrôler la durée, en millisecondes, au delà de laquelle la liste S3 est susceptible d'être cohérente. Les fichiers dont l'horodatage des modifications se situe dans les dernières millisecondes de MaxBand sont suivis, en particulier lors de leur utilisation, afin de tenir compte de la cohérence éventuelle JobBookmarks d'Amazon S3. La plupart des utilisateurs n'ont pas besoin de définir cette option. La valeur par défaut est 900 000 millisecondes, soit 15 minutes.

- `MaxFilesInBand` – Nombre (entier), pas plus qu'Aucun.

  Cette option Indique le nombre maximal de fichiers à enregistrer à partir des dernières secondes maxBand. Si ce nombre est dépassé, les fichiers supplémentaires sont ignorés et traités dans l'exécution de tâche suivante.

- `AdditionalOptions` – Un objet S3 DirectSourceAdditionalOptions.

  Indique des options de connexion supplémentaires.

- `JsonPath` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  JsonPath Chaîne définissant les données JSON.

- `Multiline` : booléen.

  Une valeur booléenne qui indique si un même registre peut couvrir plusieurs lignes. Cela peut se produire lorsqu'un champ contient un caractère de nouvelle ligne. Vous devez définir cette option sur « VRAI » si aucun registre ne s'étend sur plusieurs lignes. La valeur par défaut est `False`, qui permet un fractionnement en fichiers plus intense pendant l'analyse.

- `OutputSchemas` : un tableau d'objets GlueSchema.

  Indique le schéma de données de la source S3 JSON.

ParquetSource Structure S3

Indique un stocker de données Apache Parquet stocké dans Amazon S3.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom du stocker de données.

- `Paths` – Obligatoire : Tableau de chaînes UTF-8.

  Une liste de chemins Amazon S3 à lire.

- `CompressionType` – Chaîne UTF-8 (valeurs valides: `snappy="SNAPPY"` | `lzo="LZO"` | `gzip="GZIP"` | `brotli="BROTLI"` | `lz4="LZ4"` | `uncompressed="UNCOMPRESSED"` | `none="NONE"`).

  Indique la manière dont les données sont comprimées. Ce n'est généralement pas nécessaire si le fichier de données a une extension standard. Les valeurs possibles sont `"gzip"` et `"bzip"`).

- `Exclusions` – Tableau de chaînes UTF-8.

  Une chaîne contenant une liste JSON des modèles glob de style Unix à exclure. Par exemple, [\"**.pdf\"] permet d'exclure tous les fichiers PDF.

- `GroupSize` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  La dimension du groupe cible, en octets. La valeur par défaut est calculée en fonction de la dimension des données en entrée et de la dimension de votre cluster. Lorsqu'il y a moins de 50 000 fichiers en entrée, `"groupFiles"` doit être défini sur `"inPartition"` pour que cela prenne effet.

- `GroupFiles` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le groupement de fichiers est activé par défaut lorsque l'entrée contient plus de 50 000 fichiers. Pour activer le groupement lorsqu'il y a moins de 50 000 fichiers, définissez ce paramètre sur « inPartition ». Pour désactiver le groupement lorsqu'il y a plus de 50 000 fichiers, définissez ce paramètre sur `"none"`.

- `Recurse` : booléen.

  Si ce paramètre est défini sur « VRAI », les fichiers sont lus de manière récursive dans tous les sous-répertoires des chemins spécifiés.

- `MaxBand` – Nombre (entier), pas plus qu'Aucun.

  Cette option permet de contrôler la durée, en millisecondes, au delà de laquelle la liste S3 est susceptible d'être cohérente. Les fichiers dont l'horodatage des modifications se situe dans les

dernières millisecondes de MaxBand sont suivis, en particulier lors de leur utilisation, afin de tenir compte de la cohérence éventuelle JobBookmarks d'Amazon S3. La plupart des utilisateurs n'ont pas besoin de définir cette option. La valeur par défaut est 900 000 millisecondes, soit 15 minutes.

- `MaxFilesInBand` – Nombre (entier), pas plus qu'Aucun.

  Cette option Indique le nombre maximal de fichiers à enregistrer à partir des dernières secondes maxBand. Si ce nombre est dépassé, les fichiers supplémentaires sont ignorés et traités dans l'exécution de tâche suivante.

- `AdditionalOptions` – Un objet S3 DirectSourceAdditionalOptions.

  Indique des options de connexion supplémentaires.

- `OutputSchemas` : un tableau d'objets GlueSchema.

  Indique le schéma de données de la source S3 Parquet.

DeltaSource Structure S3

Spécifie une source de données Delta Lake stockée dans Amazon S3.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Le nom de la source de Delta Lake.

- `Paths` – Obligatoire : Tableau de chaînes UTF-8.

  Une liste de chemins Amazon S3 à lire.

- `AdditionalDeltaOptions` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, correspondant au Custom string pattern #61.

  Chaque valeur est une chaîne UTF-8, correspondant au Custom string pattern #61.

  Indique des options de connexion supplémentaires.

- `AdditionalOptions` – Un objet S3 DirectSourceAdditionalOptions.

  Indique les options supplémentaires du connecteur.

- `OutputSchemas` : un tableau d'objets GlueSchema.

Indique le schéma de données de la source Delta Lake.

CatalogDeltaSource Structure S3

Spécifie une source de données Delta Lake enregistrée dans le catalogue de AWS Glue données. La source de données doit être stockée dans Amazon S3.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Le nom de la source de données Delta Lake.
- `Database` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la base de données à partir de laquelle lire les données.
- `Table` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la table dans la base de données à partir de laquelle lire les données.
- `AdditionalDeltaOptions` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, correspondant au Custom string pattern #61.

  Chaque valeur est une chaîne UTF-8, correspondant au Custom string pattern #61.

  Indique des options de connexion supplémentaires.
- `OutputSchemas` : un tableau d'objets GlueSchema.

  Indique le schéma de données de la source Delta Lake.

CatalogDeltaSource structure

Spécifie une source de données Delta Lake enregistrée dans le catalogue de AWS Glue données.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Le nom de la source de données Delta Lake.

- `Database` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la base de données à partir de laquelle lire les données.
- `Table` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la table dans la base de données à partir de laquelle lire les données.
- `AdditionalDeltaOptions` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, correspondant au Custom string pattern #61.

  Chaque valeur est une chaîne UTF-8, correspondant au Custom string pattern #61.

  Indique des options de connexion supplémentaires.
- `OutputSchemas` : un tableau d'objets GlueSchema.

  Indique le schéma de données de la source Delta Lake.

HudiSource Structure S3

Spécifie une source de données Hudi stockée dans. Amazon S3

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom de la source Hudi.
- `Paths` – Obligatoire : Tableau de chaînes UTF-8.

  Une liste de chemins Amazon S3 à lire.
- `AdditionalHudiOptions` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, correspondant au Custom string pattern #61.

  Chaque valeur est une chaîne UTF-8, correspondant au Custom string pattern #61.

  Indique des options de connexion supplémentaires.
- `AdditionalOptions` – Un objet S3 DirectSourceAdditionalOptions.

  Indique les options supplémentaires du connecteur.
- `OutputSchemas` : un tableau d'objets GlueSchema.

Indique le schéma de données de la source Hudi.

CatalogHudiSource Structure S3

Spécifie une source de données Hudi enregistrée dans le catalogue de AWS Glue données. La source de données Hudi doit être stockée dans Amazon S3.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom de la source de données Hudi.
- `Database` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la base de données à partir de laquelle lire les données.
- `Table` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la table dans la base de données à partir de laquelle lire les données.
- `AdditionalHudiOptions` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, correspondant au Custom string pattern #61.

  Chaque valeur est une chaîne UTF-8, correspondant au Custom string pattern #61.

  Indique des options de connexion supplémentaires.
- `OutputSchemas` : un tableau d'objets GlueSchema.

  Indique le schéma de données de la source Hudi.

ExcelSource Structure S3

Spécifie une source de données S3 Excel.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom de la source de données S3 Excel.

- `Paths` – Obligatoire : Tableau de chaînes UTF-8.

  Chemins S3 où se trouvent les fichiers Excel.

- `CompressionType` – Chaîne UTF-8 (valeurs valides: `snappy="SNAPPY"` | `lzo="LZO"` | `gzip="GZIP"` | `brotli="BROTLI"` | `lz4="LZ4"` | `uncompressed="UNCOMPRESSED"` | `none="NONE"`).

  Format de compression utilisé pour les fichiers Excel.

- `Exclusions` – Tableau de chaînes UTF-8.

  Modèles permettant d'exclure des fichiers ou des chemins spécifiques du traitement.

- `GroupSize` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Définit la taille des groupes de fichiers pour le traitement par lots.

- `GroupFiles` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Spécifie la manière dont les fichiers doivent être regroupés pour être traités.

- `Recurse` : booléen.

  Indique s'il faut traiter les sous-répertoires de manière récursive.

- `MaxBand` – Nombre (entier), pas plus qu'Aucun.

  Nombre maximal de bandes de traitement à utiliser.

- `MaxFilesInBand` – Nombre (entier), pas plus qu'Aucun.

  Nombre maximal de fichiers à traiter dans chaque bande.

- `AdditionalOptions` – Un objet S3 DirectSourceAdditionalOptions.

  Options de configuration supplémentaires pour le traitement direct des sources dans S3.

- `NumberRows` – Nombre (long).

  Nombre de lignes à traiter dans chaque fichier Excel.

- `SkipFooter` – Nombre (entier), pas plus qu'Aucun.

  Nombre de lignes à ignorer à la fin de chaque fichier Excel.

- `OutputSchemas` : un tableau d'objets GlueSchema.

  Les AWS Glue schémas à appliquer aux données traitées.

CatalogHudiSource structure

Spécifie une source de données Hudi enregistrée dans le catalogue de AWS Glue données.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom de la source de données Hudi.
- `Database` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la base de données à partir de laquelle lire les données.
- `Table` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la table dans la base de données à partir de laquelle lire les données.
- `AdditionalHudiOptions` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, correspondant au Custom string pattern #61.

  Chaque valeur est une chaîne UTF-8, correspondant au Custom string pattern #61.

  Indique des options de connexion supplémentaires.
- `OutputSchemas` : un tableau d'objets GlueSchema.

  Indique le schéma de données de la source Hudi.

Structure de Dynamo DBCatalog Source

Spécifie une source de données DynamoDB dans AWS Glue le catalogue de données.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom de la source de données.
- `Database` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la base de données à partir de laquelle lire les données.
- `Table` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

Le nom de la table dans la base de données à partir de laquelle lire les données.

- `PitrEnabled` : booléen.

  Spécifie si Point-in-Time la restauration (PITR) est activée pour la table DynamoDB. Lorsque ce paramètre est défini sur `true`, cela permet la lecture à un moment précis dans le temps. La valeur par défaut est `false`.
- `AdditionalOptions` – Un objet DDBELTCatalogAdditionalOptions.

  Indique des options de connexion supplémentaires pour la source de données DynamoDB.

RelationalCatalogSource structure

Indique une source de données de base de données relationnelle dans le catalogue de données AWS Glue .

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom de la source de données.
- `Database` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la base de données à partir de laquelle lire les données.
- `Table` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la table dans la base de données à partir de laquelle lire les données.

JDBCConnectorStructure de la cible

Indique une cible de données qui écrit sur Amazon S3 dans un stockage en colonnes Apache Parquet.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Le nom de la cible de données.
- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 1 chaîne maximum.

Les nœuds qui constituent des entrées pour la cible de données.

- `ConnectionName` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la connexion associée au connecteur.

- `ConnectionTable` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la table dans la cible de données.

- `ConnectorName` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom d'un connecteur qui sera utilisé.

- `ConnectionType` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Type de connexion, tel que marketplace.jdbc ou custom.jdbc, désignant une connexion à une cible de données JDBC.

- `AdditionalOptions` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, correspondant au Custom string pattern #61.

  Chaque valeur est une chaîne UTF-8, correspondant au Custom string pattern #61.

  Options de connexion supplémentaires pour le connecteur.

- `OutputSchemas` : un tableau d'objets GlueSchema.

  Indique le schéma de données de la cible JDBC.

SparkConnectorTarget structure

Indique une cible qui utilise un connecteur Apache Spark.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Le nom de la cible de données.

- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 1 chaîne maximum.

  Les nœuds qui constituent des entrées pour la cible de données.

- `ConnectionName` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

Le nom d'une connexion pour un connecteur Apache Spark.

- `ConnectorName` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom d'un connecteur Apache Spark.

- `ConnectionType` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Type de connexion, tel que marketplace.spark ou custom.spark, désignant une connexion à un stocker de données Apache Spark.

- `AdditionalOptions` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, correspondant au Custom string pattern #61.

  Chaque valeur est une chaîne UTF-8, correspondant au Custom string pattern #61.

  Options de connexion supplémentaires pour le connecteur.

- `OutputSchemas` : un tableau d'objets GlueSchema.

  Indique le schéma de données de la cible Spark personnalisée.

BasicCatalogTarget structure

Spécifie une cible qui utilise une table AWS Glue de catalogue de données.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Le nom de votre cible de données.

- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 1 chaîne maximum.

  Les nœuds qui constituent des entrées pour la cible de données.

- `PartitionKeys` – Tableau de chaînes UTF-8.

  Les clés de partition utilisées pour distribuer les données sur plusieurs partitions en fonction d'une clé ou d'un ensemble de clés spécifique.

- `Database` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

La base de données où se trouve la table que vous souhaitez utiliser comme cible. Cette base de données doit déjà exister dans le catalogue de données.

- `Table` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  La table qui définit le schéma de vos données de sortie. Cette table doit déjà exister dans le catalogue de données..

Structure de My SQLCatalog Target

Indique une cible qui utilise MySQL.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Le nom de la cible de données.
- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 1 chaîne maximum.

  Les nœuds qui constituent des entrées pour la cible de données.
- `Database` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la base de données dans laquelle écrire les données.
- `Table` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la table de la base de données dans laquelle écrire les données.

Structure de SQLCatalog PostgreTarget

Indique une cible qui utilise Postgre SQL.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Le nom de la cible de données.
- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 1 chaîne maximum.

  Les nœuds qui constituent des entrées pour la cible de données.

- `Database` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la base de données dans laquelle écrire les données.
- `Table` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la table de la base de données dans laquelle écrire les données.

Structure d'Oracle SQLCatalog Target

Indique une cible qui utilise Oracle SQL.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Le nom de la cible de données.
- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 1 chaîne maximum.

  Les nœuds qui constituent des entrées pour la cible de données.
- `Database` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la base de données dans laquelle écrire les données.
- `Table` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la table de la base de données dans laquelle écrire les données.

SQLServerCatalogTarget Structure de Microsoft

Indique une cible qui utilise Microsoft SQL.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Le nom de la cible de données.
- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 1 chaîne maximum.

  Les nœuds qui constituent des entrées pour la cible de données.
- `Database` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

Le nom de la base de données dans laquelle écrire les données.

- `Table` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la table de la base de données dans laquelle écrire les données.

RedshiftTarget structure

Indique une cible qui utilise Amazon Redshift.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Le nom de la cible de données.

- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 1 chaîne maximum.

  Les nœuds qui constituent des entrées pour la cible de données.

- `Database` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la base de données dans laquelle écrire les données.

- `Table` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la table de la base de données dans laquelle écrire les données.

- `RedshiftTmpDir` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Chemin Amazon S3 où les données temporaires peuvent être stockées lors de la copie à partir de la base de données.

- `TmpDirIAMRole` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le rôle IAM avec les autorisations.

- `UpsertRedshiftOptions` : un objet UpsertRedshiftTargetOptions.

  Jeu d'options permettant de configurer une opération de mise à jour/insertion lors de l'écriture vers une cible Redshift.

AmazonRedshiftTarget structure

Indique une cible Amazon Redshift.

Champs

- `Name` – Chaîne UTF-8, correspondant au Custom string pattern #63.

  Le nom de la cible Amazon Redshift.
- `Data` : un objet AmazonRedshiftNodeData.

  Indique les données du nœud cible Amazon Redshift.
- `Inputs` : tableau de chaînes UTF-8, avec une chaîne minimum et une chaîne maximum.

  Les nœuds qui constituent des entrées pour la cible de données.

UpsertRedshiftTargetOptions structure

Options permettant de configurer une opération de mise à jour/insertion lors de l'écriture vers une cible Redshift.

Champs

- `TableLocation` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Emplacement physique de la table Redshift.
- `ConnectionName` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Nom de la connexion à utiliser pour écrire dans Redshift.
- `UpsertKeys` – Tableau de chaînes UTF-8.

  Clés utilisées pour déterminer si une opération de mise à jour ou d'insertion est nécessaire.

CatalogTarget Structure S3

Spécifie une cible de données qui écrit sur Amazon S3 à l'aide du catalogue de AWS Glue données.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

Le nom de la cible de données.

- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 1 chaîne maximum.

  Les nœuds qui constituent des entrées pour la cible de données.

- `PartitionKeys` – Tableau de chaînes UTF-8.

  Indique le partitionnement natif à l'aide d'une séquence de clés.

- `Table` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la table de la base de données dans laquelle écrire les données.

- `Database` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la base de données dans laquelle écrire les données.

- `SchemaChangePolicy` – Un objet CatalogSchemaChangePolicy.

  Une politique qui indique les évolutions de mise à jour pour le crawler.

- `AutoDataQuality` : un objet AutoDataQuality.

  Indique s'il faut activer automatiquement l'évaluation de la qualité des données pour la cible du catalogue S3. Lorsque ce paramètre est défini sur `true`, les contrôles de qualité des données sont effectués automatiquement pendant l'opération d'écriture.

GlueParquetTarget Structure S3

Indique une cible de données qui écrit sur Amazon S3 dans un stockage en colonnes Apache Parquet.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Le nom de la cible de données.

- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 1 chaîne maximum.

  Les nœuds qui constituent des entrées pour la cible de données.

- `PartitionKeys` – Tableau de chaînes UTF-8.

  Indique le partitionnement natif à l'aide d'une séquence de clés.

- `Path` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Un seul chemin Amazon S3 sur lequel écrire.

- `Compression` – Chaîne UTF-8 (valeurs valides: `snappy="SNAPPY"` | `lzo="LZO"` | `gzip="GZIP"` | `brotli="BROTLI"` | `lz4="LZ4"` | `uncompressed="UNCOMPRESSED"` | `none="NONE"`).

  Indique la manière dont les données sont comprimées. Ce n'est généralement pas nécessaire si le fichier de données a une extension standard. Les valeurs possibles sont `"gzip"` et `"bzip"`).

- `NumberTargetPartitions` – Chaîne UTF-8.

  Indique le nombre de partitions cible pour les fichiers Parquet lorsque vous écrivez vers Amazon S3 à l'aide d' AWS Glue.

- `SchemaChangePolicy` – Un objet DirectSchemaChangePolicy.

  Une politique qui indique les évolutions de mise à jour pour le crawler.

- `AutoDataQuality` : un objet AutoDataQuality.

  Spécifie s'il faut activer automatiquement l'évaluation de la qualité des données pour la cible S3 AWS Glue Parquet. Lorsque ce paramètre est défini sur `true`, les contrôles de qualité des données sont effectués automatiquement pendant l'opération d'écriture.

CatalogSchemaChangePolicy structure

Une politique qui indique des comportements de mise à jour pour l'crawler.

Champs

- `EnableUpdateCatalog` : booléen.

  S'il faut utiliser ou non le comportement de mise à jour spécifié lorsque l'crawler détecte un schéma modifié.

- `UpdateBehavior` – Chaîne UTF-8 (valeurs valides : `UPDATE_IN_DATABASE` | `LOG`).

  Comportement de mise à jour lorsque le crawler détecte un schéma modifié.

DirectTarget Structure S3

Indique une cible de données qui écrit dans Amazon S3.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Le nom de la cible de données.

- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 1 chaîne maximum.

  Les nœuds qui constituent des entrées pour la cible de données.

- `PartitionKeys` – Tableau de chaînes UTF-8.

  Indique le partitionnement natif à l'aide d'une séquence de clés.

- `Path` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Un seul chemin Amazon S3 sur lequel écrire.

- `Compression` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Indique la manière dont les données sont comprimées. Ce n'est généralement pas nécessaire si le fichier de données a une extension standard. Les valeurs possibles sont `"gzip"` et `"bzip"`).

- `NumberTargetPartitions` – Chaîne UTF-8.

  Indique le nombre de partitions cible lorsque vous écrivez directement vers Amazon S3.

- `Format` – Obligatoire : chaîne UTF-8 (valeurs valides : `json="JSON"` | `csv="CSV"` | `avro="AVRO"` | `orc="ORC"` | `parquet="PARQUET"` | `hudi="HUDI"` | `delta="DELTA"` | `iceberg="ICEBERG"` | `hyper="HYPER"` | `xml="XML"`).

  Définit le format de sortie des données pour la cible.

- `SchemaChangePolicy` – Un objet DirectSchemaChangePolicy.

  Une politique qui indique les évolutions de mise à jour pour le crawler.

- `AutoDataQuality` : un objet AutoDataQuality.

  Indique s'il faut activer automatiquement l'évaluation de la qualité des données pour la cible directe S3. Lorsque ce paramètre est défini sur `true`, les contrôles de qualité des données sont effectués automatiquement pendant l'opération d'écriture.

- `OutputSchemas` : un tableau d'objets GlueSchema.

  Indique le schéma de données de la cible directe S3.

HudiCatalogTarget Structure S3

Spécifie une cible qui écrit dans une source de données Hudi du catalogue de AWS Glue données.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Le nom de la cible de données.
- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 1 chaîne maximum.

  Les nœuds qui constituent des entrées pour la cible de données.
- `PartitionKeys` – Tableau de chaînes UTF-8.

  Indique le partitionnement natif à l'aide d'une séquence de clés.
- `Table` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la table de la base de données dans laquelle écrire les données.
- `Database` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la base de données dans laquelle écrire les données.
- `AdditionalOptions` – obligatoire : tableau de mappage de paires clé-valeur.

  Chaque clé est une chaîne UTF-8, correspondant au Custom string pattern #61.

  Chaque valeur est une chaîne UTF-8, correspondant au Custom string pattern #61.

  Indique des options de connexion supplémentaires pour le connecteur.
- `SchemaChangePolicy` – Un objet CatalogSchemaChangePolicy.

  Une politique qui indique les évolutions de mise à jour pour le crawler.
- `AutoDataQuality` : un objet AutoDataQuality.

Indique s'il faut activer automatiquement l'évaluation de la qualité des données pour la cible du catalogue S3 Hudi. Lorsque ce paramètre est défini sur `true`, les contrôles de qualité des données sont effectués automatiquement pendant l'opération d'écriture.

- `OutputSchemas` : un tableau d'objets GlueSchema.

  Indique le schéma de données de la cible du catalogue S3 Hudi.

HudiDirectTarget Structure S3

Spécifie une cible qui écrit dans une source de données Hudi en Amazon S3.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Le nom de la cible de données.
- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 1 chaîne maximum.

  Les nœuds qui constituent des entrées pour la cible de données.
- `Path` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le chemin d'accès Amazon S3 de votre source de données Hudi sur laquelle écrire.
- `Compression` – Obligatoire :Chaîne UTF-8 (valeurs valides : `gzip="GZIP"` | `lzo="LZO"` | `uncompressed="UNCOMPRESSED"` | `snappy="SNAPPY"`).

  Indique la manière dont les données sont comprimées. Ce n'est généralement pas nécessaire si le fichier de données a une extension standard. Les valeurs possibles sont `"gzip"` et `"bzip"`).
- `NumberTargetPartitions` – Chaîne UTF-8.

  Indique le nombre de partitions cible pour distribuer les fichiers de jeux de données Hudi sur Amazon S3.
- `PartitionKeys` – Tableau de chaînes UTF-8.

  Indique le partitionnement natif à l'aide d'une séquence de clés.
- `Format` – Obligatoire : chaîne UTF-8 (valeurs valides : `json="JSON"` | `csv="CSV"` | `avro="AVRO"` | `orc="ORC"` | `parquet="PARQUET"` | `hudi="HUDI"` | `delta="DELTA"` | `iceberg="ICEBERG"` | `hyper="HYPER"` | `xml="XML"`).

Définit le format de sortie des données pour la cible.

- `AdditionalOptions` – obligatoire : tableau de mappage de paires clé-valeur.

  Chaque clé est une chaîne UTF-8, correspondant au Custom string pattern #61.

  Chaque valeur est une chaîne UTF-8, correspondant au Custom string pattern #61.

  Indique des options de connexion supplémentaires pour le connecteur.

- `SchemaChangePolicy` – Un objet DirectSchemaChangePolicy.

  Une politique qui indique les évolutions de mise à jour pour le crawler.

- `AutoDataQuality` : un objet AutoDataQuality.

  Indique s'il faut activer automatiquement l'évaluation de la qualité des données pour la cible directe S3 Hudi. Lorsque ce paramètre est défini sur `true`, les contrôles de qualité des données sont effectués automatiquement pendant l'opération d'écriture.

DeltaCatalogTarget Structure S3

Spécifie une cible qui écrit dans une source de données Delta Lake dans le catalogue de AWS Glue données.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Le nom de la cible de données.

- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 1 chaîne maximum.

  Les nœuds qui constituent des entrées pour la cible de données.

- `PartitionKeys` – Tableau de chaînes UTF-8.

  Indique le partitionnement natif à l'aide d'une séquence de clés.

- `Table` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la table de la base de données dans laquelle écrire les données.

- `Database` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

Le nom de la base de données dans laquelle écrire les données.

- `AdditionalOptions` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, correspondant au Custom string pattern #61.

  Chaque valeur est une chaîne UTF-8, correspondant au Custom string pattern #61.

  Indique des options de connexion supplémentaires pour le connecteur.

- `SchemaChangePolicy` – Un objet CatalogSchemaChangePolicy.

  Une politique qui indique les évolutions de mise à jour pour le crawler.

- `AutoDataQuality` : un objet AutoDataQuality.

  Indique s'il faut activer automatiquement l'évaluation de la qualité des données pour la cible du catalogue S3 Delta. Lorsque ce paramètre est défini sur `true`, les contrôles de qualité des données sont effectués automatiquement pendant l'opération d'écriture.

- `OutputSchemas` : un tableau d'objets GlueSchema.

  Indique le schéma de données de la cible du catalogue S3 Delta.

DeltaDirectTarget Structure S3

Spécifie une cible qui écrit dans une source de données de Delta Lake en Amazon S3.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Le nom de la cible de données.

- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 1 chaîne maximum.

  Les nœuds qui constituent des entrées pour la cible de données.

- `PartitionKeys` – Tableau de chaînes UTF-8.

  Indique le partitionnement natif à l'aide d'une séquence de clés.

- `Path` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le chemin d'accès Amazon S3 de votre source de données Delta Lake sur laquelle écrire.

- `Compression` – Obligatoire : Chaîne UTF-8 (valeurs valides : `uncompressed="UNCOMPRESSED"` | `snappy="SNAPPY"`).

  Indique la manière dont les données sont comprimées. Ce n'est généralement pas nécessaire si le fichier de données a une extension standard. Les valeurs possibles sont `"gzip"` et `"bzip"`).

- `NumberTargetPartitions` – Chaîne UTF-8.

  Indique le nombre de partitions cible pour distribuer les fichiers de jeux de données Delta Lake sur Amazon S3.

- `Format` – Obligatoire : chaîne UTF-8 (valeurs valides : `json="JSON"` | `csv="CSV"` | `avro="AVRO"` | `orc="ORC"` | `parquet="PARQUET"` | `hudi="HUDI"` | `delta="DELTA"` | `iceberg="ICEBERG"` | `hyper="HYPER"` | `xml="XML"`).

  Définit le format de sortie des données pour la cible.

- `AdditionalOptions` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, correspondant au Custom string pattern #61.

  Chaque valeur est une chaîne UTF-8, correspondant au Custom string pattern #61.

  Indique des options de connexion supplémentaires pour le connecteur.

- `SchemaChangePolicy` – Un objet DirectSchemaChangePolicy.

  Une politique qui indique les évolutions de mise à jour pour le crawler.

- `AutoDataQuality` : un objet AutoDataQuality.

  Indique s'il faut activer automatiquement l'évaluation de la qualité des données pour la cible directe S3 Delta. Lorsque ce paramètre est défini sur `true`, les contrôles de qualité des données sont effectués automatiquement pendant l'opération d'écriture.

HyperDirectTarget Structure S3

Spécifie une cible de HyperDirect données qui écrit sur Amazon S3.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Identifiant unique du nœud HyperDirect cible.

- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 1 chaîne maximum.

  Spécifie la source d'entrée pour la HyperDirect cible.

- `Format`— Chaîne UTF-8 (valeurs valides : `json="JSON"` | | `csv="CSV"` | `avro="AVRO"` | `orc="ORC"` | `parquet="PARQUET"` | `hudi="HUDI"` | `delta="DELTA"` `iceberg="ICEBERG"` `hyper="HYPER"` |`xml="XML"`).

  Spécifie le format de sortie des données pour la HyperDirect cible.

- `PartitionKeys` – Tableau de chaînes UTF-8.

  Définit la stratégie de partitionnement des données de sortie.

- `Path` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Emplacement S3 dans lequel les données de sortie seront écrites.

- `Compression` – Chaîne UTF-8 (valeurs valides : `uncompressed="UNCOMPRESSED"`).

  Type de compression à appliquer aux données de sortie.

- `SchemaChangePolicy` – Un objet DirectSchemaChangePolicy.

  Définit la manière dont les modifications de schéma sont gérées lors des opérations d'écriture.

- `AutoDataQuality` : un objet AutoDataQuality.

  Indique s'il faut activer automatiquement l'évaluation de la qualité des données pour la cible directe S3 Hyper. Lorsque ce paramètre est défini sur `true`, les contrôles de qualité des données sont effectués automatiquement pendant l'opération d'écriture.

- `OutputSchemas` : un tableau d'objets GlueSchema.

  Indique le schéma de données de la cible directe S3 Hyper.

IcebergDirectTarget Structure S3

Indique une cible qui écrit dans une source de données Iceberg de Amazon S3.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Indique l'identifiant unique du nœud cible Iceberg dans votre pipeline de données.

- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 1 chaîne maximum.

  Définit la source d'entrée unique qui fournit des données à cette cible Iceberg.

- `PartitionKeys` – Tableau de chaînes UTF-8.

  Indique les colonnes utilisées pour partitionner les données de la table Iceberg dans S3.

- `Path` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Indique l'emplacement S3 dans lequel les données de la table Iceberg seront stockées.

- `Format` – Obligatoire : chaîne UTF-8 (valeurs valides : `json="JSON"` | `csv="CSV"` | `avro="AVRO"` | `orc="ORC"` | `parquet="PARQUET"` | `hudi="HUDI"` | `delta="DELTA"` | `iceberg="ICEBERG"` | `hyper="HYPER"` | `xml="XML"`).

  Indique le format de fichier utilisé pour stocker les données des tables Iceberg (par exemple, Parquet, ORC).

- `AdditionalOptions` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, correspondant au Custom string pattern #61.

  Chaque valeur est une chaîne UTF-8, correspondant au Custom string pattern #61.

  Fournit des options de configuration supplémentaires pour personnaliser le comportement de la table Iceberg.

- `SchemaChangePolicy` – Un objet DirectSchemaChangePolicy.

  Définit la manière dont les modifications de schéma sont gérées lors de l'écriture de données dans la table Iceberg.

- `Compression` – Obligatoire :Chaîne UTF-8 (valeurs valides : `gzip="GZIP"` | `lzo="LZO"` | `uncompressed="UNCOMPRESSED"` | `snappy="SNAPPY"`).

  Indique le codec de compression utilisé pour les fichiers de table Iceberg dans S3.

- `NumberTargetPartitions` – Chaîne UTF-8.

  Définit le nombre de partitions cible pour la distribution des fichiers de table Iceberg dans S3.

- `OutputSchemas` : un tableau d'objets GlueSchema.

  Indique le schéma de données de la cible directe S3 Iceberg.

DirectSchemaChangePolicy structure

Une politique qui indique des comportements de mise à jour pour l'crawler.

Champs

- `EnableUpdateCatalog` : booléen.

  S'il faut utiliser ou non le comportement de mise à jour spécifié lorsque l'crawler détecte un schéma modifié.

- `UpdateBehavior` – Chaîne UTF-8 (valeurs valides : `UPDATE_IN_DATABASE` | `LOG`).

  Comportement de mise à jour lorsque l'crawler détecte un schéma modifié.

- `Table` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Indique la table de la base de données à laquelle s'applique la politique de modification du schéma.

- `Database` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Indique la base de données à laquelle s'applique la politique de modification du schéma.

ApplyMapping structure

Indique une transformation qui mappe les clés de propriétés de données de la source de données aux clés de propriété de données de la cible de données. Vous pouvez renommer les clés, modifier leur type de données et choisir les clés à supprimer du jeu de données.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom du nœud de transformation.

- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 1 chaîne maximum.

  Les entrées de données identifiées par leurs noms de nœuds.

- `Mapping` – Obligatoire : un tableau d'objets Mappage.

  Indique le mappage des clés de propriétés de données de la source de données avec les clés de propriétés de données de la cible de données.

Structure de mappage

Indique le mappage des clés de propriété de données.

Champs

- `ToKey` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Après le mappage d'application, quel nom donner à la colonne. Peut être similaire à `FromPath`.

- `FromPath` – Tableau de chaînes UTF-8.

  La table ou la colonne à modifier.

- `FromType` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le type des données à modifier.

- `ToType` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le type de données sous lequel les données doivent être modifiées.

- `Dropped` : booléen.

  Si ce paramètre est défini sur « VRAI », la colonne est supprimée.

- `Children` : un tableau d'objets Mappage.

  S'applique uniquement aux structures de données imbriquées. Si vous souhaitez modifier la structure parente, mais également l'un de ses enfants, vous pouvez remplir cette structure de données. C'est aussi `Mapping`, mais son `FromPath` sera le `FromPath` du parent plus le `FromPath` provenant de cette structure.

  Pour la partie enfants, supposons que vous ayez la structure suivante :

```
{ "FromPath": "OuterStructure", "ToKey": "OuterStructure", "ToType":
"Struct", "Dropped": false, "Chidlren": [{ "FromPath": "inner", "ToKey":
"inner", "ToType": "Double", "Dropped": false, }] }
```

  Vous pouvez Indiquer un `Mapping` qui se présente sous la forme suivante :

```
{ "FromPath": "OuterStructure", "ToKey": "OuterStructure", "ToType":
"Struct", "Dropped": false, "Chidlren": [{ "FromPath": "inner", "ToKey":
"inner", "ToType": "Double", "Dropped": false, }] }
```

SelectFields structure

Indique une transformation qui choisit les clés de propriété de données que vous souhaitez conserver.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom du nœud de transformation.

- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 1 chaîne maximum.

  Les entrées de données identifiées par leurs noms de nœuds.

- `Paths` – Obligatoire : Tableau de chaînes UTF-8.

  Un chemin JSON vers une variable de la structure de données.

DropFields structure

Indique une transformation qui choisit les clés de propriété de données que vous souhaitez supprimer.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom du nœud de transformation.

- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 1 chaîne maximum.

  Les entrées de données identifiées par leurs noms de nœuds.

- `Paths` – Obligatoire : Tableau de chaînes UTF-8.

  Un chemin JSON vers une variable de la structure de données.

RenameField structure

Indique une transformation qui renomme une clé de propriété de données unique.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom du nœud de transformation.
- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 1 chaîne maximum.

  Les entrées de données identifiées par leurs noms de nœuds.
- `SourcePath` – Obligatoire : Tableau de chaînes UTF-8.

  Un chemin JSON vers une variable de la structure de données pour les données source.
- `TargetPath` – Obligatoire : Tableau de chaînes UTF-8.

  Un chemin JSON vers une variable de la structure de données pour les données cibles.

Structure Spigot

Indique une transformation qui écrit des échantillons de données dans un compartiment Amazon S3.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom du nœud de transformation.
- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 1 chaîne maximum.

  Les entrées de données identifiées par leurs noms de nœuds.
- `Path` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Un chemin dans Amazon S3 où la transformation écrira un sous-ensemble de registres à partir du jeu de données dans un fichier JSON situé dans un compartiment Amazon S3.
- `Topk` – Nombre (entier), 100 au maximum.

  Indique un certain nombre de registres à écrire à partir du début du jeu de données.
- `Prob` – Nombre (double), 1 au maximum.

  La probabilité (valeur décimale ayant une valeur maximale de 1) de prélèvement d'un registre donné. La valeur 1 indique que chaque ligne lue à partir du jeu de données doit être incluse dans l'exemple de sortie.

Structure Join

Indique une transformation qui joint deux jeux de données en un jeu de données à l'aide d'une phrase de comparaison sur les clés de propriété de données spécifiées. Vous pouvez utiliser des jointures internes (ou intérieures), externes (ou extérieures), gauche, droite, semi gauche et anti gauche.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Le nom du nœud de transformation.

- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 2 chaînes minimum et 2 chaînes maximum.

  Les entrées de données identifiées par leurs noms de nœuds.

- `JoinType` – Obligatoire : Chaîne UTF-8 (valeurs valides : `equijoin="EQUIJOIN"` | `left="LEFT"` | `right="RIGHT"` | `outer="OUTER"` | `leftsemi="LEFT_SEMI"` | `leftanti="LEFT_ANTI"`).

  Indique le type de jointure à effectuer sur les jeux de données.

- `Columns` – Obligatoire : Tableau d'objets JoinColumn, 2 structures minimum et 2 structures maximum.

  Liste des deux colonnes à joindre.

JoinColumn structure

Indique une colonne à joindre.

Champs

- `From` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  La colonne à joindre.

- `Keys` – Obligatoire : Tableau de chaînes UTF-8.

  La clé de la colonne à joindre.

SplitFields structure

Indique une transformation qui divise les clés de propriété de données en deux `DynamicFrames`. Le résultat est une collection de `DynamicFrames` : une avec les clés de propriété de données sélectionnées, et une autre avec les clés de propriété de données restantes.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom du nœud de transformation.
- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 1 chaîne maximum.

  Les entrées de données identifiées par leurs noms de nœuds.
- `Paths` – Obligatoire : Tableau de chaînes UTF-8.

  Un chemin JSON vers une variable de la structure de données.

SelectFromCollection structure

Indique une transformation qui en choisit une `DynamicFrame` provenant d'une collection de `DynamicFrames`. Le résultat est le `DynamicFrame` sélectionné

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom du nœud de transformation.
- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 1 chaîne maximum.

  Les entrées de données identifiées par leurs noms de nœuds.
- `Index` – Obligatoire : Nombre (entier), pas plus qu'Aucun.

  L'index du DynamicFrame à sélectionner.

FillMissingValues structure

Précise une transformation qui localise les registres dans le jeu de données dont les valeurs sont manquantes et ajoute un nouveau champ avec une valeur déterminée par imputation. Le jeu de

données source est utilisé pour entraîner le modèle de machine learning (ML) qui détermine la valeur manquante.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom du nœud de transformation.
- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 1 chaîne maximum.

  Les entrées de données identifiées par leurs noms de nœuds.
- `ImputedPath` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Un chemin JSON vers une variable de la structure de données pour le jeu de données attribué.
- `FilledPath` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Un chemin JSON vers une variable de la structure de données pour le jeu de données rempli.

Structure Filtre

Indique une transformation qui divise un jeu de données en deux, en fonction d'une condition de filtre.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom du nœud de transformation.
- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 1 chaîne maximum.

  Les entrées de données identifiées par leurs noms de nœuds.
- `LogicalOperator` – Obligatoire : Chaîne UTF-8 (valeurs valides : `AND` | `OR`).

  L'opérateur utilisé pour filtrer les lignes en comparant la valeur clé à une valeur spécifiée.
- `Filters` – Obligatoire : un tableau d'objets FilterExpression.

  Indique une expression de filtre.

FilterExpression structure

Indique une expression de filtre.

Champs

- `Operation` – Obligatoire : Chaîne UTF-8 (valeurs valides : `EQ` | `LT` | `GT` | `LTE` | `GTE` | `REGEX` | `ISNULL`).

  Le type d'opération à effectuer dans l'expression.
- `Negated` : booléen.

  Indique si l'expression doit être annulée.
- `Values` – Obligatoire : un tableau d'objets FilterValue.

  Une liste de valeurs de filtre.

FilterValue structure

Représente une entrée unique dans la liste de valeurs de `FilterExpression`.

Champs

- `Type` – Obligatoire : Chaîne UTF-8 (valeurs valides : `COLUMNEXTRACTED` | `CONSTANT`).

  Le type de valeur de filtre.
- `Value` – Obligatoire : Tableau de chaînes UTF-8.

  La valeur à associer.

CustomCode structure

Indique une transformation qui utilise le code personnalisé que vous fournissez pour effectuer la transformation des données. La sortie est une collection de DynamicFrames.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom du nœud de transformation.

- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, au moins 1 chaîne.

  Les entrées de données identifiées par leurs noms de nœuds.

- `Code` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #54.

  Le code personnalisé utilisé pour effectuer la transformation des données.

- `ClassName` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom défini pour la classe de nœuds de code personnalisée.

- `OutputSchemas` : un tableau d'objets GlueSchema.

  Indique le schéma de données de la transformation du code personnalisé.

Structure SparkSQL

Indique une transformation dans laquelle vous saisissez une requête SQL à l'aide de la syntaxe Spark SQL pour transformer les données. Le résultat est un `DynamicFrame` unique.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom du nœud de transformation.

- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, au moins 1 chaîne.

  Les entrées de données identifiées par leurs noms de nœuds. Vous pouvez associer un nom de table à chaque nœud d'entrée à utiliser dans la requête SQL. Le nom que vous choisissez doit respecter les restrictions de dénomination de Spark SQL.

- `SqlQuery` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #62.

  Une requête SQL qui doit utiliser la syntaxe Spark SQL et renvoyer un jeu de données unique.

- `SqlAliases` – Obligatoire : un tableau d'objets SqlAlias.

  Liste d'alias Un alias vous autorise de spécifier le nom à utiliser dans SQL pour une entrée donnée. Par exemple, vous avez une source de données nommée « MyDataSource ». Si vous spécifiez `From` as MyDataSource et `Alias` as SqlName, alors dans votre code SQL, vous pouvez faire :

```
select * from SqlName
```

et qui obtient des données de MyDataSource.

- `OutputSchemas` : un tableau d'objets GlueSchema.

  Indique le schéma de données de la transformation SparkSQL.

SqlAlias structure

Représente une entrée unique dans la liste de valeurs de `SqlAliases`.

Champs

- `From` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #60.

  Une table ou une colonne d'une table.
- `Alias` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Un nom temporaire donné à une table ou à une colonne d'une table.

DropNullFields structure

Indique une transformation qui supprime les colonnes du jeu de données si toutes les valeurs de la colonne sont « nulles ». Par défaut, AWS Glue Studio reconnaît les objets nuls, mais certaines valeurs telles que les chaînes vides, les chaînes « nulles », les entiers -1 ou d'autres espaces réservés tels que les zéros ne sont pas automatiquement reconnues comme nulles.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom du nœud de transformation.
- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 1 chaîne maximum.

  Les entrées de données identifiées par leurs noms de nœuds.
- `NullCheckBoxList` – Un objet NullCheckBoxList.

  Une structure qui indique si certaines valeurs sont reconnues comme des valeurs null en vue de la suppression.
- `NullTextList` – Un tableau d'objets NullValueField, 50 structures maximum.

Structure qui spécifie une liste de NullValueField structures représentant une valeur nulle personnalisée telle que zéro ou une autre valeur utilisée comme espace réservé nul propre à l'ensemble de données.

La transformation `DropNullFields` supprime les valeurs nulles personnalisées uniquement, si la valeur de l'espace réservé nul et du type de données correspondent aux données.

NullCheckBoxList structure

Indique si certaines valeurs sont reconnues comme des valeurs null en vue de la suppression.

Champs

- `IsEmpty` : booléen.

  Indique qu'une chaîne vide est considérée comme une valeur null.

- `IsNullString` : booléen.

  Indique qu'une valeur révélant le mot « null » est considérée comme une valeur null.

- `IsNegOne` : booléen.

  Indique qu'une valeur entière de -1 est considérée comme une valeur null.

NullValueField structure

Représente une valeur null personnalisée telle qu'un zéro ou une autre valeur utilisée comme espace réservé null unique pour le jeu de données.

Champs

- `Value` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  La valeur de l'espace réservé null.

- `Datatype` – Obligatoire : un objet Datatype.

  Le type de données de la valeur.

Structure Datatype

Une structure représentant le type de données de la valeur.

Champs

- `Id` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #60.

  Le type de données de la valeur.
- `Label` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #60.

  Une étiquette affectée au type de données.

Structure Fusion

Indique une transformation qui fusionne une `DynamicFrame` avec une `DynamicFrame` intermédiaire basée sur les clés primaires spécifiées pour identifier les registres. Les registres en double (registres avec les mêmes clés primaires) ne sont pas dédupliqués.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Le nom du nœud de transformation.
- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 2 chaînes minimum et 2 chaînes maximum.

  Les entrées de données identifiées par leurs noms de nœuds.
- `Source` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #60.

  La source `DynamicFrame` qui sera fusionnée avec une `DynamicFrame` intermédiaire.
- `PrimaryKeys` – Obligatoire : Tableau de chaînes UTF-8.

  La liste des champs de clé primaire permettant de faire correspondre les registres des trames dynamiques source et intermédiaire.

Structure Union

Indique une transformation qui combine les lignes de deux jeux de données ou plus en un seul résultat.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Le nom du nœud de transformation.
- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 2 chaîne minimum et 2 chaîne maximum.

  Les entrées de l'ID du nœud dans la transformation.
- `UnionType` – Obligatoire : Chaîne UTF-8 (valeurs valides : `ALL` | `DISTINCT`).

  Indique le type de transformation Union.

  Spécifiez `ALL` de joindre toutes les lignes des sources de données au résultat DynamicFrame. L'union qui en résulte ne supprime pas les lignes en double.

  Spécifiez `DISTINCT` de supprimer les lignes dupliquées dans le résultat DynamicFrame.

PIIDetection structure

Indique une transformation qui identifie, supprime ou masque les données d'informations personnelles identifiables (PII).

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom du nœud de transformation.
- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 1 chaîne maximum.

  Les entrées de l'ID du nœud dans la transformation.
- `PiiType` – Obligatoire : Chaîne UTF-8 (valeurs valides : `RowAudit` | `RowHashing` | `RowMasking` | `RowPartialMasking` | `ColumnAudit` | `ColumnHashing` | `ColumnMasking`).

  Indique le type de PIIDetection transformation.

- `EntityTypesToDetect` – Obligatoire : Tableau de chaînes UTF-8.

  Indique les types d'entités que la PIIDetection transformation identifiera en tant que données PII.

  Les entités de type PII comprennent : PERSON_NAME, DATE, USA_SNN, EMAIL, USA_ITIN, USA_PASSPORT_NUMBER, PHONE_NUMBER, BANK_ACCOUNT, IP_ADDRESS, MAC_ADDRESS, USA_CPT_CODE, USA_HCPCS_CODE, USA_NATIONAL_DRUG_CODE, USA_MEDICARE_BENEFICIARY_IDENTIFIER, USA_HEALTH_INSURANCE_CLAIM_NUMBER, CREDIT_CARD, USA_NATIONAL_PROVIDER_IDENTIFIER, USA_DEA_NUMBER, USA_DRIVING_LICENSE

- `OutputColumnName` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Indique le nom de la colonne de sortie qui contiendra tout type d'entité détecté dans cette ligne.

- `SampleFraction` – Nombre (double), 1 au maximum.

  Indique la fraction des données à échantillonner lors de la recherche d'entités PII.

- `ThresholdFraction` – Nombre (double), 1 au maximum.

  Indique la fraction des données qui doit être satisfaite pour qu'une colonne soit identifiée comme données PII.

- `MaskValue` – chaîne UTF-8, d'une longueur ne dépassant pas 256 octets, correspondant au Custom string pattern #58.

  Indique la valeur qui remplacera l'entité détectée.

- `RedactText` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Indique s'il faut expurger le texte des données d'identification personnelle (PII) détecté. Lorsqu'il est défini sur `true`, le contenu des PII est remplacé par des caractères de rédaction.

- `RedactChar` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le caractère utilisé pour remplacer le contenu des PII détecté lorsque la rédaction est activée. Le caractère de rédaction par défaut est `*`.

- `MatchPattern` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Modèle d'expression régulière utilisé pour identifier un contenu des PII supplémentaire au-delà des algorithmes de détection standard.

- `NumLeftCharsToExclude` – Nombre (entier), pas plus qu'Aucun.

Le nombre de caractères à exclure de la rédaction sur le côté gauche du contenu des PII détecté. Cela permet de préserver le contexte des données sensibles.

- `NumRightCharsToExclude` – Nombre (entier), pas plus qu'Aucun.

  Le nombre de caractères à exclure de la rédaction sur le côté droit du contenu des PII détecté. Cela permet de préserver le contexte des données sensibles.

- `DetectionParameters` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Paramètres supplémentaires pour configurer le comportement de détection des PII et les paramètres de sensibilité.

- `DetectionSensitivity` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Niveau de sensibilité pour la détection des PII. Des niveaux de sensibilité plus élevés détectent un plus grand nombre de PII potentielles, mais peuvent entraîner un plus grand nombre de faux positifs.

Structure Aggregate

Indique une transformation qui regroupe les lignes par champs choisis et calcule la valeur agrégée par fonction spécifiée.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom du nœud de transformation.

- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 1 chaîne maximum.

  Indique les champs et les lignes à utiliser comme entrées pour la transformation agrégée.

- `Groups` – Obligatoire : Tableau de chaînes UTF-8.

  Indique les champs à regrouper.

- `Aggs` – Obligatoire : tableau d'objets AggregateOperation, 1 structure minimum et 30 structures maximum.

  Indique les fonctions d'agrégation à exécuter sur des champs spécifiés.

DropDuplicates structure

Indique une transformation qui supprime des lignes de données répétitives d'un jeu de données.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom du nœud de transformation.
- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 1 chaîne maximum.

  Les entrées de données identifiées par leurs noms de nœuds.
- `Columns` – Tableau de chaînes UTF-8.

  Nom des colonnes à fusionner ou à supprimer en cas de répétition.

GovernedCatalogTarget structure

Spécifie une cible de données qui écrit sur Amazon S3 à l'aide du catalogue de AWS Glue données.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Le nom de la cible de données.
- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 1 chaîne maximum.

  Les nœuds qui constituent des entrées pour la cible de données.
- `PartitionKeys` – Tableau de chaînes UTF-8.

  Indique le partitionnement natif à l'aide d'une séquence de clés.
- `Table` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la table de la base de données dans laquelle écrire les données.
- `Database` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la base de données dans laquelle écrire les données.
- `SchemaChangePolicy` – Un objet CatalogSchemaChangePolicy.

  Politique qui indique des comportements de mise à jour pour le catalogue gouverné.

GovernedCatalogSource structure

Spécifie le magasin de données dans le catalogue de AWS Glue données régi.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom du stocker de données.
- `Database` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  La base de données à partir de laquelle lire les données.
- `Table` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Table de base de données à lire.
- `PartitionPredicate` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Les partitions satisfaisant à ce prédicat sont supprimées. Les fichiers qui se situent dans la période de conservation pour ces partitions ne sont pas supprimés. Valeur définie sur `""` – vide par défaut.
- `AdditionalOptions` – Un objet S3 SourceAdditionalOptions.

  Indique des options de connexion supplémentaires.

AggregateOperation structure

Indique l'ensemble de paramètres permettant d'effectuer l'agrégation de la transformation agrégée.

Champs

- `Column` – Obligatoire : Tableau de chaînes UTF-8.

  Indique la colonne du jeu de données sur lequel la fonction d'agrégation sera appliquée.
- `AggFunc` – Obligatoire :chaîne UTF-8 (valeurs valides : `avg` | `countDistinct` | `count` | `first` | `last` | `kurtosis` | `max` | `min` | `skewness` | `stddev_samp` | `stddev_pop` | `sum` | `sumDistinct` | `var_samp` | `var_pop`).

  Indique la fonction d'agrégation à appliquer.

  Les fonctions d'agrégation possibles incluent : avg countDistinct, count, first, last, kurtosis, max, min, skewness, stddev_samp, stddev_pop, sum, sumDistinct, var_samp, var_pop

GlueSchema structure

Indique un schéma défini par l'utilisateur lorsqu'un schéma ne peut pas être déterminé par AWS Glue.

Champs

- `Columns` : un tableau d'objets GlueStudioSchemaColumn.

  Spécifie les définitions de colonnes qui constituent un AWS Glue schéma.

GlueStudioSchemaColumn structure

Spécifie une seule colonne dans une définition de AWS Glue schéma.

Champs

- `Name` : Requis : chaîne UTF-8, d'une longueur maximale de 1 024 octets, correspondant au Single-line string pattern.

  Nom de la colonne dans le schéma AWS Glue Studio.
- `Type` – Chaîne UTF-8, d'une longueur maximale de 131 072 octets, correspondant au Single-line string pattern.

  Type de ruche pour cette colonne dans le schéma AWS Glue Studio.
- `GlueStudioType` – Chaîne UTF-8, d'une longueur maximale de 131 072 octets, correspondant au Single-line string pattern.

  Type de données de la colonne tel que défini dans AWS Glue Studio.

GlueStudioColumn structure

Spécifie une seule colonne dans AWS Glue Studio.

Champs

- `Key` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  La clé de la colonne dans AWS Glue Studio.
- `FullPath` – Obligatoire : Tableau de chaînes UTF-8.

TThe URL complète de la colonne dans AWS Glue Studio.

- Type : requis : chaîne UTF-8 (valeurs valides : array="ARRAY" | bigint="BIGINT" | bigint array="BIGINT_ARRAY" | binary="BINARY" | binary array="BINARY_ARRAY" | boolean="BOOLEAN" | boolean array="BOOLEAN_ARRAY" | byte="BYTE" | byte array="BYTE_ARRAY" | char="CHAR" | char array="CHAR_ARRAY" | choice="CHOICE" | choice array="CHOICE_ARRAY" | date="DATE" | date array="DATE_ARRAY" | decimal="DECIMAL" | decimal array="DECIMAL_ARRAY" | double="DOUBLE" | double array="DOUBLE_ARRAY" | enum="ENUM" | enum array="ENUM_ARRAY" | float="FLOAT" | float array="FLOAT_ARRAY" | int="INT" | int array="INT_ARRAY" | interval="INTERVAL" | interval array="INTERVAL_ARRAY" | long="LONG" | long array="LONG_ARRAY" | object="OBJECT" | short="SHORT" | short array="SHORT_ARRAY" | smallint="SMALLINT" | smallint array="SMALLINT_ARRAY" | string="STRING" | string array="STRING_ARRAY" | timestamp="TIMESTAMP" | timestamp array="TIMESTAMP_ARRAY" | tinyint="TINYINT" | tinyint array="TINYINT_ARRAY" | varchar="VARCHAR" | varchar array="VARCHAR_ARRAY" | null="NULL" | unknown="UNKNOWN" | unknown array="UNKNOWN_ARRAY").

  TThe type de colonne dans AWS Glue Studio.

- Children : tableau d'une structure.

  TThe enfants de la colonne parent dans AWS Glue Studio.

- GlueStudioType : chaîne UTF-8 (valeurs valides : array="ARRAY" | bigint="BIGINT" | bigint array="BIGINT_ARRAY" | binary="BINARY" | binary array="BINARY_ARRAY" | boolean="BOOLEAN" | boolean array="BOOLEAN_ARRAY" | byte="BYTE" | byte array="BYTE_ARRAY" | char="CHAR" | char array="CHAR_ARRAY" | choice="CHOICE" | choice array="CHOICE_ARRAY" | date="DATE" | date array="DATE_ARRAY" | decimal="DECIMAL" | decimal array="DECIMAL_ARRAY" | double="DOUBLE" | double array="DOUBLE_ARRAY" | enum="ENUM" | enum array="ENUM_ARRAY" | float="FLOAT" | float array="FLOAT_ARRAY" | int="INT" | int array="INT_ARRAY" | interval="INTERVAL" | interval array="INTERVAL_ARRAY" | long="LONG" | long array="LONG_ARRAY" | object="OBJECT" | short="SHORT" | short array="SHORT_ARRAY" | smallint="SMALLINT" | smallint array="SMALLINT_ARRAY" | string="STRING" | string array="STRING_ARRAY" | timestamp="TIMESTAMP" | timestamp array="TIMESTAMP_ARRAY" | tinyint="TINYINT" | tinyint array="TINYINT_ARRAY" | varchar="VARCHAR" | varchar array="VARCHAR_ARRAY" | null="NULL" | unknown="UNKNOWN" | unknown array="UNKNOWN_ARRAY").

Type de données de la colonne tel que défini dans AWS Glue Studio.

DynamicTransform structure

Spécifie l'ensemble de paramètres permettant d'effectuer la transformation dynamique.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Spécifie le nom de la transformation dynamique.
- `TransformName` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Spécifie le nom de la transformation dynamique tel qu'il apparaît dans l'éditeur visuel de AWS Glue Studio.
- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 1 chaîne maximum.

  Spécifie les entrées requises pour la transformation dynamique.
- `Parameters` : un tableau d'objets TransformConfigParameter.

  Spécifie les paramètres de la transformation dynamique.
- `FunctionName` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Spécifie le nom de la fonction de la transformation dynamique.
- `Path` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Spécifie le chemin de la source de transformation dynamique et des fichiers de configuration.
- `Version` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Ce champ n'est pas utilisé et sera obsolète dans une version ultérieure.
- `OutputSchemas` : un tableau d'objets GlueSchema.

  Indique le schéma de données de la transformation dynamique.

TransformConfigParameter structure

Spécifie les paramètres du fichier de configuration de la transformation dynamique.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Spécifie le nom du paramètre dans le fichier de configuration de la transformation dynamique.
- `Type` – Obligatoire : Chaîne UTF-8 (valeurs valides : `str="STR"` | `int="INT"` | `float="FLOAT"` | `complex="COMPLEX"` | `bool="BOOL"` | `list="LIST"` | `null="NULL"`).

  Spécifie le type de paramètre dans le fichier de configuration de la transformation dynamique.
- `ValidationRule` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Spécifie la règle de validation dans le fichier de configuration de la transformation dynamique.
- `ValidationMessage` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Spécifie le message de validation dans le fichier de configuration de la transformation dynamique.
- `Value` – Tableau de chaînes UTF-8.

  Spécifie la valeur du paramètre dans le fichier de configuration de la transformation dynamique.
- `ListType` – Chaîne UTF-8 (valeurs valides: `str="STR"` | `int="INT"` | `float="FLOAT"` | `complex="COMPLEX"` | `bool="BOOL"` | `list="LIST"` | `null="NULL"`).

  Spécifie le type de liste du paramètre dans le fichier de configuration de la transformation dynamique.
- `IsOptional` : booléen.

  Spécifie si le paramètre est facultatif ou non dans le fichier de configuration de la transformation dynamique.

EvaluateDataQuality structure

Spécifie vos critères d'évaluation de la qualité des données.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom de l'évaluation de la qualité des données.
- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 1 chaîne maximum.

Entrées de votre évaluation de la qualité des données.

- `Ruleset` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 65536 octets, correspondant au Custom string pattern #59.

  Ensemble de règles pour l'évaluation de la qualité de vos données.

- `Output` – Chaîne UTF-8 (valeurs valides : `PrimaryInput` | `EvaluationResults`).

  Résultat de votre évaluation de la qualité des données.

- `PublishingOptions` – Un objet DQResultsPublishingOptions.

  Options permettant de configurer le mode de publication de vos résultats.

- `StopJobOnFailureOptions` – Un objet DQStopJobOnFailureOptions.

  Options permettant de configurer le mode d'interruption de votre tâche en cas d'échec de l'évaluation de la qualité des données.

DQResultsPublishingOptions structure

Options permettant de configurer le mode de publication des résultats de votre évaluation de la qualité des données.

Champs

- `EvaluationContext` – Chaîne UTF-8, correspondant au Custom string pattern #60.

  Contexte de l'évaluation.

- `ResultsS3Prefix` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Préfixe Amazon S3 ajouté aux résultats.

- `CloudWatchMetricsEnabled` : booléen.

  Activez les métriques pour vos résultats de qualité des données.

- `ResultsPublishingEnabled` : booléen.

  Activez la publication pour vos résultats de qualité des données.

DQStopJobOnFailureOptions structure

Options permettant de configurer le mode d'interruption de votre tâche en cas d'échec de l'évaluation de la qualité des données.

Champs

- `StopJobOnFailureTiming` – Chaîne UTF-8 (valeurs valides : `Immediate` | `AfterDataLoad`).

  Quand arrêter la tâche en cas d'échec de votre évaluation de la qualité des données. Les options sont immédiates ou AfterDataLoad.

EvaluateDataQualityMultiFrame structure

Spécifie vos critères d'évaluation de la qualité des données.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom de l'évaluation de la qualité des données.
- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, au moins 1 chaîne.

  Entrées de votre évaluation de la qualité des données. La première entrée de cette liste est la source de données principale.
- `AdditionalDataSources` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, correspondant au Custom string pattern #63.

  Chaque valeur est une chaîne UTF-8, correspondant au Custom string pattern #61.

  Les alias de toutes les sources de données, à l'exception de la source principale.
- `Ruleset` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 65536 octets, correspondant au Custom string pattern #59.

  Ensemble de règles pour l'évaluation de la qualité de vos données.
- `PublishingOptions` – Un objet DQResultsPublishingOptions.

  Options permettant de configurer le mode de publication de vos résultats.

- `AdditionalOptions` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8 (valeurs valides : `performanceTuning.caching="CacheOption"` | `observations.scope="ObservationsOption"` |`compositeRuleEvaluation.method="CompositeOption"`).

  Chaque valeur est une chaîne UTF-8.

  Options permettant de configurer le comportement d'exécution de la transformation.

- `StopJobOnFailureOptions` – Un objet DQStopJobOnFailureOptions.

  Options permettant de configurer le mode d'interruption de votre tâche en cas d'échec de l'évaluation de la qualité des données.

Structure de la recette

Un nœud AWS Glue Studio qui utilise une AWS Glue DataBrew recette dans les AWS Glue tâches.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom du nœud AWS Glue Studio.

- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 1 chaîne maximum.

  Les nœuds qui sont des entrées du nœud de recette, identifiés par un identifiant.

- `RecipeReference` – Un objet RecipeReference.

  Référence à la DataBrew recette utilisée par le nœud.

- `RecipeSteps` : un tableau d'objets RecipeStep.

  Étapes de transformation utilisées dans le nœud de recette.

RecipeReference structure

Référence à une AWS Glue DataBrew recette.

Champs

- `RecipeArn` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  L'ARN de la DataBrew recette.
- `RecipeVersion` – obligatoire : chaîne UTF-8, d'une longueur comprise entre 1 et 16 octets.

  Celui RecipeVersion de la DataBrew recette.

SnowflakeNodeData structure

Spécifie la configuration des nœuds Snowflake dans Studio. AWS Glue

Champs

- `SourceType` – Chaîne UTF-8, correspondant au Custom string pattern #60.

  Indique la manière dont les données extraites sont spécifiées. Valeurs valides : `"table"`, `"query"`.
- `Connection` : un objet Option.

  Spécifie une connexion au catalogue de AWS Glue données à un point de terminaison Snowflake.
- `Schema` – Chaîne UTF-8.

  Indique un schéma de base de données Snowflake que votre nœud doit utiliser.
- `Table` – Chaîne UTF-8.

  Indique une table Snowflake que votre nœud doit utiliser.
- `Database` – Chaîne UTF-8.

  Indique une base de données Snowflake que votre nœud doit utiliser.
- `TempDir` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Non utilisé actuellement.
- `IamRole` : un objet Option.

  Non utilisé actuellement.
- `AdditionalOptions` – Tableau de mappage de paires valeur-clé.

Chaque clé est une chaîne UTF-8, correspondant au Custom string pattern #61.

Chaque valeur est une chaîne UTF-8, correspondant au Custom string pattern #61.

Indique les options supplémentaires transmises au connecteur Snowflake. Si des options sont spécifiées ailleurs dans ce nœud, elles seront prioritaires.

- `SampleQuery` – Chaîne UTF-8.

  Une chaîne SQL utilisée pour récupérer des données avec le type source `query`.

- `PreAction` – Chaîne UTF-8.

  Une chaîne SQL exécutée avant que le connecteur Snowflake n'exécute ses actions standard.

- `PostAction` – Chaîne UTF-8.

  Une chaîne SQL exécutée après que le connecteur Snowflake a exécuté ses actions standard.

- `Action` – Chaîne UTF-8.

  Indique l'action à effectuer lors de l'écriture dans une table contenant des données préexistantes. Valeurs valides : `append`, `merge`, `truncate`, `drop`.

- `Upsert` : booléen.

  Utilisé lorsque action est `append`. Indique le comportement de résolution lorsqu'une ligne existe déjà. Si la valeur est vraie, les lignes préexistantes seront mises à jour. Si la valeur est fausse, ces lignes seront insérées.

- `MergeAction` – Chaîne UTF-8, correspondant au Custom string pattern #60.

  Indique une action de fusion. Valeurs valides : `simple`, `custom`. S'il la valeur est simple, le comportement de fusion est défini par `MergeWhenMatched` et `MergeWhenNotMatched`. Si la valeur est personnalisée, il est défini par `MergeClause`.

- `MergeWhenMatched` – Chaîne UTF-8, correspondant au Custom string pattern #60.

  Indique comment résoudre les enregistrements qui correspondent à des données préexistantes lors de la fusion. Valeurs valides : `update`, `delete`.

- `MergeWhenNotMatched` – Chaîne UTF-8, correspondant au Custom string pattern #60.

  Indique comment traiter les enregistrements qui ne correspondent pas aux données préexistantes lors de la fusion. Valeurs valides : `insert`, `none`.

- `MergeClause` – Chaîne UTF-8.

  Une instruction SQL qui spécifie un comportement de fusion personnalisé.

- `StagingTable` – Chaîne UTF-8.

  Le nom d'une table intermédiaire utilisée lors de l'exécution de l'action `merge` ou d'actions d'insertion `append`. Les données sont écrites dans cette table, puis déplacées vers `table` par une postaction générée.

- `SelectedColumns` : un tableau d'objets Option.

  Indique les colonnes combinées pour identifier un enregistrement lors de la détection des correspondances pour les fusions et les insertions. Une liste de structures avec des clés `value`, `label` et `description`. Chaque structure décrit une colonne.

- `AutoPushdown` : booléen.

  Indique si le pushdown automatique des requêtes est activée. Lorsque l'option pushdown est activée, si une partie de la requête peut être « poussée vers le bas » sur le serveur Snowflake, elle est poussée vers le bas au moment de l'exécution de la requête sur Spark. Cela améliore les performances de certaines requêtes.

- `TableSchema` : un tableau d'objets Option.

  Définit manuellement le schéma cible du nœud. Une liste de structures avec des clés `value`, `label` et `description`. Chaque structure définit une colonne.

SnowflakeSource structure

Indique une source de données Snowflake.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Le nom de la source de données Snowflake.

- `Data` – Obligatoire : un objet SnowflakeNodeData.

  Configuration de la source de données Snowflake.

- `OutputSchemas` : un tableau d'objets GlueSchema.

Indique les schémas définis par l'utilisateur pour vos données de sortie.

SnowflakeTarget structure

Indique une cible Snowflake.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Le nom de la cible Snowflake.
- `Data` – Obligatoire : un objet SnowflakeNodeData.

  Indique les données du nœud cible Snowflake.
- `Inputs` : tableau de chaînes UTF-8, avec une chaîne minimum et une chaîne maximum.

  Les nœuds qui constituent des entrées pour la cible de données.

ConnectorDataSource structure

Spécifie une source générée avec des options de connexion standard.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom de ce nœud source.
- `ConnectionType` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  LeconnectionType, tel que fourni à la AWS Glue bibliothèque sous-jacente. Ce type de nœud prend en charge les types de connexion suivants :
  - `opensearch`
  - `azuresql`
  - `azurecosmos`
  - `bigquery`
  - `saphana`
  - `teradata`

  - `vertica`
- `Data` – obligatoire : tableau de mappage de paires clé-valeur.

  Chaque clé est une chaîne UTF-8.

  Chaque valeur est une chaîne UTF-8.

  Carte indiquant des options de connexion pour le nœud. Vous trouverez les options de connexion standard pour le type de connexion correspondant dans la section Paramètres de connexion de la AWS Glue documentation.
- `OutputSchemas` : un tableau d'objets GlueSchema.

  Spécifie le schéma de données pour cette source.

ConnectorDataTarget structure

Spécifie une cible générée avec des options de connexion standard.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom de ce nœud cible.
- `ConnectionType` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  LeconnectionType, tel que fourni à la AWS Glue bibliothèque sous-jacente. Ce type de nœud prend en charge les types de connexion suivants :
  - `opensearch`
  - `azuresql`
  - `azurecosmos`
  - `bigquery`
  - `saphana`
  - `teradata`
  - `vertica`
- `Data` – obligatoire : tableau de mappage de paires clé-valeur.

  Chaque clé est une chaîne UTF-8.

Chaque valeur est une chaîne UTF-8.

Carte indiquant des options de connexion pour le nœud. Vous trouverez les options de connexion standard pour le type de connexion correspondant dans la section Paramètres de connexion de la AWS Glue documentation.

- `Inputs` : tableau de chaînes UTF-8, avec une chaîne minimum et une chaîne maximum.

  Les nœuds qui constituent des entrées pour la cible de données.

RecipeStep structure

Étape de recette utilisée dans un nœud de recette de préparation de données AWS Glue Studio.

Champs

- `Action` – Obligatoire : un objet RecipeAction.

  Action de transformation de l'étape de recette.
- `ConditionExpressions` : un tableau d'objets ConditionExpression.

  Expressions de condition pour l'étape de recette.

RecipeAction structure

Actions définies dans le nœud de recette de préparation des données de AWS Glue Studio.

Champs

- `Operation` – obligatoire : chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets, correspondant au Custom string pattern #56.

  Fonctionnement de l'action de recette.
- `Parameters` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets, correspondant au Custom string pattern #57.

  Chaque valeur est une chaîne UTF-8, d'une longueur comprise entre 1 et 32 768 octets.

Paramètres de l'action de recette.

ConditionExpression structure

Expression de condition définie dans le nœud de recette de préparation des données de AWS Glue Studio.

Champs

- `Condition` – obligatoire : chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets, correspondant au Custom string pattern #56.

  Condition de l'expression de condition.

- `Value` – Chaîne UTF-8, d'une longueur maximale de 1 024 octets.

  Valeur de l'expression de condition.

- `TargetColumn` – Obligatoire : chaîne UTF-8, d'une longueur comprise entre 1 et 1 024 octets.

  Colonne cible des expressions de condition.

CatalogIcebergSource Structure S3

Spécifie une source de données Apache Iceberg enregistrée dans le catalogue de AWS Glue données. La source de données Iceberg doit être stockée dans Amazon S3.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom de la source de données Iceberg.

- `Database` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la base de données à partir de laquelle lire les données.

- `Table` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la table dans la base de données à partir de laquelle lire les données.

- `AdditionalIcebergOptions` – Tableau de mappage de paires valeur-clé.

Chaque clé est une chaîne UTF-8, correspondant au Custom string pattern #61.

Chaque valeur est une chaîne UTF-8, correspondant au Custom string pattern #61.

Indique des options de connexion supplémentaires pour la source de données Iceberg.

- `OutputSchemas` : un tableau d'objets GlueSchema.

  Indique le schéma de données de la source Iceberg.

CatalogIcebergSource structure

Spécifie une source de données Apache Iceberg enregistrée dans le catalogue de AWS Glue données.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom de la source de données Iceberg.
- `Database` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la base de données à partir de laquelle lire les données.
- `Table` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la table dans la base de données à partir de laquelle lire les données.
- `AdditionalIcebergOptions` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, correspondant au Custom string pattern #61.

  Chaque valeur est une chaîne UTF-8, correspondant au Custom string pattern #61.

  Indique des options de connexion supplémentaires pour la source de données Iceberg.
- `OutputSchemas` : un tableau d'objets GlueSchema.

  Indique le schéma de données de la source Iceberg.

IcebergCatalogTarget Structure S3

Spécifie une cible de catalogue Apache Iceberg qui écrit des données Amazon S3 et enregistre la table dans le catalogue de AWS Glue données.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom de la cible du catalogue Iceberg.

- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 1 chaîne maximum.

  Connexion d'entrée pour la cible du catalogue Iceberg.

- `PartitionKeys` – Tableau de chaînes UTF-8.

  Liste de clés de partition pour la table Iceberg.

- `Table` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Nom de la table dans laquelle écrire dans le catalogue.

- `Database` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le nom de la base de données dans laquelle écrire les données.

- `AdditionalOptions` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, correspondant au Custom string pattern #61.

  Chaque valeur est une chaîne UTF-8, correspondant au Custom string pattern #61.

  Indique des options de connexion supplémentaires pour la cible du catalogue Iceberg.

- `SchemaChangePolicy` – Un objet CatalogSchemaChangePolicy.

  Politique de gestion des modifications de schéma dans la cible du catalogue.

- `AutoDataQuality` : un objet AutoDataQuality.

  Indique s'il faut activer automatiquement l'évaluation de la qualité des données pour la cible du catalogue S3 Iceberg. Lorsque ce paramètre est défini sur `true`, les contrôles de qualité des données sont effectués automatiquement pendant l'opération d'écriture.

Structure de Dynamo DBELTConnector Source

Spécifie une source de connecteur DynamoDB ELT pour extraire les données des tables DynamoDB.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom de la source du connecteur DynamoDB ELT.
- `ConnectionOptions` – Un objet DDBELTConnectionOptions.

  Options de connexion pour la source du connecteur DynamoDB ELT.
- `OutputSchemas` : un tableau d'objets GlueSchema.

  Indique le schéma de données de la source du connecteur DynamoDB ELT.

DDBELTConnectionStructure des options

Indique les options de connexion pour les opérations DynamoDB ELT (extraction, chargement, transformation). Cette structure contient les paramètres de configuration permettant de se connecter aux tables DynamoDB et d'en extraire les données à l'aide du connecteur ELT.

Champs

- `DynamodbExport` – Chaîne UTF-8 (valeurs valides : `ddb` | `s3`).

  Indique le type d'exportation pour l'extraction de données DynamoDB. Ce paramètre détermine la manière dont les données sont exportées depuis la table DynamoDB pendant le processus ELT.
- `DynamodbUnnestDDBJson` : booléen.

  Valeur booléenne qui indique s'il faut désimbriquer le format DynamoDB JSON lors de l'extraction des données. Lorsque cette valeur est définie sur `true`, le connecteur aplatit les structures JSON imbriquées à partir d'éléments DynamoDB. Lorsque cette valeur est définie sur `false`, la structure JSON DynamoDB d'origine est préservée.
- `DynamodbTableArn` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #61.

  Amazon Resource Name (ARN) de la table DynamoDB à partir de laquelle extraire les données. Ce paramètre spécifie la table source pour l'opération ELT.
- `DynamodbS3Bucket` – Chaîne UTF-8, correspondant au Custom string pattern #61.

Nom du compartiment Amazon S3 utilisé pour le stockage intermédiaire pendant le processus DynamoDB ELT. Ce compartiment est utilisé pour stocker temporairement les données DynamoDB exportées avant qu'elles ne soient traitées par la tâche ELT.

- `DynamodbS3Prefix` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Préfixe de clé d'objet S3 pour les fichiers stockés dans le compartiment S3 intermédiaire pendant le processus DynamoDB ELT. Ce préfixe permet d'organiser et d'identifier les fichiers temporaires créés lors de l'extraction des données.

- `DynamodbS3BucketOwner` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  L'ID de AWS compte du propriétaire du compartiment S3 spécifié dans`DynamodbS3Bucket`. Ce paramètre est obligatoire lorsque le compartiment S3 appartient à un AWS compte différent de celui qui exécute la tâche ELT, ce qui permet un accès entre comptes au compartiment de stockage intermédiaire.

- `DynamodbStsRoleArn` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Nom de ressource Amazon (ARN) du rôle de AWS Security Token Service (STS) à assumer pour accéder aux ressources DynamoDB et S3 pendant l'opération ELT. Ce rôle doit disposer des autorisations nécessaires pour lire dans la table DynamoDB et écrire dans le compartiment S3 intermédiaire.

DDBELTCatalogAdditionalOptions structure

Indique des options supplémentaires pour les opérations de catalogue DynamoDB ELT.

Champs

- `DynamodbExport` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Indique la configuration d'exportation DynamoDB pour l'opération ELT.

- `DynamodbUnnestDDBJson` : booléen.

  Indique s'il faut annuler l'imbrication du format JSON DynamoDB. Lorsque cette valeur est définie sur `true`, les structures JSON imbriquées dans les éléments DynamoDB sont aplaties.

Structure de route

Spécifie un nœud de route qui dirige les données vers différents chemins de sortie en fonction de conditions de filtrage définies.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #63.

  Nom du nœud de route.

- `Inputs` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 1 chaîne maximum.

  Connexion d'entrée pour le nœud de route.

- `GroupFiltersList` – Obligatoire : un tableau d'objets GroupFilters.

  Liste de filtres de groupe qui définissent les conditions de routage et les critères permettant de diriger les données vers différents chemins de sortie.

GroupFilters structure

Spécifie un groupe de filtres avec un opérateur logique qui détermine la manière dont les filtres sont combinés pour évaluer les conditions de routage.

Champs

- `GroupName` – Obligatoire : Chaîne UTF-8, correspondant au Custom string pattern #60.

  Nom du groupe de filtres.

- `Filters` – Obligatoire : un tableau d'objets FilterExpression.

  Liste d'expressions de filtre qui définissent les conditions pour ce groupe.

- `LogicalOperator` – Obligatoire : Chaîne UTF-8 (valeurs valides : `AND` | `OR`).

  Opérateur logique utilisé pour combiner les filtres de ce groupe. Détermine si tous les filtres doivent correspondre (AND) ou si n'importe quel filtre peut correspondre (OR).

AutoDataQuality structure

Spécifie les options de configuration pour l'évaluation automatique de la qualité des données dans les AWS Glue tâches. Cette structure permet des contrôles et une surveillance automatisés de la qualité des données pendant les opérations ETL, ce qui contribue à garantir l'intégrité et la fiabilité des données sans intervention manuelle.

Champs

- `IsEnabled` : booléen.

  Indique si l'évaluation automatique de la qualité des données est activée. Lorsque ce paramètre est défini sur `true`, les contrôles de qualité des données sont effectués automatiquement.
- `EvaluationContext` – Chaîne UTF-8, correspondant au Custom string pattern #61.

  Le contexte d'évaluation pour les contrôles automatiques de qualité des données. Cela définit la portée et les paramètres de l'évaluation de la qualité des données.

API de tâches

L'API Jobs décrit les types de données des tâches et contient des informations APIs permettant d'utiliser les tâches, les exécutions de tâches et les déclencheurs AWS Glue.

Rubriques

- Tâches
- Exécutions de tâches
- Triggers

Tâches

L'API Jobs décrit les types de données et l'API liés à la création, à la mise à jour, à la suppression ou à l'affichage des jobs dans AWS Glue.

Types de données

- Structure Job
- ExecutionProperty structure

- NotificationProperty structure
- JobCommand structure
- ConnectionsList structure
- JobUpdate structure
- SourceControlDetails structure

Structure Job

Spécifie une définition de la tâche.

Champs

- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom que vous affectez à la définition de la tâche.
- `JobMode` – Chaîne UTF-8 (valeurs valides : `SCRIPT=""` | `VISUAL=""` | `NOTEBOOK=""`).

  Mode qui décrit la façon dont une tâche a été créée. Les valeurs valides sont :

  - `SCRIPT`- La tâche a été créée à l'aide de l'éditeur de script AWS Glue Studio.
  - `VISUAL`- La tâche a été créée à l'aide de l'éditeur visuel AWS Glue Studio.
  - `NOTEBOOK` : la tâche a été créée à l'aide d'un bloc-notes de sessions interactives.

  Lorsque le champ `JobMode` est manquant ou a la valeur null, `SCRIPT` est attribué comme valeur par défaut.
- `JobRunQueuingEnabled` : booléen.

  Indique si la mise en file d'attente des exécutions de tâches est activée pour les exécutions de cette tâche.

  La valeur true signifie que la mise en file d'attente des tâches est activée pour les exécutions de tâches. Si la valeur est false ou si elle n'est pas renseignée, les exécutions de tâches ne seront pas prises en compte pour la mise en file d'attente.

  Si ce champ ne correspond pas à la valeur définie dans l'exécution de la tâche, la valeur du champ d'exécution de la tâche sera utilisée.

- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description de la tâche.

- `LogUri` – Chaîne UTF-8.

  Ce champ est réservé pour un usage futur.

- `Role` – Chaîne UTF-8.

  Nom ou Amazon Resource Name (ARN) du rôle IAM associé à cette tâche.

- `CreatedOn` – Horodatage.

  Date et heure de création de la définition de tâche.

- `LastModifiedOn` – Horodatage.

  Dernier moment où la définition de tâche a été modifiée.

- `ExecutionProperty` – Un objet ExecutionProperty.

  Objet `ExecutionProperty` spécifiant le nombre maximal d'exécutions simultanées autorisées pour cette tâche.

- `Command` – Un objet JobCommand.

  La commande `JobCommand` qui exécute cette tâche.

- `DefaultArguments` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8.

  Chaque valeur est une chaîne UTF-8.

  Les arguments par défaut pour chaque exécution de cette tâche, spécifiés en tant que paires nom-valeur.

  Vous pouvez spécifier ici les arguments que votre propre script d'exécution de tâches consomme, ainsi que les arguments qu'il consomme AWS Glue lui-même.

  Les arguments de la tâche peuvent être consignés. Ne transmettez pas de secrets en texte clair comme arguments. Récupérez les secrets d'une AWS Glue connexion AWS Secrets Manager ou d'un autre mécanisme de gestion des secrets si vous avez l'intention de les conserver dans le cadre du Job.

Pour plus d'informations sur la façon de spécifier et d'utiliser vos propres arguments Job, consultez la rubrique Calling AWS Glue APIs in Python du guide du développeur.

Pour plus d'informations sur les arguments que vous pouvez fournir dans ce champ lors de la configuration des tâches Spark, consultez la rubrique Special Parameters Used by AWS Glue dans le guide du développeur.

Pour plus d'informations sur les arguments que vous pouvez fournir dans ce champ lors de la configuration des tâches Ray, consultez Utilisation des paramètres de tâches dans les tâches Ray dans le guide du développeur.

- `NonOverridableArguments` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8.

  Chaque valeur est une chaîne UTF-8.

  Les arguments de cette tâche qui ne sont pas remplacés lorsque vous fournissez des arguments de tâche dans le cadre d'une exécution de tâche, spécifiés sous forme de paires nom-valeur.

- `Connections` – Un objet ConnectionsList.

  Connexions utilisées pour la tâche.

- `MaxRetries` : nombre (entier).

  Nombre maximal de fois que vous pouvez réessayer cette tâche après un JobRun échec.

- `AllocatedCapacity` : nombre (entier).

  Ce champ est obsolète. Utilisez `MaxCapacity` à la place.

  Nombre d'unités de traitement de AWS Glue données (DPUs) allouées aux exécutions de cette tâche. Vous pouvez en allouer au moins 2 DPUs ; la valeur par défaut est 10. Un DPU est une mesure relative de la puissance de traitement composée de 4 V de capacité CPUs de calcul et de 16 Go de mémoire. Pour plus d'informations, consultez la page de tarification AWS Glue.

- `Timeout` – Nombre (entier), au moins égal à 1.

  Délai d'expiration de la tâche en minutes. Durée maximale durant laquelle une tâche exécutée peut consommer des ressources avant qu'elle ne se termine et n'entre dans l'état `TIMEOUT`.

Les tâches doivent avoir des valeurs de délai d'expiration inférieures à 7 jours ou 10 080 minutes. Dans le cas contraire, les tâches déclencheront une exception.

Si aucune valeur n'est indiquée, le délai d'expiration est défini par défaut sur 2 880 minutes.

Toutes les AWS Glue tâches existantes dont le délai d'expiration est supérieur à 7 jours seront définies par défaut sur 7 jours. Par exemple, si vous avez spécifié un délai d'expiration de 20 jours pour un traitement par lots, celui-ci sera arrêté le 7e jour.

Pour les tâches de streaming, si vous avez défini une fenêtre de maintenance, elles seront redémarrées au cours de celle-ci au bout de sept jours.

- `MaxCapacity` – Nombre (double).

  Pour les tâches Glue version 1.0 ou antérieures, en utilisant le type de travailleur standard, le nombre d'unités de traitement des AWS Glue données (DPUs) qui peuvent être allouées lors de l'exécution de cette tâche. Un DPU est une mesure relative de la puissance de traitement composée de 4 V de capacité CPUs de calcul et de 16 Go de mémoire. Pour plus d'informations, consultez la page de tarification AWS Glue.

  Pour les tâches des versions 2.0 ou ultérieures de Glue, vous ne pouvez pas spécifier de `Maximum capacity`. Au lieu de cela, vous devez spécifier `Worker type` et `Number of workers`.

  Ne définissez pas `MaxCapacity` si vous utilisez `WorkerType` et `NumberOfWorkers`.

  La valeur pouvant être attribuée à `MaxCapacity` varie selon que vous exécutez une tâche shell Python, une tâche ETL Apache Spark ou une tâche ETL Apache Spark Streaming :

  - Lorsque vous spécifiez une tâche shell Python (`JobCommand.Name="pythonshell"`), vous pouvez allouer 0,0625 ou 1 DPU. La valeur par défaut correspond à 0,0625 DPU.
  - Lorsque vous spécifiez une tâche ETL Apache Spark (`JobCommand.Name="glueetl »`) ou une tâche ETL de streaming Apache Spark (`JobCommand.Name="gluestreaming »`), vous pouvez en allouer de 2 à 100. DPUs La valeur par défaut est 10 DPUs. Ce type de tâche ne peut pas avoir une allocation DPU fractionnée.

- `WorkerType`— Chaîne UTF-8 (valeurs valides : `Standard=""` | | `G.025X=""` | `G.1X=""` | `G.2X=""` | `G.4X=""` | `G.8X=""` | `G.12X=""` | `G.16X=""` | `R.1X=""` | `R.2X=""` `R.4X=""` `R.8X=""` |`Z.2X=""`).

  Type de travail prédéfini qui est alloué lorsqu'une tâche est exécutée.

AWS Glue propose plusieurs types de travailleurs pour répondre aux différentes exigences de charge de travail :

Types de travailleurs G (travailleurs de calcul à usage général) :

- G.025X : 0,25 DPU (2 VCPUs, 4 Go de mémoire, 84 Go de disque)
- G.1 X : 1 processeur (4 VCPUs, 16 Go de mémoire, 94 Go de disque)
- G.2X : 2 processeurs (8 VCPUs, 32 Go de mémoire, 138 Go de disque)
- G.4X : 4 processeurs (16 VCPUs, 64 Go de mémoire, 256 Go de disque)
- G.8X : 8 DPU (32 VCPUs, 128 Go de mémoire, 512 Go de disque)
- G.12X : 12 DPU (48 VCPUs, 192 Go de mémoire, 768 Go de disque)
- G.16X : 16 DPU (64 VCPUs, 256 Go de mémoire, 1024 Go de disque)

Types de travailleurs R (travailleurs optimisés pour la mémoire) :

- R.1X : 1 processeur (4 VCPUs, 32 Go de mémoire, 94 Go de disque)
- R.2X : 2 processeurs (8 VCPUs, 64 Go de mémoire, 128 Go de disque)
- R.4X : 4 processeurs (16 VCPUs, 128 Go de mémoire, 256 Go de disque)
- R.8X : 8 DPU (32 VCPUs, 256 Go de mémoire, 512 Go de disque)

- `NumberOfWorkers` : nombre (entier).

  Nombre de travaux d'un `workerType` défini qui sont attribués lorsqu'une tâche est exécutée.

- `SecurityConfiguration` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la structure `SecurityConfiguration` à utiliser avec cette tâche.

- `NotificationProperty` – Un objet NotificationProperty.

  Spécifie les propriétés de configuration d'une notification de tâche.

- `Running` : booléen.

  Ce champ est réservé pour un usage futur.

- `GlueVersion` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #48.

Dans les tâches Spark, `GlueVersion` détermine les versions d'Apache Spark et de Python AWS Glue disponibles dans une tâche. La version de Python indique la version prise en charge pour les tâches de type Spark.

Les tâches Ray doivent définir `GlueVersion` sur `4.0` ou supérieur. Toutefois, les versions de Ray, de Python et des bibliothèques supplémentaires disponibles dans votre tâche Ray sont déterminées par le paramètre `Runtime` de la commande de tâche.

Pour plus d'informations sur les AWS Glue versions disponibles et les versions correspondantes de Spark et Python, consultez la version de Glue dans le guide du développeur.

Les tâches créées sans que la version de Glue soit spécifiée sont des tâches Glue 0.9 par défaut.

- `CodeGenConfigurationNodes` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, correspondant au Custom string pattern #60.

  Chaque valeur est un objet CodeGenConfigurationNode.

  Représentation d'un graphe orienté acyclique sur lequel le composant visuel Glue Studio et la génération de code Glue Studio sont basés.

- `ExecutionClass` – Chaîne UTF-8, d'une longueur maximale de 16 octets (valeurs valides : `FLEX=""` | `STANDARD=""`).

  Indique si la tâche est exécutée avec une classe d'exécution standard ou flexible. La classe d'exécution standard est idéale pour les charges de travail sensibles au temps qui nécessitent un démarrage rapide des tâches et des ressources dédiées.

  La classe d'exécution flexible, adaptée aux tâches non urgentes dont les heures de début et de fin peuvent varier.

  Seules les tâches dotées de AWS Glue la version 3.0 ou supérieure et du type de commande `glueetl` seront autorisées à être définies `ExecutionClass` sur`FLEX`. La classe d'exécution flexible est disponible pour les tâches Spark.

- `SourceControlDetails` – Un objet SourceControlDetails.

  Les détails d'une configuration de contrôle source pour une tâche, permettant la synchronisation des artefacts de la tâche vers ou depuis un référentiel distant.

- `MaintenanceWindow` – Chaîne UTF-8, correspondant au Custom string pattern #34.

Ce champ indique un jour de la semaine et une heure pour une fenêtre de maintenance pour les tâches de streaming. AWS Glue effectue périodiquement des activités de maintenance. Au cours de ces fenêtres de maintenance, vous AWS Glue devrez redémarrer vos tâches de streaming.

AWS Glue redémarrera le travail dans les 3 heures suivant la période de maintenance spécifiée. Par exemple, si vous configurez la fenêtre de maintenance pour le lundi à 10h00 GMT, vos tâches seront redémarrées entre 10h00 GMT et 13h00 GMT.

- `ProfileName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom d'un profil AWS Glue d'utilisation associé à la tâche.

ExecutionProperty structure

Propriété d'exécution d'une tâche.

Champs

- `MaxConcurrentRuns` : nombre (entier).

  Nombre maximal d'exécutions simultanées autorisées pour la tâche. La valeur par défaut est 1. Une erreur est renvoyée lorsque ce seuil est atteint. La valeur maximale que vous pouvez spécifier est contrôlée par une limite de service.

NotificationProperty structure

Spécifie les propriétés de configuration d'une notification.

Champs

- `NotifyDelayAfter` – Nombre (entier), au moins égal à 1.

  Après le démarrage d'une exécution de tâche, nombre de minutes d'attente avant l'envoi d'une notification de délai d'exécution de tâche.

JobCommand structure

Spécifie le code exécuté lorsqu'une tâche est exécutée.

Champs

- `Name` – Chaîne UTF-8.

  Nom de la commande de tâche. Pour une tâche ETL Apache Spark, cette valeur doit être `glueetl`. Pour un shell Python, elle doit être `pythonshell`. Pour une tâche ETL Apache Spark Streaming, elle doit correspondre à `gluestreaming`. Pour une tâche Ray, cela doit être `glueray`.
- `ScriptLocation` – Chaîne UTF-8, d'une longueur maximale de 400 000 octets.

  Spécifie le chemin d'accès Amazon Simple Storage Service (Amazon S3) à un script qui exécute une tâche.
- `PythonVersion` – Chaîne UTF-8, correspondant au Custom string pattern #49.

  Version Python utilisée pour exécuter une tâche shell Python. Les valeurs autorisées sont 2 ou 3.
- `Runtime` : chaîne UTF-8, d'une longueur ne dépassant pas 64 octets, correspondant au Custom string pattern #33.

  Dans les tâches Ray, l'exécution est utilisée pour spécifier les versions de Ray, de Python et des bibliothèques supplémentaires disponibles dans votre environnement. Ce champ n'est pas utilisé dans les autres types de tâches. Pour connaître les valeurs d'environnement d'exécution prises en charge, consultez la section Environnements d'exécution Ray pris en charge dans le manuel du AWS Glue développeur.

ConnectionsList structure

Spécifie les connexions utilisées par une tâche.

Champs

- `Connections` – Tableau de chaînes UTF-8, avec 1 000 chaînes maximum.

  Liste de connexions utilisées par la tâche.

JobUpdate structure

Spécifie les informations utilisées pour mettre à jour une définition de tâche. La définition de tâche précédente est entièrement remplacée par ces informations.

Champs

- `JobMode` – Chaîne UTF-8 (valeurs valides : `SCRIPT=""` | `VISUAL=""` | `NOTEBOOK=""`).

  Mode qui décrit la façon dont une tâche a été créée. Les valeurs valides sont :

  - `SCRIPT`- La tâche a été créée à l'aide de l'éditeur de script AWS Glue Studio.
  - `VISUAL`- La tâche a été créée à l'aide de l'éditeur visuel AWS Glue Studio.
  - `NOTEBOOK` : la tâche a été créée à l'aide d'un bloc-notes de sessions interactives.

  Lorsque le champ `JobMode` est manquant ou a la valeur null, `SCRIPT` est attribué comme valeur par défaut.
- `JobRunQueuingEnabled` : booléen.

  Indique si la mise en file d'attente des exécutions de tâches est activée pour les exécutions de cette tâche.

  La valeur true signifie que la mise en file d'attente des tâches est activée pour les exécutions de tâches. Si la valeur est false ou si elle n'est pas renseignée, les exécutions de tâches ne seront pas prises en compte pour la mise en file d'attente.

  Si ce champ ne correspond pas à la valeur définie dans l'exécution de la tâche, la valeur du champ d'exécution de la tâche sera utilisée.
- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description de la tâche définie.
- `LogUri` – Chaîne UTF-8.

  Ce champ est réservé pour un usage futur.
- `Role` – Chaîne UTF-8.

  Nom ou Amazon Resource Name (ARN) du rôle IAM associé à cette tâche (obligatoire).
- `ExecutionProperty` – Un objet ExecutionProperty.

  Objet `ExecutionProperty` spécifiant le nombre maximal d'exécutions simultanées autorisées pour cette tâche.
- `Command` – Un objet JobCommand.

Objet `JobCommand` qui exécute cette tâche (obligatoire).

- `DefaultArguments` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8.

  Chaque valeur est une chaîne UTF-8.

  Les arguments par défaut pour chaque exécution de cette tâche, spécifiés en tant que paires nom-valeur.

  Vous pouvez spécifier ici les arguments que votre propre script d'exécution de tâches consomme, ainsi que les arguments qu'il consomme AWS Glue lui-même.

  Les arguments de la tâche peuvent être consignés. Ne transmettez pas de secrets en texte clair comme arguments. Récupérez les secrets d'une AWS Glue connexion AWS Secrets Manager ou d'un autre mécanisme de gestion des secrets si vous avez l'intention de les conserver dans le cadre du Job.

  Pour plus d'informations sur la façon de spécifier et d'utiliser vos propres arguments Job, consultez la rubrique Calling AWS Glue APIs in Python du guide du développeur.

  Pour plus d'informations sur les arguments que vous pouvez fournir dans ce champ lors de la configuration des tâches Spark, consultez la rubrique Special Parameters Used by AWS Glue dans le guide du développeur.

  Pour plus d'informations sur les arguments que vous pouvez fournir dans ce champ lors de la configuration des tâches Ray, consultez Utilisation des paramètres de tâches dans les tâches Ray dans le guide du développeur.

- `NonOverridableArguments` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8.

  Chaque valeur est une chaîne UTF-8.

  Les arguments de cette tâche qui ne sont pas remplacés lorsque vous fournissez des arguments de tâche dans le cadre d'une exécution de tâche, spécifiés sous forme de paires nom-valeur.

- `Connections` – Un objet ConnectionsList.

  Connexions utilisées pour la tâche.

- `MaxRetries` : nombre (entier).

  Nombre maximum de tentatives de cette tâche en cas d'échec.

- `AllocatedCapacity` : nombre (entier).

  Ce champ est obsolète. Utilisez `MaxCapacity` à la place.

  Nombre d'unités de traitement de AWS Glue données (DPUs) à allouer à cette tâche. Vous pouvez en allouer au moins 2 DPUs ; la valeur par défaut est 10. Un DPU est une mesure relative de la puissance de traitement composée de 4 V de capacité CPUs de calcul et de 16 Go de mémoire. Pour plus d'informations, consultez la page de tarification AWS Glue.

- `Timeout` – Nombre (entier), au moins égal à 1.

  Délai d'expiration de la tâche en minutes. Durée maximale durant laquelle une tâche exécutée peut consommer des ressources avant qu'elle ne se termine et n'entre dans l'état `TIMEOUT`.

  Les tâches doivent avoir des valeurs de délai d'expiration inférieures à 7 jours ou 10 080 minutes. Dans le cas contraire, les tâches déclencheront une exception.

  Si aucune valeur n'est indiquée, le délai d'expiration est défini par défaut sur 2 880 minutes.

  Toutes les AWS Glue tâches existantes dont le délai d'expiration est supérieur à 7 jours seront définies par défaut sur 7 jours. Par exemple, si vous avez spécifié un délai d'expiration de 20 jours pour un traitement par lots, celui-ci sera arrêté le 7e jour.

  Pour les tâches de streaming, si vous avez défini une fenêtre de maintenance, elles seront redémarrées au cours de celle-ci au bout de sept jours.

- `MaxCapacity` – Nombre (double).

  Pour les tâches Glue version 1.0 ou antérieures, en utilisant le type de travailleur standard, le nombre d'unités de traitement des AWS Glue données (DPUs) qui peuvent être allouées lors de l'exécution de cette tâche. Un DPU est une mesure relative de la puissance de traitement composée de 4 V de capacité CPUs de calcul et de 16 Go de mémoire. Pour plus d'informations, consultez la page de tarification AWS Glue.

  Pour les tâches des versions 2.0 et ultérieures de Glue, vous ne pouvez pas spécifier un `Maximum capacity`. Au lieu de cela, vous devez spécifier `Worker type` et `Number of workers`.

  Ne définissez pas `MaxCapacity` si vous utilisez `WorkerType` et `NumberOfWorkers`.

La valeur pouvant être attribuée à `MaxCapacity` varie selon que vous exécutez une tâche shell Python, une tâche ETL Apache Spark ou une tâche ETL Apache Spark Streaming :

- Lorsque vous spécifiez une tâche shell Python (`JobCommand.Name="pythonshell"`), vous pouvez allouer 0,0625 ou 1 DPU. La valeur par défaut correspond à 0,0625 DPU.
- Lorsque vous spécifiez une tâche ETL Apache Spark (`JobCommand.Name="glueetl »`) ou une tâche ETL de streaming Apache Spark (`JobCommand.Name="gluestreaming »`), vous pouvez en allouer de 2 à 100. DPUs La valeur par défaut est 10 DPUs. Ce type de tâche ne peut pas avoir une allocation DPU fractionnée.

- `WorkerType`— Chaîne UTF-8 (valeurs valides : `Standard="" | | G.025X="" | G.1X="" | G.2X="" | G.4X="" | G.8X="" | G.12X="" | G.16X="" | R.1X="" | R.2X="" R.4X="" R.8X="" |Z.2X=""`).

  Type de travail prédéfini qui est alloué lorsqu'une tâche est exécutée. Accepte une valeur de G.025X, G-1X, G.2X, G.4X, G.8X, G.12X, G.16X, R.1X, R.2X, R.4X ou R.8X pour les tâches Spark. Accepte la valeur Z.2X pour les tâches Ray. Pour plus d'informations, consultez Defining job properties for Spark jobs.

- `NumberOfWorkers` : nombre (entier).

  Nombre de travaux d'un `workerType` défini qui sont attribués lorsqu'une tâche est exécutée.

- `SecurityConfiguration` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la structure `SecurityConfiguration` à utiliser avec cette tâche.

- `NotificationProperty` – Un objet NotificationProperty.

  Spécifie les propriétés de configuration d'une notification de tâche.

- `GlueVersion` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #48.

  Dans les tâches Spark, `GlueVersion` détermine les versions d'Apache Spark et de Python AWS Glue disponibles dans une tâche. La version de Python indique la version prise en charge pour les tâches de type Spark.

  Les tâches Ray doivent définir `GlueVersion` sur `4.0` ou supérieur. Toutefois, les versions de Ray, de Python et des bibliothèques supplémentaires disponibles dans votre tâche Ray sont déterminées par le paramètre `Runtime` de la commande de tâche.

Pour plus d'informations sur les AWS Glue versions disponibles et les versions correspondantes de Spark et Python, consultez la version de Glue dans le guide du développeur.

Les tâches créées sans que la version de Glue soit spécifiée sont des tâches Glue 0.9 par défaut.

- `CodeGenConfigurationNodes` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, correspondant au Custom string pattern #60.

  Chaque valeur est un objet CodeGenConfigurationNode.

  Représentation d'un graphe orienté acyclique sur lequel le composant visuel Glue Studio et la génération de code Glue Studio sont basés.

- `ExecutionClass` – Chaîne UTF-8, d'une longueur maximale de 16 octets (valeurs valides : `FLEX=""` | `STANDARD=""`).

  Indique si la tâche est exécutée avec une classe d'exécution standard ou flexible. La classe d'exécution standard est idéale pour les charges de travail urgentes qui nécessitent un démarrage rapide des tâches et des ressources dédiées.

  La classe d'exécution flexible, adaptée aux tâches non urgentes dont les heures de début et de fin peuvent varier.

  Seules les tâches dotées de AWS Glue la version 3.0 ou supérieure et du type de commande `glueetl` seront autorisées à être définies `ExecutionClass` sur`FLEX`. La classe d'exécution flexible est disponible pour les tâches Spark.

- `SourceControlDetails` – Un objet SourceControlDetails.

  Les détails d'une configuration de contrôle source pour une tâche, permettant la synchronisation des artefacts de la tâche vers ou depuis un référentiel distant.

- `MaintenanceWindow` – Chaîne UTF-8, correspondant au Custom string pattern #34.

  Ce champ indique un jour de la semaine et une heure pour une fenêtre de maintenance pour les tâches de streaming. AWS Glue effectue périodiquement des activités de maintenance. Au cours de ces fenêtres de maintenance, vous AWS Glue devrez redémarrer vos tâches de streaming.

  AWS Glue redémarrera le travail dans les 3 heures suivant la période de maintenance spécifiée. Par exemple, si vous configurez la fenêtre de maintenance pour le lundi à 10h00 GMT, vos tâches seront redémarrées entre 10h00 GMT et 13h00 GMT.

- `ProfileName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom d'un profil AWS Glue d'utilisation associé à la tâche.

SourceControlDetails structure

Les détails d'une configuration de contrôle source pour une tâche, permettant la synchronisation des artefacts de la tâche vers ou depuis un référentiel distant.

Champs

- `Provider` – Chaîne UTF-8 (valeurs valides : `GITHUB` | `AWS_CODE_COMMIT`).

  Le fournisseur du référentiel distant.
- `Repository` – chaîne UTF-8, d'une longueur comprise entre 1 et 512 octets.

  Le nom du référentiel distant qui contient les artefacts de la tâche.
- `Owner` – chaîne UTF-8, d'une longueur comprise entre 1 et 512 octets.

  Le propriétaire du référentiel distant qui contient les artefacts de la tâche.
- `Branch` – chaîne UTF-8, d'une longueur comprise entre 1 et 512 octets.

  Une branche facultative dans le référentiel distant.
- `Folder` – chaîne UTF-8, d'une longueur comprise entre 1 et 512 octets.

  Un dossier facultatif dans le référentiel distant.
- `LastCommitId` – chaîne UTF-8, d'une longueur comprise entre 1 et 512 octets.

  Le dernier identifiant de validation pour une validation dans le référentiel distant.
- `LastSyncTimestamp` – chaîne UTF-8, d'une longueur comprise entre 1 et 512 octets.

  La date et l'heure auxquelles la dernière synchronisation de la tâche a été effectuée.
- `AuthStrategy` – Chaîne UTF-8 (valeurs valides : `PERSONAL_ACCESS_TOKEN` | `AWS_SECRETS_MANAGER`).

  Le type d'authentification, qui peut être un jeton d'authentification stocké dans AWS Secrets Manager ou un jeton d'accès personnel.
- `AuthToken` – chaîne UTF-8, d'une longueur comprise entre 1 et 512 octets.

La valeur d'un jeton d'autorisation.

Opérations

- CreateJob action (Python : create_job)
- UpdateJob action (Python : update_job)
- GetJob action (Python : get_job)
- GetJobs action (Python : get_jobs)
- DeleteJob action (Python : supprimer_tâche)
- ListJobs action (Python : list_jobs)
- BatchGetJobs action (Python : batch_get_jobs)

CreateJob action (Python : create_job)

Crée une nouvelle définition de tâche.

Demande

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom que vous affectez à la définition de la tâche. Doit être unique au sein de votre compte .
- `JobMode` – Chaîne UTF-8 (valeurs valides : `SCRIPT=""` | `VISUAL=""` | `NOTEBOOK=""`).

  Mode qui décrit la façon dont une tâche a été créée. Les valeurs valides sont :

  - `SCRIPT`- La tâche a été créée à l'aide de l'éditeur de script AWS Glue Studio.
  - `VISUAL`- La tâche a été créée à l'aide de l'éditeur visuel AWS Glue Studio.
  - `NOTEBOOK` : la tâche a été créée à l'aide d'un bloc-notes de sessions interactives.

  Lorsque le champ `JobMode` est manquant ou a la valeur null, `SCRIPT` est attribué comme valeur par défaut.
- `JobRunQueuingEnabled` : booléen.

  Indique si la mise en file d'attente des exécutions de tâches est activée pour les exécutions de cette tâche.

La valeur true signifie que la mise en file d'attente des tâches est activée pour les exécutions de tâches. Si la valeur est false ou si elle n'est pas renseignée, les exécutions de tâches ne seront pas prises en compte pour la mise en file d'attente.

Si ce champ ne correspond pas à la valeur définie dans l'exécution de la tâche, la valeur du champ d'exécution de la tâche sera utilisée.

- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description de la tâche définie.

- `LogUri` – Chaîne UTF-8.

  Ce champ est réservé pour un usage futur.

- `Role` – Obligatoire : chaîne UTF-8.

  Nom ou Amazon Resource Name (ARN) du rôle IAM associé à cette tâche.

- `ExecutionProperty` – Un objet ExecutionProperty.

  Objet `ExecutionProperty` spécifiant le nombre maximal d'exécutions simultanées autorisées pour cette tâche.

- `Command` – Obligatoire : un objet JobCommand.

  La commande `JobCommand` qui exécute cette tâche.

- `DefaultArguments` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8.

  Chaque valeur est une chaîne UTF-8.

  Les arguments par défaut pour chaque exécution de cette tâche, spécifiés en tant que paires nom-valeur.

  Vous pouvez spécifier ici les arguments que votre propre script d'exécution de tâches consomme, ainsi que les arguments qu'il consomme AWS Glue lui-même.

  Les arguments de la tâche peuvent être consignés. Ne transmettez pas de secrets en texte clair comme arguments. Récupérez les secrets d'une AWS Glue connexion AWS Secrets Manager

ou d'un autre mécanisme de gestion des secrets si vous avez l'intention de les conserver dans le cadre du Job.

Pour plus d'informations sur la façon de spécifier et d'utiliser vos propres arguments Job, consultez la rubrique Calling AWS Glue APIs in Python du guide du développeur.

Pour plus d'informations sur les arguments que vous pouvez fournir dans ce champ lors de la configuration des tâches Spark, consultez la rubrique Special Parameters Used by AWS Glue dans le guide du développeur.

Pour plus d'informations sur les arguments que vous pouvez fournir dans ce champ lors de la configuration des tâches Ray, consultez Utilisation des paramètres de tâches dans les tâches Ray dans le guide du développeur.

- `NonOverridableArguments` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8.

  Chaque valeur est une chaîne UTF-8.

  Les arguments de cette tâche qui ne sont pas remplacés lorsque vous fournissez des arguments de tâche dans le cadre d'une exécution de tâche, spécifiés sous forme de paires nom-valeur.

- `Connections` – Un objet ConnectionsList.

  Connexions utilisées pour la tâche.

- `MaxRetries` : nombre (entier).

  Nombre maximum de tentatives de cette tâche en cas d'échec.

- `AllocatedCapacity` : nombre (entier).

  Ce paramètre est obsolète. Utilisez `MaxCapacity` à la place.

  Le nombre d'unités de traitement de AWS Glue données (DPUs) à allouer à ce Job. Vous pouvez en allouer au moins 2 DPUs ; la valeur par défaut est 10. Un DPU est une mesure relative de la puissance de traitement composée de 4 V de capacité CPUs de calcul et de 16 Go de mémoire. Pour plus d'informations, consultez la page de tarification AWS Glue.

- `Timeout` – Nombre (entier), au moins égal à 1.

  Délai d'expiration de la tâche en minutes. Durée maximale durant laquelle une tâche exécutée peut consommer des ressources avant qu'elle ne se termine et n'entre dans l'état `TIMEOUT`.

Les tâches doivent avoir des valeurs de délai d'expiration inférieures à 7 jours ou 10 080 minutes. Dans le cas contraire, les tâches déclencheront une exception.

Si aucune valeur n'est indiquée, le délai d'expiration est défini par défaut sur 2 880 minutes.

Toutes les AWS Glue tâches existantes dont le délai d'expiration est supérieur à 7 jours seront définies par défaut sur 7 jours. Par exemple, si vous avez spécifié un délai d'expiration de 20 jours pour un traitement par lots, celui-ci sera arrêté le 7e jour.

Pour les tâches de streaming, si vous avez défini une fenêtre de maintenance, elles seront redémarrées au cours de celle-ci au bout de sept jours.

- `MaxCapacity` – Nombre (double).

  Pour les tâches Glue version 1.0 ou antérieures, en utilisant le type de travailleur standard, le nombre d'unités de traitement des AWS Glue données (DPUs) qui peuvent être allouées lors de l'exécution de cette tâche. Un DPU est une mesure relative de la puissance de traitement composée de 4 V de capacité CPUs de calcul et de 16 Go de mémoire. Pour plus d'informations, consultez la page de tarification AWS Glue.

  Pour les tâches des versions 2.0 et ultérieures de Glue, vous ne pouvez pas spécifier un `Maximum capacity`. Au lieu de cela, vous devez spécifier `Worker type` et `Number of workers`.

  Ne définissez pas `MaxCapacity` si vous utilisez `WorkerType` et `NumberOfWorkers`.

  La valeur pouvant être attribuée à `MaxCapacity` varie selon que vous exécutez une tâche shell Python, une tâche ETL Apache Spark ou une tâche ETL Apache Spark Streaming :

  - Lorsque vous spécifiez une tâche shell Python (`JobCommand.Name`="pythonshell"), vous pouvez allouer 0,0625 ou 1 DPU. La valeur par défaut correspond à 0,0625 DPU.
  - Lorsque vous spécifiez une tâche ETL Apache Spark (`JobCommand.Name`="glueetl ») ou une tâche ETL de streaming Apache Spark (`JobCommand.Name`="gluestreaming »), vous pouvez en allouer de 2 à 100. DPUs La valeur par défaut est 10 DPUs. Ce type de tâche ne peut pas avoir une allocation DPU fractionnée.
- `SecurityConfiguration` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la structure `SecurityConfiguration` à utiliser avec cette tâche.
- `Tags` – tableau de mappage de paires clé-valeur, avec 50 paires au maximum.

Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 256 octets.

Balises à utiliser avec cette tâche. Vous pouvez utiliser des balises pour limiter l'accès à la tâche. Pour plus d'informations sur les tags in AWS Glue, voir AWS Tags in AWS Glue dans le guide du développeur.

- `NotificationProperty` – Un objet NotificationProperty.

  Spécifie les propriétés de configuration d'une notification de tâche.

- `GlueVersion` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #48.

  Dans les tâches Spark, `GlueVersion` détermine les versions d'Apache Spark et de Python AWS Glue disponibles dans une tâche. La version de Python indique la version prise en charge pour les tâches de type Spark.

  Les tâches Ray doivent définir `GlueVersion` sur `4.0` ou supérieur. Toutefois, les versions de Ray, de Python et des bibliothèques supplémentaires disponibles dans votre tâche Ray sont déterminées par le paramètre `Runtime` de la commande de tâche.

  Pour plus d'informations sur les AWS Glue versions disponibles et les versions correspondantes de Spark et Python, consultez la version de Glue dans le guide du développeur.

  Les tâches créées sans que la version de Glue soit spécifiée sont des tâches Glue 0.9 par défaut.

- `NumberOfWorkers` : nombre (entier).

  Nombre de travaux d'un `workerType` défini qui sont attribués lorsqu'une tâche est exécutée.

- `WorkerType`— Chaîne UTF-8 (valeurs valides : `Standard=""` | | `G.025X=""` | `G.1X=""` | `G.2X=""` | `G.4X=""` | `G.8X=""` | `G.12X=""` | `G.16X=""` | `R.1X=""` | `R.2X=""` `R.4X=""` `R.8X=""` |`Z.2X=""`).

  Type de travail prédéfini qui est alloué lorsqu'une tâche est exécutée. Accepte une valeur de G.025X, G-1X, G.2X, G.4X, G.8X, G.12X, G.16X, R.1X, R.2X, R.4X ou R.8X pour les tâches Spark. Accepte la valeur Z.2X pour les tâches Ray.

  - Pour le type de `G.025X` worker, chaque worker est mappé sur 0,25 DPU (2 vCPUs, 4 Go de mémoire) avec un disque de 84 Go, et fournit 1 exécuteur par worker. Nous recommandons

ce type d'employé pour les travaux de streaming à faible volume. Ce type de travailleur n'est disponible que pour les tâches de streaming en AWS Glue version 3.0 ou ultérieure.

- Pour le type de `G.1X` worker, chaque worker est mappé à 1 DPU (4 vCPUs, 16 Go de mémoire) avec un disque de 94 Go, et fournit 1 exécuteur par worker. Nous recommandons ce type de travailleur pour les charges de travail telles que les transformations de données, les jointures et les requêtes, afin de proposer un moyen évolutif et rentable d'exécuter la plupart des tâches.
- Pour le type de `G.2X` travailleur, chaque utilisateur est mappé sur 2 DPU (8 VCPUs, 32 Go de mémoire) avec un disque de 138 Go et fournit un exécuteur par travailleur. Nous recommandons ce type de travailleur pour les charges de travail telles que les transformations de données, les jointures et les requêtes, afin de proposer un moyen évolutif et rentable d'exécuter la plupart des tâches.
- Pour le type de `G.4X` worker, chaque worker est mappé sur 4 DPU (16 vCPUs, 64 Go de mémoire) avec un disque de 256 Go, et fournit 1 exécuteur par worker. Nous recommandons ce type de travailleur pour les tâches dont les charges de travail contiennent les transformations, les agrégations, les jointures et les requêtes les plus exigeantes. Ce type de travailleur est disponible uniquement pour les tâches Spark ETL AWS Glue version 3.0 ou ultérieure dans les AWS régions suivantes : USA Est (Ohio), USA Est (Virginie du Nord), USA Ouest (Californie du Nord), USA Ouest (Oregon), Asie-Pacifique (Mumbai), Asie-Pacifique (Séoul), Asie-Pacifique (Sydney), Asie-Pacifique (Tokyo), Canada (Centre), Europe (Francfort), Europe (Irlande), Europe (Londres), Europe (Espagne), Europe (Stockholm), et
- Pour le type de `G.8X` travailleur, chaque travailleur est mappé sur 8 DPU (32 vCPUs, 128 Go de mémoire) avec un disque de 512 Go et fournit un exécuteur par travailleur. Nous recommandons ce type de travailleur pour les tâches dont les charges de travail contiennent les transformations, les agrégations, les jointures et les requêtes les plus exigeantes. Ce type de travailleur n'est disponible que pour les tâches Spark ETL de AWS Glue version 3.0 ou ultérieure, dans les mêmes AWS régions que celles prises en charge pour le type de `G.4X` travailleur.
- Pour le type de `G.12X` travailleur, chaque travailleur est mappé sur 12 DPU (48 vCPUs, 192 Go de mémoire) avec 768 Go de disque, et fournit un exécuteur par travailleur. Nous recommandons ce type de travailleur pour les tâches comportant de très lourdes charges de travail. Ce type de travailleur n'est disponible que pour les tâches Spark ETL de AWS Glue version 4.0 ou ultérieure.
- Pour le type de `G.16X` worker, chaque worker est mappé sur 16 DPU (64 vCPUs, 256 Go de mémoire) avec un disque de 1024 Go, et fournit 1 exécuteur par worker. Nous recommandons ce type de travailleur pour les tâches comportant de très lourdes charges de travail. Ce type

de travailleur n'est disponible que pour les tâches Spark ETL de AWS Glue version 4.0 ou ultérieure.

- Pour le type de `R.1X` worker, chaque worker est mappé à 1 DPU (4 vCPUs, 32 Go de mémoire) avec un disque de 94 Go, et fournit 1 exécuteur par worker. Nous recommandons ce type de travailleur pour les charges de travail gourmandes en mémoire. Ce type de travailleur n'est disponible que pour les tâches Spark ETL de AWS Glue version 4.0 ou ultérieure.
- Pour le type de `R.2X` worker, chaque worker est mappé sur 2 DPU (8 vCPUs, 64 Go de mémoire) avec un disque de 128 Go, et fournit 1 exécuteur par worker. Nous recommandons ce type de travailleur pour les charges de travail gourmandes en mémoire. Ce type de travailleur n'est disponible que pour les tâches Spark ETL de AWS Glue version 4.0 ou ultérieure.
- Pour le type de `R.4X` travailleur, chaque travailleur est mappé sur 4 DPU (16 vCPUs, 128 Go de mémoire) avec un disque de 256 Go et fournit un exécuteur par travailleur. Nous recommandons ce type de travailleur pour les charges de travail gourmandes en mémoire. Ce type de travailleur n'est disponible que pour les tâches Spark ETL de AWS Glue version 4.0 ou ultérieure.
- Pour le type de `R.8X` travailleur, chaque utilisateur est mappé sur 8 DPU (32 VCPUs, 256 Go de mémoire) avec un disque de 512 Go et fournit un exécuteur par travailleur. Nous recommandons ce type de travailleur pour les charges de travail gourmandes en mémoire. Ce type de travailleur n'est disponible que pour les tâches Spark ETL de AWS Glue version 4.0 ou ultérieure.
- Pour le type de `Z.2X` travailleur, chaque utilisateur est mappé sur 2 M-DPU (8 VCPUs, 64 Go de mémoire) avec un disque de 128 Go, et fournit jusqu'à 8 processeurs Ray sur la base de l'autoscaler.

- `CodeGenConfigurationNodes` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, correspondant au Custom string pattern #60.

  Chaque valeur est un objet CodeGenConfigurationNode.

  Représentation d'un graphe orienté acyclique sur lequel le composant visuel Glue Studio et la génération de code Glue Studio sont basés.

- `ExecutionClass` – Chaîne UTF-8, d'une longueur maximale de 16 octets (valeurs valides : `FLEX=""` | `STANDARD=""`).

  Indique si la tâche est exécutée avec une classe d'exécution standard ou flexible. La classe d'exécution standard est idéale pour les charges de travail urgentes qui nécessitent un démarrage rapide des tâches et des ressources dédiées.

La classe d'exécution flexible, adaptée aux tâches non urgentes dont les heures de début et de fin peuvent varier.

Seules les tâches dotées de AWS Glue la version 3.0 ou supérieure et du type de commande `glueetl` seront autorisées à être définies `ExecutionClass` sur`FLEX`. La classe d'exécution flexible est disponible pour les tâches Spark.

- `SourceControlDetails` – Un objet SourceControlDetails.

  Les détails d'une configuration de contrôle source pour une tâche, permettant la synchronisation des artefacts de la tâche vers ou depuis un référentiel distant.

- `MaintenanceWindow` – Chaîne UTF-8, correspondant au Custom string pattern #34.

  Ce champ indique un jour de la semaine et une heure pour une fenêtre de maintenance pour les tâches de streaming. AWS Glue effectue périodiquement des activités de maintenance. Au cours de ces fenêtres de maintenance, vous AWS Glue devrez redémarrer vos tâches de streaming.

  AWS Glue redémarrera le travail dans les 3 heures suivant la période de maintenance spécifiée. Par exemple, si vous configurez la fenêtre de maintenance pour le lundi à 10h00 GMT, vos tâches seront redémarrées entre 10h00 GMT et 13h00 GMT.

- `ProfileName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom d'un profil AWS Glue d'utilisation associé à la tâche.

Réponse

- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Le nom unique qui a été fourni pour cette définition de tâche.

Erreurs

- `InvalidInputException`
- `IdempotentParameterMismatchException`
- `AlreadyExistsException`
- `InternalServiceException`

- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`
- `ConcurrentModificationException`

UpdateJob action (Python : update_job)

Met à jour une définition de tâche. La définition de tâche précédente est entièrement remplacée par ces informations.

Demande

- `JobName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la définition de tâche à mettre à jour.
- `JobUpdate` – Obligatoire : un objet JobUpdate.

  Spécifie les valeurs avec lesquelles mettre à jour la définition de la tâche. Toute configuration non spécifiée est supprimée ou réinitialisée aux valeurs par défaut.
- `ProfileName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom d'un profil AWS Glue d'utilisation associé à la tâche.

Réponse

- `JobName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Retourne le nom de la définition de tâche mise à jour.

Erreurs

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`

- `ConcurrentModificationException`

GetJob action (Python : get_job)

Extrait une définition de tâche.

Demande

- `JobName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la définition de tâche à extraire.

Réponse

- `Job` – Un objet Tâche.

  Définition de tâche requise.

Erreurs

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`

GetJobs action (Python : get_jobs)

Récupère toutes les définitions de tâche actuelles.

Demande

- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, s'il s'agit d'un appel de continuation.
- `MaxResults` – Nombre (entier), compris entre 1 et 1 000.

  Taille maximale de la réponse.

Réponse

- `Jobs` : un tableau d'objets Tâche.

  Liste des définitions de tâche.
- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, si toutes les définitions de tâche n'ont pas encore été renvoyées.

Erreurs

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`

DeleteJob action (Python : supprimer_tâche)

Supprime une définition de tâche spécifiée. Si la définition de tâche est introuvable, aucune exception n'est levée.

Demande

- `JobName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la définition de tâche à supprimer.

Réponse

- `JobName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la définition de tâche qui a été supprimée.

Erreurs

- `InvalidInputException`

- `InternalServiceException`
- `OperationTimeoutException`

ListJobs action (Python : list_jobs)

Récupère les noms de toutes les ressources de travail de ce AWS compte ou des ressources portant le tag spécifié. Cette opération vous permet de voir quelles ressources sont disponibles dans votre compte, et leurs noms.

Cette opération accepte le champ `Tags` facultatif que vous pouvez utiliser comme filtre sur la réponse, afin que les ressources balisées puissent être récupérées en tant que groupe. Si vous choisissez d'utiliser le filtrage des balises, seules les ressources avec la balise sont récupérées.

Demande

- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, s'il s'agit d'une requête de continuation.
- `MaxResults` – Nombre (entier), compris entre 1 et 1 000.

  La taille maximale d'une liste à renvoyer.
- `Tags` – tableau de mappage de paires clé-valeur, avec 50 paires au maximum.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 256 octets.

  Spécifie de renvoyer uniquement les ressources balisées.

Réponse

- `JobNames` – Tableau de chaînes UTF-8.

  Noms de toutes les tâches dans le compte ou des tâches avec les balises spécifiées.
- `NextToken` – Chaîne UTF-8.

  Jeton continuation, si la liste renvoyée ne contient pas la dernière métrique disponible.

Erreurs

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`

BatchGetJobs action (Python : batch_get_jobs)

Renvoie la liste des métadonnées de ressource pour une liste donnée de noms de tâche. Après avoir appelé l'opération `ListJobs`, vous pouvez appeler cette opération pour accéder aux données sur lesquelles des autorisations vous ont été octroyées. Cette opération prend en charge toutes les autorisations IAM, y compris les conditions d'autorisation qui utilisent des balises.

Demande

- `JobNames` – obligatoire : tableau de chaînes UTF-8.

  Liste des noms de tâche, qui peuvent être les noms renvoyés à partir de l'opération `ListJobs`.

Réponse

- `Jobs` : un tableau d'objets Tâche.

  Liste des définitions de tâche.
- `JobsNotFound` – Tableau de chaînes UTF-8.

  Liste de noms de tâches introuvables.

Erreurs

- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`

Exécutions de tâches

L'API Jobs Runs décrit les types de données et l'API liés au démarrage, à l'arrêt ou à la visualisation des exécutions de tâches, ainsi qu'à la réinitialisation des signets de tâches, dans. AWS Glue L'historique d'exécution des tâches est accessible pendant 90 jours pour votre flux de travail et l'exécution des tâches.

Types de données

- JobRun structure
- Structure de Predecessor
- JobBookmarkEntry structure
- BatchStopJobRunSuccessfulSubmission structure
- BatchStopJobRunError structure
- NotificationProperty structure

JobRun structure

Contient des informations à propos d'une exécution de tâche.

Champs

- `Id` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID de cette exécution de tâche.
- `Attempt` : nombre (entier).

  Nombre de tentatives d'exécution de cette tâche.
- `PreviousRunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID de l'exécution précédente de cette tâche. Par exemple, `JobRunId` spécifié dans l'action `StartJobRun`.
- `TriggerName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

Nom du déclencheur ayant démarré cette exécution de tâche.

- `JobName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la définition de tâche utilisée dans cette exécution.

- `JobMode` – Chaîne UTF-8 (valeurs valides : `SCRIPT=""` | `VISUAL=""` | `NOTEBOOK=""`).

  Mode qui décrit la façon dont une tâche a été créée. Les valeurs valides sont :

  - `SCRIPT`- La tâche a été créée à l'aide de l'éditeur de script AWS Glue Studio.
  - `VISUAL`- La tâche a été créée à l'aide de l'éditeur visuel AWS Glue Studio.
  - `NOTEBOOK` : la tâche a été créée à l'aide d'un bloc-notes de sessions interactives.

  Lorsque le champ `JobMode` est manquant ou a la valeur null, `SCRIPT` est attribué comme valeur par défaut.

- `JobRunQueuingEnabled` : booléen.

  Indique si la mise en file d'attente des tâches est activée pour l'exécution de la tâche.

  La valeur true signifie que la mise en file d'attente des tâches est activée pour l'exécution de la tâche. Si la valeur est false ou si elle n'est pas renseignée, l'exécution de la tâche ne sera pas prise en compte pour la mise en file d'attente.

- `StartedOn` – Horodatage.

  Date et heure auxquelles cette exécution de tâche a démarré.

- `LastModifiedOn` – Horodatage.

  Heure de la dernière modification de cette exécution de tâche.

- `CompletedOn` – Horodatage.

  Date et heure auxquelles cette exécution de tâche s'est terminée.

- `JobRunState`— Chaîne UTF-8 (valeurs valides : `STARTING` | | `RUNNING` | `STOPPING` | `STOPPED` | `SUCCEEDED` | `FAILED` | `TIMEOUT` `ERROR` `WAITING` |`EXPIRED`).

  État actuel de l'exécution de tâche. Pour plus d'informations sur les statuts des tâches qui se sont terminées anormalement, consultez AWS Glue Job Run Statuses.

- `Arguments` – Tableau de mappage de paires valeur-clé.

Chaque clé est une chaîne UTF-8.

Chaque valeur est une chaîne UTF-8.

Arguments de la tâche associés à cette exécution. Pour cette exécution de tâche, ils remplacent les arguments par défaut définis pour la tâche elle-même.

Vous pouvez spécifier ici les arguments que votre propre script d'exécution de tâches consomme, ainsi que les arguments qu'il consomme AWS Glue lui-même.

Les arguments de la tâche peuvent être consignés. Ne transmettez pas de secrets en texte clair comme arguments. Récupérez les secrets d'une AWS Glue connexion AWS Secrets Manager ou d'un autre mécanisme de gestion des secrets si vous avez l'intention de les conserver dans le cadre du Job.

Pour plus d'informations sur la façon de spécifier et d'utiliser vos propres arguments Job, consultez la rubrique Calling AWS Glue APIs in Python du guide du développeur.

Pour plus d'informations sur les arguments que vous pouvez fournir dans ce champ lors de la configuration des tâches Spark, consultez la rubrique Special Parameters Used by AWS Glue dans le guide du développeur.

Pour plus d'informations sur les arguments que vous pouvez fournir dans ce champ lors de la configuration des tâches Ray, consultez Utilisation des paramètres de tâches dans les tâches Ray dans le guide du développeur.

- `ErrorMessage` – Chaîne UTF-8.

  Message d'erreur associé à cette exécution de tâche.

- `PredecessorRuns` : un tableau d'objets Predecessor.

  Liste des prédécesseurs de cette exécution de tâche.

- `AllocatedCapacity` : nombre (entier).

  Ce champ est obsolète. Utilisez `MaxCapacity` à la place.

  Le nombre d'unités de traitement de AWS Glue données (DPUs) qui y sont allouées JobRun. De 2 à 100 DPUs peuvent être alloués ; la valeur par défaut est 10. Un DPU est une mesure relative de la puissance de traitement composée de 4 V de capacité CPUs de calcul et de 16 Go de mémoire. Pour plus d'informations, consultez la page de tarification AWS Glue.

- `ExecutionTime` : nombre (entier).

  Durée (en secondes) pendant laquelle la tâche exécutée a consommé des ressources.

- `Timeout` – Nombre (entier), au moins égal à 1.

  Délai d'expiration de `JobRun` en minutes. Durée maximale durant laquelle une tâche exécutée peut consommer des ressources avant qu'elle ne se termine et n'entre dans l'état `TIMEOUT`. Cette valeur remplace la valeur définie dans la tâche parent.

  Les tâches doivent avoir des valeurs de délai d'expiration inférieures à 7 jours ou 10 080 minutes. Dans le cas contraire, les tâches déclencheront une exception.

  Si aucune valeur n'est indiquée, le délai d'expiration est défini par défaut sur 2 880 minutes.

  Toutes les AWS Glue tâches existantes dont le délai d'expiration est supérieur à 7 jours seront définies par défaut sur 7 jours. Par exemple, si vous avez spécifié un délai d'expiration de 20 jours pour un traitement par lots, celui-ci sera arrêté le 7e jour.

  Pour les tâches de streaming, si vous avez défini une fenêtre de maintenance, elles seront redémarrées au cours de celle-ci au bout de sept jours.

- `MaxCapacity` – Nombre (double).

  Pour les tâches Glue version 1.0 ou antérieures, en utilisant le type de travailleur standard, le nombre d'unités de traitement des AWS Glue données (DPUs) pouvant être allouées lors de l'exécution de cette tâche. Un DPU est une mesure relative de la puissance de traitement composée de 4 V de capacité CPUs de calcul et de 16 Go de mémoire. Pour plus d'informations, consultez la page de tarification AWS Glue.

  Pour les tâches des versions 2.0 et ultérieures de Glue, vous ne pouvez pas spécifier un `Maximum capacity`. Au lieu de cela, vous devez spécifier `Worker type` et `Number of workers`.

  Ne définissez pas `MaxCapacity` si vous utilisez `WorkerType` et `NumberOfWorkers`.

  La valeur pouvant être attribuée à `MaxCapacity` varie selon que vous exécutez une tâche shell Python, une tâche ETL Apache Spark ou une tâche ETL Apache Spark Streaming :

  - Lorsque vous spécifiez une tâche shell Python (`JobCommand.Name="pythonshell"`), vous pouvez allouer 0,0625 ou 1 DPU. La valeur par défaut correspond à 0,0625 DPU.
  - Lorsque vous spécifiez une tâche ETL Apache Spark (`JobCommand.Name="glueetl »`) ou une tâche ETL de streaming Apache Spark (`JobCommand.Name="gluestreaming »`), vous pouvez en

allouer de 2 à 100. DPUs La valeur par défaut est 10 DPUs. Ce type de tâche ne peut pas avoir une allocation DPU fractionnée.

- `WorkerType`— Chaîne UTF-8 (valeurs valides : `Standard=""` | | `G.025X=""` | `G.1X=""` | `G.2X=""` | `G.4X=""` | `G.8X=""` | `G.12X=""` | `G.16X=""` | `R.1X=""` | `R.2X=""` `R.4X=""` `R.8X=""` |`Z.2X=""`).

  Type de travail prédéfini qui est alloué lorsqu'une tâche est exécutée. Accepte une valeur de G.025X, G-1X, G.2X, G.4X, G.8X, G.12X, G.16X, R.1X, R.2X, R.4X ou R.8X pour les tâches Spark. Accepte la valeur Z.2X pour les tâches Ray.

  - Pour le type de `G.1X` worker, chaque worker est mappé à 1 DPU (4 vCPUs, 16 Go de mémoire) avec un disque de 94 Go, et fournit 1 exécuteur par worker. Nous recommandons ce type de travailleur pour les charges de travail telles que les transformations de données, les jointures et les requêtes, afin de proposer un moyen évolutif et rentable d'exécuter la plupart des tâches.
  - Pour le type de `G.2X` travailleur, chaque utilisateur est mappé sur 2 DPU (8 VCPUs, 32 Go de mémoire) avec un disque de 138 Go et fournit un exécuteur par travailleur. Nous recommandons ce type de travailleur pour les charges de travail telles que les transformations de données, les jointures et les requêtes, afin de proposer un moyen évolutif et rentable d'exécuter la plupart des tâches.
  - Pour le type de `G.4X` worker, chaque worker est mappé sur 4 DPU (16 vCPUs, 64 Go de mémoire) avec un disque de 256 Go, et fournit 1 exécuteur par worker. Nous recommandons ce type de travailleur pour les tâches dont les charges de travail contiennent les transformations, les agrégations, les jointures et les requêtes les plus exigeantes. Ce type de travailleur n'est disponible que pour les tâches Spark ETL AWS Glue version 3.0 ou ultérieure dans les AWS régions suivantes : USA Est (Ohio), USA Est (Virginie du Nord), USA Ouest (Oregon), Asie-Pacifique (Singapour), Asie-Pacifique (Sydney), Asie-Pacifique (Tokyo), Canada (Centre), Europe (Francfort), Europe (Irlande) et Europe (Stockholm).
  - Pour le type de `G.8X` travailleur, chaque travailleur est mappé sur 8 DPU (32 vCPUs, 128 Go de mémoire) avec un disque de 512 Go et fournit un exécuteur par travailleur. Nous recommandons ce type de travailleur pour les tâches dont les charges de travail contiennent les transformations, les agrégations, les jointures et les requêtes les plus exigeantes. Ce type de travailleur n'est disponible que pour les tâches Spark ETL de AWS Glue version 3.0 ou ultérieure, dans les mêmes AWS régions que celles prises en charge pour le type de `G.4X` travailleur.
  - Pour le type de `G.12X` travailleur, chaque travailleur est mappé sur 12 DPU (48 vCPUs, 192 Go de mémoire) avec 768 Go de disque, et fournit un exécuteur par travailleur. Nous recommandons ce type de travailleur pour les tâches comportant de très lourdes charges de

travail. Ce type de travailleur n'est disponible que pour les tâches Spark ETL de AWS Glue version 4.0 ou ultérieure.

- Pour le type de `G.16X` worker, chaque worker est mappé sur 16 DPU (64 vCPUs, 256 Go de mémoire) avec un disque de 1024 Go, et fournit 1 exécuteur par worker. Nous recommandons ce type de travailleur pour les tâches comportant de très lourdes charges de travail. Ce type de travailleur n'est disponible que pour les tâches Spark ETL de AWS Glue version 4.0 ou ultérieure.
- Pour le type de `G.025X` worker, chaque worker est mappé sur 0,25 DPU (2 vCPUs, 4 Go de mémoire) avec un disque de 84 Go, et fournit 1 exécuteur par worker. Nous recommandons ce type d'employé pour les travaux de streaming à faible volume. Ce type de travailleur n'est disponible que pour les tâches de streaming en AWS Glue version 3.0 ou ultérieure.
- Pour le type de `R.1X` worker, chaque worker est mappé à 1 DPU (4 vCPUs, 32 Go de mémoire) avec un disque de 94 Go, et fournit 1 exécuteur par worker. Nous recommandons ce type de travailleur pour les charges de travail gourmandes en mémoire. Ce type de travailleur n'est disponible que pour les tâches Spark ETL de AWS Glue version 4.0 ou ultérieure.
- Pour le type de `R.2X` worker, chaque worker est mappé sur 2 DPU (8 vCPUs, 64 Go de mémoire) avec un disque de 128 Go, et fournit 1 exécuteur par worker. Nous recommandons ce type de travailleur pour les charges de travail gourmandes en mémoire. Ce type de travailleur n'est disponible que pour les tâches Spark ETL de AWS Glue version 4.0 ou ultérieure.
- Pour le type de `R.4X` travailleur, chaque travailleur est mappé sur 4 DPU (16 vCPUs, 128 Go de mémoire) avec un disque de 256 Go et fournit un exécuteur par travailleur. Nous recommandons ce type de travailleur pour les charges de travail gourmandes en mémoire. Ce type de travailleur n'est disponible que pour les tâches Spark ETL de AWS Glue version 4.0 ou ultérieure.
- Pour le type de `R.8X` travailleur, chaque utilisateur est mappé sur 8 DPU (32 VCPUs, 256 Go de mémoire) avec un disque de 512 Go et fournit un exécuteur par travailleur. Nous recommandons ce type de travailleur pour les charges de travail gourmandes en mémoire. Ce type de travailleur n'est disponible que pour les tâches Spark ETL de AWS Glue version 4.0 ou ultérieure.
- Pour le type de `Z.2X` travailleur, chaque utilisateur est mappé sur 2 M-DPU (8 VCPUs, 64 Go de mémoire) avec un disque de 128 Go, et fournit jusqu'à 8 processeurs Ray sur la base de l'autoscaler.

- `NumberOfWorkers` : nombre (entier).

  Nombre de travaux d'un `workerType` défini qui sont attribués lorsqu'une tâche est exécutée.

- `SecurityConfiguration` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

Nom de la structure `SecurityConfiguration` à utiliser avec cette exécution de tâche.

- `LogGroupName` – Chaîne UTF-8.

  Le nom du groupe de journaux pour la journalisation sécurisée qui peut être chiffré côté serveur sur Amazon CloudWatch à l'aide de. AWS KMS Ce nom peut avoir pour valeur `/aws-glue/jobs/`, auquel cas le chiffrement par défaut est `NONE`. Si vous ajoutez un nom de rôle et un nom `SecurityConfiguration` (en d'autres termes, `/aws-glue/jobs-yourRoleName-yourSecurityConfigurationName/`), cette configuration de sécurité est utilisée pour chiffrer le groupe de journaux.

- `NotificationProperty` – Un objet NotificationProperty.

  Spécifie les propriétés de configuration d'une notification d'exécution de tâche.

- `GlueVersion` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #48.

  Dans les tâches Spark, `GlueVersion` détermine les versions d'Apache Spark et de Python AWS Glue disponibles dans une tâche. La version de Python indique la version prise en charge pour les tâches de type Spark.

  Les tâches Ray doivent définir `GlueVersion` sur `4.0` ou supérieur. Toutefois, les versions de Ray, de Python et des bibliothèques supplémentaires disponibles dans votre tâche Ray sont déterminées par le paramètre `Runtime` de la commande de tâche.

  Pour plus d'informations sur les AWS Glue versions disponibles et les versions correspondantes de Spark et Python, consultez la version de Glue dans le guide du développeur.

  Les tâches créées sans que la version de Glue soit spécifiée sont des tâches Glue 0.9 par défaut.

- `DPUSeconds` – Nombre (double).

  Ce champ peut être défini pour les exécutions de tâche avec la classe d'exécution `FLEX` ou lorsque autoscaling est activé, et représente la durée totale d'exécution de chaque exécuteur pendant le cycle de vie d'une tâche exécutée en secondes, multipliée par un facteur DPU (1 pour `G.1X`, 2 pour `G.2X` ou 0,25 pour `G.025X` travailleurs). Cette valeur peut être différente de la valeur `executionEngineRuntime` * `MaxCapacity` comme dans le cas des tâches Auto Scaling, car le nombre d'exécuteurs exécutés à un moment donné peut être inférieur à la `MaxCapacity`. Par conséquent, il est possible que la valeur de `DPUSeconds` soit inférieure à `executionEngineRuntime` * `MaxCapacity`.

- `ExecutionClass` – Chaîne UTF-8, d'une longueur maximale de 16 octets (valeurs valides : `FLEX=""` | `STANDARD=""`).

  Indique si la tâche est exécutée avec une classe d'exécution standard ou flexible. La classe d'exécution standard est idéale pour les charges de travail urgentes qui nécessitent un démarrage rapide des tâches et des ressources dédiées.

  La classe d'exécution flexible, adaptée aux tâches non urgentes dont les heures de début et de fin peuvent varier.

  Seules les tâches dotées de AWS Glue la version 3.0 ou supérieure et du type de commande `glueetl` seront autorisées à être définies `ExecutionClass` sur`FLEX`. La classe d'exécution flexible est disponible pour les tâches Spark.
- `MaintenanceWindow` – Chaîne UTF-8, correspondant au Custom string pattern #34.

  Ce champ indique un jour de la semaine et une heure pour une fenêtre de maintenance pour les tâches de streaming. AWS Glue effectue périodiquement des activités de maintenance. Au cours de ces fenêtres de maintenance, vous AWS Glue devrez redémarrer vos tâches de streaming.

  AWS Glue redémarrera le travail dans les 3 heures suivant la période de maintenance spécifiée. Par exemple, si vous configurez la fenêtre de maintenance pour le lundi à 10h00 GMT, vos tâches seront redémarrées entre 10h00 GMT et 13h00 GMT.
- `ProfileName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom d'un profil AWS Glue d'utilisation associé à la tâche exécutée.
- `StateDetail` – Chaîne UTF-8, d'une longueur maximale de 400 000 octets.

  Ce champ contient des informations relatives à l'état d'une exécution de tâche. Le champ autorise les valeurs Null.

  Par exemple, lorsqu'une tâche est exécutée dans un état WAITING à la suite d'une mise en file d'attente, le champ indique la raison pour laquelle la tâche exécutée est dans cet état.
- `ExecutionRoleSessionPolicy` : chaîne UTF-8, d'une longueur comprise entre 2 et 2 048 octets.

  Cette politique de session intégrée à l' StartJobRun API vous permet de restreindre de manière dynamique les autorisations du rôle d'exécution spécifié pour l'étendue de la tâche, sans avoir à créer de rôles IAM supplémentaires.

Structure de Predecessor

Exécution de tâche utilisée dans le prédicat d'un déclencheur conditionnel ayant déclenché cette exécution de tâche.

Champs

- `JobName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la définition de tâche utilisée par la tâche exécutée précédente.
- `RunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID d'exécution de tâche de l'exécution de tâche précédente.

JobBookmarkEntry structure

Définit un point à partir duquel une tâche peut reprendre le traitement.

Champs

- `JobName` – Chaîne UTF-8.

  Nom de la tâche en question.
- `Version` : nombre (entier).

  Version de la tâche.
- `Run` : nombre (entier).

  Numéro d'identification de l'exécution.
- `Attempt` : nombre (entier).

  Numéro d'identification de la tentative.
- `PreviousRunId` – Chaîne UTF-8.

  Identifiant d'exécution unique associé à l'exécution de tâche précédente.
- `RunId` – Chaîne UTF-8.

  Numéro d'identification de l'exécution.

- `JobBookmark` – Chaîne UTF-8.

  Le marque-page lui-même.

BatchStopJobRunSuccessfulSubmission structure

Enregistre une requête réussie visant à arrêter une `JobRun` spécifiée.

Champs

- `JobName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la définition de tâche utilisée par la tâche exécutée qui a été arrêtée.
- `JobRunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  `JobRunId` de la tâche qui a été arrêtée.

BatchStopJobRunError structure

Enregistre une erreur qui s'est produite lors de la tentative d'arrêt d'une tâche exécutée spécifiée.

Champs

- `JobName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la définition de tâche utilisée par la tâche exécutée en question.
- `JobRunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  `JobRunId` de l'exécution de tâche en question.
- `ErrorDetail` – Un objet ErrorDetail.

  Spécifie les détails de l'erreur qui s'est produite.

NotificationProperty structure

Spécifie les propriétés de configuration d'une notification.

Champs

- `NotifyDelayAfter` – Nombre (entier), au moins égal à 1.

  Après le démarrage d'une exécution de tâche, nombre de minutes d'attente avant l'envoi d'une notification de délai d'exécution de tâche.

Opérations

- StartJobRun action (Python : start_job_run)
- BatchStopJobRun action (Python : batch_stop_job_run)
- GetJobRun action (Python : get_job_run)
- GetJobRuns action (Python : get_job_runs)
- GetJobBookmark action (Python : get_job_bookmark)
- GetJobBookmarks action (Python : get_job_bookmarks)
- ResetJobBookmark action (Python : reset_job_bookmark)

StartJobRun action (Python : start_job_run)

Démarre une exécution de tâche à l'aide d'une définition de tâche.

Demande

- `JobName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la définition de tâche à utiliser.
- `JobRunQueuingEnabled` : booléen.

  Indique si la mise en file d'attente des tâches est activée pour l'exécution de la tâche.

  La valeur true signifie que la mise en file d'attente des tâches est activée pour l'exécution de la tâche. Si la valeur est false ou si elle n'est pas renseignée, l'exécution de la tâche ne sera pas prise en compte pour la mise en file d'attente.

- `JobRunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID d'une précédente `JobRun` pour une nouvelle tentative.

- `Arguments` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8.

  Chaque valeur est une chaîne UTF-8.

  Arguments de la tâche associés à cette exécution. Pour cette exécution de tâche, ils remplacent les arguments par défaut définis pour la tâche elle-même.

  Vous pouvez spécifier ici les arguments que votre propre script d'exécution de tâches consomme, ainsi que les arguments qu'il consomme AWS Glue lui-même.

  Les arguments de la tâche peuvent être consignés. Ne transmettez pas de secrets en texte clair comme arguments. Récupérez les secrets d'une AWS Glue connexion AWS Secrets Manager ou d'un autre mécanisme de gestion des secrets si vous avez l'intention de les conserver dans le cadre du Job.

  Pour plus d'informations sur la façon de spécifier et d'utiliser vos propres arguments Job, consultez la rubrique Calling AWS Glue APIs in Python du guide du développeur.

  Pour plus d'informations sur les arguments que vous pouvez fournir dans ce champ lors de la configuration des tâches Spark, consultez la rubrique Special Parameters Used by AWS Glue dans le guide du développeur.

  Pour plus d'informations sur les arguments que vous pouvez fournir dans ce champ lors de la configuration des tâches Ray, consultez Utilisation des paramètres de tâches dans les tâches Ray dans le guide du développeur.

- `AllocatedCapacity` : nombre (entier).

  Ce champ est obsolète. Utilisez `MaxCapacity` à la place.

  Le nombre d'unités de traitement de AWS Glue données (DPUs) à y affecter JobRun. Vous pouvez en allouer au moins 2 DPUs ; la valeur par défaut est 10. Un DPU est une mesure relative de la puissance de traitement composée de 4 V de capacité CPUs de calcul et de 16 Go de mémoire. Pour plus d'informations, consultez la page de tarification AWS Glue.

- `Timeout` – Nombre (entier), au moins égal à 1.

  Délai d'expiration de `JobRun` en minutes. Durée maximale durant laquelle une tâche exécutée peut consommer des ressources avant qu'elle ne se termine et n'entre dans l'état `TIMEOUT`. Cette valeur remplace la valeur définie dans la tâche parent.

  Les tâches doivent avoir des valeurs de délai d'expiration inférieures à 7 jours ou 10 080 minutes. Dans le cas contraire, les tâches déclencheront une exception.

  Si aucune valeur n'est indiquée, le délai d'expiration est défini par défaut sur 2 880 minutes.

  Toutes les AWS Glue tâches existantes dont le délai d'expiration est supérieur à 7 jours seront définies par défaut sur 7 jours. Par exemple, si vous avez spécifié un délai d'expiration de 20 jours pour un traitement par lots, celui-ci sera arrêté le 7e jour.

  Pour les tâches de streaming, si vous avez défini une fenêtre de maintenance, elles seront redémarrées au cours de celle-ci au bout de sept jours.

- `MaxCapacity` – Nombre (double).

  Pour les tâches Glue version 1.0 ou antérieures, en utilisant le type de travailleur standard, le nombre d'unités de traitement des AWS Glue données (DPUs) pouvant être allouées lors de l'exécution de cette tâche. Un DPU est une mesure relative de la puissance de traitement composée de 4 V de capacité CPUs de calcul et de 16 Go de mémoire. Pour plus d'informations, consultez la page de tarification AWS Glue.

  Pour les tâches des versions 2.0 et ultérieures de Glue, vous ne pouvez pas spécifier un `Maximum capacity`. Au lieu de cela, vous devez spécifier `Worker type` et `Number of workers`.

  Ne définissez pas `MaxCapacity` si vous utilisez `WorkerType` et `NumberOfWorkers`.

  La valeur pouvant être attribuée à `MaxCapacity` varie selon que vous exécutez une tâche shell Python, une tâche ETL Apache Spark ou une tâche ETL Apache Spark Streaming :

  - Lorsque vous spécifiez une tâche shell Python (`JobCommand.Name="pythonshell"`), vous pouvez allouer 0,0625 ou 1 DPU. La valeur par défaut correspond à 0,0625 DPU.
  - Lorsque vous spécifiez une tâche ETL Apache Spark (`JobCommand.Name="glueetl »`) ou une tâche ETL de streaming Apache Spark (`JobCommand.Name="gluestreaming »`), vous pouvez en allouer de 2 à 100. DPUs La valeur par défaut est 10 DPUs. Ce type de tâche ne peut pas avoir une allocation DPU fractionnée.

- `SecurityConfiguration` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la structure `SecurityConfiguration` à utiliser avec cette exécution de tâche.

- `NotificationProperty` – Un objet NotificationProperty.

  Spécifie les propriétés de configuration d'une notification d'exécution de tâche.

- `WorkerType` – Chaîne UTF-8 (valeurs valides: `Standard=""` | `G.1X=""` | `G.2X=""` | `G.025X=""` | `G.4X=""` | `G.8X=""` | `Z.2X=""`).

  Type de travail prédéfini qui est alloué lorsqu'une tâche est exécutée. Accepte une valeur de G.1X, G.2X, G.4X, G.8X ou G.025X pour les tâches Spark. Accepte la valeur Z.2X pour les tâches Ray.

  - Pour le type de `G.1X` worker, chaque worker est mappé à 1 DPU (4 vCPUs, 16 Go de mémoire) avec un disque de 94 Go, et fournit 1 exécuteur par worker. Nous recommandons ce type de travailleur pour les charges de travail telles que les transformations de données, les jointures et les requêtes, afin de proposer un moyen évolutif et rentable d'exécuter la plupart des tâches.
  - Pour le type de `G.2X` travailleur, chaque utilisateur est mappé sur 2 DPU (8 VCPUs, 32 Go de mémoire) avec un disque de 138 Go et fournit un exécuteur par travailleur. Nous recommandons ce type de travailleur pour les charges de travail telles que les transformations de données, les jointures et les requêtes, afin de proposer un moyen évolutif et rentable d'exécuter la plupart des tâches.
  - Pour le type de `G.4X` worker, chaque worker est mappé sur 4 DPU (16 vCPUs, 64 Go de mémoire) avec un disque de 256 Go, et fournit 1 exécuteur par worker. Nous recommandons ce type de travailleur pour les tâches dont les charges de travail contiennent les transformations, les agrégations, les jointures et les requêtes les plus exigeantes. Ce type de travailleur n'est disponible que pour les tâches Spark ETL AWS Glue version 3.0 ou ultérieure dans les AWS régions suivantes : USA Est (Ohio), USA Est (Virginie du Nord), USA Ouest (Oregon), Asie-Pacifique (Singapour), Asie-Pacifique (Sydney), Asie-Pacifique (Tokyo), Canada (Centre), Europe (Francfort), Europe (Irlande) et Europe (Stockholm).
  - Pour le type de `G.8X` travailleur, chaque travailleur est mappé sur 8 DPU (32 vCPUs, 128 Go de mémoire) avec un disque de 512 Go et fournit un exécuteur par travailleur. Nous recommandons ce type de travailleur pour les tâches dont les charges de travail contiennent les transformations, les agrégations, les jointures et les requêtes les plus exigeantes. Ce type de travailleur n'est disponible que pour les tâches Spark ETL de AWS Glue version 3.0 ou ultérieure, dans les mêmes AWS régions que celles prises en charge pour le type de `G.4X` travailleur.

- Pour le type de `G.12X` travailleur, chaque travailleur est mappé sur 12 DPU (48 vCPUs, 192 Go de mémoire) avec 768 Go de disque, et fournit un exécuteur par travailleur. Nous recommandons ce type de travailleur pour les tâches comportant de très lourdes charges de travail. Ce type de travailleur n'est disponible que pour les tâches Spark ETL de AWS Glue version 4.0 ou ultérieure.
- Pour le type de `G.16X` worker, chaque worker est mappé sur 16 DPU (64 vCPUs, 256 Go de mémoire) avec un disque de 1024 Go, et fournit 1 exécuteur par worker. Nous recommandons ce type de travailleur pour les tâches comportant de très lourdes charges de travail. Ce type de travailleur n'est disponible que pour les tâches Spark ETL de AWS Glue version 4.0 ou ultérieure.
- Pour le type de `G.025X` worker, chaque worker est mappé sur 0,25 DPU (2 vCPUs, 4 Go de mémoire) avec un disque de 84 Go, et fournit 1 exécuteur par worker. Nous recommandons ce type d'employé pour les travaux de streaming à faible volume. Ce type de travailleur n'est disponible que pour les tâches de streaming en AWS Glue version 3.0 ou ultérieure.
- Pour le type de `R.1X` worker, chaque worker est mappé à 1 DPU (4 vCPUs, 32 Go de mémoire) avec un disque de 94 Go, et fournit 1 exécuteur par worker. Nous recommandons ce type de travailleur pour les charges de travail gourmandes en mémoire. Ce type de travailleur n'est disponible que pour les tâches Spark ETL de AWS Glue version 4.0 ou ultérieure.
- Pour le type de `R.2X` worker, chaque worker est mappé sur 2 DPU (8 vCPUs, 64 Go de mémoire) avec un disque de 128 Go, et fournit 1 exécuteur par worker. Nous recommandons ce type de travailleur pour les charges de travail gourmandes en mémoire. Ce type de travailleur n'est disponible que pour les tâches Spark ETL de AWS Glue version 4.0 ou ultérieure.
- Pour le type de `R.4X` travailleur, chaque travailleur est mappé sur 4 DPU (16 vCPUs, 128 Go de mémoire) avec un disque de 256 Go et fournit un exécuteur par travailleur. Nous recommandons ce type de travailleur pour les charges de travail gourmandes en mémoire. Ce type de travailleur n'est disponible que pour les tâches Spark ETL de AWS Glue version 4.0 ou ultérieure.
- Pour le type de `R.8X` travailleur, chaque utilisateur est mappé sur 8 DPU (32 VCPUs, 256 Go de mémoire) avec un disque de 512 Go et fournit un exécuteur par travailleur. Nous recommandons ce type de travailleur pour les charges de travail gourmandes en mémoire. Ce type de travailleur n'est disponible que pour les tâches Spark ETL de AWS Glue version 4.0 ou ultérieure.
- Pour le type de `Z.2X` travailleur, chaque utilisateur est mappé sur 2 M-DPU (8 VCPUs, 64 Go de mémoire) avec un disque de 128 Go, et fournit jusqu'à 8 processeurs Ray sur la base de l'autoscaler.

- `NumberOfWorkers` : nombre (entier).

Nombre de travaux d'un `workerType` défini qui sont attribués lorsqu'une tâche est exécutée.

- `ExecutionClass` – Chaîne UTF-8, d'une longueur maximale de 16 octets (valeurs valides : `FLEX=""` | `STANDARD=""`).

  Indique si la tâche est exécutée avec une classe d'exécution standard ou flexible. La classe d'exécution standard est idéale pour les charges de travail urgentes qui nécessitent un démarrage rapide des tâches et des ressources dédiées.

  La classe d'exécution flexible, adaptée aux tâches non urgentes dont les heures de début et de fin peuvent varier.

  Seules les tâches dotées de AWS Glue la version 3.0 ou supérieure et du type de commande `glueetl` seront autorisées à être définies `ExecutionClass` sur`FLEX`. La classe d'exécution flexible est disponible pour les tâches Spark.
- `ProfileName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom d'un profil AWS Glue d'utilisation associé à la tâche exécutée.
- `ExecutionRoleSessionPolicy` : chaîne UTF-8, d'une longueur comprise entre 2 et 2 048 octets.

  Cette politique de session intégrée à l' StartJobRun API vous permet de restreindre de manière dynamique les autorisations du rôle d'exécution spécifié pour l'étendue de la tâche, sans avoir à créer de rôles IAM supplémentaires.

Réponse

- `JobRunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID attribuée à cette exécution de tâche.

Erreurs

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`

- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`
- `ConcurrentRunsExceededException`

BatchStopJobRun action (Python : batch_stop_job_run)

Arrête une ou plusieurs exécutions de tâche pour une définition de tâche spécifiée.

Demande

- `JobName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la définition de tâche pour laquelle arrêter les exécutions de tâche.
- `JobRunIds` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 25 chaînes maximum.

  Liste des `JobRunIds` qui doivent être arrêtés pour cette définition de tâche.

Réponse

- `SuccessfulSubmissions` : un tableau d'objets BatchStopJobRunSuccessfulSubmission.

  Une liste de ceux JobRuns qui ont été soumis avec succès pour arrêt.
- `Errors` – Un tableau BatchStopJobRunError d'objets.

  Liste des erreurs qui se sont produites lors de la tentative d'arrêt de `JobRuns`, incluant la `JobRunId` pour laquelle chaque erreur s'est produite et les détails de l'erreur.

Erreurs

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

GetJobRun action (Python : get_job_run)

Récupère les métadonnées d'une exécution de tâche donnée. L'historique d'exécution des tâches est accessible pendant 365 jours pour votre flux de travail et l'exécution des tâches.

Demande

- `JobName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la définition de tâche en cours d'exécution.
- `RunId` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID de l'exécution de tâche.
- `PredecessorsIncluded` : booléen.

  True si une liste d'exécutions précédentes doit être renvoyée.

Réponse

- `JobRun` – Un objet JobRun.

  Métadonnées de l'exécution de tâche demandée.

Erreurs

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`

GetJobRuns action (Python : get_job_runs)

Récupère les métadonnées de toutes les exécutions d'une définition de tâche donnée.

`GetJobRuns` renvoie les tâches exécutées par ordre chronologique, les tâches les plus récentes étant renvoyées en premier.

Demande

- `JobName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la définition de tâche pour laquelle récupérer toutes les exécutions de tâche.
- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, s'il s'agit d'un appel de continuation.
- `MaxResults`— Nombre (entier), pas moins de 1 ou plus de 200.

  Taille maximale de la réponse.

Réponse

- `JobRuns` : un tableau d'objets JobRun.

  Liste des objets de métadonnées de l'exécution de tâche.
- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, si toutes les exécutions de tâche demandées ne sont pas renvoyées.

Erreurs

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`

GetJobBookmark action (Python : get_job_bookmark)

Renvoie des informations sur une entrée de marque-page de tâche.

Pour plus d'informations sur l'activation et l'utilisation des marque-pages de tâche, consultez :

- Suivi des données traitées à l'aide de signets de tâche
- Paramètres du job utilisés par AWS Glue

- Structure de tâche

Demande

- `JobName` – Obligatoire : chaîne UTF-8.

  Nom de la tâche en question.
- `Version` : nombre (entier).

  Version de la tâche.
- `RunId` – Chaîne UTF-8.

  Identifiant d'exécution unique associé à l'exécution de cette tâche.

Réponse

- `JobBookmarkEntry` – Un objet JobBookmarkEntry.

  Structure qui définit un point à partir duquel une tâche peut reprendre le traitement.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ValidationException`

GetJobBookmarks action (Python : get_job_bookmarks)

Renvoie des informations sur les données de marque-page de tâche. La liste est classée par numéros de version décroissants.

Pour plus d'informations sur l'activation et l'utilisation des marque-pages de tâche, consultez :

- Suivi des données traitées à l'aide de signets de tâche
- Paramètres du job utilisés par AWS Glue

- Structure de tâche

Demande

- `JobName` – Obligatoire : chaîne UTF-8.

  Nom de la tâche en question.
- `MaxResults` : nombre (entier).

  Taille maximale de la réponse.
- `NextToken` : nombre (entier).

  Jeton de continuation, s'il s'agit d'un appel de continuation.

Réponse

- `JobBookmarkEntries` : un tableau d'objets JobBookmarkEntry.

  Liste de données de marque-page de tâche qui définit un point à partir duquel une tâche peut reprendre son exécution.
- `NextToken` : nombre (entier).

  Jeton de continuation, qui prend la valeur 1 si toutes les entrées sont renvoyées, ou > 1 si toutes les exécutions de tâches demandées n'ont pas été renvoyées.

Erreurs

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`

ResetJobBookmark action (Python : reset_job_bookmark)

Réinitialise une donnée de marque-page.

Pour plus d'informations sur l'activation et l'utilisation des marque-pages de tâche, consultez :

- Suivi des données traitées à l'aide de signets de tâche
- Paramètres du job utilisés par AWS Glue
- Structure de tâche

Demande

- `JobName` – Obligatoire : chaîne UTF-8.

  Nom de la tâche en question.
- `RunId` – Chaîne UTF-8.

  Identifiant d'exécution unique associé à l'exécution de cette tâche.

Réponse

- `JobBookmarkEntry` – Un objet JobBookmarkEntry.

  Donnée de marque-page de réinitialisation.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

Triggers

L'API Triggers décrit les types de données et l'API liés à la création, à la mise à jour ou à la suppression, ainsi qu'au démarrage et à l'arrêt des déclencheurs de tâches dans AWS Glue.

Types de données

- Structure Trigger
- TriggerUpdate structure
- Structure Predicate

- Structure Condition
- Structure Action
- EventBatchingCondition structure

Structure Trigger

Informations concernant un déclencheur spécifique.

Champs

- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Le nom du déclencheur.
- `WorkflowName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du flux de travail associé au déclencheur.
- `Id` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Réservé pour un usage futur.
- `Type` – Chaîne UTF-8 (valeurs valides : `SCHEDULED` | `CONDITIONAL` | `ON_DEMAND` | `EVENT`).

  Type de déclencheur.
- `State` – Chaîne UTF-8 (valeurs valides : `CREATING` | `CREATED` | `ACTIVATING` | `ACTIVATED` | `DEACTIVATING` | `DEACTIVATED` | `DELETING` | `UPDATING`).

  État actuel du déclencheur.
- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description de ce déclencheur.
- `Schedule` – Chaîne UTF-8.

  Une expression `cron` utilisée pour spécifier la planification (consultez Time-Based Schedules for Jobs and Crawlers (Planifications temporelles pour les tâches et les crawlers)). Par exemple, pour exécuter un élément tous les jours à 12h15 UTC, vous devez spécifier : `cron(15 12 * * ? *)`.

- `Actions` – Un tableau Action d'objets.

  Actions initiées par ce déclencheur.

- `Predicate` – Un objet Prédicat.

  Prédicat de ce déclencheur, qui définit le moment du déclenchement.

- `EventBatchingCondition` – Un objet EventBatchingCondition.

  Condition de lot qui doit être remplie (nombre spécifié d'événements reçus ou délai de traitement du lot expiré) avant que EventBridge l'événement ne se déclenche.

TriggerUpdate structure

Structure utilisée pour fournir des informations utilisées pour mettre à jour un déclencheur. Cet objet met à jour la définition précédente du déclencheur en la remplaçant complètement.

Champs

- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Réservé pour un usage futur.

- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description de ce déclencheur.

- `Schedule` – Chaîne UTF-8.

  Une expression `cron` utilisée pour spécifier la planification (consultez Time-Based Schedules for Jobs and Crawlers (Planifications temporelles pour les tâches et les crawlers)). Par exemple, pour exécuter un élément tous les jours à 12h15 UTC, vous devez spécifier : `cron(15 12 * * ? *)`.

- `Actions` – Un tableau Action d'objets.

  Actions initiées par ce déclencheur.

- `Predicate` – Un objet Prédicat.

  Prédicat de ce déclencheur, qui définit le moment du déclenchement.

- `EventBatchingCondition` – Un objet EventBatchingCondition.

Condition de lot qui doit être remplie (nombre spécifié d'événements reçus ou délai de traitement du lot expiré) avant que EventBridge l'événement ne se déclenche.

Structure Predicate

Définit le prédicat du déclencheur, qui détermine le moment du déclenchement.

Champs

- `Logical` – Chaîne UTF-8 (valeurs valides : `AND` | `ANY`).

  Champ facultatif si une seule condition est répertoriée. Si plusieurs conditions sont répertoriées, ce champ est obligatoire.
- `Conditions`— Un ensemble d'Conditionobjets, pas plus de 500 structures.

  Liste des conditions qui déterminent le moment du déclenchement.

Structure Condition

Définit une condition selon laquelle un déclencheur s'exécute.

Champs

- `LogicalOperator` – Chaîne UTF-8 (valeurs valides : `EQUALS`).

  Opérateur logique.
- `JobName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la tâche pour laquelle cette condition s'applique à `JobRuns` et sur laquelle ce déclencheur attend.
- `State`— Chaîne UTF-8 (valeurs valides : `STARTING` | | `RUNNING` | `STOPPING` | `STOPPED` | `SUCCEEDED` | `FAILED` | `TIMEOUT` `ERROR` `WAITING` |`EXPIRED`).

  État de la condition. Actuellement, les seuls états de tâche qu'un déclencheur peut écouter sont `SUCCEEDED`, `STOPPED`, `FAILED` et `TIMEOUT`. Les seuls états d'crawler qu'un déclencheur peut écouter sont `SUCCEEDED`, `FAILED` et `CANCELLED`.

- `CrawlerName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de l'crawler à laquelle cette condition s'applique.

- `CrawlState` – Chaîne UTF-8 (valeurs valides : `RUNNING` | `CANCELLING` | `CANCELLED` | `SUCCEEDED` | `FAILED` | `ERROR`).

  Nom de l'crawler à laquelle cette condition s'applique.

Structure Action

Définit une action qui doit être initiée par un déclencheur.

Champs

- `JobName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la tâche à exécuter.

- `Arguments` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8.

  Chaque valeur est une chaîne UTF-8.

  Arguments de tâche utilisés lorsque ce déclencheur s'exécute. Pour cette exécution de tâche, ils remplacent les arguments par défaut définis pour la tâche elle-même.

  Vous pouvez spécifier ici les arguments que votre propre script d'exécution de tâches consomme, ainsi que les arguments qu'il consomme AWS Glue lui-même.

  Pour plus d'informations sur la façon de spécifier et d'utiliser vos propres arguments Job, consultez la rubrique Calling AWS Glue APIs in Python du guide du développeur.

  Pour plus d'informations sur les paires clé-valeur nécessaires AWS Glue à la configuration de votre tâche, consultez la AWS Glue rubrique Paramètres spéciaux utilisés par dans le guide du développeur.

- `Timeout` – Nombre (entier), au moins égal à 1.

Délai d'expiration de `JobRun` en minutes. Durée maximale durant laquelle une tâche exécutée peut consommer des ressources avant qu'elle ne se termine et n'entre dans l'état `TIMEOUT`. Cette valeur remplace la valeur définie dans la tâche parent.

Les tâches doivent avoir des valeurs de délai d'expiration inférieures à 7 jours ou 10 080 minutes. Dans le cas contraire, les tâches déclencheront une exception.

Si aucune valeur n'est indiquée, le délai d'expiration est défini par défaut sur 2 880 minutes.

Toutes les AWS Glue tâches existantes dont le délai d'expiration est supérieur à 7 jours seront définies par défaut sur 7 jours. Par exemple, si vous avez spécifié un délai d'expiration de 20 jours pour un traitement par lots, celui-ci sera arrêté le 7e jour.

Pour les tâches de streaming, si vous avez défini une fenêtre de maintenance, elles seront redémarrées au cours de celle-ci au bout de sept jours.

- `SecurityConfiguration` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la structure `SecurityConfiguration` à utiliser avec cette action.

- `NotificationProperty` – Un objet NotificationProperty.

  Spécifie les propriétés de configuration d'une notification d'exécution de tâche.

- `CrawlerName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de l'crawler à utiliser avec cette action.

EventBatchingCondition structure

Condition de lot qui doit être remplie (nombre spécifié d'événements reçus ou délai de traitement du lot expiré) avant que EventBridge l'événement ne se déclenche.

Champs

- `BatchSize` – obligatoire : nombre (entier), compris entre 1 et 100.

  Nombre d'événements qui doivent être reçus d'Amazon EventBridge avant que l' EventBridge événement ne déclenche un incendie.

- `BatchWindow` – nombre (entier), compris entre 1 et 900.

  Fenêtre de temps en secondes après laquelle EventBridge l'événement déclenche des incendies. La fenêtre démarre dès la réception du premier événement.

Opérations

- CreateTrigger action (Python : create_trigger)
- StartTrigger action (Python : start_trigger)
- GetTrigger action (Python : get_trigger)
- GetTriggers action (Python : get_triggers)
- UpdateTrigger action (Python : update_trigger)
- StopTrigger action (Python : stop_trigger)
- DeleteTrigger action (Python : delete_trigger)
- ListTriggers action (Python : list_triggers)
- BatchGetTriggers action (Python : batch_get_triggers)

CreateTrigger action (Python : create_trigger)

Crée un nouveau déclencheur.

Les arguments de la tâche peuvent être consignés. Ne transmettez pas de secrets en texte clair comme arguments. Récupérez les secrets d'une AWS Glue connexion, d'un Gestionnaire de AWS Secrets ou d'un autre mécanisme de gestion des secrets si vous avez l'intention de les conserver dans le cadre du Job.

Demande

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Le nom du déclencheur.

- `WorkflowName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du flux de travail associé au déclencheur.

- `Type` – obligatoire : chaîne UTF-8 (valeurs valides : `SCHEDULED` | `CONDITIONAL` | `ON_DEMAND` | `EVENT`).

  Type du nouveau déclencheur.

- `Schedule` – Chaîne UTF-8.

  Une expression `cron` utilisée pour spécifier la planification (consultez Time-Based Schedules for Jobs and Crawlers (Planifications temporelles pour les tâches et les crawlers)). Par exemple, pour exécuter un élément tous les jours à 12h15 UTC, vous devez spécifier : `cron(15 12 * * ? *)`.

  Ce champ est obligatoire lorsque le type de déclencheur est SCHEDULED.

- `Predicate` – Un objet Prédicat.

  Prédicat pour spécifier le moment du déclenchement.

  Ce champ est obligatoire lorsque le type de déclencheur est `CONDITIONAL`.

- `Actions` – Obligatoire : un tableau d'objets Action.

  Actions initiées par ce déclencheur lorsqu'il s'exécute.

- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description du nouveau déclencheur.

- `StartOnCreation` : booléen.

  Définissez ce paramètre sur `true` pour démarrer les déclencheurs `SCHEDULED` et `CONDITIONAL` lorsqu'ils sont créés. True n'est pas pris en charge pour les déclencheurs `ON_DEMAND`.

- `Tags` – tableau de mappage de paires clé-valeur, avec 50 paires au maximum.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 256 octets.

  Balises à utiliser avec ce déclencheur. Vous pouvez utiliser des balises pour limiter l'accès au déclencheur. Pour plus d'informations sur les tags in AWS Glue, voir AWS Tags in AWS Glue dans le guide du développeur.

- `EventBatchingCondition` : un objet EventBatchingCondition.

Condition de lot qui doit être remplie (nombre spécifié d'événements reçus ou délai de traitement du lot expiré) avant que EventBridge l'événement ne se déclenche.

Réponse

- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Le nom du déclencheur.

Erreurs

- `AlreadyExistsException`
- `EntityNotFoundException`
- `InvalidInputException`
- `IdempotentParameterMismatchException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`
- `ConcurrentModificationException`

StartTrigger action (Python : start_trigger)

Lance un déclencheur existant. Consultez Déclenchement de tâches pour obtenir des informations sur la manière dont différents types de déclencheurs sont lancés.

Demande

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du déclencheur à lancer.

Réponse

- Name – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du déclencheur qui a été lancé.

Erreurs

- `InvalidInputException`
- `InternalServiceException`
- `EntityNotFoundException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`
- `ConcurrentRunsExceededException`

GetTrigger action (Python : get_trigger)

Extrait la définition d'un déclencheur.

Demande

- Name – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du déclencheur à extraire.

Réponse

- `Trigger` – Un objet Déclencheur.

  Définition du déclencheur demandé.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`

- `InternalServiceException`
- `OperationTimeoutException`

GetTriggers action (Python : get_triggers)

Permet d'obtenir tous les déclencheurs associés à une tâche.

Demande

- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, s'il s'agit d'un appel de continuation.

- `DependentJobName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la tâche pour laquelle récupérer des déclencheurs. Le déclencheur qui peut lancer cette tâche est renvoyé. Si aucun déclencheur de ce type n'existe, tous les déclencheurs sont renvoyés.

- `MaxResults`— Nombre (entier), pas moins de 1 ou plus de 200.

  Taille maximale de la réponse.

Réponse

- `Triggers` : un tableau d'objets Déclencheur.

  Liste des déclencheurs pour la tâche spécifiée.

- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, si tous les déclencheurs demandés ne sont pas encore renvoyés.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

UpdateTrigger action (Python : update_trigger)

Met à jour la définition d'un déclencheur.

Les arguments de la tâche peuvent être consignés. Ne transmettez pas de secrets en texte clair comme arguments. Récupérez les secrets d'une AWS Glue connexion, d'un Gestionnaire de AWS Secrets ou d'un autre mécanisme de gestion des secrets si vous avez l'intention de les conserver dans le cadre du Job.

Demande

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du déclencheur à mettre à jour.

- `TriggerUpdate` – Obligatoire : un objet TriggerUpdate.

  Nouvelles valeurs pour mettre à jour le déclencheur.

Réponse

- `Trigger` – Un objet Déclencheur.

  Définition du déclencheur obtenu.

Erreurs

- `InvalidInputException`
- `InternalServiceException`
- `EntityNotFoundException`
- `OperationTimeoutException`
- `ConcurrentModificationException`

StopTrigger action (Python : stop_trigger)

Arrête un déclencheur spécifié.

Demande

- Name – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du déclencheur à arrêter.

Réponse

- Name – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du déclencheur qui a été arrêté.

Erreurs

- `InvalidInputException`
- `InternalServiceException`
- `EntityNotFoundException`
- `OperationTimeoutException`
- `ConcurrentModificationException`

DeleteTrigger action (Python : delete_trigger)

Supprime un déclencheur spécifié. Si le déclencheur est introuvable, aucune exception n'est levée.

Demande

- Name – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du déclencheur à supprimer.

Réponse

- Name – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

Nom du déclencheur qui a été supprimé.

Erreurs

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ConcurrentModificationException`

ListTriggers action (Python : list_triggers)

Récupère les noms de toutes les ressources de déclenchement de ce AWS compte ou des ressources portant le tag spécifié. Cette opération vous permet de voir quelles ressources sont disponibles dans votre compte, et leurs noms.

Cette opération accepte le champ `Tags` facultatif que vous pouvez utiliser comme filtre sur la réponse, afin que les ressources balisées puissent être récupérées en tant que groupe. Si vous choisissez d'utiliser le filtrage des balises, seules les ressources avec la balise sont récupérées.

Demande

- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, s'il s'agit d'une requête de continuation.
- `DependentJobName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la tâche pour laquelle récupérer les déclencheurs. Le déclencheur qui peut lancer cette tâche est renvoyé. Si aucun déclencheur de ce type n'existe, tous les déclencheurs sont renvoyés.
- `MaxResults`— Nombre (entier), pas moins de 1 ou plus de 200.

  La taille maximale d'une liste à renvoyer.
- `Tags` – tableau de mappage de paires clé-valeur, avec 50 paires au maximum.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 256 octets.

Spécifie de renvoyer uniquement les ressources balisées.

Réponse

- `TriggerNames` – Tableau de chaînes UTF-8.

  Noms de tous les déclencheurs dans le compte ou des déclencheurs avec les balises spécifiées.
- `NextToken` – Chaîne UTF-8.

  Jeton continuation, si la liste renvoyée ne contient pas la dernière métrique disponible.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

BatchGetTriggers action (Python : batch_get_triggers)

Renvoie la liste des métadonnées de ressource pour une liste donnée de noms de déclencheur. Après avoir appelé l'opération `ListTriggers`, vous pouvez appeler cette opération pour accéder aux données sur lesquelles des autorisations vous ont été octroyées. Cette opération prend en charge toutes les autorisations IAM, y compris les conditions d'autorisation qui utilisent des balises.

Demande

- `TriggerNames` – obligatoire : tableau de chaînes UTF-8.

  Liste des noms de déclencheur, qui peuvent être les noms renvoyés à partir de l'opération `ListTriggers`.

Réponse

- `Triggers` : un tableau d'objets Déclencheur.

  Liste des définitions de déclencheur.

- `TriggersNotFound` – Tableau de chaînes UTF-8.

  Liste des noms des déclencheurs introuvables.

Erreurs

- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`

Intégration APIs dans AWS Glue

Types de données

- Structure d'intégration
- IntegrationConfig structure
- IntegrationPartition structure
- IntegrationError structure
- IntegrationFilter structure
- InboundIntegration structure
- SourceProcessingProperties structure
- TargetProcessingProperties structure
- SourceTableConfig structure
- TargetTableConfig structure

Structure d'intégration

Décrit une intégration zéro ETL.

Champs

- `SourceArn`— Obligatoire : chaîne UTF-8 d'une longueur d'au moins 1 ou de plus de 128 octets.

  ARN de la source pour l'intégration.

- `TargetArn`— Obligatoire : chaîne UTF-8 d'une longueur d'au moins 1 ou de plus de 128 octets.

  ARN de la cible pour l'intégration.
- `Description` – chaîne UTF-8, d'une longueur ne dépassant pas 1 000 octets, correspondant au Custom string pattern #12.

  Description de l'intégration.
- `IntegrationName`— Obligatoire : chaîne UTF-8 d'une longueur d'au moins 1 ou de plus de 128 octets.

  Nom unique de l'intégration.
- `IntegrationArn`— Obligatoire : chaîne UTF-8 d'une longueur d'au moins 1 ou de plus de 128 octets.

  Amazon Resource Name (ARN) de l'intégration.
- `KmsKeyId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 2 048 octets.

  ARN d'une clé KMS utilisée pour chiffrer le canal.
- `AdditionalEncryptionContext` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8.

  Chaque valeur est une chaîne UTF-8.

  Ensemble facultatif de paires clé-valeur non secrètes qui contiennent des informations contextuelles supplémentaires pour le chiffrement. Il ne peut être fourni que si `KMSKeyId` est indiqué.
- `Tags` : un tableau d'objets Balise.

  Métadonnée attribuée à la ressource composée d'une liste de paires clé-valeur.
- `Status` – Obligatoire : Chaîne UTF-8 (valeurs valides : `CREATING` | `ACTIVE` | `MODIFYING` | `FAILED` | `DELETING` | `SYNCING` | `NEEDS_ATTENTION`).

  Les statuts possibles sont les suivants :
  - CREATING : l'intégration est en cours de création.
  - ACTIVE : la création de l'intégration a réussi..
  - MODIFYING : l'intégration est en cours de modification.
  - FAILED : la création de l'intégration a échoué.

  - DELETING : l'intégration est supprimée.
  - SYNCING : l'intégration est en cours de synchronisation.
  - NEEDS_ATTENTION : l'intégration requiert votre attention, par exemple pour la synchronisation.
- `CreateTime` – obligatoire : horodatage.

  Heure (UTC) de création de l'intégration.
- `IntegrationConfig` : un objet IntegrationConfig.

  Propriétés associées à l'intégration.
- `Errors` : un tableau d'objets IntegrationError.

  Liste des erreurs associées à l'intégration.
- `DataFilter` – Chaîne UTF-8, d'une longueur comprise entre 1 et 2 048 octets.

  Sélectionne les tables source pour l'intégration à l'aide de la syntaxe du filtre Maxwell.

IntegrationConfig structure

Propriétés associées à l'intégration.

Champs

- `RefreshInterval` – Chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Spécifie la fréquence à laquelle CDC (Change Data Capture) doit effectuer des extractions ou des chargements incrémentiels. Ce paramètre offre la flexibilité nécessaire pour aligner le taux de rafraîchissement sur vos modèles de mise à jour des données spécifiques, vos considérations relatives à la charge du système et vos objectifs d'optimisation des performances. L'incrément de temps peut être réglé de 15 minutes à 8 640 minutes (six jours).
- `SourceProperties` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8.

  Chaque valeur est une chaîne UTF-8.

  Une collection de paires clé-valeur qui spécifient des propriétés supplémentaires pour la source d'intégration. Ces propriétés fournissent des options de configuration qui peuvent être utilisées

pour personnaliser le comportement de la source ODB lors des opérations d'intégration de données.

- `ContinuousSync` : booléen.

  Permet une synchronisation continue pour les extractions de données à la demande depuis : 1) des applications SaaS vers AWS des services de données tels qu'Amazon Redshift Amazon S3 et, et 2) DynamoDB vers. Amazon S3

IntegrationPartition structure

Structure qui décrit la manière dont les données sont partitionnées sur la cible.

Champs

- `FieldName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Nom du champ utilisé pour partitionner les données sur la cible. Évitez d'utiliser des colonnes qui ont des valeurs uniques pour chaque ligne (par exemple, LastModifiedTimestamp « », SystemModTimeStamp « ») comme colonne de partition. Ces colonnes ne sont pas adaptées au partitionnement, car elles créent un grand nombre de petites partitions, ce qui peut entraîner des problèmes de performances.

- `FunctionSpec` – Chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Spécifie la fonction utilisée pour partitionner les données sur la cible. Les valeurs acceptées pour ce paramètre sont les suivantes :

  - `identity` : utilise les valeurs source directement sans transformation
  - `year` : extrait l'année à partir des valeurs d'horodatage (par exemple, 2023)
  - `month` : extrait le mois à partir des valeurs d'horodatage (par exemple, 2023-01)
  - `day` : extrait le jour à partir des valeurs d'horodatage (par exemple, 2023-01-15)
  - `hour` : extrait l'heure à partir des valeurs d'horodatage (par exemple, 2023-01-15-14)

- `ConversionSpec` – Chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Indique le format d'horodatage des données source. Les valeurs valides sont :

  - `epoch_sec` : horodatage d'époque Unix en secondes
  - `epoch_milli` : horodatage de l'époque Unix en millisecondes
  - `iso` : horodatage au format ISO 8601

> **Note**
>
> Spécifiez uniquement `ConversionSpec` lorsque vous utilisez des fonctions de partition basées sur l'horodatage (année, mois, jour ou heure). AWS Glue Zero-ETL utilise ce paramètre pour transformer correctement les données source au format d'horodatage avant le partitionnement.
> N'utilisez pas de colonnes à cardinalité élevée avec la fonction de partition `identity`. Les colonnes à haute cardinalité comprennent :
>
> - Clés primaires
> - Champs d'horodatage (tels que `LastModifiedTimestamp` et `CreatedDate`)
> - Horodatages générés par le système
>
> L'utilisation de colonnes à haute cardinalité avec partitionnement des identités crée de nombreuses petites partitions, ce qui peut considérablement dégrader les performances d'ingestion.

IntegrationError structure

Erreur associée à une intégration zéro ETL.

Champs

- `ErrorCode` – Chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Code associé à cette erreur.
- `ErrorMessage` – Chaîne UTF-8, d'une longueur comprise entre 1 et 2 048 octets.

  Message décrivant l'erreur.

IntegrationFilter structure

Filtre qui peut être utilisé lors de l'invocation d'une demande `DescribeIntegrations`.

Champs

- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

Nom du filtre.

- `Values` – Tableau de chaînes UTF-8.

  Une liste de valeurs de filtre.

InboundIntegration structure

Structure pour une intégration qui écrit des données dans une ressource.

Champs

- `SourceArn`— Obligatoire : chaîne UTF-8 d'une longueur d'au moins 1 ou de plus de 128 octets.

  ARN de la ressource source pour l'intégration.
- `TargetArn`— Obligatoire : chaîne UTF-8 d'une longueur d'au moins 1 ou de plus de 128 octets.

  ARN de la ressource cible pour l'intégration..
- `IntegrationArn`— Obligatoire : chaîne UTF-8 d'une longueur d'au moins 1 ou de plus de 128 octets.

  ARN de l'intégration zéro ETL.
- `Status` – Obligatoire : Chaîne UTF-8 (valeurs valides : `CREATING` | `ACTIVE` | `MODIFYING` | `FAILED` | `DELETING` | `SYNCING` | `NEEDS_ATTENTION`).

  Les statuts possibles sont les suivants :
  - CREATING : l'intégration est en cours de création.
  - ACTIVE : la création de l'intégration a réussi..
  - MODIFYING : l'intégration est en cours de modification.
  - FAILED : la création de l'intégration a échoué.
  - DELETING : l'intégration est supprimée.
  - SYNCING : l'intégration est en cours de synchronisation.
  - NEEDS_ATTENTION : l'intégration requiert votre attention, par exemple pour la synchronisation.
- `CreateTime` – obligatoire : horodatage.

  Heure (UTC) de création de l'intégration.
- `IntegrationConfig` : un objet IntegrationConfig.

Propriétés associées à l'intégration.

- `Errors` : un tableau d'objets IntegrationError.

  Liste des erreurs associées à l'intégration.

SourceProcessingProperties structure

Propriétés des ressources associées à la source d'intégration.

Champs

- `RoleArn` – Chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Rôle IAM permettant d'accéder à la AWS Glue connexion.

TargetProcessingProperties structure

Propriétés des ressources associées à la cible d'intégration.

Champs

- `RoleArn` – Chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Rôle IAM permettant d'accéder à la AWS Glue base de données.
- `KmsArn` – Chaîne UTF-8, d'une longueur comprise entre 1 et 2 048 octets.

  ARN de la clé KMS utilisée pour le chiffrement.
- `ConnectionName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Connexion AWS Glue réseau permettant de configurer la AWS Glue tâche exécutée dans le VPC du client.
- `EventBusArn` – Chaîne UTF-8, d'une longueur comprise entre 1 et 2 048 octets.

  ARN d'un bus d'événements Eventbridge pour recevoir la notification d'état de l'intégration.

SourceTableConfig structure

Propriétés utilisées par l'étape source pour traiter les données de la source.

Champs

- `Fields` – Tableau de chaînes UTF-8.

  Liste des champs utilisés pour le filtrage au niveau des colonnes. Non prise en charge actuellement.
- `FilterPredicate` – Chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Clause de condition utilisée pour le filtrage au niveau des lignes. Non prise en charge actuellement.
- `PrimaryKey` – Tableau de chaînes UTF-8.

  Fournissez le jeu de clés primaires pour cette table. Actuellement pris en charge spécifiquement pour les entités SAP `Entity0f` sur demande. Contactez AWS le Support pour rendre cette fonctionnalité disponible.
- `RecordUpdateField` – Chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Champ d'extraction incrémentielle basée sur l'horodatage. Non prise en charge actuellement.

TargetTableConfig structure

Propriétés utilisées par l'étape cible pour partitionner les données sur la cible.

Champs

- `UnnestSpec` – Chaîne UTF-8 (valeurs valides : `TOPLEVEL` | `FULL` | `NOUNNEST`).

  Spécifie la manière dont les objets imbriqués sont aplatis par rapport aux éléments de niveau supérieur. Les valeurs valides sont : « TOPLEVEL », « FULL » ou « NOUNNEST ».
- `PartitionSpec` : un tableau d'objets IntegrationPartition.

  Détermine la disposition du fichier sur la cible.
- `TargetTableName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Nom facultatif d'une table cible.

Opérations

- CreateIntegration action (Python : create_integration)

- ModifyIntegration action (Python : modifier_intégration)
- DescribeIntegrations action (Python : describe_integrations)
- DeleteIntegration action (Python : delete_integration)
- DescribeInboundIntegrations action (Python : describe_inbound_integrations)
- CreateIntegrationTableProperties action (Python : create_integration_table_properties)
- UpdateIntegrationTableProperties action (Python : update_integration_table_properties)
- GetIntegrationTableProperties action (Python : get_integration_table_properties)
- DeleteIntegrationTableProperties action (Python : delete_integration_table_properties)
- CreateIntegrationResourceProperty action (Python : create_integration_resource_property)
- UpdateIntegrationResourceProperty action (Python : update_integration_resource_property)
- GetIntegrationResourceProperty action (Python : get_integration_resource_property)
- UntagResource action (Python : untag_resource)
- ListTagsForResource action (Python : list_tags_for_resource)

CreateIntegration action (Python : create_integration)

Crée une intégration zéro ETL dans le compte de l'appelant entre deux ressources portant Amazon Resource Names (ARNs) : le `SourceArn` et. `TargetArn`

Demande

- `IntegrationName`— Obligatoire : chaîne UTF-8 d'une longueur d'au moins 1 ou de plus de 128 octets.

  Nom unique pour une intégration dans AWS Glue.
- `SourceArn`— Obligatoire : chaîne UTF-8 d'une longueur d'au moins 1 ou de plus de 128 octets.

  ARN de la ressource source pour l'intégration.
- `TargetArn`— Obligatoire : chaîne UTF-8 d'une longueur d'au moins 1 ou de plus de 128 octets.

  ARN de la ressource cible pour l'intégration..
- `Description` – chaîne UTF-8, d'une longueur ne dépassant pas 1 000 octets, correspondant au Custom string pattern #12.

Description de l'intégration.

- `DataFilter` – Chaîne UTF-8, d'une longueur comprise entre 1 et 2 048 octets.

  Sélectionne les tables source pour l'intégration à l'aide de la syntaxe du filtre Maxwell.

- `KmsKeyId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 2 048 octets.

  ARN d'une clé KMS utilisée pour chiffrer le canal.

- `AdditionalEncryptionContext` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8.

  Chaque valeur est une chaîne UTF-8.

  Ensemble facultatif de paires clé-valeur non secrètes qui contiennent des informations contextuelles supplémentaires pour le chiffrement. Il ne peut être fourni que si `KMSKeyId` est indiqué.

- `Tags` : un tableau d'objets Balise.

  Métadonnée attribuée à la ressource composée d'une liste de paires clé-valeur.

- `IntegrationConfig` : un objet IntegrationConfig.

  Paramètres de configuration.

Réponse

- `SourceArn`— Obligatoire : chaîne UTF-8 d'une longueur d'au moins 1 ou de plus de 128 octets.

  ARN de la ressource source pour l'intégration.

- `TargetArn`— Obligatoire : chaîne UTF-8 d'une longueur d'au moins 1 ou de plus de 128 octets.

  ARN de la ressource cible pour l'intégration..

- `IntegrationName`— Obligatoire : chaîne UTF-8 d'une longueur d'au moins 1 ou de plus de 128 octets.

  Nom unique pour une intégration dans AWS Glue.

- `Description` – chaîne UTF-8, d'une longueur ne dépassant pas 1 000 octets, correspondant au Custom string pattern #12.

Description de l'intégration.

- `IntegrationArn`— Obligatoire : chaîne UTF-8 d'une longueur d'au moins 1 ou de plus de 128 octets.

  Amazon Resource Name (ARN) de l'intégration créée.

- `KmsKeyId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 2 048 octets.

  ARN d'une clé KMS utilisée pour chiffrer le canal.

- `AdditionalEncryptionContext` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8.

  Chaque valeur est une chaîne UTF-8.

  Ensemble facultatif de paires clé-valeur non secrètes qui contiennent des informations contextuelles supplémentaires pour le chiffrement.

- `Tags` : un tableau d'objets Balise.

  Métadonnée attribuée à la ressource composée d'une liste de paires clé-valeur.

- `Status` – Obligatoire : Chaîne UTF-8 (valeurs valides : `CREATING` | `ACTIVE` | `MODIFYING` | `FAILED` | `DELETING` | `SYNCING` | `NEEDS_ATTENTION`).

  État de l'intégration en cours de création.

  Les statuts possibles sont les suivants :

  - CREATING : l'intégration est en cours de création.
  - ACTIVE : la création de l'intégration a réussi..
  - MODIFYING : l'intégration est en cours de modification.
  - FAILED : la création de l'intégration a échoué.
  - DELETING : l'intégration est supprimée.
  - SYNCING : l'intégration est en cours de synchronisation.
  - NEEDS_ATTENTION : l'intégration requiert votre attention, par exemple pour la synchronisation.

- `CreateTime` – obligatoire : horodatage.

  Heure (UTC) de création de l'intégration.

- `Errors` : un tableau d'objets IntegrationError.

Liste des erreurs associées à la création de l'intégration.

- `DataFilter` – Chaîne UTF-8, d'une longueur comprise entre 1 et 2 048 octets.

  Sélectionne les tables source pour l'intégration à l'aide de la syntaxe du filtre Maxwell.

- `IntegrationConfig` : un objet IntegrationConfig.

  Paramètres de configuration.

Erreurs

- `ValidationException`
- `AccessDeniedException`
- `ResourceNotFoundException`
- `InternalServerException`
- `IntegrationConflictOperationFault`
- `IntegrationQuotaExceededFault`
- `KMSKeyNotAccessibleFault`
- `EntityNotFoundException`
- `InternalServiceException`
- `ConflictException`
- `ResourceNumberLimitExceededException`
- `InvalidInputException`

ModifyIntegration action (Python : modifier_intégration)

Modifie une intégration zéro ETL dans le compte de l'appelant.

Demande

- `IntegrationIdentifier`— Obligatoire : chaîne UTF-8 d'une longueur d'au moins 1 ou de plus de 128 octets.

  Amazon Resource Name (ARN) de l'intégration.

- `Description` – chaîne UTF-8, d'une longueur ne dépassant pas 1 000 octets, correspondant au Custom string pattern #12.

  Description de l'intégration.

- `DataFilter` – Chaîne UTF-8, d'une longueur comprise entre 1 et 2 048 octets.

  Sélectionne les tables source pour l'intégration à l'aide de la syntaxe du filtre Maxwell.

- `IntegrationConfig` : un objet IntegrationConfig.

  Paramètres de configuration pour l'intégration. Actuellement, seuls les RefreshInterval peuvent être modifiés.

- `IntegrationName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Nom unique pour une intégration dans AWS Glue.

Réponse

- `SourceArn`— Obligatoire : chaîne UTF-8 d'une longueur d'au moins 1 ou de plus de 128 octets.

  ARN de la source pour l'intégration.

- `TargetArn`— Obligatoire : chaîne UTF-8 d'une longueur d'au moins 1 ou de plus de 128 octets.

  ARN de la cible pour l'intégration.

- `IntegrationName`— Obligatoire : chaîne UTF-8 d'une longueur d'au moins 1 ou de plus de 128 octets.

  Nom unique pour une intégration dans AWS Glue.

- `Description` – chaîne UTF-8, d'une longueur ne dépassant pas 1 000 octets, correspondant au Custom string pattern #12.

  Description de l'intégration.

- `IntegrationArn`— Obligatoire : chaîne UTF-8 d'une longueur d'au moins 1 ou de plus de 128 octets.

  Amazon Resource Name (ARN) de l'intégration.

- `KmsKeyId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 2 048 octets.

  ARN d'une clé KMS utilisée pour chiffrer le canal.

- `AdditionalEncryptionContext` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8.

  Chaque valeur est une chaîne UTF-8.

  Ensemble facultatif de paires clé-valeur non secrètes qui contiennent des informations contextuelles supplémentaires pour le chiffrement.

- `Tags` : un tableau d'objets Balise.

  Métadonnée attribuée à la ressource composée d'une liste de paires clé-valeur.

- `Status` – Obligatoire : Chaîne UTF-8 (valeurs valides : `CREATING` | `ACTIVE` | `MODIFYING` | `FAILED` | `DELETING` | `SYNCING` | `NEEDS_ATTENTION`).

  État de l'intégration en cours de modification.

  Les statuts possibles sont les suivants :

  - CREATING : l'intégration est en cours de création.
  - ACTIVE : la création de l'intégration a réussi..
  - MODIFYING : l'intégration est en cours de modification.
  - FAILED : la création de l'intégration a échoué.
  - DELETING : l'intégration est supprimée.
  - SYNCING : l'intégration est en cours de synchronisation.
  - NEEDS_ATTENTION : l'intégration requiert votre attention, par exemple pour la synchronisation.

- `CreateTime` – obligatoire : horodatage.

  Heure (UTC) de création de l'intégration.

- `Errors` : un tableau d'objets IntegrationError.

  Liste des erreurs associées à la modification de l'intégration.

- `DataFilter` – Chaîne UTF-8, d'une longueur comprise entre 1 et 2 048 octets.

  Sélectionne les tables source pour l'intégration à l'aide de la syntaxe du filtre Maxwell.

- `IntegrationConfig` : un objet IntegrationConfig.

  Paramètres de configuration mis à jour pour l'intégration.

Erreurs

- `ValidationException`
- `AccessDeniedException`
- `InternalServerException`
- `IntegrationNotFoundFault`
- `IntegrationConflictOperationFault`
- `InvalidIntegrationStateFault`
- `EntityNotFoundException`
- `InternalServiceException`
- `ConflictException`
- `InvalidStateException`
- `InvalidInputException`

DescribeIntegrations action (Python : describe_integrations)

L'API est utilisée pour récupérer une liste d'intégrations.

Demande

- `IntegrationIdentifier` – Chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Amazon Resource Name (ARN) de l'intégration.
- `Marker` – Chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Valeur qui indique le point de départ de la prochaine série d'enregistrements de réponses dans une demande ultérieure.
- `MaxRecords` : nombre (entier).

  Nombre total d'éléments à renvoyer dans la sortie.
- `Filters` : un tableau d'objets IntegrationFilter.

  Liste de clés et de valeurs, pour filtrer les résultats. Les touches prises en charge sont « Status », IntegrationName « » et « SourceArn ». IntegrationName est limité à une seule valeur.

Réponse

- `Integrations` : un tableau d'objets Integration.

  Liste des intégrations zéro ETL.
- `Marker` – Chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Valeur qui indique le point de départ de la prochaine série d'enregistrements de réponses dans une demande ultérieure.

Erreurs

- `ValidationException`
- `AccessDeniedException`
- `InternalServerException`
- `IntegrationNotFoundFault`
- `EntityNotFoundException`
- `InternalServiceException`
- `InvalidInputException`

DeleteIntegration action (Python : delete_integration)

Supprime l'intégration zéro ETL spécifiée.

Demande

- `IntegrationIdentifier`— Obligatoire : chaîne UTF-8 d'une longueur d'au moins 1 ou de plus de 128 octets.

  Amazon Resource Name (ARN) de l'intégration.

Réponse

- `SourceArn`— Obligatoire : chaîne UTF-8 d'une longueur d'au moins 1 ou de plus de 128 octets.

  ARN de la source pour l'intégration.
- `TargetArn`— Obligatoire : chaîne UTF-8 d'une longueur d'au moins 1 ou de plus de 128 octets.

ARN de la cible pour l'intégration.

- `IntegrationName`— Obligatoire : chaîne UTF-8 d'une longueur d'au moins 1 ou de plus de 128 octets.

  Nom unique pour une intégration dans AWS Glue.

- `Description` – chaîne UTF-8, d'une longueur ne dépassant pas 1 000 octets, correspondant au Custom string pattern #12.

  Description de l'intégration.

- `IntegrationArn`— Obligatoire : chaîne UTF-8 d'une longueur d'au moins 1 ou de plus de 128 octets.

  Amazon Resource Name (ARN) de l'intégration.

- `KmsKeyId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 2 048 octets.

  ARN d'une clé KMS utilisée pour chiffrer le canal.

- `AdditionalEncryptionContext` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8.

  Chaque valeur est une chaîne UTF-8.

  Ensemble facultatif de paires clé-valeur non secrètes qui contiennent des informations contextuelles supplémentaires pour le chiffrement.

- `Tags` : un tableau d'objets Balise.

  Métadonnée attribuée à la ressource composée d'une liste de paires clé-valeur.

- `Status` – Obligatoire : Chaîne UTF-8 (valeurs valides : `CREATING` | `ACTIVE` | `MODIFYING` | `FAILED` | `DELETING` | `SYNCING` | `NEEDS_ATTENTION`).

  État de l'intégration en cours de suppression.

  Les statuts possibles sont les suivants :

  - CREATING : l'intégration est en cours de création.
  - ACTIVE : la création de l'intégration a réussi..
  - MODIFYING : l'intégration est en cours de modification.
  - FAILED : la création de l'intégration a échoué.

  - DELETING : l'intégration est supprimée.
  - SYNCING : l'intégration est en cours de synchronisation.
  - NEEDS_ATTENTION : l'intégration requiert votre attention, par exemple pour la synchronisation.
- `CreateTime` – obligatoire : horodatage.

  Heure (UTC) de création de l'intégration.
- `Errors` : un tableau d'objets IntegrationError.

  Liste des erreurs associées à l'intégration.
- `DataFilter` – Chaîne UTF-8, d'une longueur comprise entre 1 et 2 048 octets.

  Sélectionne les tables source pour l'intégration à l'aide de la syntaxe du filtre Maxwell.

Erreurs

- `ValidationException`
- `AccessDeniedException`
- `InternalServerException`
- `IntegrationNotFoundFault`
- `IntegrationConflictOperationFault`
- `InvalidIntegrationStateFault`
- `EntityNotFoundException`
- `InternalServiceException`
- `ConflictException`
- `InvalidStateException`
- `InvalidInputException`

DescribeInboundIntegrations action (Python : describe_inbound_integrations)

Renvoie une liste des intégrations entrantes pour l'intégration spécifiée.

Demande

- `IntegrationArn` – Chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Amazon Resource Name (ARN) de l'intégration.
- `Marker` – Chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Jeton permettant de spécifier où commencer la pagination. Il s'agit du marqueur provenant d'une réponse tronquée précédemment.
- `MaxRecords` : nombre (entier).

  Nombre total d'éléments à renvoyer dans la sortie.
- `TargetArn` – Chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Amazon Resource Name (ARN) de la ressource cible dans l'intégration.

Réponse

- `InboundIntegrations` : un tableau d'objets InboundIntegration.

  Liste des intégrations entrantes.
- `Marker` – Chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Valeur qui indique le point de départ de la prochaine série d'enregistrements de réponses dans une demande ultérieure.

Erreurs

- `ValidationException`
- `AccessDeniedException`
- `InternalServerException`
- `IntegrationNotFoundFault`
- `TargetResourceNotFound`
- `OperationNotSupportedException`
- `EntityNotFoundException`
- `InternalServiceException`

- `InvalidInputException`

CreateIntegrationTableProperties action (Python : create_integration_table_properties)

Cette API est utilisée pour fournir des propriétés de remplacement facultatives pour les tables qui doivent être répliquées. Ces propriétés peuvent inclure des propriétés de filtrage et de partitionnement pour les tables source et cible. Pour définir à la fois les propriétés source et cible, la même API doit être invoquée avec l'ARN de AWS Glue connexion comme `ResourceArn` avec`SourceTableConfig`, et l'ARN AWS Glue de base de données comme `ResourceArn` avec `TargetTableConfig` respectivement.

Demande

- `ResourceArn`— Obligatoire : chaîne UTF-8 d'une longueur d'au moins 1 ou de plus de 128 octets.

  Amazon Resource Name (ARN) de la table cible pour laquelle vous souhaitez créer les propriétés de la table d'intégration. Actuellement, cette API prend uniquement en charge la création de propriétés de table d'intégration pour les tables cibles, et l'ARN fourni doit être l'ARN de la table cible dans le catalogue de AWS Glue données. La prise en charge de la création de propriétés de table d'intégration pour les connexions source (à l'aide de l'ARN de connexion) n'est pas encore mise en œuvre et sera ajoutée dans une future version.
- `TableName`— Obligatoire : chaîne UTF-8 d'une longueur d'au moins 1 ou de plus de 128 octets.

  Nom de la table à répliquer.
- `SourceTableConfig` – Un objet SourceTableConfig.

  Structure pour la configuration de table source. Consultez la structure `SourceTableConfig` pour voir la liste des propriétés source prises en charge.
- `TargetTableConfig` – Un objet TargetTableConfig.

  Structure pour la configuration de table cible.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `ValidationException`
- `AccessDeniedException`
- `ResourceNotFoundException`
- `InternalServerException`
- `EntityNotFoundException`
- `InternalServiceException`
- `InvalidInputException`

UpdateIntegrationTableProperties action (Python : update_integration_table_properties)

Cette API est utilisée pour fournir des propriétés de remplacement facultatives pour les tables qui doivent être répliquées. Ces propriétés peuvent inclure des propriétés de filtrage et de partitionnement pour les tables source et cible. Pour définir à la fois les propriétés source et cible, la même API doit être invoquée avec l'ARN de AWS Glue connexion comme `ResourceArn` avec`SourceTableConfig`, et l'ARN AWS Glue de base de données comme `ResourceArn` avec `TargetTableConfig` respectivement.

Le remplacement sera reflété dans toutes les intégrations utilisant le même `ResourceArn` et la même table source.

Demande

- `ResourceArn`— Obligatoire : chaîne UTF-8 d'une longueur d'au moins 1 ou de plus de 128 octets.

  ARN de connexion de la source ou ARN de base de données de la cible.
- `TableName`— Obligatoire : chaîne UTF-8 d'une longueur d'au moins 1 ou de plus de 128 octets.

  Nom de la table à répliquer.
- `SourceTableConfig` – Un objet SourceTableConfig.

  Structure pour la configuration de table source.
- `TargetTableConfig` – Un objet TargetTableConfig.

  Structure pour la configuration de table cible.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `ValidationException`
- `AccessDeniedException`
- `ResourceNotFoundException`
- `InternalServerException`
- `EntityNotFoundException`
- `InternalServiceException`
- `InvalidInputException`

GetIntegrationTableProperties action (Python : get_integration_table_properties)

Cette API est utilisée pour extraire des propriétés de remplacement facultatives pour les tables qui doivent être répliquées. Ces propriétés peuvent inclure des propriétés de filtrage et de partition pour les tables source et cible.

Demande

- `ResourceArn`— Obligatoire : chaîne UTF-8 d'une longueur d'au moins 1 ou de plus de 128 octets.

  Amazon Resource Name (ARN) de la table cible pour laquelle vous souhaitez récupérer les propriétés de la table d'intégration. Actuellement, cette API prend uniquement en charge la récupération des propriétés des tables cibles, et l'ARN fourni doit être l'ARN de la table cible dans le catalogue de AWS Glue données. La prise en charge de la récupération de propriétés de table d'intégration pour les connexions source (à l'aide de l'ARN de connexion) n'est pas encore mise en œuvre et sera ajoutée dans une future version.

- `TableName`— Obligatoire : chaîne UTF-8 d'une longueur d'au moins 1 ou de plus de 128 octets.

  Nom de la table à répliquer.

Réponse

- `ResourceArn` – Chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Amazon Resource Name (ARN) de la table cible pour laquelle vous souhaitez récupérer les propriétés de la table d'intégration. Actuellement, cette API prend uniquement en charge la récupération des propriétés des tables cibles, et l'ARN fourni doit être l'ARN de la table cible dans le catalogue de AWS Glue données. La prise en charge de la récupération de propriétés de table d'intégration pour les connexions source (à l'aide de l'ARN de connexion) n'est pas encore mise en œuvre et sera ajoutée dans une future version.
- `TableName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Nom de la table à répliquer.
- `SourceTableConfig` – Un objet SourceTableConfig.

  Structure pour la configuration de table source.
- `TargetTableConfig` – Un objet TargetTableConfig.

  Structure pour la configuration de table cible.

Erreurs

- `ValidationException`
- `AccessDeniedException`
- `ResourceNotFoundException`
- `InternalServerException`
- `EntityNotFoundException`
- `InternalServiceException`
- `InvalidInputException`

DeleteIntegrationTableProperties action (Python : delete_integration_table_properties)

Supprime les propriétés de table créées pour les tables devant être répliquées.

Demande

- `ResourceArn`— Obligatoire : chaîne UTF-8 d'une longueur d'au moins 1 ou de plus de 128 octets.

  ARN de connexion de la source ou ARN de base de données de la cible.
- `TableName`— Obligatoire : chaîne UTF-8 d'une longueur d'au moins 1 ou de plus de 128 octets.

  Nom de la table à répliquer.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `ValidationException`
- `AccessDeniedException`
- `ResourceNotFoundException`
- `InternalServerException`
- `EntityNotFoundException`
- `InternalServiceException`
- `InvalidInputException`

CreateIntegrationResourceProperty action (Python : create_integration_resource_property)

Cette API peut être utilisée pour configurer la AWS Glue connexion (pour `ResourceProperty` la source) ou l'ARN de la AWS Glue base de données (pour la cible). Ces propriétés peuvent inclure le rôle permettant d'accéder à la connexion ou à la base de données. Pour définir les propriétés source et cible, la même API doit être invoquée avec l'ARN de AWS Glue connexion comme `ResourceArn` avec `SourceProcessingProperties` et l'ARN AWS Glue de base de données comme `ResourceArn` avec `TargetProcessingProperties` respectivement.

Demande

- `ResourceArn`— Obligatoire : chaîne UTF-8 d'une longueur d'au moins 1 ou de plus de 128 octets.

ARN de connexion de la source ou ARN de base de données de la cible.

- `SourceProcessingProperties` – Un objet SourceProcessingProperties.

  Propriétés des ressources associées à la source d'intégration.

- `TargetProcessingProperties` – Un objet TargetProcessingProperties.

  Propriétés des ressources associées à la cible d'intégration.

Réponse

- `ResourceArn`— Obligatoire : chaîne UTF-8 d'une longueur d'au moins 1 ou de plus de 128 octets.

  ARN de connexion de la source ou ARN de base de données de la cible.

- `SourceProcessingProperties` – Un objet SourceProcessingProperties.

  Propriétés des ressources associées à la source d'intégration.

- `TargetProcessingProperties` – Un objet TargetProcessingProperties.

  Propriétés des ressources associées à la cible d'intégration.

Erreurs

- `ValidationException`
- `AccessDeniedException`
- `ConflictException`
- `InternalServerException`
- `ResourceNotFoundException`
- `EntityNotFoundException`
- `InternalServiceException`
- `InvalidInputException`

UpdateIntegrationResourceProperty action (Python : update_integration_resource_property)

Cette API peut être utilisée pour mettre à jour `ResourceProperty` la AWS Glue connexion (pour la source) ou l'ARN de la AWS Glue base de données (pour la cible). Ces propriétés peuvent inclure le rôle permettant d'accéder à la connexion ou à la base de données. Comme la même ressource peut être utilisée dans plusieurs intégrations, la mise à jour des propriétés des ressources aura un impact sur toutes les intégrations qui l'utilisent.

Demande

- `ResourceArn`— Obligatoire : chaîne UTF-8 d'une longueur d'au moins 1 ou de plus de 128 octets.

  ARN de connexion de la source ou ARN de base de données de la cible.
- `SourceProcessingProperties` – Un objet SourceProcessingProperties.

  Propriétés des ressources associées à la source d'intégration.
- `TargetProcessingProperties` – Un objet TargetProcessingProperties.

  Propriétés des ressources associées à la cible d'intégration.

Réponse

- `ResourceArn` – Chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  ARN de connexion de la source ou ARN de base de données de la cible.
- `SourceProcessingProperties` – Un objet SourceProcessingProperties.

  Propriétés des ressources associées à la source d'intégration.
- `TargetProcessingProperties` – Un objet TargetProcessingProperties.

  Propriétés des ressources associées à la cible d'intégration.

Erreurs

- `ValidationException`
- `AccessDeniedException`
- `InternalServerException`

- `ResourceNotFoundException`
- `EntityNotFoundException`
- `InternalServiceException`
- `InvalidInputException`

GetIntegrationResourceProperty action (Python : get_integration_resource_property)

Cette API est utilisée pour récupérer la AWS Glue connexion (pour `ResourceProperty` la source) ou l'ARN de la AWS Glue base de données (pour la cible)

Demande

- `ResourceArn`— Obligatoire : chaîne UTF-8 d'une longueur d'au moins 1 ou de plus de 128 octets.

  ARN de connexion de la source ou ARN de base de données de la cible.

Réponse

- `ResourceArn` – Chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  ARN de connexion de la source ou ARN de base de données de la cible.
- `SourceProcessingProperties` – Un objet SourceProcessingProperties.

  Propriétés des ressources associées à la source d'intégration.
- `TargetProcessingProperties` – Un objet TargetProcessingProperties.

  Propriétés des ressources associées à la cible d'intégration.

Erreurs

- `ValidationException`
- `AccessDeniedException`
- `InternalServerException`
- `ResourceNotFoundException`
- `EntityNotFoundException`

- `InternalServiceException`
- `InvalidInputException`

UntagResource action (Python : untag_resource)

Supprime les balises spécifiées d'une ressource d'intégration.

Demande

- `ResourceArn` – Obligatoire : chaîne UTF-8, d'une longueur comprise entre 1 et 10 240 octets, correspondant au Custom string pattern #50.

  Amazon Resource Name (ARN) de la ressource d'intégration.
- `TagsToRemove` – Obligatoire : Tableau de chaînes UTF-8, avec 50 chaînes maximum.

  Liste des balises de métadonnés à supprimer de la ressource.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `ResourceNotFoundException`

ListTagsForResource action (Python : list_tags_for_resource)

Répertorie les balises de métadonnées attribuées à la ressource spécifiée.

Demande

- `ResourceARN` – Obligatoire : chaîne UTF-8, d'une longueur comprise entre 1 et 10 240 octets, correspondant au Custom string pattern #50.

  ARN de ressource pour la ressource.

Réponse

- `Tags` – Un tableau d'objets Balise, 10 structures au maximum.

  Une liste de balises.

Erreurs

- `ResourceNotFoundException`

Exceptions

- ResourceNotFoundException structure
- InternalServerException structure
- IntegrationAlreadyExistsFault structure
- IntegrationConflictOperationFault structure
- IntegrationQuotaExceededFault structure
- KMSKeyNotAccessibleFault structure
- IntegrationNotFoundFault structure
- TargetResourceNotFound structure
- InvalidIntegrationStateFault structure

ResourceNotFoundException structure

La ressource est introuvable.

Champs

- `Message` – Chaîne UTF-8.

  Un message décrivant le problème.

InternalServerException structure

Une erreur interne du serveur s'est produite.

Champs

- `Message` – Chaîne UTF-8.

  Un message décrivant le problème.

IntegrationAlreadyExistsFault structure

L'intégration spécifiée existe déjà.

Champs

- `Message` – Chaîne UTF-8.

  Un message décrivant le problème.

IntegrationConflictOperationFault structure

L'opération demandée est en conflit avec une autre opération.

Champs

- `Message` – Chaîne UTF-8.

  Un message décrivant le problème.

IntegrationQuotaExceededFault structure

Les données traitées dans le cadre de votre intégration ont dépassé votre quota.

Champs

- `Message` – Chaîne UTF-8.

  Un message décrivant le problème.

KMSKeyNotAccessibleFault structure

La clé KMS spécifiée n'est pas accessible.

Champs

- `Message` – Chaîne UTF-8.

  Un message décrivant le problème.

IntegrationNotFoundFault structure

L'intégration spécifiée est introuvable.

Champs

- `Message` – Chaîne UTF-8.

  Un message décrivant le problème.

TargetResourceNotFound structure

La ressource cible est introuvable.

Champs

- `Message` – Chaîne UTF-8.

  Un message décrivant le problème.

InvalidIntegrationStateFault structure

L'intégration est dans un état non valide.

Champs

- `Message` – Chaîne UTF-8.

  Un message décrivant le problème.

Séances interactives API

L'API des sessions interactives décrit l' AWS Glue API liée à l'utilisation de sessions AWS Glue interactives pour créer et tester des scripts d'extraction, de transformation et de chargement (ETL) pour l'intégration de données.

Types de données

- Structure de séance
- SessionCommand structure
- Structure de la déclaration
- StatementOutput structure
- StatementOutputData structure
- ConnectionsList structure

Structure de séance

Période pendant laquelle un environnement d'exécution Spark distant est en cours d'exécution.

Champs

- `Id` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  L'ID de la séance.
- `CreatedOn` – Horodatage.

  Date et heure de création de la séance.
- `Status` – Chaîne UTF-8 (valeurs valides : `PROVISIONING` | `READY` | `FAILED` | `TIMEOUT` | `STOPPING` | `STOPPED`).

  Statut d'une séance.
- `ErrorMessage` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Le message d'erreur affiché pendant la séance.

- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description de la séance.

- `Role` – chaîne UTF-8, d'une longueur comprise entre 20 et 2048 octets, correspondant au Custom string pattern #30.

  Nom ou Amazon Resource Name (ARN) du rôle IAM associé à la séance.

- `Command` – Un objet SessionCommand.

  L'objet de commande. Voir. SessionCommand

- `DefaultArguments` – Tableau de mappage de paires valeur-clé, avec 75 paires au maximum.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets, correspondant au Custom string pattern #31.

  Chaque valeur est une chaîne UTF-8, d'une longueur ne dépassant pas 4096 octets, correspondant au URI address multi-line string pattern.

  Tableau de mappage de paires valeur-clé. Le maximum est de 75 paires.

- `Connections` – Un objet ConnectionsList.

  Nombre de connexions utilisées pour la séance.

- `Progress` – Nombre (double).

  La progression de l'exécution du code de la séance.

- `MaxCapacity` – Nombre (double).

  Le nombre d'unités de traitement de AWS Glue données (DPUs) qui peuvent être allouées lors de l'exécution de la tâche. Un DPU est une mesure relative de la puissance de traitement composée de 4 V de capacité CPUs de calcul et de 16 Go de mémoire.

- `SecurityConfiguration` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la SecurityConfiguration structure à utiliser avec la session.

- `GlueVersion` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #48.

La AWS Glue version détermine les versions d'Apache Spark et de Python prises AWS Glue en charge. Le GlueVersion doit être supérieur à 2,0.

- `DataAccessId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 36 octets.

  ID d'accès aux données de la séance.

- `PartitionId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 36 octets.

  L'ID de partition de la séance.

- `NumberOfWorkers` : nombre (entier).

  Nombre d'utilisateurs d'une `WorkerType` définie pour la séance.

- `WorkerType` – Chaîne UTF-8 (valeurs valides: `Standard=""` | `G.1X=""` | `G.2X=""` | `G.025X=""` | `G.4X=""` | `G.8X=""` | `Z.2X=""`).

  Type de travail prédéfini qui est alloué lorsqu'une session est exécutée. Accepte une valeur de `G.1X`, `G.2X`, `G.4X` ou `G.8X` pour les sessions Spark. Accepte la valeur `Z.2X` pour les sessions Ray.

- `CompletedOn` – Horodatage.

  La date et heure à laquelle cette session s'est terminée.

- `ExecutionTime` – Nombre (double).

  La durée totale de la session.

- `DPUSeconds` – Nombre (double).

  La DPUs consommation par la séance (formule : ExecutionTime * MaxCapacity).

- `IdleTimeout` : nombre (entier).

  Le nombre de minutes d'inactivité avant l'expiration de la session.

- `ProfileName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom d'un profil AWS Glue d'utilisation associé à la session.

SessionCommand structure

La commande `SessionCommand` qui exécute cette tâche.

Champs

- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Spécifie le nom du SessionCommand. Il peut s'agir de « gluetel » ou de « gluestreaming ».
- `PythonVersion` – Chaîne UTF-8, correspondant au Custom string pattern #49.

  Spécifie la version de Python utilisée. La version de Python indique la version prise en charge pour les tâches de type Spark.

Structure de la déclaration

La déclaration ou la demande pour qu'une action particulière se produise dans une séance.

Champs

- `Id` : nombre (entier).

  ID de la déclaration.
- `Code` – Chaîne UTF-8.

  Code d'exécution de la déclaration.
- `State` – Chaîne UTF-8 (valeurs valides: `WAITING` | `RUNNING` | `AVAILABLE` | `CANCELLING` | `CANCELLED` | |).

  État pendant que la demande est exécutée.
- `Output` – Un objet StatementOutput.

  Le résultat en JSON.
- `Progress` – Nombre (double).

  La progression de l'exécution du code.
- `StartedOn` – Nombre (long).

  Date et heure Unix de démarrage de la définition de tâche.
- `CompletedOn` – Nombre (long).

  Date et heure Unix auxquelles la définition de tâche a été terminée.

StatementOutput structure

Résultat de l'exécution de code au format JSON.

Champs

- `Data` – Un objet StatementOutputData.

  Résultat de l'exécution de code.
- `ExecutionCount` : nombre (entier).

  Nombre d'exécution du résultat.
- `Status` – Chaîne UTF-8 (valeurs valides : `WAITING` | `RUNNING` | `AVAILABLE` | `CANCELLING` | `CANCELLED` | `ERROR`).

  Statut du résultat de l'exécution de code.
- `ErrorName` – Chaîne UTF-8.

  Nom de l'erreur dans le résultat.
- `ErrorValue` – Chaîne UTF-8.

  Valeur de l'erreur du résultat.
- `Traceback` – Tableau de chaînes UTF-8.

  Le retraçage du résultat.

StatementOutputData structure

Résultat de l'exécution de code au format JSON.

Champs

- `TextPlain` – Chaîne UTF-8.

  Résultat de l'exécution de code au format texte.

ConnectionsList structure

Spécifie les connexions utilisées par une tâche.

Champs

- `Connections` – Tableau de chaînes UTF-8, avec 1 000 chaînes maximum.

  Liste de connexions utilisées par la tâche.

Opérations

- CreateSession action (Python : créer_session)
- StopSession action (Python : stop_session)
- DeleteSession action (Python : supprimer_session)
- GetSession action (Python : get_session)
- ListSessions action (Python : list_sessions)
- RunStatement action (Python : run_statement)
- CancelStatement action (Python : annuler_statement)
- GetStatement action (Python : get_statement)
- ListStatements action (Python : list_statements)
- GetGlueIdentityCenterConfiguration action (Python : get_glue_identity_center_configuration)
- UpdateGlueIdentityCenterConfiguration action (Python : update_glue_identity_center_configuration)
- CreateGlueIdentityCenterConfiguration action (Python : create_glue_identity_center_configuration)
- DeleteGlueIdentityCenterConfiguration action (Python : delete_glue_identity_center_configuration)

CreateSession action (Python : créer_session)

Crée une nouvelle séance.

Demande

Demande de création d'une nouvelle séance.

- `Id` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  L'ID de la séance demandée
- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

Description de la séance.

- `Role` – Obligatoire : chaîne UTF-8, d'une longueur comprise entre 20 et 2048 octets, correspondant au Custom string pattern #30.

  Un ARN de rôle IAM

- `Command` – Obligatoire : un objet SessionCommand.

  La commande `SessionCommand` qui exécute cette tâche.

- `Timeout` – Nombre (entier), au moins égal à 1.

  Nombre de minutes avant l'expiration de la séance. La valeur par défaut pour les tâches ETL Spark est de 48 heures (2 880 minutes). Consultez la documentation pour les autres types de tâches.

- `IdleTimeout` – Nombre (entier), au moins égal à 1.

  Nombre de minutes d'inactivité avant l'expiration de la séance. La valeur par défaut pour les tâches ETL Spark est la valeur du délai d'expiration. Consultez la documentation pour les autres types de tâches.

- `DefaultArguments` – Tableau de mappage de paires valeur-clé, avec 75 paires au maximum.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets, correspondant au Custom string pattern #31.

  Chaque valeur est une chaîne UTF-8, d'une longueur ne dépassant pas 4096 octets, correspondant au URI address multi-line string pattern.

  Tableau de mappage de paires valeur-clé. Le maximum est de 75 paires.

- `Connections` – Un objet ConnectionsList.

  Nombre de connexions à utiliser pour la séance.

- `MaxCapacity` – Nombre (double).

  Le nombre d'unités de traitement de AWS Glue données (DPUs) qui peuvent être allouées lors de l'exécution de la tâche. Un DPU est une mesure relative de la puissance de traitement composée de 4 V de capacité CPUs de calcul et de 16 Go de mémoire.

- `NumberOfWorkers` : nombre (entier).

  Nombre d'utilisateurs d'une `WorkerType` définie pour la séance.

- `WorkerType` – Chaîne UTF-8 (valeurs valides: `Standard=""` | `G.1X=""` | `G.2X=""` | `G.025X=""` | `G.4X=""` | `G.8X=""` | `Z.2X=""`).

  Type de travail prédéfini qui est alloué lorsqu'une tâche est exécutée. Accepte une valeur de G.1X, G.2X, G.4X ou G.8X pour les tâches Spark. Accepte la valeur Z.2X pour les blocs-notes Ray.

  - Pour le type de `G.1X` worker, chaque worker est mappé à 1 DPU (4 vCPUs, 16 Go de mémoire) avec un disque de 94 Go, et fournit 1 exécuteur par worker. Nous recommandons ce type de travailleur pour les charges de travail telles que les transformations de données, les jointures et les requêtes, afin de proposer un moyen évolutif et rentable d'exécuter la plupart des tâches.
  - Pour le type de `G.2X` travailleur, chaque utilisateur est mappé sur 2 DPU (8 VCPUs, 32 Go de mémoire) avec un disque de 138 Go et fournit un exécuteur par travailleur. Nous recommandons ce type de travailleur pour les charges de travail telles que les transformations de données, les jointures et les requêtes, afin de proposer un moyen évolutif et rentable d'exécuter la plupart des tâches.
  - Pour le type de `G.4X` worker, chaque worker est mappé sur 4 DPU (16 vCPUs, 64 Go de mémoire) avec un disque de 256 Go, et fournit 1 exécuteur par worker. Nous recommandons ce type de travailleur pour les tâches dont les charges de travail contiennent les transformations, les agrégations, les jointures et les requêtes les plus exigeantes. Ce type de travailleur n'est disponible que pour les tâches Spark ETL AWS Glue version 3.0 ou ultérieure dans les AWS régions suivantes : USA Est (Ohio), USA Est (Virginie du Nord), USA Ouest (Oregon), Asie-Pacifique (Singapour), Asie-Pacifique (Sydney), Asie-Pacifique (Tokyo), Canada (Centre), Europe (Francfort), Europe (Irlande) et Europe (Stockholm).
  - Pour le type de `G.8X` travailleur, chaque travailleur est mappé sur 8 DPU (32 vCPUs, 128 Go de mémoire) avec un disque de 512 Go et fournit un exécuteur par travailleur. Nous recommandons ce type de travailleur pour les tâches dont les charges de travail contiennent les transformations, les agrégations, les jointures et les requêtes les plus exigeantes. Ce type de travailleur n'est disponible que pour les tâches Spark ETL de AWS Glue version 3.0 ou ultérieure, dans les mêmes AWS régions que celles prises en charge pour le type de `G.4X` travailleur.
  - Pour le type de `Z.2X` travailleur, chaque utilisateur est mappé sur 2 M-DPU (8 VCPUs, 64 Go de mémoire) avec un disque de 128 Go et fournit jusqu'à 8 processeurs de rayons sur la base de l'autoscaler.
- `SecurityConfiguration` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Le nom de la SecurityConfiguration structure à utiliser avec la session

- `GlueVersion` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #48.

  La AWS Glue version détermine les versions d'Apache Spark et de Python prises AWS Glue en charge. Le GlueVersion doit être supérieur à 2,0.

- `DataAccessId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 36 octets.

  ID d'accès aux données de la séance.

- `PartitionId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 36 octets.

  ID de partition de la séance.

- `Tags` – Tableau de mappage de paires clé-valeur, avec 50 paires au maximum.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 256 octets.

  Carte des paires de valeurs clés (balises) appartenant à la séance.

- `RequestOrigin` – chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets, correspondant au Custom string pattern #31.

  Origine de la demande.

- `ProfileName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom d'un profil AWS Glue d'utilisation associé à la session.

Réponse

- `Session` – Un objet Session.

  Retourne l'objet de séance dans la réponse.

Erreurs

- `AccessDeniedException`
- `IdempotentParameterMismatchException`
- `InternalServiceException`

- `OperationTimeoutException`
- `InvalidInputException`
- `ValidationException`
- `AlreadyExistsException`
- `ResourceNumberLimitExceededException`

StopSession action (Python : stop_session)

Arrête la séance.

Demande

- `Id` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID de la séance à arrêter.
- `RequestOrigin` – chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets, correspondant au Custom string pattern #31.

  Origine de la demande.

Réponse

- `Id` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Retourne l'ID de la séance arrêtée.

Erreurs

- `AccessDeniedException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`
- `IllegalSessionStateException`

- `ConcurrentModificationException`

DeleteSession action (Python : supprimer_session)

Supprime la séance.

Demande

- `Id` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID de la séance à supprimer.
- `RequestOrigin` – chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets, correspondant au Custom string pattern #31.

  Nom de l'origine de la demande de suppression de séance.

Réponse

- `Id` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Renvoie l'ID de la séance supprimée.

Erreurs

- `AccessDeniedException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`
- `IllegalSessionStateException`
- `ConcurrentModificationException`

GetSession action (Python : get_session)

Récupère la séance.

Demande

- `Id` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  L'ID de la séance.
- `RequestOrigin` – chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets, correspondant au Custom string pattern #31.

  Origine de la demande.

Réponse

- `Session` – Un objet Session.

  L'objet récupéré est renvoyé dans la réponse.

Erreurs

- `AccessDeniedException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`

ListSessions action (Python : list_sessions)

Extrait une liste de séances.

Demande

- `NextToken` – Chaîne UTF-8, d'une longueur maximale de 400 000 octets.

  Jeton pour l'ensemble de résultats suivant, ou null s'il n'y a pas de résultats supplémentaires.
- `MaxResults` – Nombre (entier), compris entre 1 et 1 000.

  Nombre maximal de résultats.
- `Tags` – Tableau de mappage de paires clé-valeur, avec 50 paires au maximum.

Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 256 octets.

Balises appartenant à la séance.

- `RequestOrigin` – chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets, correspondant au Custom string pattern #31.

  Origine de la demande.

Réponse

- `Ids` – Tableau de chaînes UTF-8.

  Retourne l'ID de la séance

- `Sessions` : un tableau d'objets Session.

  Retourne l'objet de la séance.

- `NextToken` – Chaîne UTF-8, d'une longueur maximale de 400 000 octets.

  Jeton pour l'ensemble de résultats suivant, ou null s'il n'y a pas de résultats supplémentaires.

Erreurs

- `AccessDeniedException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

RunStatement action (Python : run_statement)

Exécute la déclaration.

Demande

- `SessionId` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

ID de séance de la déclaration à exécuter.

- `Code` – Obligatoire : Chaîne UTF-8, d'une longueur ne dépassant pas 68000 octets.

  Code de la déclaration à exécuter.

- `RequestOrigin` – chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets, correspondant au Custom string pattern #31.

  Origine de la demande.

Réponse

- `Id` : nombre (entier).

  Renvoie l'ID de la déclaration exécutée.

Erreurs

- `EntityNotFoundException`
- `AccessDeniedException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`
- `ValidationException`
- `ResourceNumberLimitExceededException`
- `IllegalSessionStateException`

CancelStatement action (Python : annuler_statement)

Annule la déclaration.

Demande

- `SessionId` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID de séance de la déclaration à annuler.

- `Id` – Obligatoire : nombre (entier).

  ID de la déclaration à annuler.

- `RequestOrigin` – chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets, correspondant au Custom string pattern #31.

  Origine de la demande d'annuler la déclaration.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `AccessDeniedException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`
- `IllegalSessionStateException`

GetStatement action (Python : get_statement)

Récupère la déclaration.

Demande

- `SessionId` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID de séance de la déclaration.

- `Id` – Obligatoire : nombre (entier).

  L'ID de la déclaration.

- `RequestOrigin` – chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets, correspondant au Custom string pattern #31.

Origine de la demande.

Réponse

- `Statement` – Un objet Instruction.

  Renvoie la déclaration.

Erreurs

- `AccessDeniedException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`
- `IllegalSessionStateException`

ListStatements action (Python : list_statements)

Répertorie les déclarations de la séance.

Demande

- `SessionId` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID de séance des déclarations.

- `RequestOrigin` – chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets, correspondant au Custom string pattern #31.

  Origine de la demande de liste des déclarations.

- `NextToken` – Chaîne UTF-8, d'une longueur maximale de 400 000 octets.

  Jeton de continuation, s'il s'agit d'un appel de continuation.

Réponse

- `Statements` : un tableau d'objets Instruction.

  Renvoie la liste des déclarations.
- `NextToken` – Chaîne UTF-8, d'une longueur maximale de 400 000 octets.

  Jeton de continuation, si toutes les déclarations n'ont pas encore été retournées.

Erreurs

- `AccessDeniedException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`
- `IllegalSessionStateException`

GetGlueIdentityCenterConfiguration action (Python : get_glue_identity_center_configuration)

Récupère les détails de configuration actuels AWS Glue d'Identity Center, y compris les informations relatives à l'instance et à l'application Identity Center associées.

Demande

- Aucun paramètre de demande.

Réponse

Réponse contenant les détails de configuration AWS Glue d'Identity Center.

- `ApplicationArn` – Chaîne UTF-8, d'une longueur comprise entre 10 et 1 224 octets.

  Le nom de ressource Amazon (ARN) de l'application Identity Center associée à la AWS Glue configuration.
- `InstanceArn` – Chaîne UTF-8, d'une longueur comprise entre 10 et 1 224 octets.

Le nom de ressource Amazon (ARN) de l'instance Identity Center associée à la AWS Glue configuration.

- `Scopes` – Tableau de chaînes UTF-8.

  Liste des champs d'application d'Identity Center qui définissent les autorisations et les niveaux d'accès pour la AWS Glue configuration.

Erreurs

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `AccessDeniedException`
- `ConcurrentModificationException`

UpdateGlueIdentityCenterConfiguration action (Python : update_glue_identity_center_configuration)

Met à jour la configuration existante de AWS Glue l'Identity Center, permettant de modifier les étendues et les autorisations de l'intégration.

Demande

Demande de mise à jour d'une configuration AWS Glue Identity Center existante.

- `Scopes` – Tableau de chaînes UTF-8, entre 1 et 50 chaînes.

  Liste des étendues d'Identity Center qui définissent les autorisations et les niveaux d'accès mis à jour pour la AWS Glue configuration.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `AccessDeniedException`
- `ConcurrentModificationException`

CreateGlueIdentityCenterConfiguration action (Python : create_glue_identity_center_configuration)

Crée une nouvelle configuration AWS Glue Identity Center pour permettre l'intégration entre AWS Glue AWS IAM Identity Center à des fins d'authentification et d'autorisation.

Demande

Demande de création d'une nouvelle configuration AWS Glue d'Identity Center.

- `InstanceArn`— Obligatoire : chaîne UTF-8 d'une longueur d'au moins 10 octets ou supérieure à 1224 octets.

  Le nom de ressource Amazon (ARN) de l'instance Identity Center à associer à la AWS Glue configuration.
- `Scopes` – Tableau de chaînes UTF-8, entre 1 et 50 chaînes.

  Liste des champs d'application d'Identity Center qui définissent les autorisations et les niveaux d'accès pour la AWS Glue configuration.

Réponse

Réponse à la création d'une nouvelle configuration AWS Glue d'Identity Center.

- `ApplicationArn` – Chaîne UTF-8, d'une longueur comprise entre 10 et 1 224 octets.

  Le nom de ressource Amazon (ARN) de l'application Identity Center créée pour la AWS Glue configuration.

Erreurs

- `InvalidInputException`
- `AlreadyExistsException`
- `InternalServiceException`
- `OperationTimeoutException`
- `AccessDeniedException`
- `ConcurrentModificationException`

DeleteGlueIdentityCenterConfiguration action (Python : delete_glue_identity_center_configuration)

Supprime la configuration existante de l' AWS Glue Identity Center, supprimant ainsi l'intégration entre IAM Identity Center AWS Glue et AWS IAM Identity Center.

Demande

- Aucun paramètre de demande.

Réponse

- Paramètres d’absence de réponse.

Erreurs

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `AccessDeniedException`
- `ConcurrentModificationException`

API de points de terminaison de développement

L'API Development endpoints décrit l' AWS Glue API liée aux tests à l'aide d'une personnalisation DevEndpoint.

Types de données

- DevEndpoint structure
- DevEndpointCustomLibraries structure

DevEndpoint structure

Point de terminaison de développement où un développeur peut déboguer à distance des scripts ETL (extraction, transformation et chargement).

Champs

- `EndpointName` – Chaîne UTF-8.

  Le nom de l'`DevEndpoint`.
- `RoleArn` – Chaîne UTF-8, correspondant au AWS IAM ARN string pattern.

  Amazon Resource Name (ARN) du rôle IAM utilisé dans ce `DevEndpoint`.
- `SecurityGroupIds` – Tableau de chaînes UTF-8.

  Liste d'identifiants de groupe de sécurité utilisés dans ce `DevEndpoint`.
- `SubnetId` – Chaîne UTF-8.

  ID de sous-réseau pour ce `DevEndpoint`.
- `YarnEndpointAddress` – Chaîne UTF-8.

  Adresse du point de terminaison YARN utilisée par ce `DevEndpoint`.
- `PrivateAddress` – Chaîne UTF-8.

  Adresse IP privée pour accéder au `DevEndpoint` au sein d'un VPC si le `DevEndpoint` est créé dans celui-ci. Le champ `PrivateAddress` est présent uniquement lorsque vous créez le `DevEndpoint` au sein de votre VPC.
- `ZeppelinRemoteSparkInterpreterPort` : nombre (entier).

Port Apache Zeppelin pour l'interpréteur Apache Spark à distance.

- `PublicAddress` – Chaîne UTF-8.

  Adresse IP publique utilisée par ce `DevEndpoint`. Le champ `PublicAddress` est présent uniquement lorsque vous créez un `DevEndpoint` de cloud privé non virtuel.

- `Status` – Chaîne UTF-8.

  Statut actuel de ce `DevEndpoint`.

- `WorkerType` – Chaîne UTF-8 (valeurs valides: `Standard=""` | `G.1X=""` | `G.2X=""` | `G.025X=""` | `G.4X=""` | `G.8X=""` | `Z.2X=""`).

  Type de travail prédéfini qui est alloué au point de terminaison de développement. Accepte la valeur Standard, G.1X ou G.2X.

  - Pour le type de travail `Standard`, chaque travail fournit 4 vCPU, 16 Go de mémoire et 50 Go de disque, ainsi que 2 exécuteurs par travail.
  - Pour le type de travail `G.1X`, chaque travail mappe vers 1 DPU (4 vCPU, 16 Go de mémoire et 64 Go de disque), et fournit 1 exécuteur par travail. Nous vous recommandons ce type d'employé pour les tâches utilisant beaucoup de mémoire.
  - Pour le type de travail `G.2X`, chaque travail mappe vers 2 DPU (8 vCPU, 32 Go de mémoire et 128 Go de disque), et fournit 1 exécuteur par travail. Nous vous recommandons ce type d'employé pour les tâches utilisant beaucoup de mémoire.

  Problème connu : lorsqu'un point de terminaison de développement est créé avec la configuration `G.2X` `WorkerType`, les pilotes Spark pour le point de terminaison de développement s'exécutent sur 4 vCPU, 16 Go de mémoire et un disque de 64 Go.

- `GlueVersion` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #48.

  La version de Glue détermine les versions d'Apache Spark et Python prises AWS Glue en charge. La version Python indique la version prise en charge pour l'exécution de vos scripts ETL sur les points de terminaison de développement.

  Pour plus d'informations sur les AWS Glue versions disponibles et les versions correspondantes de Spark et Python, consultez la version de Glue dans le guide du développeur.

  Les points de terminaison de développement créés sans spécifier de version de Glue sont par défaut Glue 0.9.

Vous pouvez spécifier une version du support Python pour les points de terminaison de développement en utilisant le `Arguments` paramètre dans le `CreateDevEndpoint` ou `UpdateDevEndpoint` APIs. Si aucun argument n'est fourni, la version utilisée par défaut est Python 2.

- `NumberOfWorkers` : nombre (entier).

  Nombre de travaux d'un `workerType` défini qui sont alloués au point de terminaison de développement.

  Le nombre maximal de travaux que vous pouvez définir est de 299 pour `G.1X` et de 149 pour `G.2X`.

- `NumberOfNodes` : nombre (entier).

  Le nombre d'unités de traitement des AWS Glue données (DPUs) allouées à cela`DevEndpoint`.

- `AvailabilityZone` – Chaîne UTF-8.

  La zone de AWS disponibilité dans laquelle `DevEndpoint` il se trouve.

- `VpcId` – Chaîne UTF-8.

  ID du cloud privé virtuel (VPC) utilisé par ce `DevEndpoint`.

- `ExtraPythonLibsS3Path` – Chaîne UTF-8.

  Chemin(s) d'accès à une ou plusieurs bibliothèques Python dans un compartiment Amazon S3 qui doit être chargé dans votre `DevEndpoint`. Plusieurs valeurs doivent être des chemins entiers séparés par une virgule.

  > **Note**
  >
  > Vous pouvez uniquement utiliser des bibliothèques Python pur avec un `DevEndpoint`. Les bibliothèques reposant sur des extensions C, par exemple la bibliothèque d'analyse des données Python pandas, ne sont actuellement pas prises en charge.

- `ExtraJarsS3Path` – Chaîne UTF-8.

  Chemin d'accès à un ou plusieurs fichiers Java `.jar` dans un compartiment S3 qui doit être chargé dans votre `DevEndpoint`.

> **Note**
>
> Vous ne pouvez utiliser que des Java/Scala bibliothèques pures avec un`DevEndpoint`.

- `FailureReason` – Chaîne UTF-8.

  Raison d'un échec actuel dans ce `DevEndpoint`.

- `LastUpdateStatus` – Chaîne UTF-8.

  Statut de la dernière mise à jour.

- `CreatedTimestamp` – Horodatage.

  Le moment où cela DevEndpoint a été créé.

- `LastModifiedTimestamp` – Horodatage.

  Moment auquel ce `DevEndpoint` a été modifié pour la dernière fois.

- `PublicKey` – Chaîne UTF-8.

  Clé publique à utiliser par ce `DevEndpoint` pour l'authentification. Cet attribut est fourni à des fins de rétrocompatibilité, car l'attribut dont l'utilisation est recommandée sont les clés publiques.

- `PublicKeys` – Tableau de chaînes UTF-8, avec 5 chaînes maximum.

  Liste de clés publiques à utiliser par le `DevEndpoints` pour authentification. L'utilisation de cet attribut est plus recommandé qu'une simple clé publique, car les clés publiques vous permettent d'avoir une clé privée différente pour chaque client.

> **Note**
>
> Si vous avez déjà créé un point de terminaison avec une clé publique, vous devez supprimer cette clé pour être en mesure de définir une liste de clés publiques. Appelez l'opération d'API `UpdateDevEndpoint` avec le contenu de la clé publique dans l'attribut `deletePublicKeys` et la liste des nouvelles clés dans l'attribut `addPublicKeys`.

- `SecurityConfiguration` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la structure `SecurityConfiguration` à utiliser avec ce `DevEndpoint`.

- `Arguments` – tableau de mappage de paires clé-valeur, avec 100 paires au maximum.

  Chaque clé est une chaîne UTF-8.

  Chaque valeur est une chaîne UTF-8.

  Mappage des arguments utilisés pour configurer le `DevEndpoint`.

  Les arguments valides sont les suivants :

  - `"--enable-glue-datacatalog": ""`

  Vous pouvez spécifier une version du support Python pour les points de terminaison de développement en utilisant le `Arguments` paramètre dans le `CreateDevEndpoint` ou `UpdateDevEndpoint` APIs. Si aucun argument n'est fourni, la version utilisée par défaut est Python 2.

DevEndpointCustomLibraries structure

Bibliothèques personnalisées à charger dans un point de terminaison de développement.

Champs

- `ExtraPythonLibsS3Path` – Chaîne UTF-8.

  Chemins d'accès à une ou plusieurs bibliothèques Python dans un compartiment Amazon Simple Storage Service (Amazon S3) qui doivent être chargées dans votre `DevEndpoint`. Plusieurs valeurs doivent être des chemins entiers séparés par une virgule.

  > **Note**
  >
  > Vous pouvez uniquement utiliser des bibliothèques Python pur avec un `DevEndpoint`. Les bibliothèques reposant sur des extensions C, par exemple la bibliothèque d'analyse des données Python pandas, ne sont actuellement pas prises en charge.

- `ExtraJarsS3Path` – Chaîne UTF-8.

  Chemin d'accès à un ou plusieurs fichiers Java `.jar` dans un compartiment S3 qui doit être chargé dans votre `DevEndpoint`.

> **Note**
>
> Vous ne pouvez utiliser que des Java/Scala bibliothèques pures avec un`DevEndpoint`.

Opérations

- CreateDevEndpoint action (Python : create_dev_endpoint)
- UpdateDevEndpoint action (Python : update_dev_endpoint)
- DeleteDevEndpoint action (Python : delete_dev_endpoint)
- GetDevEndpoint action (Python : get_dev_endpoint)
- GetDevEndpoints action (Python : get_dev_endpoints)
- BatchGetDevEndpoints action (Python : batch_get_dev_endpoints)
- ListDevEndpoints action (Python : list_dev_endpoints)

CreateDevEndpoint action (Python : create_dev_endpoint)

Crée un nouveau point de terminaison de développement.

Demande

- `EndpointName` – Obligatoire : chaîne UTF-8.

  Nom à affecter au nouveau `DevEndpoint`.
- `RoleArn` – Obligatoire : Chaîne UTF-8, correspondant au AWS IAM ARN string pattern.

  Rôle IAM pour le `DevEndpoint`.
- `SecurityGroupIds` – Tableau de chaînes UTF-8.

  Groupe de sécurité IDs pour les groupes de sécurité à utiliser par le nouveau`DevEndpoint`.
- `SubnetId` – Chaîne UTF-8.

  ID de sous-réseau pour le nouveau `DevEndpoint` à utiliser.
- `PublicKey` – Chaîne UTF-8.

Clé publique à utiliser par ce `DevEndpoint` pour l'authentification. Cet attribut est fourni à des fins de rétrocompatibilité, car l'attribut dont l'utilisation est recommandée sont les clés publiques.

- `PublicKeys` – Tableau de chaînes UTF-8, avec 5 chaînes maximum.

  Liste de clés publiques à utiliser par les points de terminaison de développement pour authentification. L'utilisation de cet attribut est plus recommandé qu'une simple clé publique, car les clés publiques vous permettent d'avoir une clé privée différente pour chaque client.

  > **Note**
  >
  > Si vous avez déjà créé un point de terminaison avec une clé publique, vous devez supprimer cette clé pour être en mesure de définir une liste de clés publiques. Appelez l'API `UpdateDevEndpoint` avec le contenu de la clé publique dans l'attribut `deletePublicKeys` et la liste des nouvelles clés dans l'attribut `addPublicKeys`.

- `NumberOfNodes` : nombre (entier).

  Le nombre d'unités de traitement des AWS Glue données (DPUs) à y affecter`DevEndpoint`.

- `WorkerType` – Chaîne UTF-8 (valeurs valides: `Standard=""` | `G.1X=""` | `G.2X=""` | `G.025X=""` | `G.4X=""` | `G.8X=""` | `Z.2X=""`).

  Type de travail prédéfini qui est alloué au point de terminaison de développement. Accepte la valeur Standard, G.1X ou G.2X.

  - Pour le type de travail `Standard`, chaque travail fournit 4 vCPU, 16 Go de mémoire et 50 Go de disque, ainsi que 2 exécuteurs par travail.
  - Pour le type de travail `G.1X`, chaque travail mappe vers 1 DPU (4 vCPU, 16 Go de mémoire et 64 Go de disque), et fournit 1 exécuteur par travail. Nous vous recommandons ce type d'employé pour les tâches utilisant beaucoup de mémoire.
  - Pour le type de travail `G.2X`, chaque travail mappe vers 2 DPU (8 vCPU, 32 Go de mémoire et 128 Go de disque), et fournit 1 exécuteur par travail. Nous vous recommandons ce type d'employé pour les tâches utilisant beaucoup de mémoire.

  Problème connu : lorsqu'un point de terminaison de développement est créé avec la configuration `G.2X` `WorkerType`, les pilotes Spark pour le point de terminaison de développement s'exécutent sur 4 vCPU, 16 Go de mémoire et un disque de 64 Go.

- `GlueVersion` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #48.

  La version de Glue détermine les versions d'Apache Spark et Python prises AWS Glue en charge. La version Python indique la version prise en charge pour l'exécution de vos scripts ETL sur les points de terminaison de développement.

  Pour plus d'informations sur les AWS Glue versions disponibles et les versions correspondantes de Spark et Python, consultez la version de Glue dans le guide du développeur.

  Les points de terminaison de développement créés sans spécifier de version de Glue sont par défaut Glue 0.9.

  Vous pouvez spécifier une version du support Python pour les points de terminaison de développement en utilisant le `Arguments` paramètre dans le `CreateDevEndpoint` ou `UpdateDevEndpoint` APIs. Si aucun argument n'est fourni, la version utilisée par défaut est Python 2.

- `NumberOfWorkers` : nombre (entier).

  Nombre de travaux d'un `workerType` défini qui sont alloués au point de terminaison de développement.

  Le nombre maximal de travaux que vous pouvez définir est de 299 pour `G.1X` et de 149 pour `G.2X`.

- `ExtraPythonLibsS3Path` – Chaîne UTF-8.

  Chemin(s) d'accès à une ou plusieurs bibliothèques Python dans un compartiment Amazon S3 qui doit être chargé dans votre `DevEndpoint`. Plusieurs valeurs doivent être des chemins entiers séparés par une virgule.

  > **Note**
  >
  > Vous pouvez uniquement utiliser des bibliothèques Python pur avec un `DevEndpoint`. Les bibliothèques reposant sur des extensions C, par exemple la bibliothèque d'analyse des données Python pandas, ne sont pas encore prises en charge.

- `ExtraJarsS3Path` – Chaîne UTF-8.

Chemin d'accès à un ou plusieurs fichiers Java `.jar` dans un compartiment S3 qui doit être chargé dans votre `DevEndpoint`.

- `SecurityConfiguration` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la structure `SecurityConfiguration` à utiliser avec ce `DevEndpoint`.

- `Tags` – tableau de mappage de paires clé-valeur, avec 50 paires au maximum.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 256 octets.

  Balises à utiliser avec ce DevEndpoint. Vous pouvez utiliser des balises pour limiter l'accès au DevEndpoint. Pour plus d'informations sur les tags in AWS Glue, voir AWS Tags in AWS Glue dans le guide du développeur.

- `Arguments` – tableau de mappage de paires clé-valeur, avec 100 paires au maximum.

  Chaque clé est une chaîne UTF-8.

  Chaque valeur est une chaîne UTF-8.

  Mappage des arguments utilisés pour configurer le `DevEndpoint`.

Réponse

- `EndpointName` – Chaîne UTF-8.

  Nom attribué au nouveau `DevEndpoint`.

- `Status` – Chaîne UTF-8.

  Statut actuel du nouveau `DevEndpoint`.

- `SecurityGroupIds` – Tableau de chaînes UTF-8.

  Groupes de sécurité affectés au nouveau `DevEndpoint`.

- `SubnetId` – Chaîne UTF-8.

  ID de sous-réseau affecté au nouveau `DevEndpoint`.

- `RoleArn` – Chaîne UTF-8, correspondant au AWS IAM ARN string pattern.

Amazon Resource Name (ARN) du rôle affecté au nouveau `DevEndpoint`.

- `YarnEndpointAddress` – Chaîne UTF-8.

  Adresse du point de terminaison YARN utilisé par ce `DevEndpoint`.

- `ZeppelinRemoteSparkInterpreterPort` : nombre (entier).

  Port Apache Zeppelin pour l'interpréteur Apache Spark à distance.

- `NumberOfNodes` : nombre (entier).

  Le nombre d'unités de traitement des AWS Glue données (DPUs) allouées à cela DevEndpoint.

- `WorkerType` – Chaîne UTF-8 (valeurs valides: `Standard=""` | `G.1X=""` | `G.2X=""` | `G.025X=""` | `G.4X=""` | `G.8X=""` | `Z.2X=""`).

  Type de travail prédéfini qui est alloué au point de terminaison de développement. La valeur peut être Standard, G.1X ou G.2X.

- `GlueVersion` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #48.

  La version de Glue détermine les versions d'Apache Spark et Python prises AWS Glue en charge. La version Python indique la version prise en charge pour l'exécution de vos scripts ETL sur les points de terminaison de développement.

  Pour plus d'informations sur les AWS Glue versions disponibles et les versions correspondantes de Spark et Python, consultez la version de Glue dans le guide du développeur.

- `NumberOfWorkers` : nombre (entier).

  Nombre de travaux d'un `workerType` défini qui sont alloués au point de terminaison de développement.

- `AvailabilityZone` – Chaîne UTF-8.

  La zone de AWS disponibilité dans laquelle `DevEndpoint` il se trouve.

- `VpcId` – Chaîne UTF-8.

  ID du cloud privé virtuel (VPC) utilisé par ce `DevEndpoint`.

- `ExtraPythonLibsS3Path` – Chaîne UTF-8.

Chemin(s) d'accès à une ou plusieurs bibliothèques Python dans un compartiment S3 qui sera chargé dans votre `DevEndpoint`.

- `ExtraJarsS3Path` – Chaîne UTF-8.

  Chemi(s)n d'accès à un ou plusieurs fichiers Java `.jar` dans un compartiment S3 qui sera chargé dans votre `DevEndpoint`.

- `FailureReason` – Chaîne UTF-8.

  Raison d'un échec actuel dans ce `DevEndpoint`.

- `SecurityConfiguration` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la structure `SecurityConfiguration` utilisée avec ce `DevEndpoint`.

- `CreatedTimestamp` – Horodatage.

  Moment où ce `DevEndpoint` a été créé.

- `Arguments` – tableau de mappage de paires clé-valeur, avec 100 paires au maximum.

  Chaque clé est une chaîne UTF-8.

  Chaque valeur est une chaîne UTF-8.

  Mappage des arguments utilisés pour configurer ce `DevEndpoint`.

  Les arguments valides sont les suivants :

  - `"--enable-glue-datacatalog": ""`

  Vous pouvez spécifier une version du support Python pour les points de terminaison de développement en utilisant le `Arguments` paramètre dans le `CreateDevEndpoint` ou `UpdateDevEndpoint` APIs. Si aucun argument n'est fourni, la version utilisée par défaut est Python 2.

Erreurs

- `AccessDeniedException`
- `AlreadyExistsException`
- `IdempotentParameterMismatchException`

- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`
- `ValidationException`
- `ResourceNumberLimitExceededException`

UpdateDevEndpoint action (Python : update_dev_endpoint)

Met à jour un point de terminaison de développement spécifié.

Demande

- `EndpointName` – Obligatoire : chaîne UTF-8.

  Nom du `DevEndpoint` à mettre à jour.
- `PublicKey` – Chaîne UTF-8.

  Clé publique pour le `DevEndpoint` à utiliser.
- `AddPublicKeys` – Tableau de chaînes UTF-8, avec 5 chaînes maximum.

  Liste des clés publiques pour le `DevEndpoint` à utiliser.
- `DeletePublicKeys` – Tableau de chaînes UTF-8, avec 5 chaînes maximum.

  Liste de clés publiques à supprimer du `DevEndpoint`.
- `CustomLibraries` – Un objet DevEndpointCustomLibraries.

  Bibliothèques Python ou Java personnalisées à charger dans le `DevEndpoint`.
- `UpdateEtlLibraries` : booléen.

  `True` si la liste des bibliothèques personnalisées à charger dans le point de terminaison de développement doit être mise à jour, sinon `False`.
- `DeleteArguments` – Tableau de chaînes UTF-8.

  Liste des clés d'argument à supprimer du mappage d'arguments utilisé pour configurer le `DevEndpoint`.
- `AddArguments` – tableau de mappage de paires clé-valeur, avec 100 paires au maximum.

Chaque clé est une chaîne UTF-8.

Chaque valeur est une chaîne UTF-8.

Mappage d'arguments à ajouter au mappage d'arguments utilisé pour configurer le `DevEndpoint`.

Les arguments valides sont les suivants :

- `"--enable-glue-datacatalog": ""`

Vous pouvez spécifier une version du support Python pour les points de terminaison de développement en utilisant le `Arguments` paramètre dans le `CreateDevEndpoint` ou `UpdateDevEndpoint` APIs. Si aucun argument n'est fourni, la version utilisée par défaut est Python 2.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`
- `ValidationException`

DeleteDevEndpoint action (Python : delete_dev_endpoint)

Supprime un point de terminaison de développement spécifié.

Demande

- `EndpointName` – Obligatoire : chaîne UTF-8.

  Le nom de l'`DevEndpoint`.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`

GetDevEndpoint action (Python : get_dev_endpoint)

Récupère des informations sur un point de terminaison de développement spécifié.

> **Note**
>
> Lorsque vous créez un point de terminaison de développement dans un cloud privé virtuel (VPC), AWS Glue renvoie uniquement une adresse IP privée et le champ de l'adresse IP publique n'est pas renseigné. Lorsque vous créez un point de terminaison de développement non VPC, il AWS Glue renvoie uniquement une adresse IP publique.

Demande

- `EndpointName` – Obligatoire : chaîne UTF-8.

  Nom du `DevEndpoint` pour lequel des informations seront récupérées.

Réponse

- `DevEndpoint` – Un objet DevEndpoint.

  Définition d'un `DevEndpoint`.

Erreurs

- `EntityNotFoundException`

- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`

GetDevEndpoints action (Python : get_dev_endpoints)

Récupère tous les points de terminaison de développement de ce AWS compte.

> **Note**
>
> Lorsque vous créez un point de terminaison de développement dans un Virtual Private Cloud (VPC), AWS Glue renvoie uniquement une adresse IP privée et le champ de l'adresse IP publique n'est pas renseigné. Lorsque vous créez un point de terminaison de développement non VPC, il AWS Glue renvoie uniquement une adresse IP publique.

Demande

- `MaxResults` – Nombre (entier), compris entre 1 et 1 000.

  Taille maximale des informations à renvoyer.
- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, s'il s'agit d'un appel de continuation.

Réponse

- `DevEndpoints` : un tableau d'objets DevEndpoint.

  Liste des définitions de `DevEndpoint`.
- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, si toutes les définitions de `DevEndpoint` n'ont pas encore été renvoyées.

Erreurs

- `EntityNotFoundException`

- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`

BatchGetDevEndpoints action (Python : batch_get_dev_endpoints)

Renvoie une liste des métadonnées de ressource pour une liste donnée de noms de point de terminaison de développement. Après avoir appelé l'opération `ListDevEndpoints`, vous pouvez appeler cette opération pour accéder aux données sur lesquelles des autorisations vous ont été octroyées. Cette opération prend en charge toutes les autorisations IAM, y compris les conditions d'autorisation qui utilisent des balises.

Demande

- `customerAccountId` – Chaîne UTF-8.

  L'identifiant AWS du compte.
- `DevEndpointNames` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 25 chaînes maximum.

  Liste de noms de `DevEndpoint`, qui peuvent être les noms renvoyés à partir de l'opération `ListDevEndpoint`.

Réponse

- `DevEndpoints` : un tableau d'objets DevEndpoint.

  Liste des définitions de `DevEndpoint`.
- `DevEndpointsNotFound` – tableau de chaînes UTF-8, avec 1 chaîne minimum et 25 chaînes maximum.

  Liste des `DevEndpoints` introuvables.

Erreurs

- `AccessDeniedException`
- `InternalServiceException`

- `OperationTimeoutException`
- `InvalidInputException`

ListDevEndpoints action (Python : list_dev_endpoints)

Récupère les noms de toutes les ressources `DevEndpoint` dans ce compte AWS , ou les ressources avec la balise spécifiée. Cette opération vous permet de voir quelles ressources sont disponibles dans votre compte, et leurs noms.

Cette opération accepte le champ `Tags` facultatif que vous pouvez utiliser comme filtre sur la réponse, afin que les ressources balisées puissent être récupérées en tant que groupe. Si vous choisissez d'utiliser le filtrage des balises, seules les ressources avec la balise sont récupérées.

Demande

- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, s'il s'agit d'une requête de continuation.
- `MaxResults` – Nombre (entier), compris entre 1 et 1 000.

  La taille maximale d'une liste à renvoyer.
- `Tags` – tableau de mappage de paires clé-valeur, avec 50 paires au maximum.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 256 octets.

  Spécifie de renvoyer uniquement les ressources balisées.

Réponse

- `DevEndpointNames` – Tableau de chaînes UTF-8.

  Noms de tous les `DevEndpoint` dans le compte ou les `DevEndpoint` avec les balises spécifiées.
- `NextToken` – Chaîne UTF-8.

  Jeton continuation, si la liste renvoyée ne contient pas la dernière métrique disponible.

Erreurs

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`

Registre de schémas

L'API de registre des schémas décrit les types de données et l'API liés à l'utilisation des schémas dans AWS Glue.

Types de données

- RegistryId structure
- RegistryListItem structure
- MetadataInfo structure
- OtherMetadataValueListItem structure
- SchemaListItem structure
- SchemaVersionListItem structure
- MetadataKeyValuePair structure
- SchemaVersionErrorItem structure
- ErrorDetails structure
- SchemaVersionNumber structure
- SchemaId structure

RegistryId structure

Structure d'encapsulation qui peut contenir le nom et l'Amazon Resource Name (ARN) du registre.

Champs

- `RegistryName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #46.

Nom du registre. Utilisé uniquement pour la recherche. `RegistryArn` ou `RegistryName` doit être fourni.

- `RegistryArn` – Chaîne UTF-8, d'une longueur comprise entre 1 et 10 240 octets, correspondant au Custom string pattern #50.

  ARN du registre à mettre à jour. `RegistryArn` ou `RegistryName` doit être fourni.

RegistryListItem structure

Une structure contenant les détails d'un registre.

Champs

- `RegistryName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #46.

  Nom du registre.

- `RegistryArn` – Chaîne UTF-8, d'une longueur comprise entre 1 et 10 240 octets, correspondant au Custom string pattern #50.

  L'Amazon Resource Name (ARN) du registre.

- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description du registre.

- `Status` – Chaîne UTF-8 (valeurs valides : `AVAILABLE` | `DELETING`).

  Statut du registre.

- `CreatedTime` – Chaîne UTF-8.

  Données que le registre a été créé.

- `UpdatedTime` – Chaîne UTF-8.

  Date à laquelle le registre a été mis à jour.

MetadataInfo structure

Structure contenant des informations de métadonnées pour une version de schéma.

Champs

- `MetadataValue` – Chaîne UTF-8, d'une longueur comprise entre 1 et 256 octets, correspondant au Custom string pattern #14.

  Valeur correspondant à une clé de métadonnées.

- `CreatedTime` – Chaîne UTF-8.

  Heure à laquelle l'entrée a été créée.

- `OtherMetadataValueList` : un tableau d'objets OtherMetadataValueListItem.

  Autres métadonnées appartenant à la même clé de métadonnées.

OtherMetadataValueListItem structure

Structure contenant d'autres métadonnées pour une version de schéma appartenant à la même clé de métadonnées.

Champs

- `MetadataValue` – Chaîne UTF-8, d'une longueur comprise entre 1 et 256 octets, correspondant au Custom string pattern #14.

  Valeur correspondante de la clé de métadonnées pour les autres métadonnées appartenant à la même clé de métadonnées.

- `CreatedTime` – Chaîne UTF-8.

  Heure à laquelle l'entrée a été créée.

SchemaListItem structure

Objet qui contient des détails minimaux pour un schéma.

Champs

- `RegistryName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #46.

  Nom du registre dans lequel se trouve le schéma.

- `SchemaName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #46.

  Nom du schéma.

- `SchemaArn` – Chaîne UTF-8, d'une longueur comprise entre 1 et 10 240 octets, correspondant au Custom string pattern #50.

  Amazon Resource Name (ARN) du schéma.

- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description du schéma.

- `SchemaStatus` – Chaîne UTF-8 (valeurs valides : `AVAILABLE` | `PENDING` | `DELETING`).

  Statut du schéma.

- `CreatedTime` – Chaîne UTF-8.

  Date et heure de création d'un schéma.

- `UpdatedTime` – Chaîne UTF-8.

  Date et heure de mise à jour d'un schéma.

SchemaVersionListItem structure

Objet contenant les détails d'une version de schéma.

Champs

- `SchemaArn` – Chaîne UTF-8, d'une longueur comprise entre 1 et 10 240 octets, correspondant au Custom string pattern #50.

  L'Amazon Resource Name (ARN) du filtre.

- `SchemaVersionId` – Chaîne UTF-8, d'une longueur exacte de 36 octets, correspondant au Custom string pattern #45.

  Identifiant unique de la version du schéma.
- `VersionNumber` – Nombre (long), compris entre 1 et 100 000.

  Numéro de version du schéma.
- `Status` – Chaîne UTF-8 (valeurs valides : `AVAILABLE` | `PENDING` | `FAILURE` | `DELETING`).

  Statut de la version du schéma.
- `CreatedTime` – Chaîne UTF-8.

  Date et heure de création à laquelle la version de schéma a été créée.

MetadataKeyValuePair structure

Structure contenant une paire de valeurs clé pour les métadonnées.

Champs

- `MetadataKey` – chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets, correspondant au Custom string pattern #14.

  Clé de métadonnées.
- `MetadataValue` – Chaîne UTF-8, d'une longueur comprise entre 1 et 256 octets, correspondant au Custom string pattern #14.

  Valeur correspondant à une clé de métadonnées.

SchemaVersionErrorItem structure

Objet contenant les détails d'erreur d'une opération sur une version de schéma.

Champs

- `VersionNumber` – Nombre (long), compris entre 1 et 100 000.

  Numéro de version du schéma.
- `ErrorDetails` – Un objet ErrorDetails.

Détails de l'erreur pour la version du schéma.

ErrorDetails structure

Objet contenant des détails d'erreur.

Champs

- `ErrorCode` – Chaîne UTF-8.

  Code d'erreur correspondant à une erreur.
- `ErrorMessage` – Chaîne UTF-8.

  Message d'erreur pour une erreur.

SchemaVersionNumber structure

Structure contenant les informations de version du schéma.

Champs

- `LatestVersion` : booléen.

  Dernière version disponible pour le schéma.
- `VersionNumber` – Nombre (long), compris entre 1 et 100 000.

  Numéro de version du schéma.

SchemaId structure

ID unique du schéma dans le registre des AWS Glue schémas.

Champs

- `SchemaArn` – Chaîne UTF-8, d'une longueur comprise entre 1 et 10 240 octets, correspondant au Custom string pattern #50.

  L'Amazon Resource Name (ARN) du filtre. `SchemaArn` ou `SchemaName` doit être fourni.

- `SchemaName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #46.

  Nom du schéma. `SchemaArn` ou `SchemaName` doit être fourni.

- `RegistryName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #46.

  Nom du registre de schémas contenant le schéma.

Opérations

- CreateRegistry action (Python : create_registry)
- CreateSchema action (Python : créer_schéma)
- GetSchema action (Python : get_schema)
- ListSchemaVersions action (Python : list_schema_versions)
- GetSchemaVersion action (Python : get_schema_version)
- GetSchemaVersionsDiff action (Python : get_schema_versions_diff)
- ListRegistries action (Python : list_registries)
- ListSchemas action (Python : list_schemas)
- RegisterSchemaVersion action (Python : register_schema_version)
- UpdateSchema action (Python : update_schema)
- CheckSchemaVersionValidity action (Python : check_schema_version_validity)
- UpdateRegistry action (Python : update_registry)
- GetSchemaByDefinition action (Python : get_schema_by_definition)
- GetRegistry action (Python : get_registry)
- PutSchemaVersionMetadata action (Python : put_schema_version_metadata)
- QuerySchemaVersionMetadata action (Python : query_schema_version_metadata)
- RemoveSchemaVersionMetadata action (Python : remove_schema_version_metadata)
- DeleteRegistry action (Python : supprimer_registre)
- DeleteSchema action (Python : supprimer_schéma)
- DeleteSchemaVersions action (Python : delete_schema_versions)

CreateRegistry action (Python : create_registry)

Crée un registre qui peut être utilisé pour contenir une collection de schémas.

Demande

- `RegistryName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #46.

  Nom du registre à créer, d'une longueur maximale de 255, et ne pouvant contenir que des lettres, des chiffres, trait d'union, trait de soulignement, symbole dollar ou dièse. Pas d'espace.
- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description du registre. Si la description n'est pas fournie, aucune valeur par défaut ne sera attribuée.
- `Tags` – Tableau de mappage de paires clé-valeur, avec 50 paires au maximum.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 256 octets.

  AWS balises contenant une paire clé-valeur et pouvant être recherchées par console, ligne de commande ou API.

Réponse

- `RegistryArn` – Chaîne UTF-8, d'une longueur comprise entre 1 et 10 240 octets, correspondant au Custom string pattern #50.

  Amazon Resource Name (ARN) du registre nouvellement créé.
- `RegistryName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #46.

  Nom du registre.
- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description du registre.

- `Tags` – Tableau de mappage de paires clé-valeur, avec 50 paires au maximum.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 256 octets.

  Balises pour le registre.

Erreurs

- `InvalidInputException`
- `AccessDeniedException`
- `AlreadyExistsException`
- `ResourceNumberLimitExceededException`
- `ConcurrentModificationException`
- `ThrottlingException`
- `InternalServiceException`

CreateSchema action (Python : créer_schéma)

Crée un jeu de schémas et enregistre la définition de schéma. Renvoie une erreur si le jeu de schémas existe déjà sans pour autant enregistrer la version.

Lorsque le jeu de schémas est créé, un point de contrôle de version est défini sur la première version. Le mode de compatibilité « DISABLED » empêche toute version de schéma supplémentaire d'être ajoutée après la première version de schéma. Pour tous les autres modes de compatibilité, la validation des paramètres de compatibilité ne sera appliquée qu'à partir de la deuxième version lorsque l'API `RegisterSchemaVersion` est utilisée.

Lorsque cette API est appelée sans `RegistryId`, cela créera une entrée pour un « registre par défaut » dans les tables de base de données du registre, si elle n'est pas déjà présente.

Demande

- `RegistryId` – Un objet RegistryId.

Il s'agit d'une forme d'encapsulation qui contient les champs d'identité du registre. S'il n'est pas fourni, le registre par défaut sera utilisé. Le format ARN pour celui-ci sera : `arn:aws:glue:us-east-2:<customer id>:registry/default-registry:random-5-letter-id`.

- `SchemaName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #46.

  Nom du schéma à créer d'une longueur maximale de 255. Il ne peut contenir que des lettres, chiffres, tirets, traits de soulignement, signes dollar ou signes de hachage. Pas d'espace.

- `DataFormat` – Obligatoire : Chaîne UTF-8 (valeurs valides : `AVRO` | `JSON` | `PROTOBUF`).

  Format de données de la définition de schéma. À l'heure actuelle, `AVRO`, `JSON` et `PROTOBUF` sont pris en charge.

- `Compatibility` – Chaîne UTF-8 (valeurs valides : `NONE` | `DISABLED` | `BACKWARD` | `BACKWARD_ALL` | `FORWARD` | `FORWARD_ALL` | `FULL` | `FULL_ALL`).

  Mode de compatibilité du schéma. Les valeurs possibles sont :

  - NONE (AUCUN) : aucun mode de compatibilité ne s'applique. Vous pouvez utiliser ce choix dans les scénarios de développement ou si vous ne connaissez pas le mode de compatibilité que vous souhaitez appliquer aux schémas. Toute nouvelle version ajoutée sera acceptée sans faire l'objet d'un contrôle de compatibilité.
  - DISABLED (DÉSACTIVÉ) : ce choix de compatibilité empêche la gestion des versions pour un schéma particulier. Vous pouvez utiliser ce choix pour empêcher la gestion des versions futures d'un schéma.
  - BACKWARD (DESCENDANT) : ce choix de compatibilité est recommandé, car il permet aux récepteurs de données de lire à la fois la version actuelle et une version antérieure du schéma. Cela signifie, par exemple, qu'une nouvelle version de schéma ne peut pas supprimer des champs de données ou modifier le type de ces champs, de sorte qu'ils ne peuvent pas être lus par les lecteurs utilisant la version précédente.
  - BACKWARD_ALL (DESCENDANT_TOUT) : ce choix de compatibilité permet aux récepteurs de données de lire à la fois la version actuelle et toutes les versions antérieures du schéma. Vous pouvez utiliser ce choix lorsque vous devez supprimer des champs ou ajouter des champs facultatifs, et vérifier la compatibilité avec toutes les versions de schéma précédentes.
  - ASCENDANT : ce choix de compatibilité permet aux récepteurs de données de lire à la fois la version actuelle et la version suivante du schéma, mais pas nécessairement les versions ultérieures. Vous pouvez utiliser ce choix lorsque vous devez ajouter des champs ou supprimer

des champs facultatifs, mais uniquement vérifier la compatibilité avec la dernière version du schéma.

- ASCENDANT_TOUT : ce choix de compatibilité permet aux récepteurs de données de lire les données écrites par les producteurs de tout nouveau schéma enregistré. Vous pouvez utiliser ce choix lorsque vous devez ajouter des champs ou supprimer des champs facultatifs, et vérifier la compatibilité avec toutes les versions de schéma précédentes.
- COMPLET : ce choix de compatibilité permet aux récepteurs de données de lire les données écrites par les producteurs à l'aide de la version précédente ou suivante du schéma, mais pas nécessairement des versions antérieures ou ultérieures. Vous pouvez utiliser ce choix lorsque vous devez ajouter ou supprimer des champs facultatifs, mais uniquement vérifier la compatibilité par rapport à la dernière version du schéma.
- COMPLET_TOUT : ce choix de compatibilité permet aux récepteurs de données de lire les données écrites par les producteurs utilisant toutes les versions de schéma précédentes. Vous pouvez utiliser ce choix lorsque vous devez ajouter ou supprimer des champs facultatifs et vérifier la compatibilité avec toutes les versions de schéma précédentes.

- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description facultative du schéma. Si la description n'est pas fournie, aucune valeur par défaut ne sera attribuée automatiquement.

- `Tags` – Tableau de mappage de paires clé-valeur, avec 50 paires au maximum.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 256 octets.

  AWS balises contenant une paire clé-valeur et pouvant être recherchées par console, ligne de commande ou API. Si spécifié, suit le AWS tags-on-create modèle.

- `SchemaDefinition` – Chaîne UTF-8, d'une longueur comprise entre 1 et 170 000 octets, correspondant au Custom string pattern #13.

  Définition de schéma à l'aide du paramètre `DataFormat` pour `SchemaName`.

Réponse

- `RegistryName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #46.

Nom du registre.

- `RegistryArn` – Chaîne UTF-8, d'une longueur comprise entre 1 et 10 240 octets, correspondant au Custom string pattern #50.

  L'Amazon Resource Name (ARN) du registre.

- `SchemaName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #46.

  Nom du schéma.

- `SchemaArn` – Chaîne UTF-8, d'une longueur comprise entre 1 et 10 240 octets, correspondant au Custom string pattern #50.

  L'Amazon Resource Name (ARN) du filtre.

- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description du schéma si spécifié lors de sa création.

- `DataFormat` – Chaîne UTF-8 (valeurs valides : `AVRO` | `JSON` | `PROTOBUF`).

  Format de données de la définition de schéma. À l'heure actuelle, `AVRO`, `JSON` et `PROTOBUF` sont pris en charge.

- `Compatibility` – Chaîne UTF-8 (valeurs valides : `NONE` | `DISABLED` | `BACKWARD` | `BACKWARD_ALL` | `FORWARD` | `FORWARD_ALL` | `FULL` | `FULL_ALL`).

  Mode de compatibilité du schéma.

- `SchemaCheckpoint` – Nombre (long), compris entre 1 et 100 000.

  Numéro de version du point de contrôle (la dernière fois que le mode de compatibilité a été modifié).

- `LatestSchemaVersion` – Nombre (long), compris entre 1 et 100 000.

  Version la plus récente du schéma associé à la définition de schéma renvoyée.

- `NextSchemaVersion` – Nombre (long), compris entre 1 et 100 000.

  Version suivante du schéma associé à la définition de schéma renvoyée.

- `SchemaStatus` – Chaîne UTF-8 (valeurs valides : `AVAILABLE` | `PENDING` | `DELETING`).

Statut du schéma.

- `Tags` – Tableau de mappage de paires clé-valeur, avec 50 paires au maximum.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 256 octets.

  Balises pour le schéma.

- `SchemaVersionId` – Chaîne UTF-8, d'une longueur exacte de 36 octets, correspondant au Custom string pattern #45.

  Identifiant unique de la première version du schéma.

- `SchemaVersionStatus` – Chaîne UTF-8 (valeurs valides : `AVAILABLE` | `PENDING` | `FAILURE` | `DELETING`).

  Statut de la première version du schéma créée.

Erreurs

- `InvalidInputException`
- `AccessDeniedException`
- `EntityNotFoundException`
- `AlreadyExistsException`
- `ResourceNumberLimitExceededException`
- `ConcurrentModificationException`
- `ThrottlingException`
- `InternalServiceException`

GetSchema action (Python : get_schema)

Décrit en détail le schéma spécifié.

Demande

- `SchemaId` – Obligatoire : un objet SchemaId.

Structure d'encapsulation contenant les champs d'identité de schéma. La structure contient :

- SchemaId$ SchemaArn : Le nom de ressource Amazon (ARN) du schéma. Il faut fournir soit `SchemaArn`, soit `SchemaName` et `RegistryName`.
- SchemaId$ SchemaName : nom du schéma. Il faut fournir soit `SchemaArn`, soit `SchemaName` et `RegistryName`.

Réponse

- `RegistryName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #46.

  Nom du registre.

- `RegistryArn` – Chaîne UTF-8, d'une longueur comprise entre 1 et 10 240 octets, correspondant au Custom string pattern #50.

  L'Amazon Resource Name (ARN) du registre.

- `SchemaName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #46.

  Nom du schéma.

- `SchemaArn` – Chaîne UTF-8, d'une longueur comprise entre 1 et 10 240 octets, correspondant au Custom string pattern #50.

  L'Amazon Resource Name (ARN) du filtre.

- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description du schéma si spécifié lors de sa création

- `DataFormat` – Chaîne UTF-8 (valeurs valides : `AVRO` | `JSON` | `PROTOBUF`).

  Format de données de la définition de schéma. À l'heure actuelle, `AVRO`, `JSON` et `PROTOBUF` sont pris en charge.

- `Compatibility` – Chaîne UTF-8 (valeurs valides : `NONE` | `DISABLED` | `BACKWARD` | `BACKWARD_ALL` | `FORWARD` | `FORWARD_ALL` | `FULL` | `FULL_ALL`).

  Mode de compatibilité du schéma.

- `SchemaCheckpoint` – Nombre (long), compris entre 1 et 100 000.

  Numéro de version du point de contrôle (la dernière fois que le mode de compatibilité a été modifié).
- `LatestSchemaVersion` – Nombre (long), compris entre 1 et 100 000.

  Version la plus récente du schéma associé à la définition de schéma renvoyée.
- `NextSchemaVersion` – Nombre (long), compris entre 1 et 100 000.

  Version suivante du schéma associé à la définition de schéma renvoyée.
- `SchemaStatus` – Chaîne UTF-8 (valeurs valides : `AVAILABLE` | `PENDING` | `DELETING`).

  Statut du schéma.
- `CreatedTime` – Chaîne UTF-8.

  Date et heure de création du schéma.
- `UpdatedTime` – Chaîne UTF-8.

  Date et heure de mise à jour du schéma.

Erreurs

- `InvalidInputException`
- `AccessDeniedException`
- `EntityNotFoundException`
- `ThrottlingException`
- `InternalServiceException`

ListSchemaVersions action (Python : list_schema_versions)

Renvoie une liste des versions de schéma que vous avez créées, avec un minimum d'informations. Les versions de schémas dont le statut est Deleted ne seront pas incluses dans les résultats. Des résultats vides seront retournés s'il n'y a pas de version de schéma disponible.

Demande

- `SchemaId` – Obligatoire : un objet SchemaId.

Structure d'encapsulation contenant les champs d'identité de schéma. La structure contient :

- SchemaId$ SchemaArn : Le nom de ressource Amazon (ARN) du schéma. Il faut fournir soit `SchemaArn`, soit `SchemaName` et `RegistryName`.
- SchemaId$ SchemaName : nom du schéma. Il faut fournir soit `SchemaArn`, soit `SchemaName` et `RegistryName`.

- `MaxResults` – Nombre (entier), compris entre 1 et 100.

  Nombre maximum de résultats requis par page. Si la valeur n'est pas fournie, elle par défaut sera de 25 par page.

- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, s'il s'agit d'un appel de continuation.

Réponse

- `Schemas` : un tableau d'objets SchemaVersionListItem.

  Tableau d'objets `SchemaVersionList` contenant les détails de chaque version de schéma.

- `NextToken` – Chaîne UTF-8.

  Jeton de continuation pour la pagination de la liste des jetons renvoyés, renvoyé si le segment actuel de la liste n'est pas le dernier.

Erreurs

- `InvalidInputException`
- `AccessDeniedException`
- `EntityNotFoundException`
- `ThrottlingException`
- `InternalServiceException`

GetSchemaVersion action (Python : get_schema_version)

Récupère le schéma spécifié par son ID unique attribué lors de la création ou de l'enregistrement d'une version du schéma. Les versions de schémas dont le statut est Deleted ne seront pas incluses dans les résultats.

Demande

- `SchemaId` – Un objet SchemaId.

  Structure d'encapsulation contenant les champs d'identité de schéma. La structure contient :

  - SchemaId$ SchemaArn : Le nom de ressource Amazon (ARN) du schéma. Il faut fournir soit `SchemaArn`, soit `SchemaName` et `RegistryName`.
  - SchemaId$ SchemaName : nom du schéma. Il faut fournir soit `SchemaArn`, soit `SchemaName` et `RegistryName`.
- `SchemaVersionId` – Chaîne UTF-8, d'une longueur exacte de 36 octets, correspondant au Custom string pattern #45.

  Le `SchemaVersionId` de la version du schéma. Ce champ est requis pour l'extraction par ID de schéma. Il faut fournir soit cet élément, soit l'encapsulateur `SchemaId`.
- `SchemaVersionNumber` – Un objet SchemaVersionNumber.

  Numéro de version du schéma.

Réponse

- `SchemaVersionId` – Chaîne UTF-8, d'une longueur exacte de 36 octets, correspondant au Custom string pattern #45.

  Le `SchemaVersionId` de la version du schéma.
- `SchemaDefinition` – Chaîne UTF-8, d'une longueur comprise entre 1 et 170 000 octets, correspondant au Custom string pattern #13.

  La définition du schéma pour l'ID du schéma.
- `DataFormat` – Chaîne UTF-8 (valeurs valides : `AVRO` | `JSON` | `PROTOBUF`).

  Format de données de la définition de schéma. À l'heure actuelle, `AVRO`, `JSON` et `PROTOBUF` sont pris en charge.

- `SchemaArn` – Chaîne UTF-8, d'une longueur comprise entre 1 et 10 240 octets, correspondant au Custom string pattern #50.

  L'Amazon Resource Name (ARN) du filtre.

- `VersionNumber` – Nombre (long), compris entre 1 et 100 000.

  Numéro de version du schéma.

- `Status` – Chaîne UTF-8 (valeurs valides : `AVAILABLE` | `PENDING` | `FAILURE` | `DELETING`).

  Statut de la version du schéma.

- `CreatedTime` – Chaîne UTF-8.

  Date et heure de création à laquelle la version de schéma a été créée.

Erreurs

- `InvalidInputException`
- `AccessDeniedException`
- `EntityNotFoundException`
- `ThrottlingException`
- `InternalServiceException`

GetSchemaVersionsDiff action (Python : get_schema_versions_diff)

Récupère la différence de version de schéma dans le type de différence spécifié entre deux versions de schéma stockées dans le registre des schémas.

Cette API vous permet de comparer deux versions de schéma entre deux définitions de schéma sous le même schéma.

Demande

- `SchemaId` – Obligatoire : un objet SchemaId.

  Structure d'encapsulation contenant les champs d'identité de schéma. La structure contient :

  - SchemaId$ SchemaArn : Le nom de ressource Amazon (ARN) du schéma. `SchemaArn` ou `SchemaName` doit être fourni.

  - SchemaId$ SchemaName : nom du schéma. `SchemaArn` ou `SchemaName` doit être fourni.
- `FirstSchemaVersionNumber` – Obligatoire : un objet SchemaVersionNumber.

  La première des deux versions de schéma à comparer.
- `SecondSchemaVersionNumber` – Obligatoire : un objet SchemaVersionNumber.

  La seconde des deux versions de schéma à comparer.
- `SchemaDiffType` – Obligatoire : Chaîne UTF-8 (valeurs valides : `SYNTAX_DIFF`).

  Fait référence à `SYNTAX_DIFF`, qui est le type diff actuellement pris en charge.

Réponse

- `Diff` – Chaîne UTF-8, d'une longueur comprise entre 1 et 340 000 octets, correspondant au Custom string pattern #13.

  La différence entre les schémas sous forme de chaîne de JsonPatch caractères.

Erreurs

- `InvalidInputException`
- `EntityNotFoundException`
- `AccessDeniedException`
- `ThrottlingException`
- `InternalServiceException`

ListRegistries action (Python : list_registries)

Renvoie une liste des registres que vous avez créés, avec des informations de registre minimales. Les registres dont le statut est `Deleting` ne seront pas inclus dans les résultats. Des résultats vides seront retournés s'il n'y a pas de registres disponibles.

Demande

- `MaxResults` – Nombre (entier), compris entre 1 et 100.

Nombre maximum de résultats requis par page. Si la valeur n'est pas fournie, elle par défaut sera de 25 par page.

- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, s'il s'agit d'un appel de continuation.

Réponse

- `Registries` : un tableau d'objets RegistryListItem.

  Tableau d'objets `RegistryDetailedListItem` contenant des détails minimaux de chaque registre.

- `NextToken` – Chaîne UTF-8.

  Jeton de continuation pour la pagination de la liste des jetons renvoyés, renvoyé si le segment actuel de la liste n'est pas le dernier.

Erreurs

- `InvalidInputException`
- `AccessDeniedException`
- `ThrottlingException`
- `InternalServiceException`

ListSchemas action (Python : list_schemas)

Renvoie une liste de schémas avec des détails minimaux. Les schémas dont le statut est Deleted ne seront pas inclus dans les résultats. Des résultats vides seront retournés s'il n'y a pas de schéma disponible.

Lorsque `RegistryId` n'est pas fourni, tous les schémas entre les registres feront partie de la réponse de l'API.

Demande

- `RegistryId` – Un objet RegistryId.

Structure d'encapsulation qui peut contenir le nom et l'Amazon Resource Name (ARN) du registre.

- `MaxResults` – Nombre (entier), compris entre 1 et 100.

  Nombre maximum de résultats requis par page. Si la valeur n'est pas fournie, elle par défaut sera de 25 par page.

- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, s'il s'agit d'un appel de continuation.

Réponse

- `Schemas` : un tableau d'objets SchemaListItem.

  Tableau d'objets `SchemaListItem` contenant les détails de chaque schéma.

- `NextToken` – Chaîne UTF-8.

  Jeton de continuation pour la pagination de la liste des jetons renvoyés, renvoyé si le segment actuel de la liste n'est pas le dernier.

Erreurs

- `InvalidInputException`
- `AccessDeniedException`
- `EntityNotFoundException`
- `ThrottlingException`
- `InternalServiceException`

RegisterSchemaVersion action (Python : register_schema_version)

Ajoute une nouvelle version au schéma existant. Renvoie une erreur si la nouvelle version du schéma ne répond pas aux exigences de compatibilité du jeu de schémas. Cette API ne crée pas de nouveau jeu de schémas et renvoie une erreur 404 si le jeu de schémas n'est pas déjà présent dans le registre de schémas.

S'il s'agit de la première définition de schéma enregistrée dans le registre de schéma, cette API stockera la version du schéma et retournera immédiatement à l'appelant. Sinon, cet appel peut

s'exécuter plus longtemps que d'autres opérations en raison des modes de compatibilité. Vous pouvez appeler l'API `GetSchemaVersion` avec le `SchemaVersionId` pour vérifier les modes de compatibilité.

Si la même définition de schéma est déjà stockée dans le registre de schéma en tant que version, l'ID de schéma du schéma existant est renvoyé à l'appelant.

Demande

- `SchemaId` – Obligatoire : un objet SchemaId.

  Structure d'encapsulation contenant les champs d'identité de schéma. La structure contient :

  - SchemaId$ SchemaArn : Le nom de ressource Amazon (ARN) du schéma. Il faut fournir soit `SchemaArn`, soit `SchemaName` et `RegistryName`.
  - SchemaId$ SchemaName : nom du schéma. Il faut fournir soit `SchemaArn`, soit `SchemaName` et `RegistryName`.
- `SchemaDefinition` – Obligatoire : chaîne UTF-8, d'une longueur comprise entre 1 et 170 000 octets, correspondant au Custom string pattern #13.

  Définition de schéma à l'aide du paramètre `DataFormat` pour `SchemaName`.

Réponse

- `SchemaVersionId` – Chaîne UTF-8, d'une longueur exacte de 36 octets, correspondant au Custom string pattern #45.

  ID unique qui représente la version de ce schéma.
- `VersionNumber` – Nombre (long), compris entre 1 et 100 000.

  Version de ce schéma (pour le flux de synchronisation uniquement, dans le cas où il s'agit de la première version).
- `Status` – Chaîne UTF-8 (valeurs valides : `AVAILABLE` | `PENDING` | `FAILURE` | `DELETING`).

  Statut de la version du schéma.

Erreurs

- `InvalidInputException`

- `AccessDeniedException`
- `EntityNotFoundException`
- `ResourceNumberLimitExceededException`
- `ThrottlingException`
- `ConcurrentModificationException`
- `InternalServiceException`

UpdateSchema action (Python : update_schema)

Met à jour la description, le paramètre de compatibilité ou le point de contrôle de version d'un jeu de schéma.

Pour mettre à jour le paramètre de compatibilité, l'appel ne valide pas la compatibilité pour l'ensemble des versions de schéma avec le nouveau paramètre de compatibilité. Si la valeur pour `Compatibility` est fournie, le `VersionNumber` (un point de contrôle) est également requis. L'API valide le numéro de version du point de contrôle à des fins de cohérence.

Si la valeur du `VersionNumber` (point de contrôle) est fournie, elle `Compatibility` est facultative et peut être utilisée comme point set/reset de contrôle pour le schéma.

Cette mise à jour se produira uniquement si le schéma est dans l'état AVAILABLE (DISPONIBLE).

Demande

- `SchemaId` – Obligatoire : un objet SchemaId.

  Structure d'encapsulation contenant les champs d'identité de schéma. La structure contient :

  - SchemaId$ SchemaArn : Le nom de ressource Amazon (ARN) du schéma. `SchemaArn` ou `SchemaName` doit être fourni.
  - SchemaId$ SchemaName : nom du schéma. `SchemaArn` ou `SchemaName` doit être fourni.
- `SchemaVersionNumber` – Un objet SchemaVersionNumber.

  Numéro de version requis pour le pointage de contrôle. `VersionNumber` ou `Compatibility` doit être fourni.
- `Compatibility` – Chaîne UTF-8 (valeurs valides : `NONE` | `DISABLED` | `BACKWARD` | `BACKWARD_ALL` | `FORWARD` | `FORWARD_ALL` | `FULL` | `FULL_ALL`).

  Nouveau paramètre de compatibilité pour le schéma.

- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Nouvelle description du schéma.

Réponse

- `SchemaArn` – Chaîne UTF-8, d'une longueur comprise entre 1 et 10 240 octets, correspondant au Custom string pattern #50.

  L'Amazon Resource Name (ARN) du filtre.

- `SchemaName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #46.

  Nom du schéma.

- `RegistryName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #46.

  Nom du registre contenant le schéma.

Erreurs

- `InvalidInputException`
- `AccessDeniedException`
- `EntityNotFoundException`
- `ConcurrentModificationException`
- `ThrottlingException`
- `InternalServiceException`

CheckSchemaVersionValidity action (Python : check_schema_version_validity)

Valide le schéma fourni. Cet appel n'a pas d'effets secondaires, il effectue simplement une validation en utilisant le schéma fourni à l'aide de `DataFormat`. Comme il ne prend pas de nom de jeu de schéma, aucune vérification de compatibilité n'est effectuée.

Demande

- `DataFormat` – Obligatoire : Chaîne UTF-8 (valeurs valides : `AVRO` | `JSON` | `PROTOBUF`).

  Format de données de la définition de schéma. À l'heure actuelle, `AVRO`, `JSON` et `PROTOBUF` sont pris en charge.
- `SchemaDefinition` – Obligatoire : chaîne UTF-8, d'une longueur comprise entre 1 et 170 000 octets, correspondant au Custom string pattern #13.

  Définition du schéma qui doit être validé.

Réponse

- `Valid` : booléen.

  Renvoie true, si le schéma est valide et false sinon.
- `Error` – Chaîne UTF-8, d'une longueur comprise entre 1 et 5 000 octets.

  Message d'erreur d'échec de validation.

Erreurs

- `InvalidInputException`
- `AccessDeniedException`
- `ThrottlingException`
- `InternalServiceException`

UpdateRegistry action (Python : update_registry)

Met à jour un registre existant qui est utilisé pour contenir une collection de schémas. Les propriétés mises à jour se rapportent au registre et ne modifient aucun des schémas de celui-ci.

Demande

- `RegistryId` – Obligatoire : un objet RegistryId.

  Structure d'encapsulation qui peut contenir le nom et l'Amazon Resource Name (ARN) du registre.

- `Description` – Obligatoire : Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description du registre. Si la description n'est pas fournie, ce champ ne sera pas mis à jour.

Réponse

- `RegistryName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #46.

  Nom du registre mis à jour.
- `RegistryArn` – Chaîne UTF-8, d'une longueur comprise entre 1 et 10 240 octets, correspondant au Custom string pattern #50.

  Amazon Resource Name (ARN) du registre mis à jour.

Erreurs

- `InvalidInputException`
- `AccessDeniedException`
- `EntityNotFoundException`
- `ConcurrentModificationException`
- `ThrottlingException`
- `InternalServiceException`

GetSchemaByDefinition action (Python : get_schema_by_definition)

Récupère un schéma à l'aide de sa `SchemaDefinition`. La définition du schéma est envoyée au registre de schéma, rendue canonique et hachée. Si le hachage correspond à la portée du `SchemaName` ou de l'ARN (ou du registre par défaut, si aucun n'est fourni), les métadonnées de ce schéma sont retournées. Dans le cas contraire, un 404 ou une NotFound erreur est renvoyée. Les versions de schémas dont le statut est `Deleted` ne seront pas incluses dans les résultats.

Demande

- `SchemaId` – Obligatoire : un objet SchemaId.

Structure d'encapsulation contenant les champs d'identité de schéma. La structure contient :

- SchemaId$ SchemaArn : Le nom de ressource Amazon (ARN) du schéma. `SchemaArn` ou `SchemaName` doit être fourni.
- SchemaId$ SchemaName : nom du schéma. `SchemaArn` ou `SchemaName` doit être fourni.

- `SchemaDefinition` – Obligatoire : chaîne UTF-8, d'une longueur comprise entre 1 et 170 000 octets, correspondant au Custom string pattern #13.

  Définition du schéma pour lequel les détails du schéma sont requis.

Réponse

- `SchemaVersionId` – Chaîne UTF-8, d'une longueur exacte de 36 octets, correspondant au Custom string pattern #45.

  ID de schéma de la version du schéma.
- `SchemaArn` – Chaîne UTF-8, d'une longueur comprise entre 1 et 10 240 octets, correspondant au Custom string pattern #50.

  L'Amazon Resource Name (ARN) du filtre.
- `DataFormat` – Chaîne UTF-8 (valeurs valides : `AVRO` | `JSON` | `PROTOBUF`).

  Format de données de la définition de schéma. À l'heure actuelle, `AVRO`, `JSON` et `PROTOBUF` sont pris en charge.
- `Status` – Chaîne UTF-8 (valeurs valides : `AVAILABLE` | `PENDING` | `FAILURE` | `DELETING`).

  Statut de la version du schéma.
- `CreatedTime` – Chaîne UTF-8.

  Date et heure de création du schéma.

Erreurs

- `InvalidInputException`
- `AccessDeniedException`
- `EntityNotFoundException`
- `ThrottlingException`

- `InternalServiceException`

GetRegistry action (Python : get_registry)

Décrit en détail le registre spécifié.

Demande

- `RegistryId` – Obligatoire : un objet RegistryId.

  Structure d'encapsulation qui peut contenir le nom et l'Amazon Resource Name (ARN) du registre.

Réponse

- `RegistryName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #46.

  Nom du registre.
- `RegistryArn` – Chaîne UTF-8, d'une longueur comprise entre 1 et 10 240 octets, correspondant au Custom string pattern #50.

  L'Amazon Resource Name (ARN) du registre.
- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description du registre.
- `Status` – Chaîne UTF-8 (valeurs valides : `AVAILABLE` | `DELETING`).

  Statut du registre.
- `CreatedTime` – Chaîne UTF-8.

  Date et heure de création du registre.
- `UpdatedTime` – Chaîne UTF-8.

  Date et heure de mise à jour du registre.

Erreurs

- `InvalidInputException`
- `AccessDeniedException`
- `EntityNotFoundException`
- `ThrottlingException`
- `InternalServiceException`

PutSchemaVersionMetadata action (Python : put_schema_version_metadata)

Place la paire de valeurs clé de métadonnées pour un ID de version de schéma spécifié. Un maximum de 10 paires de valeurs clé sera autorisé par version de schéma. Ils peuvent être ajoutés au moyen d'un ou de plusieurs appels.

Demande

- `SchemaId` – Un objet SchemaId.

  ID unique du schéma.
- `SchemaVersionNumber` – Un objet SchemaVersionNumber.

  Numéro de version du schéma.
- `SchemaVersionId` – Chaîne UTF-8, d'une longueur exacte de 36 octets, correspondant au Custom string pattern #45.

  ID de version unique de la version du schéma.
- `MetadataKeyValue` – Obligatoire : un objet MetadataKeyValuePair.

  Valeur correspondant à une clé de métadonnées.

Réponse

- `SchemaArn` – Chaîne UTF-8, d'une longueur comprise entre 1 et 10 240 octets, correspondant au Custom string pattern #50.

  Amazon Resource Name (ARN) du schéma.

- `SchemaName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #46.

  Nom du schéma.

- `RegistryName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #46.

  Nom du registre.

- `LatestVersion` : booléen.

  Dernière version du schéma.

- `VersionNumber` – Nombre (long), compris entre 1 et 100 000.

  Numéro de version du schéma.

- `SchemaVersionId` – Chaîne UTF-8, d'une longueur exacte de 36 octets, correspondant au Custom string pattern #45.

  ID de version unique de la version du schéma.

- `MetadataKey` – Chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets, correspondant au Custom string pattern #14.

  Clé de métadonnées.

- `MetadataValue` – Chaîne UTF-8, d'une longueur comprise entre 1 et 256 octets, correspondant au Custom string pattern #14.

  Valeur de la clé de métadonnées.

Erreurs

- `InvalidInputException`
- `AccessDeniedException`
- `AlreadyExistsException`
- `EntityNotFoundException`
- `ThrottlingException`
- `ResourceNumberLimitExceededException`

QuerySchemaVersionMetadata action (Python : query_schema_version_metadata)

Requêtes pour les informations de métadonnées de version du schéma.

Demande

- `SchemaId` – Un objet SchemaId.

  Structure d'encapsulation qui peut contenir le nom et l'Amazon Resource Name (ARN) du schéma.
- `SchemaVersionNumber` – Un objet SchemaVersionNumber.

  Numéro de version du schéma.
- `SchemaVersionId` – Chaîne UTF-8, d'une longueur exacte de 36 octets, correspondant au Custom string pattern #45.

  ID de version unique de la version du schéma.
- `MetadataList` : un tableau d'objets MetadataKeyValuePair.

  Rechercher des paires clé-valeur pour les métadonnées, si elles ne sont pas fournies, toutes les informations de métadonnées seront récupérées.
- `MaxResults` – Nombre (entier), compris entre 1 et 50.

  Nombre maximum de résultats requis par page. Si la valeur n'est pas fournie, elle par défaut sera de 25 par page.
- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, s'il s'agit d'un appel de continuation.

Réponse

- `MetadataInfoMap` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets, correspondant au Custom string pattern #14.

  Chaque valeur est un objet MetadataInfo.

  Carte d'une clé de métadonnées et des valeurs associées.

- `SchemaVersionId` – Chaîne UTF-8, d'une longueur exacte de 36 octets, correspondant au Custom string pattern #45.

  ID de version unique de la version du schéma.
- `NextToken` – Chaîne UTF-8.

  Jeton de continuation pour la pagination de la liste des jetons renvoyés, renvoyé si le segment actuel de la liste n'est pas le dernier.

Erreurs

- `InvalidInputException`
- `AccessDeniedException`
- `ThrottlingException`
- `EntityNotFoundException`

RemoveSchemaVersionMetadata action (Python : remove_schema_version_metadata)

Supprime une paire de valeurs clé des métadonnées de version de schéma pour l'ID de version de schéma spécifié.

Demande

- `SchemaId` – Un objet SchemaId.

  Structure d'encapsulation qui peut contenir le nom et l'Amazon Resource Name (ARN) du schéma.
- `SchemaVersionNumber` – Un objet SchemaVersionNumber.

  Numéro de version du schéma.
- `SchemaVersionId` – Chaîne UTF-8, d'une longueur exacte de 36 octets, correspondant au Custom string pattern #45.

  ID de version unique de la version du schéma.
- `MetadataKeyValue` – Obligatoire : un objet MetadataKeyValuePair.

  Valeur de la clé de métadonnées.

Réponse

- `SchemaArn` – Chaîne UTF-8, d'une longueur comprise entre 1 et 10 240 octets, correspondant au Custom string pattern #50.

  L'Amazon Resource Name (ARN) du filtre.
- `SchemaName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #46.

  Nom du schéma.
- `RegistryName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #46.

  Nom du registre.
- `LatestVersion` : booléen.

  Dernière version du schéma.
- `VersionNumber` – Nombre (long), compris entre 1 et 100 000.

  Numéro de version du schéma.
- `SchemaVersionId` – Chaîne UTF-8, d'une longueur exacte de 36 octets, correspondant au Custom string pattern #45.

  ID de version pour la version du schéma.
- `MetadataKey` – Chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets, correspondant au Custom string pattern #14.

  Clé de métadonnées.
- `MetadataValue` – Chaîne UTF-8, d'une longueur comprise entre 1 et 256 octets, correspondant au Custom string pattern #14.

  Valeur de la clé de métadonnées.

Erreurs

- `InvalidInputException`
- `AccessDeniedException`
- `ThrottlingException`

- `EntityNotFoundException`

DeleteRegistry action (Python : supprimer_registre)

Supprimer l'intégralité du registre, y compris le schéma et toutes ses versions. Pour obtenir le statut de l'opération de suppression, vous pouvez appeler l'API `GetRegistry` après l'appel asynchrone. La suppression d'un registre désactivera toutes les opérations en ligne du registre, telles que le`UpdateRegistry`, `CreateSchema``UpdateSchema`, et `RegisterSchemaVersion` APIs.

Demande

- `RegistryId` – Obligatoire : un objet RegistryId.

  Structure d'encapsulation qui peut contenir le nom et l'Amazon Resource Name (ARN) du registre.

Réponse

- `RegistryName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #46.

  Nom du registre en cours de suppression.
- `RegistryArn` – Chaîne UTF-8, d'une longueur comprise entre 1 et 10 240 octets, correspondant au Custom string pattern #50.

  Amazon Resource Name (ARN) du registre en cours de suppression.
- `Status` – Chaîne UTF-8 (valeurs valides : `AVAILABLE` | `DELETING`).

  Statut du registre. La réussite de l'opération retournera le statut `Deleting`.

Erreurs

- `InvalidInputException`
- `EntityNotFoundException`
- `AccessDeniedException`
- `ThrottlingException`
- `ConcurrentModificationException`

DeleteSchema action (Python : supprimer_schéma)

Supprime l'ensemble de schémas, y compris le jeu de schémas et toutes ses versions. Pour obtenir le statut de l'opération de suppression, vous pouvez appeler l'API `GetSchema` après l'appel asynchrone. La suppression d'un registre désactivera toutes les opérations en ligne pour le schéma, telles que le`GetSchemaByDefinition`, et `RegisterSchemaVersion` APIs.

Demande

- `SchemaId` – Obligatoire : un objet SchemaId.

  Structure d'encapsulation qui peut contenir le nom et l'Amazon Resource Name (ARN) du schéma.

Réponse

- `SchemaArn` – Chaîne UTF-8, d'une longueur comprise entre 1 et 10 240 octets, correspondant au Custom string pattern #50.

  Amazon Resource Name (ARN) du schéma en cours de suppression.
- `SchemaName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #46.

  Nom du schéma en cours de suppression.
- `Status` – Chaîne UTF-8 (valeurs valides : `AVAILABLE` | `PENDING` | `DELETING`).

  Statut du schéma.

Erreurs

- `InvalidInputException`
- `EntityNotFoundException`
- `AccessDeniedException`
- `ThrottlingException`
- `ConcurrentModificationException`

DeleteSchemaVersions action (Python : delete_schema_versions)

Supprimez des versions du schéma spécifié. Un numéro ou une plage de versions peut être fourni. Si le mode de compatibilité interdit la suppression d'une version nécessaire, telle que BACKWARDS_FULL (DESCENDANT_TOUT), une erreur est renvoyée. En appelant l'API `GetSchemaVersions` après cet appel, vous obtiendrez la liste de l'état des versions supprimées.

Lorsque la plage de numéros de version contient une version comportant un point de contrôle, l'API renvoie un conflit 409 et ne procède pas à la suppression. Vous devez d'abord supprimer le point de contrôle à l'aide de l'API `DeleteSchemaCheckpoint` avant d'utiliser cette API.

Vous ne pouvez pas utiliser l'API `DeleteSchemaVersions` pour supprimer la première version de schéma dans le jeu de schémas. La première version de schéma ne peut être supprimée que par l'appel à l'API `DeleteSchema`. Cette opération supprimera également les `SchemaVersionMetadata` attachées aux versions du schéma. Les suppressions définitives seront appliquées sur la base de données.

Si le mode de compatibilité interdit la suppression d'une version nécessaire, telle que BACKWARDS_FULL (DESCENDANT_TOUT), une erreur est renvoyée.

Demande

- `SchemaId` – Obligatoire : un objet SchemaId.

  Structure d'encapsulation qui peut contenir le nom et l'Amazon Resource Name (ARN) du schéma.
- `Versions` – Obligatoire : chaîne UTF-8, d'une longueur comprise entre 1 et 100 000 octets, correspondant au Custom string pattern #15.

  Une plage de versions peut être fournie au format suivant :

  - un numéro de version unique, 5
  - une plage, 5-8 : supprime les versions 5, 6, 7, 8

Réponse

- `SchemaVersionErrors` : un tableau d'objets SchemaVersionErrorItem.

  Une liste d'objets `SchemaVersionErrorItem`, chacun contenant une version d'erreur et de schéma.

Erreurs

- `InvalidInputException`
- `EntityNotFoundException`
- `AccessDeniedException`
- `ThrottlingException`
- `ConcurrentModificationException`

Flux de travail

L'API Workflows décrit les types de données et l'API liés à la création, à la mise à jour ou à l'affichage des flux de travail dans AWS Glue. L'historique d'exécution des tâches est accessible pendant 90 jours pour votre flux de travail et l'exécution des tâches.

Types de données

- JobNodeDetails structure
- CrawlerNodeDetails structure
- TriggerNodeDetails structure
- Structure Crawl
- Structure de nœud
- Structure Edge
- Structure de flux de travail
- WorkflowGraph structure
- WorkflowRun structure
- WorkflowRunStatistics structure
- StartingEventBatchCondition structure
- Structure du plan
- BlueprintDetails structure
- LastActiveDefinition structure
- BlueprintRun structure

JobNodeDetails structure

Détails d'un nœud de tâches présent dans le flux de travail.

Champs

- `JobRuns` : un tableau d'objets JobRun.

  Informations relatives aux exécutions de tâches représentées par le nœud de tâches.

CrawlerNodeDetails structure

Détails d'un nœud d'crawler présent dans le flux de travail.

Champs

- `Crawls` : un tableau d'objets Crawl.

  Liste de crawlers représentés par le nœud de crawler.

TriggerNodeDetails structure

Détails d'un nœud Trigger présent dans le flux de travail.

Champs

- `Trigger` – Un objet Déclencheur.

  Informations relatives au déclencheur représenté par le nœud de déclencheur.

Structure Crawl

Détails d'un crawler dans le flux de travail.

Champs

- `State` – Chaîne UTF-8 (valeurs valides : `RUNNING` | `CANCELLING` | `CANCELLED` | `SUCCEEDED` | `FAILED` | `ERROR`).

  État de l'crawler.

- `StartedOn` – Horodatage.

  Date et heure auxquelles le crawler a démarré.

- `CompletedOn` – Horodatage.

  Date et heure auxquelles l'analyse a terminé.

- `ErrorMessage` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Message d'erreur associé au crawler

- `LogGroup` – Chaîne UTF-8, d'une longueur comprise entre 1 et 512 octets, correspondant au Log group string pattern.

  Groupe de journaux associés au crawler.

- `LogStream` – Chaîne UTF-8, d'une longueur comprise entre 1 et 512 octets, correspondant au Log-stream string pattern.

  Flux de journaux associé au crawler.

Structure de nœud

Un nœud représente un AWS Glue composant (déclencheur, robot ou tâche) sur un graphe de flux de travail.

Champs

- `Type` – Chaîne UTF-8 (valeurs valides : `CRAWLER` | `JOB` | `TRIGGER`).

  Type de AWS Glue composant représenté par le nœud.

- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du AWS Glue composant représenté par le nœud.

- `UniqueId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID unique affecté au nœud au sein du flux de travail.

- `TriggerDetails` – Un objet TriggerNodeDetails.

Détails du déclencheur lorsque le nœud représente un déclencheur.

- `JobDetails` – Un objet JobNodeDetails.

  Détails de la tâche lorsque le nœud représente une tâche.

- `CrawlerDetails` – Un objet CrawlerNodeDetails.

  Détails de l'crawler lorsque le nœud représente un crawler.

Structure Edge

Un arête représente une connexion dirigée entre deux AWS Glue composants qui font partie du flux de travail auquel appartient l'arête.

Champs

- `SourceId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaîne unique du nœud au sein du flux de travail où démarre la périphérie.

- `DestinationId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaîne unique du nœud au sein du flux de travail où la périphérie se termine.

Structure de flux de travail

Un flux de travail est un ensemble de plusieurs AWS Glue tâches dépendantes et de robots d'exploration exécutés pour effectuer une tâche ETL complexe. Un flux de travail gère l'exécution et le suivi de tous ses jobs et crawlers.

Champs

- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du flux de travail.

- `Description` – Chaîne UTF-8.

Description du flux de travail.

- `DefaultRunProperties` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaque valeur est une chaîne UTF-8.

  Collection de propriétés à utiliser dans le cadre de chaque exécution du flux de travail. Les propriétés d'exécution sont mises à la disposition de chaque tâche du flux de travail. Une tâche peut modifier les propriétés des tâches suivants dans le flux de travail.

- `CreatedOn` – Horodatage.

  Date et heure auxquelles le flux de travail a été créé.

- `LastModifiedOn` – Horodatage.

  Date et heure auxquelles le flux de travail a été modifié pour la dernière fois.

- `LastRun` – Un objet WorkflowRun.

  Informations relatives à la dernière exécution du flux de travail.

- `Graph` – Un objet WorkflowGraph.

  Le graphique représente tous les AWS Glue composants appartenant au flux de travail sous forme de nœuds et les connexions dirigées entre eux sous forme d'arêtes.

- `CreationStatus` – Chaîne UTF-8 (valeurs valides : `CREATING` | `CREATED` | `CREATION_FAILED`).

  État de création du flux de travail.

- `MaxConcurrentRuns` : nombre (entier).

  Vous pouvez utiliser ce paramètre pour empêcher plusieurs mises à jour indésirables des données, pour contrôler les coûts ou, dans certains cas, pour empêcher le dépassement du nombre maximal d'exécutions simultanées de l'un des travaux de composant. Si vous laissez ce paramètre vide, le nombre d'exécutions de flux de travail simultanées est illimité.

- `BlueprintDetails` – Un objet BlueprintDetails.

  Cette structure indique les détails du plan à partir duquel ce flux de travail particulier est créé.

WorkflowGraph structure

Un graphique de flux de travail représente le flux de travail complet contenant tous les composants AWS Glue présents dans le flux de travail et toutes les connexions dirigées entre eux.

Champs

- `Nodes` : un tableau d'objets Nœud.

  Une liste des AWS Glue composants appartenant au flux de travail représentés sous forme de nœuds.

- `Edges` : un tableau d'objets Edge.

  Liste de toutes les connexions dirigées entre les nœuds appartenant au flux de travail.

WorkflowRun structure

Une exécution de flux de travail est une exécution de flux de travail qui fournit toutes les informations d'exécution.

Champs

- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du flux de travail qui a été exécuté.

- `WorkflowRunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID de cette exécution de flux de travail.

- `PreviousRunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID de la précédente exécution de flux de travail.

- `WorkflowRunProperties` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

Chaque valeur est une chaîne UTF-8.

Propriétés d'exécution de flux de travail qui ont été définies au cours de l'exécution.

- `StartedOn` – Horodatage.

  Date et heure auxquelles l'exécution de flux de travail a démarré.

- `CompletedOn` – Horodatage.

  Date et heure auxquelles l'exécution de flux de travail a s'est terminée.

- `Status` – Chaîne UTF-8 (valeurs valides : `RUNNING` | `COMPLETED` | `STOPPING` | `STOPPED` | `ERROR`).

  État de l'exécution de flux de travail.

- `ErrorMessage` – Chaîne UTF-8.

  Ce message d'erreur décrit toute erreur qui peut s'être produite lors du démarrage de l'exécution du flux de travail. Actuellement, le seul message d'erreur est « Concurrent runs exceeded for workflow: `foo` » (Exécutions simultanées dépassées pour le flux de travail : foo).

- `Statistics` – Un objet WorkflowRunStatistics.

  Statistiques de l'exécution.

- `Graph` – Un objet WorkflowGraph.

  Le graphique représente tous les AWS Glue composants appartenant au flux de travail sous forme de nœuds et les connexions dirigées entre eux sous forme d'arêtes.

- `StartingEventBatchCondition` – Un objet StartingEventBatchCondition.

  Condition de lot qui a démarré l'exécution du flux de travail.

WorkflowRunStatistics structure

Le flux de travail exécut des statistiques et fournit des statistiques sur l'exécution du flux de travail.

Champs

- `TotalActions` : nombre (entier).

  Nombre total d'actions dans l'exécution du flux de travail.

- `TimeoutActions` : nombre (entier).

  Nombre total d'actions qui ont expiré.

- `FailedActions` : nombre (entier).

  Nombre total d'actions qui ont échoué.

- `StoppedActions` : nombre (entier).

  Nombre total d'actions qui sont arrêtées.

- `SucceededActions` : nombre (entier).

  Nombre total d'actions qui ont réussi.

- `RunningActions` : nombre (entier).

  Nombre total d'actions en cours d'exécution.

- `ErroredActions` : nombre (entier).

  Indique le nombre d'exécutions de tâches dans l'état ERROR dans l'exécution du flux de travail.

- `WaitingActions` : nombre (entier).

  Indique le nombre d'exécutions de tâches en état WAITING dans l'exécution du flux de travail.

StartingEventBatchCondition structure

Condition de lot qui a démarré l'exécution du flux de travail. Soit le nombre d'événements de la taille du lot est arrivé, auquel cas le BatchSize membre est différent de zéro, soit la fenêtre du lot a expiré, auquel cas le BatchWindow membre est différent de zéro.

Champs

- `BatchSize` : nombre (entier).

  Nombre d'événements dans le lot.

- `BatchWindow` : nombre (entier).

  Durée de la fenêtre de traitement par lots en secondes.

Structure du plan

Détails d'un plan.

Champs

- `Name` – chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets, correspondant au Custom string pattern #31.

  Nom du plan.

- `Description` – chaîne UTF-8, d'une longueur comprise entre 1 et 512 octets.

  Description du plan.

- `CreatedOn` – Horodatage.

  Date et heure d'enregistrement du plan.

- `LastModifiedOn` – Horodatage.

  Date et heure auxquelles le plan a été modifié pour la dernière fois.

- `ParameterSpec` – chaîne UTF-8, d'une longueur comprise entre 1 et 131 072 octets.

  Chaîne JSON qui indique la liste des spécifications de paramètres pour le plan.

- `BlueprintLocation` – Chaîne UTF-8.

  Spécifie le chemin dans Amazon S3 où le plan est publié.

- `BlueprintServiceLocation` – Chaîne UTF-8.

  Spécifie un chemin dans Amazon S3 où le plan est copié lorsque vous appelez `CreateBlueprint/UpdateBlueprint` pour enregistrer le plan dans AWS Glue.

- `Status` – Chaîne UTF-8 (valeurs valides : `CREATING` | `ACTIVE` | `UPDATING` | `FAILED`).

  Statut de l'enregistrement du plan.

  - Création : l'enregistrement du plan est en cours.
  - Actif : le plan a été enregistré avec succès.
  - Mise à jour : une mise à jour de l'enregistrement du plan est en cours.
  - Échec : l'enregistrement du plan a échoué.

- `ErrorMessage` – Chaîne UTF-8.

Message d'erreur.

- `LastActiveDefinition` – Un objet LastActiveDefinition.

  Lorsqu'il existe plusieurs versions d'un plan et que la dernière version contient des erreurs, cet attribut indique la dernière définition de plan réussie qui est disponible avec le service.

BlueprintDetails structure

Détails d'un plan.

Champs

- `BlueprintName` – chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets, correspondant au Custom string pattern #31.

  Nom du plan.
- `RunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID d'exécution de ce plan.

LastActiveDefinition structure

Lorsqu'il existe plusieurs versions d'un plan et que la dernière version contient des erreurs, cet attribut indique la dernière définition de plan réussie qui est disponible avec le service.

Champs

- `Description` – chaîne UTF-8, d'une longueur comprise entre 1 et 512 octets.

  Description du plan.
- `LastModifiedOn` – Horodatage.

  Date et heure auxquelles le plan a été modifié pour la dernière fois.
- `ParameterSpec` – chaîne UTF-8, d'une longueur comprise entre 1 et 131 072 octets.

  Chaîne JSON spécifiant les paramètres du plan.
- `BlueprintLocation` – Chaîne UTF-8.

Spécifie un chemin dans Amazon S3 où le plan est publié par le AWS Glue développeur.

- `BlueprintServiceLocation` – Chaîne UTF-8.

  Spécifie un chemin dans Amazon S3 où le plan est copié lorsque vous créez ou mettez à jour ce dernier.

BlueprintRun structure

Détails d'une exécution de plan.

Champs

- `BlueprintName` – chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets, correspondant au Custom string pattern #31.

  Nom du plan.

- `RunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID d'exécution de cette exécution de plan.

- `WorkflowName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom d'un flux de travail créé à la suite d'une exécution réussie du plan. Si une exécution de plan comporte une erreur, aucun flux de travail ne sera créé.

- `State` – Chaîne UTF-8 (valeurs valides : `RUNNING` | `SUCCEEDED` | `FAILED` | `ROLLING_BACK`).

  État de l'exécution du plan. Les valeurs possibles sont :

  - En cours d'exécution : l'exécution du plan est en cours.
  - Réussi : l'exécution du plan s'est terminée.
  - Échec : l'exécution du plan a échoué et la restauration est terminée.
  - Annuler : l'exécution du plan a échoué et la restauration est en cours.

- `StartedOn` – Horodatage.

  Date et heure auxquelles l'exécution du plan a démarré.

- `CompletedOn` – Horodatage.

Date et heure auxquelles le modèle a été exécuté.

- `ErrorMessage` – Chaîne UTF-8.

  Indique les erreurs détectées lors de l'exécution du plan.

- `RollbackErrorMessage` – Chaîne UTF-8.

  S'il y a des erreurs lors de la création des entités d'un flux de travail, nous essayons de restaurer les entités créées jusqu'à ce point et de les supprimer. Cet attribut indique les erreurs rencontrées lors de la tentative de suppression des entités créées.

- `Parameters` – chaîne UTF-8, d'une longueur comprise entre 1 et 131 072 octets.

  Paramètres du plan sous la forme d'une chaîne. Vous devrez fournir une valeur pour chaque clé requise à partir de la spécification de paramètre définie dans `Blueprint$ParameterSpec`.

- `RoleArn` – Chaîne UTF-8, d'une longueur comprise entre 1 et 1024 octets, correspondant au Custom string pattern #30.

  ARN du rôle. Ce rôle sera assumé par le AWS Glue service et sera utilisé pour créer le flux de travail et les autres entités d'un flux de travail.

Opérations

- CreateWorkflow action (Python : create_workflow)
- UpdateWorkflow action (Python : update_workflow)
- DeleteWorkflow action (Python : delete_workflow)
- GetWorkflow action (Python : get_workflow)
- ListWorkflows action (Python : list_workflows)
- BatchGetWorkflows action (Python : batch_get_workflows)
- GetWorkflowRun action (Python : get_workflow_run)
- GetWorkflowRuns action (Python : get_workflow_runs)
- GetWorkflowRunProperties action (Python : get_workflow_run_properties)
- PutWorkflowRunProperties action (Python : put_workflow_run_properties)
- CreateBlueprint action (Python : créer_blueprint)
- UpdateBlueprint action (Python : update_blueprint)

- DeleteBlueprint action (Python : delete_blueprint)
- ListBlueprints action (Python : list_blueprints)
- BatchGetBlueprints action (Python : batch_get_blueprints)
- StartBlueprintRun action (Python : start_blueprint_run)
- GetBlueprintRun action (Python : get_blueprint_run)
- GetBlueprintRuns action (Python : get_blueprint_runs)
- StartWorkflowRun action (Python : start_workflow_run)
- StopWorkflowRun action (Python : stop_workflow_run)
- ResumeWorkflowRun action (Python : resume_workflow_run)

CreateWorkflow action (Python : create_workflow)

Crée un nouveau flux de travail.

Demande

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom à affecter ai flux de travail. Il doit être unique au sein de votre compte.
- `Description` – Chaîne UTF-8, d'une longueur maximale de 120 000 octets.

  Description du flux de travail.
- `DefaultRunProperties` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaque valeur est une chaîne UTF-8.

  Collection de propriétés à utiliser dans le cadre de chaque exécution du flux de travail.

  Les propriétés d'exécution peuvent être journalisées. Ne transmettez pas de secrets en texte clair comme propriétés. Récupérez les secrets d'une AWS Glue connexion, d'un Gestionnaire de AWS Secrets ou d'un autre mécanisme de gestion des secrets si vous avez l'intention de les utiliser dans le cadre de l'exécution du flux de travail.
- `Tags` – Tableau de mappage de paires clé-valeur, avec 50 paires au maximum.

Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 256 octets.

Balises à utiliser avec ce flux de travail.

- `MaxConcurrentRuns` : nombre (entier).

  Vous pouvez utiliser ce paramètre pour empêcher plusieurs mises à jour indésirables des données, pour contrôler les coûts ou, dans certains cas, pour empêcher le dépassement du nombre maximal d'exécutions simultanées de l'un des travaux de composant. Si vous laissez ce paramètre vide, le nombre d'exécutions de flux de travail simultanées est illimité.

Réponse

- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du flux de travail qui a été fourni dans le cadre de la requête.

Erreurs

- `AlreadyExistsException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`
- `ConcurrentModificationException`

UpdateWorkflow action (Python : update_workflow)

Met à jour un flux de travail existant.

Demande

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

Nom du flux de travail à mettre à jour.

- `Description` – Chaîne UTF-8, d'une longueur maximale de 120 000 octets.

  Description du flux de travail.

- `DefaultRunProperties` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaque valeur est une chaîne UTF-8.

  Collection de propriétés à utiliser dans le cadre de chaque exécution du flux de travail.

  Les propriétés d'exécution peuvent être journalisées. Ne transmettez pas de secrets en texte clair comme propriétés. Récupérez les secrets d'une AWS Glue connexion, d'un Gestionnaire de AWS Secrets ou d'un autre mécanisme de gestion des secrets si vous avez l'intention de les utiliser dans le cadre de l'exécution du flux de travail.

- `MaxConcurrentRuns` : nombre (entier).

  Vous pouvez utiliser ce paramètre pour empêcher plusieurs mises à jour indésirables des données, pour contrôler les coûts ou, dans certains cas, pour empêcher le dépassement du nombre maximal d'exécutions simultanées de l'un des travaux de composant. Si vous laissez ce paramètre vide, le nombre d'exécutions de flux de travail simultanées est illimité.

Réponse

- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du flux de travail qui a été spécifié dans l'entrée.

Erreurs

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`

- `ConcurrentModificationException`

DeleteWorkflow action (Python : delete_workflow)

Supprime un flux de travail.

Demande

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du flux de travail à supprimer.

Réponse

- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du flux de travail spécifié dans l'entrée.

Erreurs

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ConcurrentModificationException`

GetWorkflow action (Python : get_workflow)

Récupère les métadonnées de ressource pour un flux de travail.

Demande

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du flux de travail à récupérer.
- `IncludeGraph` : booléen.

Indique si un graphique doit être inclus lors du renvoi de métadonnées de ressource de flux de travail.

Réponse

- `Workflow` – Un objet Flux de travail.

  Métadonnées de ressource du flux de travail.

Erreurs

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`

ListWorkflows action (Python : list_workflows)

Affiche les noms des flux de travail créés dans le compte.

Demande

- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, s'il s'agit d'une requête de continuation.
- `MaxResults`— Nombre (entier), pas moins de 1 ou plus de 25.

  La taille maximale d'une liste à renvoyer.

Réponse

- `Workflows` – tableau de chaînes UTF-8, avec 1 chaîne minimum et 25 chaînes maximum.

  Liste des noms des flux de travail dans le compte.
- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, si tous les noms de flux de travail n'ont pas été retournés.

Erreurs

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

BatchGetWorkflows action (Python : batch_get_workflows)

Retourne la liste des métadonnées de ressource pour une liste donnée de noms de flux de travail. Après avoir appelé l'opération `ListWorkflows`, vous pouvez appeler cette opération pour accéder aux données sur lesquelles des autorisations vous ont été octroyées. Cette opération prend en charge toutes les autorisations IAM, y compris les conditions d'autorisation qui utilisent des balises.

Demande

- `Names` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 25 chaînes maximum.

  Liste des noms de flux de travail, qui peuvent être les noms renvoyés à partir de l'opération `ListWorkflows`.
- `IncludeGraph` : booléen.

  Indique si un graphique doit être inclus lors du renvoi de métadonnées de ressource de flux de travail.

Réponse

- `Workflows` – tableau d'objets Flux de travail, avec 1 structure minimum et 25 structures maximum.

  Liste de métadonnées de ressource de flux de travail.
- `MissingWorkflows` – tableau de chaînes UTF-8, avec 1 chaîne minimum et 25 chaînes maximum.

  Liste des noms de flux de travail non trouvés.

Erreurs

- `InternalServiceException`
- `OperationTimeoutException`

- `InvalidInputException`

GetWorkflowRun action (Python : get_workflow_run)

Extrait les métadonnées pour une exécution de flux de travail donnée. L'historique d'exécution des tâches est accessible pendant 90 jours pour votre flux de travail et l'exécution des tâches.

Demande

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du flux de travail en cours d'exécution.
- `RunId` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID de l'exécution de flux de travail.
- `IncludeGraph` : booléen.

  Indique si le graphique de flux de travail doit être inclus en réponse ou non.

Réponse

- `Run` – Un objet WorkflowRun.

  Métadonnées d'exécution de flux de travail demandées.

Erreurs

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`

GetWorkflowRuns action (Python : get_workflow_runs)

Extrait les métadonnées de toutes les exécutions d'un flux de travail donné.

Demande

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du flux de travail dont les métadonnées des exécutions doivent être renvoyées.

- `IncludeGraph` : booléen.

  Indique si le graphique de flux de travail doit être inclus en réponse ou non.

- `NextToken` – Chaîne UTF-8.

  Taille maximale de la réponse.

- `MaxResults` – Nombre (entier), compris entre 1 et 1 000.

  Nombre maximal d'exécutions de flux de travail à inclure dans la réponse.

Réponse

- `Runs` – tableau d'objets WorkflowRun, avec 1 structure minimum et 1 000 structures maximum.

  Liste d'objets des métadonnées d'exécution de flux de travail.

- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, si toutes les exécutions de tâche demandées ne sont pas renvoyées.

Erreurs

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`

GetWorkflowRunProperties action (Python : get_workflow_run_properties)

Extrait les propriétés d'exécution de flux de travail qui ont été définies au cours de l'exécution.

Demande

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du flux de travail qui a été exécuté.

- `RunId` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID de l'exécution de flux de travail dont les propriétés d'exécution doivent être renvoyées.

Réponse

- `RunProperties` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaque valeur est une chaîne UTF-8.

  Propriétés d'exécution de flux de travail qui ont été définies au cours de l'exécution spécifiée.

Erreurs

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`

PutWorkflowRunProperties action (Python : put_workflow_run_properties)

Place les propriétés d'exécution de flux de travail spécifiées pour l'exécution de flux de travail donnée. Si une propriété existe déjà pour l'exécution spécifiée, elle remplace la valeur et ajoute la propriété aux propriétés existantes.

Demande

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du flux de travail qui a été exécuté.

- `RunId` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID de l'exécution de flux de travail pour laquelle les propriétés d'exécution doivent être mises à jour.

- `RunProperties` – obligatoire : tableau de mappage de paires clé-valeur.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaque valeur est une chaîne UTF-8.

  Propriétés à placer pour l'exécution spécifiée.

  Les propriétés d'exécution peuvent être journalisées. Ne transmettez pas de secrets en texte clair comme propriétés. Récupérez les secrets d'une AWS Glue connexion, d'un Gestionnaire de AWS Secrets ou d'un autre mécanisme de gestion des secrets si vous avez l'intention de les utiliser dans le cadre de l'exécution du flux de travail.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `AlreadyExistsException`
- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`
- `ConcurrentModificationException`

CreateBlueprint action (Python : créer_blueprint)

Enregistre un plan avec AWS Glue.

Demande

- `Name` – obligatoire : chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets, correspondant au Custom string pattern #31.

  Nom du plan.

- `Description` – chaîne UTF-8, d'une longueur comprise entre 1 et 512 octets.

  Description du plan.

- `BlueprintLocation` – obligatoire : chaîne UTF-8, d'une longueur comprise entre 1 et 8 192 octets, correspondant au Custom string pattern #32.

  Spécifie un chemin dans Amazon S3 où le plan est publié.

- `Tags` – tableau de mappage de paires clé-valeur, avec 50 paires au maximum.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 256 octets.

  Balises à appliquer à ce plan.

Réponse

- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Renvoie le nom du plan qui a été enregistré.

Erreurs

- `AlreadyExistsException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

- `ResourceNumberLimitExceededException`

UpdateBlueprint action (Python : update_blueprint)

Met à jour un plan enregistré.

Demande

- `Name` – obligatoire : chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets, correspondant au Custom string pattern #31.

  Nom du plan.

- `Description` – chaîne UTF-8, d'une longueur comprise entre 1 et 512 octets.

  Description du plan.

- `BlueprintLocation` – obligatoire : chaîne UTF-8, d'une longueur comprise entre 1 et 8 192 octets, correspondant au Custom string pattern #32.

  Spécifie un chemin dans Amazon S3 où le plan est publié.

Réponse

- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Renvoie le nom du plan qui a été mis à jour.

Erreurs

- `EntityNotFoundException`
- `ConcurrentModificationException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`
- `IllegalBlueprintStateException`

DeleteBlueprint action (Python : delete_blueprint)

Supprime un plan existant.

Demande

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du plan à supprimer.

Réponse

- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Renvoie le nom du plan qui a été supprimé.

Erreurs

- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

ListBlueprints action (Python : list_blueprints)

Répertorie tous les noms de plan d'un compte.

Demande

- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, s'il s'agit d'une requête de continuation.
- `MaxResults`— Nombre (entier), pas moins de 1 ou plus de 25.

  La taille maximale d'une liste à renvoyer.
- `Tags` – tableau de mappage de paires clé-valeur, avec 50 paires au maximum.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 256 octets.

Filtre la liste en fonction d'une balise de AWS ressource.

Réponse

- `Blueprints` – Tableau de chaînes UTF-8.

  Liste des noms des plans dans le compte.
- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, si tous les noms de plan n'ont pas été renvoyés.

Erreurs

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

BatchGetBlueprints action (Python : batch_get_blueprints)

Extrait des informations sur une liste de plans.

Demande

- `Names` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 25 chaînes maximum.

  Une liste de noms de plan.
- `IncludeBlueprint` : booléen.

  Indique si le plan doit être inclus dans la réponse ou non.
- `IncludeParameterSpec` : booléen.

  Spécifie s'il faut ou non inclure les paramètres, sous forme de chaîne JSON, pour le plan dans la réponse.

Réponse

- `Blueprints` : un tableau d'objets Plan .

  Renvoie la liste des plans sous forme d'objet `Blueprints`.
- `MissingBlueprints` – Tableau de chaînes UTF-8.

  Renvoie une liste de `BlueprintNames` qui ont été introuvables.

Erreurs

- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`

StartBlueprintRun action (Python : start_blueprint_run)

Démarre une nouvelle exécution du plan spécifié.

Demande

- `BlueprintName` – obligatoire : chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets, correspondant au Custom string pattern #31.

  Nom du plan.
- `Parameters` – chaîne UTF-8, d'une longueur comprise entre 1 et 131 072 octets.

  Spécifie les paramètres en tant qu'objet `BlueprintParameters`.
- `RoleArn` – Obligatoire : chaîne UTF-8, d'une longueur comprise entre 1 et 1024 octets, correspondant au Custom string pattern #30.

  Spécifie le rôle IAM utilisé pour créer le flux de travail.

Réponse

- `RunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

ID d'exécution de cette exécution de plan.

Erreurs

- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`
- `ResourceNumberLimitExceededException`
- `EntityNotFoundException`
- `IllegalBlueprintStateException`

GetBlueprintRun action (Python : get_blueprint_run)

Récupère les détails d'une exécution de plan.

Demande

- `BlueprintName` – obligatoire : chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets, correspondant au Custom string pattern #31.

  Nom du plan.
- `RunId` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID d'exécution de l'exécution de plan que vous souhaitez récupérer.

Réponse

- `BlueprintRun` – Un objet BlueprintRun.

  Renvoie un objet `BlueprintRun`.

Erreurs

- `EntityNotFoundException`
- `InternalServiceException`

- `OperationTimeoutException`

GetBlueprintRuns action (Python : get_blueprint_runs)

Récupère les détails des exécutions de plan pour un plan spécifié.

Demande

- `BlueprintName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du plan.
- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, s'il s'agit d'une requête de continuation.
- `MaxResults` – Nombre (entier), compris entre 1 et 1 000.

  La taille maximale d'une liste à renvoyer.

Réponse

- `BlueprintRuns` : un tableau d'objets BlueprintRun.

  Envoie la liste des objets `BlueprintRun`.
- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, si toutes les exécutions de plan ne sont pas renvoyées.

Erreurs

- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`

StartWorkflowRun action (Python : start_workflow_run)

Démarre une nouvelle exécution du flux de travail spécifié.

Demande

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du flux de travail à démarrer.
- `RunProperties` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaque valeur est une chaîne UTF-8.

  Les propriétés d'exécution de flux de travail pour la nouvelle exécution de flux de travail.

  Les propriétés d'exécution peuvent être journalisées. Ne transmettez pas de secrets en texte clair comme propriétés. Récupérez les secrets d'une AWS Glue connexion, d'un Gestionnaire de AWS Secrets ou d'un autre mécanisme de gestion des secrets si vous avez l'intention de les utiliser dans le cadre de l'exécution du flux de travail.

Réponse

- `RunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID pour la nouvelle exécution.

Erreurs

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`

- `ConcurrentRunsExceededException`

StopWorkflowRun action (Python : stop_workflow_run)

Arrête l'exécution du cycle de flux de travail spécifié.

Demande

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du flux de travail à arrêter.
- `RunId` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID de du cycle de flux de travail à arrêter.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `IllegalWorkflowStateException`

ResumeWorkflowRun action (Python : resume_workflow_run)

Redémarre les nœuds sélectionnés d'une précédente exécution de flux de travail partiellement terminée et reprend l'exécution de flux de travail. Les nœuds sélectionnés et tous les nœuds en aval des nœuds sélectionnés sont exécutés.

Demande

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du flux de travail à reprendre.

- `RunId` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID de l'exécution du flux de travail à reprendre.

- `NodeIds` – obligatoire : tableau de chaînes UTF-8.

  Liste IDs des nœuds que vous souhaitez redémarrer. Les nœuds qui doivent être redémarrés doivent avoir une tentative d'exécution dans l'exécution d'origine.

Réponse

- `RunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nouvel ID attribué à l'exécution de flux de travail reprise. Chaque résumé d'exécution de flux de travail aura un nouvel ID d'exécution.

- `NodeIds` – Tableau de chaînes UTF-8.

  Liste des nœuds IDs qui ont été réellement redémarrés.

Erreurs

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ConcurrentRunsExceededException`
- `IllegalWorkflowStateException`

Profils d'utilisation

L'API des profils d'utilisation décrit les types de données et l'API liés à la création, à la mise à jour ou à l'affichage des profils d'utilisation dans AWS Glue.

Types de données

- ProfileConfiguration structure
- ConfigurationObject structure
- UsageProfileDefinition structure

ProfileConfiguration structure

Spécifie les valeurs de tâche et de session qu'un administrateur configure dans un profil AWS Glue d'utilisation.

Champs

- `SessionConfiguration` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaque valeur est un objet ConfigurationObject.

  Une carte clé-valeur des paramètres de configuration pour AWS Glue les sessions.
- `JobConfiguration` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaque valeur est un objet ConfigurationObject.

  Une carte clé-valeur des paramètres de configuration pour AWS Glue les tâches.

ConfigurationObject structure

Spécifie les valeurs définies par un administrateur pour chaque paramètre de tâche ou de session configuré dans un profil AWS Glue d'utilisation.

Champs

- `DefaultValue` – chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets, correspondant au Custom string pattern #35.

  Valeur par défaut du paramètre.

- `AllowedValues` – Tableau de chaînes UTF-8.

  Liste de valeurs autorisées pour le paramètre.

- `MinValue` – chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets, correspondant au Custom string pattern #35.

  Valeur minimale autorisée pour le paramètre.

- `MaxValue` – chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets, correspondant au Custom string pattern #35.

  Valeur maximale autorisée pour le paramètre.

UsageProfileDefinition structure

Décrit un profil AWS Glue d'utilisation.

Champs

- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du profil d'utilisation.

- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description du profil d'utilisation.

- `CreatedOn` – Horodatage.

  Date et heure de la création du profil d'utilisation.

- `LastModifiedOn` – Horodatage.

  Date et heure auxquelles le profil d'utilisation a été modifié pour la dernière fois.

Opérations

- CreateUsageProfile action (Python : create_usage_profile)
- GetUsageProfile action (Python : get_usage_profile)
- UpdateUsageProfile action (Python : update_usage_profile)
- DeleteUsageProfile action (Python : delete_usage_profile)
- ListUsageProfiles action (Python : list_usage_profiles)

CreateUsageProfile action (Python : create_usage_profile)

Crée un profil AWS Glue d'utilisation.

Demande

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du profil d'utilisation.
- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description du profil d'utilisation.
- `Configuration` – Obligatoire : un objet ProfileConfiguration.

  Objet `ProfileConfiguration` spécifiant les valeurs de tâche et de session pour le profil.
- `Tags` – Tableau de mappage de paires clé-valeur, avec 50 paires au maximum.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 256 octets.

  Liste de balises appliquées au profil d'utilisation.

Réponse

- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

Nom du profil d'utilisation créé.

Erreurs

- `InvalidInputException`
- `InternalServiceException`
- `AlreadyExistsException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`
- `OperationNotSupportedException`

GetUsageProfile action (Python : get_usage_profile)

Récupère les informations relatives au profil AWS Glue d'utilisation spécifié.

Demande

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du profil d'utilisation à récupérer.

Réponse

- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du profil d'utilisation.
- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description du profil d'utilisation.
- `Configuration` – Un objet ProfileConfiguration.

  Objet `ProfileConfiguration` spécifiant les valeurs de tâche et de session pour le profil.
- `CreatedOn` – Horodatage.

Date et heure de la création du profil d'utilisation.

- `LastModifiedOn` – Horodatage.

  Date et heure auxquelles le profil d'utilisation a été modifié pour la dernière fois.

Erreurs

- `InvalidInputException`
- `InternalServiceException`
- `EntityNotFoundException`
- `OperationTimeoutException`
- `OperationNotSupportedException`

UpdateUsageProfile action (Python : update_usage_profile)

Mettez à jour un profil AWS Glue d'utilisation.

Demande

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du profil d'utilisation.
- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description du profil d'utilisation.
- `Configuration` – Obligatoire : un objet ProfileConfiguration.

  Objet `ProfileConfiguration` spécifiant les valeurs de tâche et de session pour le profil.

Réponse

- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du profil d'utilisation qui a été mis à jour.

Erreurs

- `InvalidInputException`
- `InternalServiceException`
- `EntityNotFoundException`
- `OperationTimeoutException`
- `OperationNotSupportedException`
- `ConcurrentModificationException`

DeleteUsageProfile action (Python : delete_usage_profile)

Supprime le profil d'utilisation AWS Glue spécifié.

Demande

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du profil d'utilisation à supprimer.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `OperationNotSupportedException`

ListUsageProfiles action (Python : list_usage_profiles)

Répertoriez tous les profils AWS Glue d'utilisation.

Demande

- `NextToken` – Chaîne UTF-8, d'une longueur maximale de 400 000 octets.

  Jeton de continuation, inclus s'il s'agit d'un appel de continuation.
- `MaxResults`— Nombre (entier), pas moins de 1 ou plus de 200.

  Nombre maximal de profils d'utilisation à renvoyer dans une seule réponse.

Réponse

- `Profiles` : un tableau d'objets UsageProfileDefinition.

  Liste des objets du profil d'utilisation (`UsageProfileDefinition`).
- `NextToken` – Chaîne UTF-8, d'une longueur maximale de 400 000 octets.

  Jeton de continuation, présent si le segment de liste actuel n'est pas le dernier.

Erreurs

- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`
- `OperationNotSupportedException`

API de Machine Learning

L'API Machine learning décrit les types de données du machine learning et inclut l'API de création, suppression ou mise à jour d'une transformation, ou de démarrage de l'exécution d'une tâche de machine learning.

Types de données

- TransformParameters structure
- EvaluationMetrics structure
- MLTransform structure

- FindMatchesParameters structure
- FindMatchesMetrics structure
- ConfusionMatrix structure
- GlueTable structure
- TaskRun structure
- TransformFilterCriteria structure
- TransformSortCriteria structure
- TaskRunFilterCriteria structure
- TaskRunSortCriteria structure
- TaskRunProperties structure
- FindMatchesTaskRunProperties structure
- ImportLabelsTaskRunProperties structure
- ExportLabelsTaskRunProperties structure
- LabelingSetGenerationTaskRunProperties structure
- SchemaColumn structure
- TransformEncryption structure
- MLUserDataEncryption structure
- ColumnImportance structure

TransformParameters structure

Paramètres propres à l'algorithme associés à la transformation Machine Learning.

Champs

- `TransformType` – Obligatoire : Chaîne UTF-8 (valeurs valides : `FIND_MATCHES`).

  Type de transformation du Machine Learning.

  Pour de plus amples informations sur les types de transformation du Machine Learning, veuillez consulter Création de transformations du Machine Learning.

- `FindMatchesParameters` – Un objet FindMatchesParameters.

Paramètres de l'algorithme de recherche de correspondances.

EvaluationMetrics structure

Les métriques d'évaluation fournissent une estimation de la qualité de votre transformation du Machine Learning.

Champs

- `TransformType` – Obligatoire : Chaîne UTF-8 (valeurs valides : `FIND_MATCHES`).

  Type de transformation du Machine Learning.
- `FindMatchesMetrics` – Un objet FindMatchesMetrics.

  Métriques d'évaluation de l'algorithme de recherche de correspondances.

MLTransform structure

Structure d'une transformation du Machine Learning.

Champs

- `TransformId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID de transformation unique généré pour la transformation du Machine Learning. L'ID est garanti être unique et ne pas changer.
- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom défini par l'utilisateur de la transformation Machine Learning. Les noms ne sont pas garantis être uniques et peuvent être modifiés à tout moment.
- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Longue description définie par l'utilisateur de la transformation Machine Learning. Les descriptions ne sont pas garanties être uniques et peuvent être modifiées à tout moment.
- `Status` – Chaîne UTF-8 (valeurs valides : `NOT_READY` | `READY` | `DELETING`).

Statut actuel de la transformation du Machine Learning.

- `CreatedOn` – Horodatage.

  Horodatage. Heure et date auxquelles la transformation du Machine Learning a été créée.

- `LastModifiedOn` – Horodatage.

  Horodatage. Dernier moment auquel cette transformation du Machine Learning a été modifiée.

- `InputRecordTables` – Un tableau d'objets GlueTable, 10 structures au maximum.

  Liste des définitions de AWS Glue table utilisées par la transformation.

- `Parameters` – Un objet TransformParameters.

  Un objet `TransformParameters`. Vous pouvez utiliser des paramètres pour ajuster (personnaliser) le comportement de la transformation du Machine Learning : à cette fin, spécifiez les données apprises du Machine Learning, ainsi que vos préférences quant aux différents compromis à faire (valeur versus rappel, ou précision versus coût, par exemple).

- `EvaluationMetrics` – Un objet EvaluationMetrics.

  Un objet `EvaluationMetrics`. Les métriques d'évaluation fournissent une estimation de la qualité de votre transformation du Machine Learning.

- `LabelCount` : nombre (entier).

  Identifiant de comptage pour les fichiers d'étiquetage générés par AWS Glue cette transformation. Au fur et à mesure que vous créez une transformation de meilleure qualité, vous pouvez, de manière itérative, télécharger, étiqueter et charger le fichier d'étiquetage.

- `Schema` – Un tableau d'objets SchemaColumn, 100 structures au maximum.

  Carte de paires clé-valeur correspondant aux colonnes et aux types de données sur lesquels la transformation peut s'exécuter. Comporte une limite supérieure de 100 colonnes.

- `Role` – Chaîne UTF-8.

  Nom ou Amazon Resource Name (ARN) du rôle IAM avec les autorisations requises. Les autorisations requises incluent à la fois les autorisations de rôle de AWS Glue service sur les AWS Glue ressources et les autorisations Amazon S3 requises par la transformation.

  - Ce rôle nécessite des autorisations AWS Glue de rôle de service pour autoriser l'accès aux ressources dans AWS Glue. Veuillez consulter la rubrique Attacher une politique aux utilisateurs IAM accédant à AWS Glue.

  - Ce rôle a besoin d'une autorisation pour vos sources, vos cibles, votre répertoire temporaire et vos scripts Amazon Simple Storage Service (Amazon S3), ainsi que pour les bibliothèques utilisées par l'exécution de la tâche dans le cadre de cette transformation.

- `GlueVersion` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #48.

  Cette valeur détermine la version compatible avec laquelle AWS Glue cette transformation de machine learning est compatible. Glue 1.0 est recommandé pour la plupart des clients. Si la valeur n'est pas définie, la compatibilité Glue par défaut est Glue 0.9. Pour plus d'informations, veuillez consulter la rubrique Versions AWS Glue dans le guide du développeur.

- `MaxCapacity` – Nombre (double).

  Nombre d'unités de traitement de AWS Glue données (DPUs) allouées aux exécutions de tâches pour cette transformation. Vous pouvez allouer de 2 à 100 DPUs ; la valeur par défaut est 10. Un DPU est une mesure relative de la puissance de traitement composée de 4 V de capacité CPUs de calcul et de 16 Go de mémoire. Pour plus d'informations, consultez la page de tarification AWS Glue.

  `MaxCapacity` est une option mutuellement exclusive avec `NumberOfWorkers` et `WorkerType`.

  - Si `NumberOfWorkers` ou `WorkerType` est défini, ne `MaxCapacity` peut pas être défini.
  - Si `MaxCapacity` est défini, ni `NumberOfWorkers` ni `WorkerType` ne peuvent être définis.
  - Si `WorkerType` est défini, `NumberOfWorkers` est obligatoire (et inversement).
  - `MaxCapacity` et `NumberOfWorkers` doivent être au moins égal à 1.

  Lorsque le champ `WorkerType` est défini sur une valeur autre que `Standard`, le champ `MaxCapacity` est défini automatiquement et passe en lecture seule.

- `WorkerType` – Chaîne UTF-8 (valeurs valides: `Standard=""` | `G.1X=""` | `G.2X=""` | `G.025X=""` | `G.4X=""` | `G.8X=""` | `Z.2X=""`).

  Type d'exécuteur prédéfini qui est alloué lorsqu'une tâche de cette transformation s'exécute. Accepte la valeur Standard, G.1X ou G.2X.

  - Pour le type de travail `Standard`, chaque travail fournit 4 vCPU, 16 Go de mémoire et 50 Go de disque, ainsi que 2 exécuteurs par travail.
  - Pour le type de travail `G.1X`, chaque travail fournit 4 vCPU, 16 Go de mémoire et 64 Go de disque, ainsi que 1 exécuteur par travail.

  - Pour le type de travail `G.2X`, chaque travail fournit 8 vCPU, 32 Go de mémoire et 128 Go de disque, ainsi que 1 exécuteur par travail.

  `MaxCapacity` est une option mutuellement exclusive avec `NumberOfWorkers` et `WorkerType`.

  - Si `NumberOfWorkers` ou `WorkerType` est défini, ne `MaxCapacity` peut pas être défini.
  - Si `MaxCapacity` est défini, ni `NumberOfWorkers` ni `WorkerType` ne peuvent être définis.
  - Si `WorkerType` est défini, `NumberOfWorkers` est obligatoire (et inversement).
  - `MaxCapacity` et `NumberOfWorkers` doivent être au moins égal à 1.
- `NumberOfWorkers` : nombre (entier).

  Nombre d'exécuteurs d'un `workerType` défini qui sont alloués lorsqu'une tâche de la transformation s'exécute.

  Si `WorkerType` est défini, `NumberOfWorkers` est obligatoire (et inversement).
- `Timeout` – Nombre (entier), au moins égal à 1.

  Délai d'expiration en minutes de la transformation Machine Learning.
- `MaxRetries` : nombre (entier).

  Nombre maximal de tentatives autorisées après l'échec d'une exécution `MLTaskRun` de la transformation Machine Learning.
- `TransformEncryption` – Un objet TransformEncryption.

  Les encryption-at-rest paramètres de la transformation qui s'appliquent à l'accès aux données utilisateur. Les transformations de machine learning peuvent accéder aux données utilisateur chiffrées dans Amazon S3 à l'aide de KMS.

FindMatchesParameters structure

Paramètres de configuration de la transformation de la recherche des correspondances.

Champs

- `PrimaryKeyColumnName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 1024 octets, correspondant au Single-line string pattern.

Nom d'une colonne qui identifie de façon unique les lignes de la table source. Utilisé pour vous aider à identifier les enregistrements correspondants.

- `PrecisionRecallTradeoff` – Nombre (double), au plus égal à 1,0.

  Valeur sélectionnée lors du paramétrage de votre transformation pour un équilibre entre la précision et le rappel. Une valeur de 0,5 signifie aucune préférence ; une valeur de 1.0 signifie un écart uniquement pour la précision, et une valeur de 0,0 un écart pour le rappel. Comme il s'agit d'un compromis, le choix de valeurs proches de 1.0 signifie un rappel très faible et le choix de valeurs proches de 0.0 une très faible précision.

  La précision métrique indique la fréquence à laquelle votre modèle est correct lorsqu'il prédit une correspondance.

  La métrique de rappel indique que pour une correspondance réelle, la fréquence à laquelle votre modèle prédit la correspondance.

- `AccuracyCostTradeoff` – Nombre (double), au plus égal à 1,0.

  Valeur sélectionnée lors du paramétrage de votre transformation pour un équilibre entre la précision et le coût. Une valeur de 0,5 signifie que le système équilibre les préoccupations de précision et de coût. Une valeur de 1.0 signifie un biais uniquement pour la précision, ce qui entraîne généralement un coût plus élevé, et parfois beaucoup plus élevé. Une valeur de 0,0 signifie un biais uniquement pour le coût, ce qui se traduit par une transformation `FindMatches` moins précise, et parfois une précision inacceptable.

  La précision mesure la façon dont la transformation trouve les vrais positifs et les vrais négatifs. L'augmentation de la précision nécessite plus de ressources machine et des coûts supérieurs. Mais elle entraîne également une augmentation du rappel.

  Le coût mesure le nombre de ressources de calcul, et donc le montant, consommées pour exécuter la transformation.

- `EnforceProvidedLabels` : booléen.

  La valeur à activer ou désactiver pour forcer la sortie afin qu'elle corresponde aux étiquettes fournies par les utilisateurs. Si la valeur est `True`, la transformation `find matches` oblige la sortie à correspondre aux étiquettes fournies. Les résultats remplacent les résultats de combinaison normaux. Si la valeur est `False`, la transformation `find matches` ne garantit pas que toutes les étiquettes fournies sont respectées, et les résultats s'appuient sur le modèle formé.

Notez que la définition de cette valeur sur true peut augmenter la durée d'exécution de la combinaison.

FindMatchesMetrics structure

Métriques d'évaluation de l'algorithme de recherche de correspondances. La qualité de votre transformation Machine Learning est mesurée en obtenant que votre transformation prédise certaines correspondances et en comparant les résultats pour connaître les correspondances du même jeu de données. Les métriques de qualité sont basées sur un sous-ensemble de vos données, afin qu'ils ne soient pas précis.

Champs

- `AreaUnderPRCurve` – Nombre (double), au plus égal à 1,0.

  L'aire sous la precision/recall courbe (AUPRC) est un nombre unique qui mesure la qualité globale de la transformation, indépendamment du choix fait entre précision et rappel. Les valeurs élevées indiquent que vous avez un compromis précision vs. rappel plus attrayant.

  Pour en savoir plus, consultez Précision et rappel dans Wikipédia.

- `Precision` – Nombre (double), au plus égal à 1,0.

  La précision métrique indique à quelle fréquence votre transformation est correcte lorsqu'elle prédit une correspondance. Plus précisément, elle évalue la fréquence à laquelle la transformation trouve de vrais positifs à partir du total de vrais positifs possible.

  Pour en savoir plus, consultez Précision et rappel dans Wikipédia.

- `Recall` – Nombre (double), au plus égal à 1,0.

  La sensibilité métrique indique pour une correspondance réelle, la fréquence à laquelle votre transformation prédit la correspondance. Plus précisément, elle évalue la fréquence à laquelle la transformation trouve de vrais positifs à partir du total des enregistrements de la source de données.

  Pour en savoir plus, consultez Précision et rappel dans Wikipédia.

- `F1` – Nombre (double), au plus égal à 1,0.

La métrique F1 maximale indique la précision de la transformation entre 0 et 1, où 1 est la meilleure précision.

Pour plus d'informations, consultez la page Wikipedia relative au score F1.

- `ConfusionMatrix` – Un objet ConfusionMatrix.

  La matrice de confusion vous explique ce que votre transformation prédit avec précision et quels types d'erreurs elle effectue.

  Pour plus d'informations, consultez Matrice de confusion dans Wikipédia.

- `ColumnImportances` – Un tableau d'objets ColumnImportance, 100 structures au maximum.

  Une liste de structures `ColumnImportance` contenant des métriques d'importance de colonne, triées par ordre d'importance décroissant.

ConfusionMatrix structure

La matrice de confusion vous explique ce que votre transformation prédit avec précision et quels types d'erreurs elle effectue.

Pour plus d'informations, consultez Matrice de confusion dans Wikipédia.

Champs

- `NumTruePositives` – Nombre (long).

  Nombre de correspondances dans les données que la transformation a trouvées correctement, dans la matrice de confusion de votre transformation.

- `NumFalsePositives` – Nombre (long).

  Le nombre de non-correspondances dans les données que la transformation a classées de manière incorrecte comme correspondance, dans la matrice de confusion de votre transformation.

- `NumTrueNegatives` – Nombre (long).

  Nombre de non-correspondances dans les données que la transformation a rejetées correctement, dans la matrice de confusion de votre transformation.

- `NumFalseNegatives` – Nombre (long).

Nombre de correspondances dans les données que la transformation n'a pas trouvées, dans la matrice de confusion de votre transformation.

GlueTable structure

Base de données et table utilisées pour AWS Glue Data Catalog les données d'entrée ou de sortie.

Champs

- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la base de données dans le AWS Glue Data Catalog.

- `TableName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table dans AWS Glue Data Catalog.

- `CatalogId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant unique pour AWS Glue Data Catalog.

- `ConnectionName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la connexion à AWS Glue Data Catalog.

- `AdditionalOptions` – Tableau de mappage de paires valeur-clé, pas moins de 1 ou plus de 10 paires..

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaque valeur est une chaîne Description, d'une longueur ne dépassant pas 2048 octets, correspondant au URI address multi-line string pattern.

  Options supplémentaires pour la table. Actuellement, deux clés sont prises en charge :

  - `pushDownPredicate` : pour filtrer les partitions sans avoir à répertorier ni lire tous les fichiers de votre jeu de données.

- `catalogPartitionPredicate` : pour utiliser le nettoyage de partition côté serveur à l'aide des index de partition du AWS Glue Data Catalog.

TaskRun structure

Paramètres d'échantillonnage qui sont associés à la transformation du Machine Learning.

Champs

- `TransformId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant unique de la transformation.
- `TaskRunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant unique de la tâche exécutée.
- `Status` – Chaîne UTF-8 (valeurs valides : `RUNNING` | `FINISHED` | `FAILED` | `PENDING_EXECUTION` | `TIMED_OUT` | `CANCELING` | `CANCELED` | `RECEIVED_BY_TASKRUNNER`).

  Statut actuel de la tâche demandée.
- `LogGroupName` – Chaîne UTF-8.

  Nom du groupe de journaux pour une journalisation sécurisée, associée à la tâche exécutée.
- `Properties` – Un objet TaskRunProperties.

  Spécifie les propriétés de configuration associées à la tâche exécutée.
- `ErrorString` – Chaîne UTF-8.

  Liste des chaînes d'erreur associées à la tâche exécutée.
- `StartedOn` – Horodatage.

  Date et heure auxquelles la tâche exécutée a démarré.
- `LastModifiedOn` – Horodatage.

  Dernière date de modification de la tâche exécutée demandée.
- `CompletedOn` – Horodatage.

Heure à laquelle l'exécution de la tâche demandée s'est terminée.

- `ExecutionTime` : nombre (entier).

  Durée (en secondes) pendant laquelle la tâche exécutée a consommé des ressources.

TransformFilterCriteria structure

Critères utilisés pour filtrer les transformations du Machine Learning.

Champs

- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de transformation unique qui est utilisé pour filtrer les transformations du Machine Learning.

- `TransformType` – Chaîne UTF-8 (valeurs valides : `FIND_MATCHES`).

  Type de transformation du Machine Learning utilisé pour filtrer les transformations du Machine Learning.

- `Status` – Chaîne UTF-8 (valeurs valides : `NOT_READY` | `READY` | `DELETING`).

  Filtre la liste des transformations du Machine Learning sur le dernier état connu des transformations (pour indiquer si une transformation peut être utilisée ou non). L'un des états « NOT_READY », « READY » ou « DELETING ».

- `GlueVersion` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #48.

  Cette valeur détermine la version compatible avec laquelle AWS Glue cette transformation de machine learning est compatible. Glue 1.0 est recommandé pour la plupart des clients. Si la valeur n'est pas définie, la compatibilité Glue par défaut est Glue 0.9. Pour plus d'informations, veuillez consulter la rubrique Versions AWS Glue dans le guide du développeur.

- `CreatedBefore` – Horodatage.

  Heure et date avant lesquelles les transformations ont été créées.

- `CreatedAfter` – Horodatage.

  Heure et date après lesquelles les transformations ont été créées.

- `LastModifiedBefore` – Horodatage.

  Filtre sur les dernières transformations modifiées avant cette date.

- `LastModifiedAfter` – Horodatage.

  Filtre sur les dernières transformations modifiées après cette date.

- `Schema` – Un tableau d'objets SchemaColumn, 100 structures au maximum.

  Filtre sur les ensembles de données avec un schéma spécifique. L'objet `Map<Column, Type>` est un tableau de paires clé-valeur qui représente le schéma que cette transformation accepte, où `Column` est le nom d'une colonne, et `Type` le type de données, comme nombre entier ou chaîne. Comporte une limite supérieure de 100 colonnes.

TransformSortCriteria structure

Critères de tri associés à la transformation du Machine Learning.

Champs

- `Column` – Obligatoire : Chaîne UTF-8 (valeurs valides : `NAME` | `TRANSFORM_TYPE` | `STATUS` | `CREATED` | `LAST_MODIFIED`).

  La colonne doit être utilisée dans les critères de tri associés à la transformation du Machine Learning.

- `SortDirection` – Obligatoire : Chaîne UTF-8 (valeurs valides : `DESCENDING` | `ASCENDING`).

  La direction de tri doit être utilisée dans les critères de tri associés à la transformation du Machine Learning.

TaskRunFilterCriteria structure

Critères utilisés pour filtrer les exécutions de tâche pour la transformation du Machine Learning.

Champs

- `TaskRunType` – Chaîne UTF-8 (valeurs valides : `EVALUATION` | `LABELING_SET_GENERATION` | `IMPORT_LABELS` | `EXPORT_LABELS` | `FIND_MATCHES`).

  Type de tâche exécutée.

- `Status` – Chaîne UTF-8 (valeurs valides : `RUNNING` | `FINISHED` | `FAILED` | `PENDING_EXECUTION` | `TIMED_OUT` | `CANCELING` | `CANCELED` | `RECEIVED_BY_TASKRUNNER`).

  Statut actuel de la tâche exécutée.

- `StartedBefore` – Horodatage.

  Filtre sur les exécutions de tâche démarrées avant cette date.

- `StartedAfter` – Horodatage.

  Filtre sur les exécutions de tâche démarrées après cette date.

TaskRunSortCriteria structure

Critères de tri utilisés pour trier la liste des exécutions de tâche pour la transformation du Machine Learning.

Champs

- `Column` – Obligatoire : Chaîne UTF-8 (valeurs valides : `TASK_RUN_TYPE` | `STATUS` | `STARTED`).

  La colonne doit être utilisé pour trier la liste des exécutions de tâche pour la transformation du Machine Learning.

- `SortDirection` – Obligatoire : Chaîne UTF-8 (valeurs valides : `DESCENDING` | `ASCENDING`).

  La direction de tri doit être utilisé pour trier la liste des exécutions de tâche pour la transformation du Machine Learning.

TaskRunProperties structure

Propriétés de configuration pour la tâche exécutée.

Champs

- `TaskType` – Chaîne UTF-8 (valeurs valides : `EVALUATION` | `LABELING_SET_GENERATION` | `IMPORT_LABELS` | `EXPORT_LABELS` | `FIND_MATCHES`).

  Type de tâche exécutée.

- `ImportLabelsTaskRunProperties` : un objet ImportLabelsTaskRunProperties.

Propriétés de configuration d'une exécution de tâche d'importation des étiquettes.

- `ExportLabelsTaskRunProperties` : un objet ExportLabelsTaskRunProperties.

  Propriétés de configuration d'une exécution de tâche d'exportation des étiquettes.

- `LabelingSetGenerationTaskRunProperties` – Un objet LabelingSetGenerationTaskRunProperties.

  Propriétés de configuration pour une exécution de tâches de génération d'ensemble d'étiquetage.

- `FindMatchesTaskRunProperties` – Un objet FindMatchesTaskRunProperties.

  Propriétés de configuration pour une exécution de tâche de recherche de correspondances.

FindMatchesTaskRunProperties structure

Spécifie les propriétés de configuration d'une exécution de tâche de recherche de correspondances.

Champs

- `JobId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID de tâche d'une exécution de tâche de recherche de correspondances.

- `JobName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom attribué à la tâche pour l'exécution de tâche de recherche de correspondances.

- `JobRunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID d'exécution de tâche d'une exécution de tâche de recherche de correspondances.

ImportLabelsTaskRunProperties structure

Spécifie les propriétés de configuration d'une exécution de tâche de l'importation des étiquettes.

Champs

- `InputS3Path` – Chaîne UTF-8.

Chemin Amazon Simple Storage Service (Amazon S3) à partir d'où vous allez importer les étiquettes.

- `Replace` : booléen.

  Indique si vous souhaitez remplacer vos étiquettes existantes.

ExportLabelsTaskRunProperties structure

Spécifie les propriétés de configuration d'une exécution de tâche d'exportation des étiquettes.

Champs

- `OutputS3Path` – Chaîne UTF-8.

  Chemin Amazon Simple Storage Service (Amazon S3) où vous allez exporter les étiquettes.

LabelingSetGenerationTaskRunProperties structure

Spécifie les propriétés de configuration pour une exécution de tâches de génération d'ensemble d'étiquetage.

Champs

- `OutputS3Path` – Chaîne UTF-8.

  Chemin Amazon Simple Storage Service (Amazon S3) où vous allez générer l'ensemble d'étiquetage.

SchemaColumn structure

Paire clé-valeur qui représente une colonne et type de données sur lesquels cette transformation peut être exécuté. Le paramètre `Schema` de la `MLTransform` peut contenir jusqu'à 100 de ces structures.

Champs

- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 1024 octets, correspondant au Single-line string pattern.

Le nom de la colonne.

- `DataType` – Chaîne UTF-8, d'une longueur maximale de 131 072 octets, correspondant au Single-line string pattern.

  Type de données de la colonne.

TransformEncryption structure

Les encryption-at-rest paramètres de la transformation qui s'appliquent à l'accès aux données utilisateur. Les transformations de machine learning peuvent accéder aux données utilisateur chiffrées dans Amazon S3 à l'aide de KMS.

De plus, les étiquettes importées et les transformations formées peuvent désormais être chiffrées à l'aide d'une clé KMS fournie par le client.

Champs

- `MlUserDataEncryption` – Un objet MLUserDataEncryption.

  Objet `MLUserDataEncryption` contenant le mode de chiffrement et l'ID de la clé KMS fournie par le client.
- `TaskRunSecurityConfigurationName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la configuration de sécurité.

MLUserDataEncryption structure

Les encryption-at-rest paramètres de la transformation qui s'appliquent à l'accès aux données utilisateur.

Champs

- `MlUserDataEncryptionMode` – Obligatoire : Chaîne UTF-8 (valeurs valides : `DISABLED` | `SSE-KMS="SSEKMS"`).

  Mode de chiffrement appliqué aux données utilisateur. Les valeurs valides sont :

  - DISABLED : le chiffrement est désactivé

  - SSEKMS : utilisation du chiffrement côté serveur avec AWS Key Management Service (SSE-KMS) pour les données utilisateur stockées dans Amazon S3.
- `KmsKeyId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID de la clé KMS fournie par le client.

ColumnImportance structure

Structure contenant le nom de colonne et son score d'importance de colonne.

L'importance des colonnes vous aide à comprendre comment elles contribuent à votre modèle, en identifiant les colonnes de vos enregistrements qui sont plus importantes que les autres.

Champs

- `ColumnName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom d'une colonne.
- `Importance` – Nombre (double), au plus égal à 1,0.

  Score d'importance de colonne pour la colonne, sous la forme d'une décimale.

Opérations

- Créer une MLTransform action (Python : create_ml_transform)
- MLTransform Action de mise à jour (Python : update_ml_transform)
- MLTransform Action de suppression (Python : delete_ml_transform)
- Obtenir une MLTransform action (Python : get_ml_transform)
- Obtenir une MLTransforms action (Python : get_ml_transform)
- MLTransforms Action de liste (Python : list_ml_transform)
- Lancer MLEvaluation TaskRun l'action (Python : start_ml_evaluation_task_run)
- Lancer MLLabeling SetGenerationTaskRun l'action (Python : start_ml_labeling_set_generation_task_run)
- Action Get MLTask Run (Python : get_ml_task_run)

- Action Get MLTask Runs (Python : get_ml_task_runs)
- Annuler l'action MLTask Exécuter (Python : cancel_ml_task_run)
- StartExportLabelsTaskRun action (Python : start_export_labels_task_run)
- StartImportLabelsTaskRun action (Python : start_import_labels_task_run)

Créer une MLTransform action (Python : create_ml_transform)

Crée une transformation basée sur le AWS Glue machine learning. Cette opération crée la transformation et tous les paramètres nécessaires pour la former.

Appelez cette opération comme première étape du processus d'utilisation d'une transformation du Machine Learning (comme la transformation `FindMatches`) pour la déduplication des données. Vous pouvez fournir une `Description` facultative, en plus des paramètres que vous souhaitez utiliser pour votre algorithme.

Vous devez également définir certains paramètres pour les tâches exécutées en votre nom AWS Glue dans le cadre de l'apprentissage à partir de vos données et de la création d'une transformation d'apprentissage automatique de haute qualité. Ces paramètres incluent `Role` et, le cas échéant, `AllocatedCapacity`, `Timeout` et `MaxRetries`. Pour en savoir plus, consultez Jobs (Tâches).

Demande

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom unique que vous attribuez à la transformation lorsque vous la créez.
- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description de la transformation du Machine Learning qui est définie. La valeur par défaut est une chaîne vide.
- `InputRecordTables` – Obligatoire : Un tableau d'objets GlueTable, 10 structures au maximum.

  Liste des définitions de AWS Glue table utilisées par la transformation.
- `Parameters` – Obligatoire : un objet TransformParameters.

  Paramètres algorithmiques spécifiques au type de transformation utilisé. Dépendant de façon conditionnelle du type de transformation.

- `Role` – Obligatoire : chaîne UTF-8.

  Nom ou Amazon Resource Name (ARN) du rôle IAM avec les autorisations requises. Les autorisations requises incluent à la fois les autorisations de rôle de AWS Glue service sur les AWS Glue ressources et les autorisations Amazon S3 requises par la transformation.

  - Ce rôle nécessite des autorisations AWS Glue de rôle de service pour autoriser l'accès aux ressources dans AWS Glue. Veuillez consulter la rubrique Attacher une politique aux utilisateurs IAM accédant à AWS Glue.
  - Ce rôle a besoin d'une autorisation pour vos sources, vos cibles, votre répertoire temporaire et vos scripts Amazon Simple Storage Service (Amazon S3), ainsi que pour les bibliothèques utilisées par l'exécution de la tâche dans le cadre de cette transformation.
- `GlueVersion` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #48.

  Cette valeur détermine la version compatible avec laquelle AWS Glue cette transformation de machine learning est compatible. Glue 1.0 est recommandé pour la plupart des clients. Si la valeur n'est pas définie, la compatibilité Glue par défaut est Glue 0.9. Pour plus d'informations, veuillez consulter la rubrique Versions AWS Glue dans le guide du développeur.
- `MaxCapacity` – Nombre (double).

  Nombre d'unités de traitement de AWS Glue données (DPUs) allouées aux exécutions de tâches pour cette transformation. Vous pouvez allouer de 2 à 100 DPUs ; la valeur par défaut est 10. Un DPU est une mesure relative de la puissance de traitement composée de 4 V de capacité CPUs de calcul et de 16 Go de mémoire. Pour plus d'informations, consultez la page de tarification AWS Glue.

  `MaxCapacity` est une option mutuellement exclusive avec `NumberOfWorkers` et `WorkerType`.

  - Si `NumberOfWorkers` ou `WorkerType` est défini, ne `MaxCapacity` peut pas être défini.
  - Si `MaxCapacity` est défini, ni `NumberOfWorkers` ni `WorkerType` ne peuvent être définis.
  - Si `WorkerType` est défini, `NumberOfWorkers` est obligatoire (et inversement).
  - `MaxCapacity` et `NumberOfWorkers` doivent être au moins égal à 1.

  Lorsque le champ `WorkerType` est défini sur une valeur autre que `Standard`, le champ `MaxCapacity` est défini automatiquement et passe en lecture seule.

  Lorsque le champ `WorkerType` est défini sur une valeur autre que `Standard`, le champ `MaxCapacity` est défini automatiquement et passe en lecture seule.

- `WorkerType` – Chaîne UTF-8 (valeurs valides: `Standard=""` | `G.1X=""` | `G.2X=""` | `G.025X=""` | `G.4X=""` | `G.8X=""` | `Z.2X=""`).

  Type de travail prédéfini qui est alloué lorsqu'une tâche est exécutée. Accepte la valeur Standard, G.1X ou G.2X.

  - Pour le type de travail `Standard`, chaque travail fournit 4 vCPU, 16 Go de mémoire et 50 Go de disque, ainsi que 2 exécuteurs par travail.
  - Pour le type de travail `G.1X`, chaque travail fournit 4 vCPU, 16 Go de mémoire et 64 Go de disque, ainsi que 1 exécuteur par travail.
  - Pour le type de travail `G.2X`, chaque travail fournit 8 vCPU, 32 Go de mémoire et 128 Go de disque, ainsi que 1 exécuteur par travail.

  `MaxCapacity` est une option mutuellement exclusive avec `NumberOfWorkers` et `WorkerType`.

  - Si `NumberOfWorkers` ou `WorkerType` est défini, ne `MaxCapacity` peut pas être défini.
  - Si `MaxCapacity` est défini, ni `NumberOfWorkers` ni `WorkerType` ne peuvent être définis.
  - Si `WorkerType` est défini, `NumberOfWorkers` est obligatoire (et inversement).
  - `MaxCapacity` et `NumberOfWorkers` doivent être au moins égal à 1.
- `NumberOfWorkers` : nombre (entier).

  Nombre de travaux d'un `workerType` défini qui sont attribués lorsqu'une tâche est exécutée.

  Si `WorkerType` est défini, `NumberOfWorkers` est obligatoire (et inversement).
- `Timeout` – Nombre (entier), au moins égal à 1.

  Délai d'expiration de la tâche exécutée pour cette transformation en minutes. Durée maximale pendant laquelle l'exécution d'une tâche pour cette transformation peut utiliser des ressources avant qu'elle ne soit mise hors service et passe à l'état `TIMEOUT`. La valeur par défaut est de 2 880 minutes (48 heures).
- `MaxRetries` : nombre (entier).

  Nombre maximal de nouvelles tentatives d'une tâche de cette transformation après l'échec de l'exécution d'une tâche.
- `Tags` – Tableau de mappage de paires clé-valeur, avec 50 paires au maximum.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 256 octets.

Balises à utiliser avec cette transformation de Machine Learning. Vous pouvez utiliser des balises pour limiter l'accès à la transformation de Machine Learning. Pour plus d'informations sur les tags in AWS Glue, voir AWS Tags in AWS Glue dans le guide du développeur.

- `TransformEncryption` – Un objet TransformEncryption.

  Les encryption-at-rest paramètres de la transformation qui s'appliquent à l'accès aux données utilisateur. Les transformations de machine learning peuvent accéder aux données utilisateur chiffrées dans Amazon S3 à l'aide de KMS.

Réponse

- `TransformId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant unique généré pour la transformation.

Erreurs

- `AlreadyExistsException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`
- `AccessDeniedException`
- `ResourceNumberLimitExceededException`
- `IdempotentParameterMismatchException`

MLTransform Action de mise à jour (Python : update_ml_transform)

Met à jour une transformation du Machine Learning existante. Appelez cette opération pour ajuster les paramètres de l'algorithme afin d'obtenir de meilleurs résultats.

Après l'appel de cette opération, vous pouvez appeler l'opération `StartMLEvaluationTaskRun` pour évaluer comment vos nouveaux paramètres ont atteint vos objectifs (par exemple, améliorer la qualité de votre transformation Machine Learning, ou la rendre plus rentable).

Demande

- `TransformId` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant unique qui a été généré lorsque la transformation a été créée.

- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom unique que vous avez donné à la transformation lorsque vous l'avez créée.

- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description de la transformation. La valeur par défaut est une chaîne vide.

- `Parameters` – Un objet TransformParameters.

  Paramètres de configuration spécifiques au type de transformation (algorithme) utilisé. Dépendant de façon conditionnelle du type de transformation.

- `Role` – Chaîne UTF-8.

  Nom ou Amazon Resource Name (ARN) du rôle IAM avec les autorisations requises.

- `GlueVersion` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #48.

  Cette valeur détermine la version compatible avec laquelle AWS Glue cette transformation de machine learning est compatible. Glue 1.0 est recommandé pour la plupart des clients. Si la valeur n'est pas définie, la compatibilité Glue par défaut est Glue 0.9. Pour plus d'informations, veuillez consulter la rubrique Versions AWS Glue dans le guide du développeur.

- `MaxCapacity` – Nombre (double).

  Nombre d'unités de traitement de AWS Glue données (DPUs) allouées aux exécutions de tâches pour cette transformation. Vous pouvez allouer de 2 à 100 DPUs ; la valeur par défaut est 10. Un DPU est une mesure relative de la puissance de traitement composée de 4 V de capacité CPUs de calcul et de 16 Go de mémoire. Pour plus d'informations, consultez la page de tarification AWS Glue.

  Lorsque le champ `WorkerType` est défini sur une valeur autre que `Standard`, le champ `MaxCapacity` est défini automatiquement et passe en lecture seule.

- `WorkerType` – Chaîne UTF-8 (valeurs valides: `Standard=""` | `G.1X=""` | `G.2X=""` | `G.025X=""` | `G.4X=""` | `G.8X=""` | `Z.2X=""`).

  Type de travail prédéfini qui est alloué lorsqu'une tâche est exécutée. Accepte la valeur Standard, G.1X ou G.2X.

  - Pour le type de travail `Standard`, chaque travail fournit 4 vCPU, 16 Go de mémoire et 50 Go de disque, ainsi que 2 exécuteurs par travail.
  - Pour le type de travail `G.1X`, chaque travail fournit 4 vCPU, 16 Go de mémoire et 64 Go de disque, ainsi que 1 exécuteur par travail.
  - Pour le type de travail `G.2X`, chaque travail fournit 8 vCPU, 32 Go de mémoire et 128 Go de disque, ainsi que 1 exécuteur par travail.
- `NumberOfWorkers` : nombre (entier).

  Nombre de travaux d'un `workerType` défini qui sont attribués lorsqu'une tâche est exécutée.
- `Timeout` – Nombre (entier), au moins égal à 1.

  Délai d'attente (en minutes) d'une tâche exécutée pour cette transformation. Durée maximale pendant laquelle l'exécution d'une tâche pour cette transformation peut utiliser des ressources avant qu'elle ne soit mise hors service et passe à l'état `TIMEOUT`. La valeur par défaut est de 2 880 minutes (48 heures).
- `MaxRetries` : nombre (entier).

  Nombre maximal de nouvelles tentatives d'une tâche de cette transformation après l'échec de l'exécution d'une tâche.

Réponse

- `TransformId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant unique de la transformation qui a été mise à jour.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`

- `OperationTimeoutException`
- `InternalServiceException`
- `AccessDeniedException`

MLTransform Action de suppression (Python : delete_ml_transform)

Supprime toute transformation AWS Glue de machine learning. Les transformations Machine Learning sont un type spécial de transformation qui utilisent le Machine Learning pour découvrir les détails de la transformation à exécuter par l'apprentissage à partir d'exemples fournis par les humains. Ces transformations sont ensuite enregistrées par AWS Glue. Si vous n'avez plus besoin d'une transformation, vous pouvez la supprimer en appelant `DeleteMLTransforms`. Cependant, les AWS Glue tâches qui font toujours référence à la transformation supprimée ne seront plus couronnées de succès.

Demande

- `TransformId` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant unique de la transformation à supprimer.

Réponse

- `TransformId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant unique de la transformation qui a été supprimée.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

Obtenir une MLTransform action (Python : get_ml_transform)

Récupère un artefact de transformation basé sur le AWS Glue machine learning et toutes les métadonnées correspondantes. Les transformations Machine Learning sont un type spécial de transformation qui utilisent le Machine Learning pour découvrir les détails de la transformation à exécuter par l'apprentissage à partir d'exemples fournis par les humains. Ces transformations sont ensuite enregistrées par AWS Glue. Vous pouvez récupérer leurs métadonnées en appelant `GetMLTransform`.

Demande

- `TransformId` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant unique de la transformation, générée à l'heure à laquelle la transformation a été créée.

Réponse

- `TransformId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant unique de la transformation, générée à l'heure à laquelle la transformation a été créée.
- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom unique donné à la transformation lors de sa création.
- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description de la transformation.
- `Status` – Chaîne UTF-8 (valeurs valides : `NOT_READY` | `READY` | `DELETING`).

  Dernier état connu de la transformation (pour indiquer si elle peut être utilisée ou non). L'un des états « NOT_READY », « READY » ou « DELETING ».
- `CreatedOn` – Horodatage.

  Date et heure de création de la transformation.
- `LastModifiedOn` – Horodatage.

Date et heure de la dernière modification de la transformation.

- `InputRecordTables` – Un tableau d'objets GlueTable, 10 structures au maximum.

  Liste des définitions de AWS Glue table utilisées par la transformation.

- `Parameters` – Un objet TransformParameters.

  Paramètres de configuration spécifiques à l'algorithme utilisé.

- `EvaluationMetrics` : un objet EvaluationMetrics.

  Dernières métriques d'évaluation.

- `LabelCount` : nombre (entier).

  Nombre d'étiquettes disponibles pour cette transformation.

- `Schema` – Un tableau d'objets SchemaColumn, 100 structures au maximum.

  Objet `Map<Column, Type>` qui représente le schéma que cette transformation accepte. Comporte une limite supérieure de 100 colonnes.

- `Role` – Chaîne UTF-8.

  Nom ou Amazon Resource Name (ARN) du rôle IAM avec les autorisations requises.

- `GlueVersion` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #48.

  Cette valeur détermine la version compatible avec laquelle AWS Glue cette transformation de machine learning est compatible. Glue 1.0 est recommandé pour la plupart des clients. Si la valeur n'est pas définie, la compatibilité Glue par défaut est Glue 0.9. Pour plus d'informations, veuillez consulter la rubrique Versions AWS Glue dans le guide du développeur.

- `MaxCapacity` – Nombre (double).

  Nombre d'unités de traitement de AWS Glue données (DPUs) allouées aux exécutions de tâches pour cette transformation. Vous pouvez allouer de 2 à 100 DPUs ; la valeur par défaut est 10. Un DPU est une mesure relative de la puissance de traitement composée de 4 V de capacité CPUs de calcul et de 16 Go de mémoire. Pour plus d'informations, consultez la page de tarification AWS Glue.

  Lorsque le champ `WorkerType` est défini sur une valeur autre que `Standard`, le champ `MaxCapacity` est défini automatiquement et passe en lecture seule.

- `WorkerType` – Chaîne UTF-8 (valeurs valides: `Standard=""` | `G.1X=""` | `G.2X=""` | `G.025X=""` | `G.4X=""` | `G.8X=""` | `Z.2X=""`).

  Type de travail prédéfini qui est alloué lorsqu'une tâche est exécutée. Accepte la valeur Standard, G.1X ou G.2X.

  - Pour le type de travail `Standard`, chaque travail fournit 4 vCPU, 16 Go de mémoire et 50 Go de disque, ainsi que 2 exécuteurs par travail.
  - Pour le type de travail `G.1X`, chaque travail fournit 4 vCPU, 16 Go de mémoire et 64 Go de disque, ainsi que 1 exécuteur par travail.
  - Pour le type de travail `G.2X`, chaque travail fournit 8 vCPU, 32 Go de mémoire et 128 Go de disque, ainsi que 1 exécuteur par travail.
- `NumberOfWorkers` : nombre (entier).

  Nombre de travaux d'un `workerType` défini qui sont attribués lorsqu'une tâche est exécutée.
- `Timeout` – Nombre (entier), au moins égal à 1.

  Délai d'attente (en minutes) d'une tâche exécutée pour cette transformation. Durée maximale pendant laquelle l'exécution d'une tâche pour cette transformation peut utiliser des ressources avant qu'elle ne soit mise hors service et passe à l'état `TIMEOUT`. La valeur par défaut est de 2 880 minutes (48 heures).
- `MaxRetries` : nombre (entier).

  Nombre maximal de nouvelles tentatives d'une tâche de cette transformation après l'échec de l'exécution d'une tâche.
- `TransformEncryption` – Un objet TransformEncryption.

  Les encryption-at-rest paramètres de la transformation qui s'appliquent à l'accès aux données utilisateur. Les transformations de machine learning peuvent accéder aux données utilisateur chiffrées dans Amazon S3 à l'aide de KMS.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

Obtenir une MLTransforms action (Python : get_ml_transform)

Obtient une liste triable et filtrable des transformations d'apprentissage AWS Glue automatique existantes. Les transformations Machine Learning sont un type spécial de transformation qui utilisent le Machine Learning pour découvrir les détails de la transformation à exécuter par l'apprentissage à partir d'exemples fournis par les humains. Ces transformations sont ensuite enregistrées par AWS Glue, et vous pouvez récupérer leurs métadonnées en appelant`GetMLTransforms`.

Demande

- `NextToken` – Chaîne UTF-8.

  Jeton de pagination pour décaler les résultats.
- `MaxResults` – Nombre (entier), compris entre 1 et 1 000.

  Nombre maximal de résultats à renvoyer.
- `Filter` – Un objet TransformFilterCriteria.

  Critères de transformation du filtre.
- `Sort` – Un objet TransformSortCriteria.

  Critères de tri.

Réponse

- `Transforms` – Obligatoire : un tableau d'objets MLTransform.

  Liste de transformations du Machine Learning.
- `NextToken` – Chaîne UTF-8.

  Un jeton de pagination, si d'autres résultats sont disponibles.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

MLTransforms Action de liste (Python : list_ml_transform)

Récupère une liste triable et filtrable des transformations de AWS Glue machine learning existantes dans ce AWS compte, ou des ressources avec le tag spécifié. Cette opération accepte le champ `Tags` facultatif que vous pouvez utiliser comme filtre sur les réponses, afin que les ressources balisées puissent être récupérées en tant que groupe. Si vous choisissez d'utiliser le filtrage des balises, seules les ressources avec les balises sont récupérées.

Demande

- `NextToken` – Chaîne UTF-8.

  Jeton de continuation, s'il s'agit d'une requête de continuation.
- `MaxResults` – Nombre (entier), compris entre 1 et 1 000.

  La taille maximale d'une liste à renvoyer.
- `Filter` – Un objet TransformFilterCriteria.

  Élément `TransformFilterCriteria` utilisé pour filtrer les transformations de Machine Learning.
- `Sort` – Un objet TransformSortCriteria.

  Élément `TransformSortCriteria` utilisé pour trier les transformations de Machine Learning.
- `Tags` – Tableau de mappage de paires clé-valeur, avec 50 paires au maximum.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 256 octets.

  Spécifie de renvoyer uniquement les ressources balisées.

Réponse

- `TransformIds` – Obligatoire : Tableau de chaînes UTF-8.

  Identifiants de toutes les transformations de Machine Learning dans le compte, ou les transformations de Machine Learning avec les balises spécifiées.
- `NextToken` – Chaîne UTF-8.

  Jeton continuation, si la liste renvoyée ne contient pas la dernière métrique disponible.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

Lancer MLEvaluation TaskRun l'action (Python : start_ml_evaluation_task_run)

Démarre une tâche pour estimer la qualité de la transformation.

Lorsque vous fournissez des ensembles d'étiquettes comme exemples de vérité, l'apprentissage AWS Glue automatique utilise certains de ces exemples pour en tirer des leçons. Le reste des étiquettes est utilisé en tant que test pour estimer la qualité.

Retourne un identifiant unique pour l'exécution. Vous pouvez appeler `GetMLTaskRun` pour obtenir plus d'informations sur les statistiques de `EvaluationTaskRun`.

Demande

- `TransformId` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant unique de la transformation du Machine Learning.

Réponse

- `TaskRunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant unique associé à cette exécution.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`

- `OperationTimeoutException`
- `InternalServiceException`
- `ConcurrentRunsExceededException`
- `MLTransformNotReadyException`

Lancer MLLabeling SetGenerationTaskRun l'action (Python : start_ml_labeling_set_generation_task_run)

Démarre le flux de travail de l'apprentissage actif d'apprentissage automatique pour votre transformation de Machine Learning afin d'améliorer la qualité de la transformation en générant des ensembles d'étiquette et en ajoutant des étiquettes.

Quand `StartMLLabelingSetGenerationTaskRun` a pris fin, AWS Glue aura généré un « jeu d'étiquetage » ou un ensemble de questions auxquelles les humains devront répondre.

Dans le cas de la transformation `FindMatches`, ces questions prennent la forme : « Quel est le bon moyen de regrouper conjointement ces lignes en groupes composés d'enregistrements correspondants ? »

Une fois que le processus d'étiquetage est terminé, vous pouvez charger vos étiquettes avec un appel à `StartImportLabelsTaskRun`. Une fois `StartImportLabelsTaskRun` terminé, toutes les futures exécutions de la transformation du Machine Learning utiliseront les nouvelles étiquettes améliorées et effectueront une transformation de meilleure qualité.

Remarque : Le rôle utilisé pour écrire le jeu d'étiquetage généré sur le `OutputS3Path` est le rôle associé à la transformation du machine learning, spécifié dans l'API `CreateMLTransform`.

Demande

- `TransformId` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant unique de la transformation du Machine Learning.
- `OutputS3Path` – Obligatoire : chaîne UTF-8.

  Chemin Amazon Simple Storage Service (Amazon S3) sur lequel vous générez l'ensemble d'étiquetage.

Réponse

- `TaskRunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant d'exécution unique associé à cette exécution de tâche.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`
- `ConcurrentRunsExceededException`

Action Get MLTask Run (Python : get_ml_task_run)

Permet d'obtenir les détails d'une exécution de tâche spécifique sur une transformation du Machine Learning. Les exécutions de tâches d'apprentissage automatique sont des tâches asynchrones exécutées AWS Glue en votre nom dans le cadre de différents flux de travail d'apprentissage automatique. Vous pouvez vérifier les statistiques de toutes les tâches exécutées en appelant `GetMLTaskRun` avec le `TaskRunID` et le `TransformID` de la transformation parent.

Demande

- `TransformId` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant unique de la transformation du Machine Learning.
- `TaskRunId` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant unique de la tâche exécutée.

Réponse

- `TransformId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant unique de la tâche exécutée.

- `TaskRunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant d'exécution unique associé à cette exécution.

- `Status` – Chaîne UTF-8 (valeurs valides : `RUNNING` | `FINISHED` | `FAILED` | `PENDING_EXECUTION` | `TIMED_OUT` | `CANCELING` | `CANCELED` | `RECEIVED_BY_TASKRUNNER`).

  Statut de la tâche exécutée.

- `LogGroupName` – Chaîne UTF-8.

  Noms des groupes de journaux associés à la tâche exécutée.

- `Properties` – Un objet TaskRunProperties.

  Liste des propriétés associés à la tâche exécutée.

- `ErrorString` – Chaîne UTF-8.

  Chaînes d'erreur associées à la tâche exécutée.

- `StartedOn` – Horodatage.

  Date et heure de l'exécution de cette tâche.

- `LastModifiedOn` – Horodatage.

  Date et heure de la dernière modification de cette exécution de tâche.

- `CompletedOn` – Horodatage.

  Date et heure de fin de la dernière exécution de cette tâche.

- `ExecutionTime` : nombre (entier).

  Durée (en secondes) pendant laquelle la tâche exécutée a consommé des ressources.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

Action Get MLTask Runs (Python : get_ml_task_runs)

Permet d'obtenir la liste des exécutions d'une transformation du Machine Learning. Les exécutions de tâches d'apprentissage automatique sont des tâches asynchrones exécutées AWS Glue en votre nom dans le cadre de différents flux de travail d'apprentissage automatique. Vous pouvez obtenir une liste triable et filtrable des exécutions de tâche du Machine Learning en appelant `GetMLTaskRuns` avec le `TransformID` de sa transformation parent et d'autres paramètres facultatifs, comme décrit dans cette section.

Cette opération renvoie la liste des exécutions historiques et doit être paginée.

Demande

- `TransformId` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant unique de la transformation du Machine Learning.
- `NextToken` – Chaîne UTF-8.

  Jeton pour la pagination des résultats. La valeur par défaut est vide.
- `MaxResults` – Nombre (entier), compris entre 1 et 1 000.

  Nombre maximal de résultats à renvoyer.
- `Filter` – Un objet TaskRunFilterCriteria.

  Critères de filtre, dans la structure `TaskRunFilterCriteria`, pour la tâche exécutée.
- `Sort` – Un objet TaskRunSortCriteria.

  Critères de tri, dans la structure `TaskRunSortCriteria`, pour la tâche exécutée.

Réponse

- `TaskRuns` : un tableau d'objets TaskRun.

  Liste des exécutions de tâches associées à la transformation.

- `NextToken` – Chaîne UTF-8.

  Un jeton de pagination, si d'autres résultats sont disponibles.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

Annuler l'action MLTask Exécuter (Python : cancel_ml_task_run)

Annule (arrête) une exécution de tâche. Les exécutions de tâches d'apprentissage automatique sont des tâches asynchrones exécutées AWS Glue en votre nom dans le cadre de différents flux de travail d'apprentissage automatique. Vous pouvez annuler une exécution de tâche du Machine Learning à tout moment en appelant `CancelMLTaskRun` avec le `TransformID` d'une transformation parent de la tâche exécutée et le `TaskRunId` de la tâche exécutée.

Demande

- `TransformId` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant unique de la transformation du Machine Learning.

- `TaskRunId` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant unique de la tâche exécutée.

Réponse

- `TransformId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant unique de la transformation du Machine Learning.

- `TaskRunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant unique de l'exécution de la tâche.

- `Status` – Chaîne UTF-8 (valeurs valides : `RUNNING` | `FINISHED` | `FAILED` | `PENDING_EXECUTION` | `TIMED_OUT` | `CANCELING` | `CANCELED` | `RECEIVED_BY_TASKRUNNER`).

  Statut de cette exécution.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

StartExportLabelsTaskRun action (Python : start_export_labels_task_run)

Commence une tâche asynchrone pour exporter toutes les données étiquetées pour une transformation particulière. Cette tâche est le seul appel d'API lié à l'étiquetage qui ne fasse pas partie du flux de travail d'apprentissage actif classique. Vous utilisez généralement `StartExportLabelsTaskRun` lorsque vous souhaitez utiliser tous vos étiquettes existantes en même temps, par exemple lorsque vous souhaitez supprimer ou modifier les étiquettes qui ont été soumises précédemment comme vraies. Cette opération d'API accepte le `TransformId` dont vous souhaitez exporter les étiquettes et un chemin Amazon Simple Storage Service (Amazon S3) vers lequel exporter les étiquettes. L'opération renvoie un `TaskRunId`. Vous pouvez vérifier le statut de votre tâche d'exécuter en appelant l'API `GetMLTaskRun`.

Demande

- `TransformId` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant unique de la transformation du Machine Learning.
- `OutputS3Path` – Obligatoire : chaîne UTF-8.

  Chemin Amazon S3 où vous exportez les étiquettes.

Réponse

- `TaskRunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant unique de l'exécution de la tâche.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

StartImportLabelsTaskRun action (Python : start_import_labels_task_run)

Vous permet de fournir des étiquettes supplémentaires (exemples de vérité) à utiliser afin d'enseigner la transformation du Machine Learning et d'améliorer sa qualité. Cette opération d'API est généralement utilisée dans le cadre du processus d'apprentissage active qui commence par l'appel `StartMLLabelingSetGenerationTaskRun` et qui se traduit finalement par l'amélioration de la qualité de votre transformation de Machine Learning.

Une fois `StartMLLabelingSetGenerationTaskRun` terminé, le machine learning AWS Glue aura généré une série de questions pour que les humains y répondent. (La réponse à ces questions est souvent appelé « étiquette » dans les flux de travail du Machine Learning). Dans le cas de la transformation `FindMatches`, ces questions prennent la forme : « Quel est le bon moyen de regrouper conjointement ces lignes en groupes composés d'enregistrements correspondants ? »

Une fois le processus d'étiquetage terminé, les utilisateurs les téléchargent answers/labels en appelant`StartImportLabelsTaskRun`. Une fois `StartImportLabelsTaskRun` terminé, toutes les exécutions futures de la transformation du Machine Learning utilisent les nouvelles étiquettes améliorées et effectuent une transformation de meilleure qualité.

Par défaut, `StartMLLabelingSetGenerationTaskRun` apprend en permanence et associe toutes les étiquettes que vous chargez, sauf si vous définissez `Replace` sur true. Si vous définissez `Replace` sur true, `StartImportLabelsTaskRun` supprime et oublie toutes les étiquettes précédemment chargées et apprend uniquement à partir de l'ensemble exact que vous chargez. Le remplacement des étiquettes peuvent être utiles si vous vous rendez compte que vous avez précédemment téléchargé les étiquettes incorrectes et que vous pensez qu'elles ont une incidence négative sur la qualité de votre transformation.

Vous pouvez vérifier le statut de votre exécution de tâche en appelant l'opération `GetMLTaskRun`.

Demande

- `TransformId` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant unique de la transformation du Machine Learning.
- `InputS3Path` – Obligatoire : chaîne UTF-8.

  Chemin Amazon Simple Storage Service (Amazon S3) à partir duquel vous importez les étiquettes.
- `ReplaceAllLabels` : booléen.

  Indique si vous souhaitez remplacer vos étiquettes existantes.

Réponse

- `TaskRunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant unique de l'exécution de la tâche.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`

- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`
- `InternalServiceException`

API Qualité des données

L'API Qualité des données décrit les types de données de qualité et inclut l'API permettant de créer, supprimer ou de mettre à jour des ensembles de règles, exécutions et évaluations.

Types de données

- DataSource structure
- DataQualityRulesetListDetails structure
- DataQualityTargetTable structure
- DataQualityRulesetEvaluationRunDescription structure
- DataQualityRulesetEvaluationRunFilter structure
- DataQualityEvaluationRunAdditionalRunOptions structure
- DataQualityRuleRecommendationRunDescription structure
- DataQualityRuleRecommendationRunFilter structure
- DataQualityResult structure
- DataQualityAnalyzerResult structure
- DataQualityObservation structure
- MetricBasedObservation structure
- DataQualityMetricValues structure
- DataQualityRuleResult structure
- DataQualityResultDescription structure
- DataQualityResultFilterCriteria structure
- DataQualityRulesetFilterCriteria structure
- DataQualityAggregatedMetrics structure
- StatisticAnnotation structure
- TimestampedInclusionAnnotation structure
- AnnotationError structure

- DatapointInclusionAnnotation structure
- StatisticSummaryList liste
- StatisticSummary structure
- RunIdentifier structure
- StatisticModelResult structure
- DataQualityGlueTable structure

DataSource structure

Source de données (AWS Glue table) pour laquelle vous souhaitez obtenir des résultats de qualité.

Champs

- `GlueTable` – Un objet GlueTable.

  Et une AWS Glue table.
- `DataQualityGlueTable` – Un objet DataQualityGlueTable.

  Un AWS Glue tableau pour les opérations relatives à la qualité des données.

DataQualityRulesetListDetails structure

Décrit un ensemble de règles de qualité des données renvoyé par `GetDataQualityRuleset`.

Champs

- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de l'ensemble de règles de qualité des données.
- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description de l'ensemble de règles de qualité des données.
- `CreatedOn` – Horodatage.

  Date et heure de création de l'ensemble de règles de qualité des données.

- `LastModifiedOn` – Horodatage.

  Date et heure de la dernière modification de l'ensemble de règles de qualité des données.

- `TargetTable` – Un objet DataQualityTargetTable.

  Objet représentant une AWS Glue table.

- `RecommendationRunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Lors de la création d'un ensemble de règles à partir d'une exécution de recommandation, cet ID d'exécution est généré pour relier les deux.

- `RuleCount` : nombre (entier).

  Nombre de règles dans l'ensemble de règles.

DataQualityTargetTable structure

Objet représentant une AWS Glue table.

Champs

- `TableName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Le nom de la AWS Glue table.

- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la base de données dans laquelle se trouve la AWS Glue table.

- `CatalogId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  L'identifiant du catalogue où se trouve la AWS Glue table.

DataQualityRulesetEvaluationRunDescription structure

Décrit le résultat d'une exécution d'évaluation d'un ensemble de règles de qualité des données.

Champs

- `RunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant d'exécution unique associé à cette exécution.

- `Status` – Chaîne UTF-8 (valeurs valides : `RUNNING` | `FINISHED` | `FAILED` | `PENDING_EXECUTION` | `TIMED_OUT` | `CANCELING` | `CANCELED` | `RECEIVED_BY_TASKRUNNER`).

  Statut de cette exécution.

- `StartedOn` – Horodatage.

  Date et heure de lancement de l'exécution.

- `DataSource` – Un objet DataSource.

  La source de données (une AWS Glue table) associée à l'exécution.

DataQualityRulesetEvaluationRunFilter structure

Critères de filtrage.

Champs

- `DataSource` – Obligatoire : un objet DataSource.

  Filtrez en fonction d'une source de données (une AWS Glue table) associée à l'exécution.

- `StartedBefore` – Horodatage.

  Filtrez les résultats en fonction des exécutions qui ont débuté avant cette heure.

- `StartedAfter` – Horodatage.

  Filtrez les résultats en fonction des exécutions qui ont débuté après cette heure.

DataQualityEvaluationRunAdditionalRunOptions structure

Options d'exécution supplémentaires que vous pouvez spécifier pour une exécution d'évaluation.

Champs

- `CloudWatchMetricsEnabled` : booléen.

  Activer ou non les CloudWatch métriques.
- `ResultsS3Prefix` – Chaîne UTF-8.

  Préfixe permettant à Amazon S3 de stocker les résultats.
- `CompositeRuleEvaluationMethod` – Chaîne UTF-8 (valeurs valides : `COLUMN` | `ROW`).

  Définissez la méthode d'évaluation des règles composites du jeu de règles sur ROW/COLUMN.

DataQualityRuleRecommendationRunDescription structure

Décrit le résultat de l'exécution d'une recommandation de règle de qualité des données.

Champs

- `RunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant d'exécution unique associé à cette exécution.
- `Status` – Chaîne UTF-8 (valeurs valides : `RUNNING` | `FINISHED` | `FAILED` | `PENDING_EXECUTION` | `TIMED_OUT` | `CANCELING` | `CANCELED` | `RECEIVED_BY_TASKRUNNER`).

  Statut de cette exécution.
- `StartedOn` – Horodatage.

  Date et heure de lancement de cette exécution.
- `DataSource` – Un objet DataSource.

  Source de données (AWS Glue table) associée à l'exécution de la recommandation.

DataQualityRuleRecommendationRunFilter structure

Filtre permettant de répertorier les exécutions de recommandations relatives à la qualité des données.

Champs

- `DataSource` – Obligatoire : un objet DataSource.

  Filtrez en fonction d'une source de données spécifiée (AWS Glue table).
- `StartedBefore` – Horodatage.

  Filtrez en fonction de l'heure de début des résultats avant l'heure indiquée.
- `StartedAfter` – Horodatage.

  Filtrez en fonction de l'heure de début des résultats après l'heure indiquée.

DataQualityResult structure

Décrit un résultat sur la qualité des données.

Champs

- `ResultId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID de résultat unique pour le résultat en matière de qualité des données.
- `ProfileId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  L'ID de profil pour le résultat en matière de qualité des données.
- `Score` – Nombre (double), au plus égal à 1,0.

  Score de qualité des données agrégées. Représente le rapport entre le nombre de règles transmises et le nombre total de règles.
- `DataSource` – Un objet DataSource.

  Table associée au résultat sur la qualité des données, le cas échéant.
- `RulesetName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de l'ensemble de règles associé au résultat sur la qualité des données.
- `EvaluationContext` – Chaîne UTF-8.

Dans le contexte d'une tâche dans AWS Glue Studio, chaque nœud du canevas se voit généralement attribuer un nom et les nœuds de qualité des données porteront un nom. Dans le cas de plusieurs nœuds, `evaluationContext` peut distinguer les nœuds.

- `StartedOn` – Horodatage.

  Date et heure du début de cette exécution de l'évaluation de la qualité des données.

- `CompletedOn` – Horodatage.

  Date et heure de fin de la dernière exécution de cette évaluation de la qualité des données.

- `JobName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Le nom de tâche associé au résultat sur la qualité des données, le cas échéant.

- `JobRunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID d'exécution de la tâche associé au résultat sur la qualité des données, le cas échéant.

- `RulesetEvaluationRunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID d'exécution unique de l'évaluation de l'ensemble de règles pour ce résultat sur la qualité des données.

- `RuleResults` – Un tableau d'objets DataQualityRuleResult, 2 000 structures au maximum.

  Liste d'objets `DataQualityRuleResult` représentant les résultats de chaque règle.

- `AnalyzerResults` – Un tableau d'objets DataQualityAnalyzerResult, 2 000 structures au maximum.

  Liste d'objets `DataQualityAnalyzerResult` représentant les résultats de chaque analyseur.

- `Observations` – Un tableau d'objets DataQualityObservation, 50 structures maximum.

  Liste d'objets `DataQualityObservation` représentant les observations générées après évaluation des règles et des analyseurs.

- `AggregatedMetrics` – Un objet DataQualityAggregatedMetrics.

Récapitulatif des `DataQualityAggregatedMetrics` objets indiquant le nombre total de lignes et de règles traitées, y compris leurs pass/fail statistiques basées sur les résultats au niveau des lignes.

DataQualityAnalyzerResult structure

Décrit le résultat de l'évaluation d'un analyseur de qualité des données.

Champs

- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de l'analyseur de qualité des données.

- `Description` – Chaîne UTF-8, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description de l'analyseur de qualité des données.

- `EvaluationMessage` – Chaîne UTF-8, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Message d'évaluation.

- `EvaluatedMetrics` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaque valeur est un nombre (double).

  Carte des métriques associées à l'évaluation de l'analyseur.

DataQualityObservation structure

Décrit l'observation générée après l'évaluation des règles et des analyseurs.

Champs

- `Description` – Chaîne UTF-8, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description de l'observation de la qualité des données.

- `MetricBasedObservation` – Un objet MetricBasedObservation.

  Objet de type `MetricBasedObservation` représentant l'observation basée sur des métriques de qualité des données évaluées.

MetricBasedObservation structure

Décrit l'observation basée sur les métriques générée sur la base des métriques de qualité des données évaluées.

Champs

- `MetricName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la métrique de qualité des données utilisée pour générer l'observation.

- `StatisticId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  L'ID de statistique.

- `MetricValues` – Un objet DataQualityMetricValues.

  Objet de type `DataQualityMetricValues` représentant l'analyse de la valeur de métrique de qualité des données.

- `NewRules` – Tableau de chaînes UTF-8.

  Liste des nouvelles règles de qualité des données générées dans le cadre de l'observation basée sur la valeur de métrique de qualité des données.

DataQualityMetricValues structure

Décrit la valeur de la métrique de qualité des données en fonction de l'analyse des données historiques.

Champs

- `ActualValue` – Nombre (double).

  Valeur réelle de la métrique de qualité des données.

- `ExpectedValue` – Nombre (double).

  Valeur attendue de la métrique de qualité des données selon l'analyse des données historiques.

- `LowerLimit` – Nombre (double).

  Limite inférieure de la valeur de métrique de qualité des données selon l'analyse des données historiques.

- `UpperLimit` – Nombre (double).

  Limite supérieure de la valeur de métrique de qualité des données selon l'analyse des données historiques.

DataQualityRuleResult structure

Décrit le résultat de l'évaluation d'une règle de qualité des données.

Champs

- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la règle de qualité des données.

- `Description` – Chaîne UTF-8, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description de la règle de qualité des données.

- `EvaluationMessage` – Chaîne UTF-8, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

Message d'évaluation.

- `Result` – Chaîne UTF-8 (valeurs valides : `PASS` | `FAIL` | `ERROR`).

  État de réussite ou d'échec de la règle.

- `EvaluatedMetrics` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaque valeur est un nombre (double).

  Une carte des métriques associées à l'évaluation de la règle.

- `EvaluatedRule` – Chaîne UTF-8, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Règle évaluée.

- `RuleMetrics` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaque valeur est un nombre (double).

  Carte contenant les métriques associées à l'évaluation de la règle en fonction des résultats au niveau des lignes.

DataQualityResultDescription structure

Décrit un résultat sur la qualité des données.

Champs

- `ResultId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID de résultat unique pour ce résultat sur la qualité des données.

- `DataSource` – Un objet DataSource.

  Nom de la table associée au résultat sur la qualité des données.

- `JobName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la tâche associée au résultat en matière de qualité des données.

- `JobRunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID d'exécution de la tâche associé au résultat en matière de qualité des données.

- `StartedOn` – Horodatage.

  Heure du début de l'exécution pour ce résultat en matière de qualité des données.

DataQualityResultFilterCriteria structure

Critères permettant de renvoyer des résultats en matière de qualité des données.

Champs

- `DataSource` – Un objet DataSource.

  Filtrez les résultats en fonction de la source de données spécifiée. Par exemple, récupérer tous les résultats d'une AWS Glue table.

- `JobName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Filtrez les résultats en fonction du nom de la tâche spécifiée.

- `JobRunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Filtrez les résultats en fonction de l'ID d'exécution de la tâche spécifiée.

- `StartedAfter` – Horodatage.

  Filtrez les résultats en fonction des exécutions qui ont débuté après cette heure.

- `StartedBefore` – Horodatage.

  Filtrez les résultats en fonction des exécutions qui ont débuté avant cette heure.

DataQualityRulesetFilterCriteria structure

Critères permettant de filtrer les ensembles de règles de qualité des données.

Champs

- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom des critères de filtrage de l'ensemble de règles.
- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description des critères de filtrage de l'ensemble de règles.
- `CreatedBefore` – Horodatage.

  Filtre sur les ensembles de règles créés avant cette date.
- `CreatedAfter` – Horodatage.

  Filtre sur les ensembles de règles créés après cette date.
- `LastModifiedBefore` – Horodatage.

  Filtre sur les derniers ensembles de règles modifiés avant cette date.
- `LastModifiedAfter` – Horodatage.

  Filtre sur les derniers ensembles de règles modifiés après cette date.
- `TargetTable` – Un objet DataQualityTargetTable.

  Nom et nom de la base de données de la table cible.

DataQualityAggregatedMetrics structure

Résumé des mesures indiquant le nombre total de lignes et de règles traitées, y compris leurs pass/fail statistiques basées sur les résultats au niveau des lignes.

Champs

- `TotalRowsProcessed` – Nombre (double).

Nombre total de lignes traitées lors de l'évaluation de la qualité des données.

- `TotalRowsPassed` – Nombre (double).

  Nombre total de lignes ayant satisfait à toutes les règles de qualité des données applicables.

- `TotalRowsFailed` – Nombre (double).

  Nombre total de lignes n'ayant pas respecté une ou plusieurs règles de qualité des données.

- `TotalRulesProcessed` – Nombre (double).

  Nombre total de règles de qualité des données qui ont été évaluées.

- `TotalRulesPassed` – Nombre (double).

  Nombre total de règles de qualité des données qui ont satisfait à leurs critères d'évaluation.

- `TotalRulesFailed` – Nombre (double).

  Nombre total de règles de qualité des données qui n'ont pas satisfait à leurs critères d'évaluation.

StatisticAnnotation structure

Annotation statistique.

Champs

- `ProfileId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  L'ID de profil.

- `StatisticId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  L'ID de statistique.

- `StatisticRecorded0n` – Horodatage.

  Horodatage de l'enregistrement de la statistique annotée.

- `InclusionAnnotation` – Un objet TimestampedInclusionAnnotation.

  Annotation d'inclusion appliquée à la statistique.

TimestampedInclusionAnnotation structure

Annotation d'inclusion horodatée.

Champs

- `Value` – Chaîne UTF-8 (valeurs valides : `INCLUDE` | `EXCLUDE`).

  Valeur de l'annotation d'inclusion.
- `LastModifiedOn` – Horodatage.

  Horodatage de la dernière modification de l'annotation d'inclusion.

AnnotationError structure

Annotation qui a échoué.

Champs

- `ProfileId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID de profil de l'annotation qui a échoué.
- `StatisticId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID de statistique de l'annotation qui a échoué.
- `FailureReason` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Raison pour laquelle l'annotation a échoué.

DatapointInclusionAnnotation structure

Annotation d'inclusion.

Champs

- `ProfileId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

ID du profil de qualité des données auquel appartient la statistique.

- `StatisticId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  L'ID de statistique.

- `InclusionAnnotation` – Chaîne UTF-8 (valeurs valides : `INCLUDE` | `EXCLUDE`).

  Valeur d'annotation d'inclusion à appliquer à la statistique.

StatisticSummaryList liste

Une liste de `StatisticSummary`.

Tableau d'objets StatisticSummary.

Une liste de `StatisticSummary`.

StatisticSummary structure

Informations récapitulatives relatives à une statistique.

Champs

- `StatisticId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  L'ID de statistique.

- `ProfileId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  L'ID de profil.

- `RunIdentifier` – Un objet RunIdentifier.

  Identifiant de l'exécution.

- `StatisticName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Custom string pattern #16.

  Nom de la statistique.

- `DoubleValue` – Nombre (double).

  Valeur de la statistique.

- `EvaluationLevel` – Chaîne UTF-8 (valeurs valides : `Dataset="DATASET"` | `Column="COLUMN"` | `Multicolumn="MULTICOLUMN"`).

  Niveau d'évaluation de la statistique. Valeurs possibles :`Dataset`,`Column`,`Multicolumn`.

- `ColumnsReferenced` – Tableau de chaînes UTF-8.

  Liste des colonnes référencées par la statistique.

- `ReferencedDatasets` – Tableau de chaînes UTF-8.

  Liste des jeux de données référencés par la statistique.

- `StatisticProperties` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaque valeur est une chaîne Description, d'une longueur ne dépassant pas 2048 octets, correspondant au URI address multi-line string pattern.

  `StatisticPropertiesMap`, qui contient des valeurs `NameString` et `DescriptionString`.

- `RecordedOn` – Horodatage.

  Horodatage de l'enregistrement de la statistique.

- `InclusionAnnotation` – Un objet TimestampedInclusionAnnotation.

  Annotation d'inclusion pour la statistique.

RunIdentifier structure

Identifiant d'exécution.

Champs

- `RunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID de l'exécution.

- `JobRunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID de l'exécution de la tâche.

StatisticModelResult structure

Résultat du modèle statistique.

Champs

- `LowerBound` – Nombre (double).

  Limite inférieure.
- `UpperBound` – Nombre (double).

  Limite supérieure.
- `PredictedValue` – Nombre (double).

  Valeur prédite.
- `ActualValue` – Nombre (double).

  Valeur réelle.
- `Date` – Horodatage.

  Date.
- `InclusionAnnotation` – Chaîne UTF-8 (valeurs valides : `INCLUDE` | `EXCLUDE`).

  Annotation d'inclusion.

DataQualityGlueTable structure

Base de données et table utilisées pour AWS Glue Data Catalog les données d'entrée ou de sortie pour les opérations de qualité des données.

Champs

- `DatabaseName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

Nom de la base de données dans le AWS Glue Data Catalog.

- `TableName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la table dans AWS Glue Data Catalog.

- `CatalogId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant unique pour AWS Glue Data Catalog.

- `ConnectionName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la connexion à AWS Glue Data Catalog.

- `AdditionalOptions` – Tableau de mappage de paires valeur-clé, pas moins de 1 ou plus de 10 paires..

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaque valeur est une chaîne Description, d'une longueur ne dépassant pas 2048 octets, correspondant au URI address multi-line string pattern.

  Options supplémentaires pour la table. Actuellement, deux clés sont prises en charge :

  - `pushDownPredicate` : pour filtrer les partitions sans avoir à répertorier ni lire tous les fichiers de votre jeu de données.
  - `catalogPartitionPredicate` : pour utiliser le nettoyage de partition côté serveur à l'aide des index de partition du AWS Glue Data Catalog.

- `PreProcessingQuery`— Chaîne UTF-8, d'une longueur maximale de 51 200 octets, correspondant au. URI address multi-line string pattern

  Requête SQL au format SparkSQL qui peut être utilisée pour prétraiter les données de la table AWS Glue Data Catalog dans, avant d'exécuter l'opération de qualité des données.

Opérations

- StartDataQualityRulesetEvaluationRun action (Python : start_data_quality_ruleset_evaluation_run)

- CancelDataQualityRulesetEvaluationRun action (Python : cancel_data_quality_ruleset_evaluation_run)
- GetDataQualityRulesetEvaluationRun action (Python : get_data_quality_ruleset_evaluation_run)
- ListDataQualityRulesetEvaluationRuns action (Python : list_data_quality_ruleset_evaluation_runs)
- StartDataQualityRuleRecommendationRun action (Python : start_data_quality_rule_recommendation_run)
- CancelDataQualityRuleRecommendationRun action (Python : cancel_data_quality_rule_recommendation_run)
- GetDataQualityRuleRecommendationRun action (Python : get_data_quality_rule_recommendation_run)
- ListDataQualityRuleRecommendationRuns action (Python : list_data_quality_rule_recommendation_runs)
- GetDataQualityResult action (Python : get_data_quality_result)
- BatchGetDataQualityResult action (Python : batch_get_data_quality_result)
- ListDataQualityResults action (Python : list_data_quality_results)
- CreateDataQualityRuleset action (Python : create_data_quality_ruleset)
- DeleteDataQualityRuleset action (Python : delete_data_quality_ruleset)
- GetDataQualityRuleset action (Python : get_data_quality_ruleset)
- ListDataQualityRulesets action (Python : list_data_quality_rulesets)
- UpdateDataQualityRuleset action (Python : update_data_quality_ruleset)
- ListDataQualityStatistics action (Python : list_data_quality_statistics)
- TimestampFilter structure
- CreateDataQualityRulesetRequest structure
- GetDataQualityRulesetResponse structure
- GetDataQualityResultResponse structure
- StartDataQualityRuleRecommendationRunRequest structure
- GetDataQualityRuleRecommendationRunResponse structure
- BatchPutDataQualityStatisticAnnotation action (Python : batch_put_data_quality_statistic_annotation)
- GetDataQualityModel action (Python : get_data_quality_model)
- GetDataQualityModelResult action (Python : get_data_quality_model_result)

- ListDataQualityStatisticAnnotations action (Python : list_data_quality_statistic_annotations)
- PutDataQualityProfileAnnotation action (Python : put_data_quality_profile_annotation)

StartDataQualityRulesetEvaluationRun action (Python : start_data_quality_ruleset_evaluation_run)

Une fois que vous avez une définition d'ensemble de règles (recommandée ou la vôtre), vous appelez cette opération pour évaluer l'ensemble de règles par rapport à une source de données (AWS Glue table). L'évaluation calcule les résultats que vous pouvez récupérer à l'aide de l'API `GetDataQualityResult`.

Demande

- `DataSource` – Obligatoire : un objet DataSource.

  La source de données (AWS Glue table) associée à cette exécution.
- `Role` – Obligatoire : chaîne UTF-8.

  IAM Rôle fourni pour chiffrer les résultats de l'exécution.
- `NumberOfWorkers` : nombre (entier).

  Nombre d'employés `G.1X` à utiliser dans l'exécution. La valeur par défaut est 5.
- `Timeout` – Nombre (entier), au moins égal à 1.

  Délai d'expiration d'une exécution en minutes. Durée maximale pendant laquelle une exécution peut consommer des ressources avant qu'elle ne se termine et n'entre dans l'état `TIMEOUT`. La valeur par défaut est de 2 880 minutes (48 heures).
- `ClientToken` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Utilisée pour l'idempotence ; il est recommandé de la définir sur un ID aléatoire (tel qu'un UUID) afin d'éviter de créer ou de démarrer plusieurs instances de la même ressource.
- `AdditionalRunOptions` – Un objet DataQualityEvaluationRunAdditionalRunOptions.

  Options d'exécution supplémentaires que vous pouvez spécifier pour une exécution d'évaluation.
- `RulesetNames` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 10 chaînes maximum.

Liste de noms d'ensembles de règles.

- `AdditionalDataSources` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaque valeur est un objet DataSource.

  Une carte de chaînes de référence vers des sources de données supplémentaires que vous pouvez spécifier pour une exécution d'évaluation.

Réponse

- `RunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant d'exécution unique associé à cette exécution.

Erreurs

- `InvalidInputException`
- `EntityNotFoundException`
- `OperationTimeoutException`
- `InternalServiceException`
- `ConflictException`

CancelDataQualityRulesetEvaluationRun action (Python : cancel_data_quality_ruleset_evaluation_run)

Annule une exécution au cours de laquelle un ensemble de règles est évalué par rapport à une source de données.

Demande

- `RunId` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

Identifiant d'exécution unique associé à cette exécution.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

GetDataQualityRulesetEvaluationRun action (Python : get_data_quality_ruleset_evaluation_run)

Récupère une exécution spécifique au cours de laquelle un ensemble de règles est évalué par rapport à une source de données.

Demande

- `RunId` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant d'exécution unique associé à cette exécution.

Réponse

- `RunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant d'exécution unique associé à cette exécution.
- `DataSource` – Un objet DataSource.

  La source de données (une AWS Glue table) associée à cette exécution d'évaluation.
- `Role` – Chaîne UTF-8.

IAM Rôle fourni pour chiffrer les résultats de l'exécution.

- `NumberOfWorkers` : nombre (entier).

  Nombre d'employés `G.1X` à utiliser dans l'exécution. La valeur par défaut est 5.

- `Timeout` – Nombre (entier), au moins égal à 1.

  Délai d'expiration d'une exécution en minutes. Durée maximale pendant laquelle une exécution peut consommer des ressources avant qu'elle ne se termine et n'entre dans l'état `TIMEOUT`. La valeur par défaut est de 2 880 minutes (48 heures).

- `AdditionalRunOptions` – Un objet DataQualityEvaluationRunAdditionalRunOptions.

  Options d'exécution supplémentaires que vous pouvez spécifier pour une exécution d'évaluation.

- `Status` – Chaîne UTF-8 (valeurs valides : `RUNNING` | `FINISHED` | `FAILED` | `PENDING_EXECUTION` | `TIMED_OUT` | `CANCELING` | `CANCELED` | `RECEIVED_BY_TASKRUNNER`).

  Statut de cette exécution.

- `ErrorString` – Chaîne UTF-8.

  Chaînes d'erreur associées à l'exécution.

- `StartedOn` – Horodatage.

  Date et heure de lancement de cette exécution.

- `LastModifiedOn` – Horodatage.

  Horodatage. Dernier moment où cette exécution de recommandation de règle de qualité des données a été modifiée.

- `CompletedOn` – Horodatage.

  Date et heure de fin de cette exécution.

- `ExecutionTime` : nombre (entier).

  Durée (en secondes) pendant laquelle l'exécution a consommé des ressources.

- `RulesetNames` – Tableau de chaînes UTF-8, avec 1 chaîne minimum et 10 chaînes maximum.

  Liste des noms des ensembles de règles utilisés pour l'exécution. Actuellement, ce paramètre ne peut avoir qu'un seul nom d'ensemble de règles.

- `ResultIds` – Tableau de chaînes UTF-8, avec 1 chaîne minimum et 10 chaînes maximum.

Liste des résultats IDs relatifs à la qualité des données obtenus lors de l'exécution.

- `AdditionalDataSources` – Tableau de mappage de paires valeur-clé.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaque valeur est un objet DataSource.

  Une carte de chaînes de référence vers des sources de données supplémentaires que vous pouvez spécifier pour une exécution d'évaluation.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

ListDataQualityRulesetEvaluationRuns action (Python : list_data_quality_ruleset_evaluation_runs)

Répertorie l'ensemble des exécutions répondant aux critères de filtrage, lorsqu'un ensemble de règles est évalué par rapport à une source de données.

Demande

- `Filter` – Un objet DataQualityRulesetEvaluationRunFilter.

  Critères de filtrage.

- `NextToken` – Chaîne UTF-8.

  Jeton de pagination pour décaler les résultats.

- `MaxResults` – Nombre (entier), compris entre 1 et 1 000.

  Nombre maximal de résultats à renvoyer.

Réponse

- `Runs` : un tableau d'objets DataQualityRulesetEvaluationRunDescription.

  Liste d'objets `DataQualityRulesetEvaluationRunDescription` représentant les exécutions d'un ensemble de règles en matière de qualité des données.
- `NextToken` – Chaîne UTF-8.

  Un jeton de pagination, si d'autres résultats sont disponibles.

Erreurs

- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

StartDataQualityRuleRecommendationRun action (Python : start_data_quality_rule_recommendation_run)

Lance une série de recommandations qui est utilisée pour générer des règles lorsque vous ne savez pas quelles règles écrire. AWS Glue Data Quality analyse les données et formule des recommandations pour un ensemble de règles potentiel. Vous pouvez ensuite trier l'ensemble de règles et modifier l'ensemble de règles généré selon votre convenance.

Les exécutions de recommandations sont automatiquement supprimées après 90 jours.

Demande

La demande de recommandation d'une règle de qualité des données.

- `DataSource` – Obligatoire : un objet DataSource.

  La source de données (AWS Glue table) associée à cette exécution.
- `Role` – Obligatoire : chaîne UTF-8.

  IAM Rôle fourni pour chiffrer les résultats de l'exécution.
- `NumberOfWorkers` : nombre (entier).

  Nombre d'employés `G.1X` à utiliser dans l'exécution. La valeur par défaut est 5.

- `Timeout` – Nombre (entier), au moins égal à 1.

  Délai d'expiration d'une exécution en minutes. Durée maximale pendant laquelle une exécution peut consommer des ressources avant qu'elle ne se termine et n'entre dans l'état `TIMEOUT`. La valeur par défaut est de 2 880 minutes (48 heures).

- `CreatedRulesetName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de l'ensemble de règles.

- `DataQualitySecurityConfiguration` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la configuration de sécurité créée avec l'option de chiffrement de la qualité des données.

- `ClientToken` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Utilisée pour l'idempotence ; il est recommandé de la définir sur un ID aléatoire (tel qu'un UUID) afin d'éviter de créer ou de démarrer plusieurs instances de la même ressource.

Réponse

- `RunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant d'exécution unique associé à cette exécution.

Erreurs

- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`
- `ConflictException`

CancelDataQualityRuleRecommendationRun action (Python : cancel_data_quality_rule_recommendation_run)

Annule l'exécution de recommandation spécifiée qui était utilisée pour générer des règles.

Demande

- `RunId` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant d'exécution unique associé à cette exécution.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

GetDataQualityRuleRecommendationRun action (Python : get_data_quality_rule_recommendation_run)

Obtient l'exécution de recommandation spécifiée qui a été utilisée pour générer des règles.

Demande

- `RunId` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant d'exécution unique associé à cette exécution.

Réponse

La réponse à l'exécution de la recommandation d'une règle de qualité des données.

- `RunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant d'exécution unique associé à cette exécution.
- `DataSource` – Un objet DataSource.

  La source de données (une AWS Glue table) associée à cette exécution.
- `Role` – Chaîne UTF-8.

  IAM Rôle fourni pour chiffrer les résultats de l'exécution.
- `NumberOfWorkers` : nombre (entier).

  Nombre d'employés `G.1X` à utiliser dans l'exécution. La valeur par défaut est 5.
- `Timeout` – Nombre (entier), au moins égal à 1.

  Délai d'expiration d'une exécution en minutes. Durée maximale pendant laquelle une exécution peut consommer des ressources avant qu'elle ne se termine et n'entre dans l'état `TIMEOUT`. La valeur par défaut est de 2 880 minutes (48 heures).
- `Status` – Chaîne UTF-8 (valeurs valides : `RUNNING` | `FINISHED` | `FAILED` | `PENDING_EXECUTION` | `TIMED_OUT` | `CANCELING` | `CANCELED` | `RECEIVED_BY_TASKRUNNER`).

  Statut de cette exécution.
- `ErrorString` – Chaîne UTF-8.

  Chaînes d'erreur associées à l'exécution.
- `StartedOn` – Horodatage.

  Date et heure de lancement de cette exécution.
- `LastModifiedOn` – Horodatage.

  Horodatage. Dernier moment où cette exécution de recommandation de règle de qualité des données a été modifiée.
- `CompletedOn` – Horodatage.

  Date et heure de fin de cette exécution.

- `ExecutionTime` : nombre (entier).

  Durée (en secondes) pendant laquelle l'exécution a consommé des ressources.

- `RecommendedRuleset` – Chaîne UTF-8, d'une longueur comprise entre 1 et 65536 octets.

  À la fin de l'exécution d'une recommandation de règle de démarrage, un ensemble de règles recommandé est créé. Ce membre dispose de ces règles au format DQDL (Data Quality Definition Language).

- `CreatedRulesetName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de l'ensemble de règles créé par l'exécution.

- `DataQualitySecurityConfiguration` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la configuration de sécurité créée avec l'option de chiffrement de la qualité des données.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

ListDataQualityRuleRecommendationRuns action (Python : list_data_quality_rule_recommendation_runs)

Répertorie les exécutions de recommandation répondant aux critères de filtrage.

Demande

- `Filter` – Un objet DataQualityRuleRecommendationRunFilter.

  Critères de filtrage.

- `NextToken` – Chaîne UTF-8.

  Jeton de pagination pour décaler les résultats.

- `MaxResults` – Nombre (entier), compris entre 1 et 1 000.

  Nombre maximal de résultats à renvoyer.

Réponse

- `Runs` : un tableau d'objets DataQualityRuleRecommendationRunDescription.

  Liste d'objets `DataQualityRuleRecommendationRunDescription`.
- `NextToken` – Chaîne UTF-8.

  Un jeton de pagination, si d'autres résultats sont disponibles.

Erreurs

- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

GetDataQualityResult action (Python : get_data_quality_result)

Récupère le résultat d'une évaluation des règles de qualité des données.

Demande

- `ResultId` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID de résultat unique pour le résultat en matière de qualité des données.

Réponse

La réponse pour le résultat en matière de qualité des données.

- `ResultId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID de résultat unique pour le résultat en matière de qualité des données.

- `ProfileId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  L'ID de profil pour le résultat en matière de qualité des données.

- `Score` – Nombre (double), au plus égal à 1,0.

  Score de qualité des données agrégées. Représente le rapport entre le nombre de règles transmises et le nombre total de règles.

- `DataSource` – Un objet DataSource.

  Table associée au résultat sur la qualité des données, le cas échéant.

- `RulesetName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de l'ensemble de règles associé au résultat sur la qualité des données.

- `EvaluationContext` – Chaîne UTF-8.

  Dans le contexte d'une tâche dans AWS Glue Studio, chaque nœud du canevas se voit généralement attribuer un nom et les nœuds de qualité des données porteront un nom. Dans le cas de plusieurs nœuds, `evaluationContext` peut distinguer les nœuds.

- `StartedOn` – Horodatage.

  Date et heure du début de l'exécution de ce résultat en matière de qualité des données.

- `CompletedOn` – Horodatage.

  Date et heure de fin de l'exécution de ce résultat en matière de qualité des données.

- `JobName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Le nom de tâche associé au résultat sur la qualité des données, le cas échéant.

- `JobRunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID d'exécution de la tâche associé au résultat sur la qualité des données, le cas échéant.

- `RulesetEvaluationRunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID d'exécution unique associé à l'évaluation de l'ensemble de règles.

- `RuleResults` – Un tableau d'objets DataQualityRuleResult, 2 000 structures au maximum.

  Liste d'objets `DataQualityRuleResult` représentant les résultats de chaque règle.

- `AnalyzerResults` – Un tableau d'objets DataQualityAnalyzerResult, 2 000 structures au maximum.

  Liste d'objets `DataQualityAnalyzerResult` représentant les résultats de chaque analyseur.

- `Observations` – Un tableau d'objets DataQualityObservation, 50 structures maximum.

  Liste d'objets `DataQualityObservation` représentant les observations générées après évaluation des règles et des analyseurs.

- `AggregatedMetrics` – Un objet DataQualityAggregatedMetrics.

  Récapitulatif des `DataQualityAggregatedMetrics` objets indiquant le nombre total de lignes et de règles traitées, y compris leurs pass/fail statistiques basées sur les résultats au niveau des lignes.

Erreurs

- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`
- `EntityNotFoundException`

BatchGetDataQualityResult action (Python : batch_get_data_quality_result)

Récupère une liste des résultats de qualité des données pour le résultat IDs spécifié.

Demande

- `ResultIds` – Obligatoire : Tableau de chaînes UTF-8, avec 1 chaîne minimum et 100 chaînes maximum.

  Une liste de résultats uniques IDs pour les résultats relatifs à la qualité des données.

Réponse

- `Results` – Obligatoire : un tableau d'objets DataQualityResult.

  Liste d'objets `DataQualityResult` représentant les résultats en matière de qualité des données.
- `ResultsNotFound` – Tableau de chaînes UTF-8, avec 1 chaîne minimum et 100 chaînes maximum.

  Liste des résultats IDs pour lesquels aucun résultat n'a été trouvé.

Erreurs

- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

ListDataQualityResults action (Python : list_data_quality_results)

Renvoie tous les résultats d'exécution en matière de qualité des données pour votre compte.

Demande

- `Filter` – Un objet DataQualityResultFilterCriteria.

  Critères de filtrage.
- `NextToken` – Chaîne UTF-8.

  Jeton de pagination pour décaler les résultats.
- `MaxResults` – Nombre (entier), compris entre 1 et 1 000.

  Nombre maximal de résultats à renvoyer.

Réponse

- `Results` – Obligatoire : un tableau d'objets DataQualityResultDescription.

  Liste d'objets `DataQualityResultDescription`.
- `NextToken` – Chaîne UTF-8.

Un jeton de pagination, si d'autres résultats sont disponibles.

Erreurs

- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

CreateDataQualityRuleset action (Python : create_data_quality_ruleset)

Crée un ensemble de règles de qualité des données avec des règles DQDL appliquées à une table spécifiée AWS Glue .

Vous créez l'ensemble de règles au format DQDL (Data Quality Definition Language). Pour plus d'informations, consultez le guide du AWS Glue développeur.

Demande

Demande de création d'un ensemble de règles de qualité des données.

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom unique de l'ensemble de règles de qualité des données.
- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description de l'ensemble de règles de qualité des données.
- `Ruleset` – Obligatoire : chaîne UTF-8, d'une longueur comprise entre 1 et 65536 octets.

  Ensemble de règles DQDL (Data Quality Definition Language). Pour plus d'informations, consultez le guide du AWS Glue développeur.
- `Tags` – Tableau de mappage de paires clé-valeur, avec 50 paires au maximum.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 256 octets.

Liste de balises appliquées à l'ensemble de règles de qualité des données.

- `TargetTable` – Un objet DataQualityTargetTable.

  Table cible associée à l'ensemble de règles de qualité des données.

- `RecommendationRunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID d'exécution unique pour l'exécution recommandée.

- `DataQualitySecurityConfiguration` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la configuration de sécurité créée avec l'option de chiffrement de la qualité des données.

- `ClientToken` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Utilisée pour l'idempotence ; il est recommandé de la définir sur un ID aléatoire (tel qu'un UUID) afin d'éviter de créer ou de démarrer plusieurs instances de la même ressource.

Réponse

- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom unique de l'ensemble de règles de qualité des données.

Erreurs

- `InvalidInputException`
- `AlreadyExistsException`
- `OperationTimeoutException`
- `InternalServiceException`
- `ResourceNumberLimitExceededException`

DeleteDataQualityRuleset action (Python : delete_data_quality_ruleset)

Supprime un ensemble de règles de qualité des données.

Demande

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de l'ensemble de règles de qualité des données.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

GetDataQualityRuleset action (Python : get_data_quality_ruleset)

Renvoie un ensemble de règles existant par identifiant ou nom.

Demande

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Le nom de l'ensemble de règles.

Réponse

Renvoie la réponse de l'ensemble de règles de qualité des données.

- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Le nom de l'ensemble de règles.

- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description de l'ensemble de règles.
- `Ruleset` – Chaîne UTF-8, d'une longueur comprise entre 1 et 65536 octets.

  Ensemble de règles DQDL (Data Quality Definition Language). Pour plus d'informations, consultez le guide du AWS Glue développeur.
- `TargetTable` – Un objet DataQualityTargetTable.

  Nom et nom de la base de données de la table cible.
- `CreatedOn` – Horodatage.

  Horodatage. Date et heure de création de cet ensemble de règles de qualité des données.
- `LastModifiedOn` – Horodatage.

  Horodatage. Dernier moment où cet ensemble de règles de qualité des données a été modifié.
- `RecommendationRunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Lors de la création d'un ensemble de règles à partir d'une exécution de recommandation, cet ID d'exécution est généré pour relier les deux.
- `DataQualitySecurityConfiguration` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la configuration de sécurité créée avec l'option de chiffrement de la qualité des données.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

ListDataQualityRulesets action (Python : list_data_quality_rulesets)

Renvoie une liste paginée d'ensembles de règles pour la liste de tables spécifiée. AWS Glue

Demande

- `NextToken` – Chaîne UTF-8.

  Jeton de pagination pour décaler les résultats.
- `MaxResults` – Nombre (entier), compris entre 1 et 1 000.

  Nombre maximal de résultats à renvoyer.
- `Filter` – Un objet DataQualityRulesetFilterCriteria.

  Critères de filtrage.
- `Tags` – Tableau de mappage de paires clé-valeur, avec 50 paires au maximum.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 256 octets.

  Liste de balises de paire clé-valeur.

Réponse

- `Rulesets` : un tableau d'objets DataQualityRulesetListDetails.

  Liste paginée d'ensembles de règles pour la liste de tables spécifiée. AWS Glue
- `NextToken` – Chaîne UTF-8.

  Un jeton de pagination, si d'autres résultats sont disponibles.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

UpdateDataQualityRuleset action (Python : update_data_quality_ruleset)

Met à jour l'ensemble de règles de qualité des données spécifié.

Demande

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de l'ensemble de règles de qualité des données.

- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description de l'ensemble de règles.

- `Ruleset` – Chaîne UTF-8, d'une longueur comprise entre 1 et 65536 octets.

  Ensemble de règles DQDL (Data Quality Definition Language). Pour plus d'informations, consultez le guide du AWS Glue développeur.

Réponse

- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de l'ensemble de règles de qualité des données.

- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description de l'ensemble de règles.

- `Ruleset` – Chaîne UTF-8, d'une longueur comprise entre 1 et 65536 octets.

  Ensemble de règles DQDL (Data Quality Definition Language). Pour plus d'informations, consultez le guide du AWS Glue développeur.

Erreurs

- `EntityNotFoundException`
- `AlreadyExistsException`
- `IdempotentParameterMismatchException`
- `InvalidInputException`
- `OperationTimeoutException`

- `InternalServiceException`
- `ResourceNumberLimitExceededException`

ListDataQualityStatistics action (Python : list_data_quality_statistics)

Récupère la liste des statistiques de qualité des données.

Demande

- `StatisticId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  L'ID de statistique.
- `ProfileId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  L'ID de profil.
- `TimestampFilter` – Un objet TimestampFilter.

  Un filtre d'horodatage.
- `MaxResults` – Nombre (entier), compris entre 1 et 1 000.

  Nombre maximal de résultats à renvoyer dans cette demande.
- `NextToken` – Chaîne UTF-8.

  Un jeton de pagination pour demander la page de résultats suivante.

Réponse

- `Statistics` : un tableau d'objets StatisticSummary.

  Une `StatisticSummaryList`.
- `NextToken` – Chaîne UTF-8.

  Un jeton de pagination pour demander la page de résultats suivante.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`

TimestampFilter structure

Un filtre d'horodatage.

Champs

- `RecordedBefore` – Horodatage.

  Horodatage avant lequel les statistiques doivent être incluses dans les résultats.
- `RecordedAfter` – Horodatage.

  Horodatage après lequel les statistiques doivent être incluses dans les résultats.

CreateDataQualityRulesetRequest structure

Demande de création d'un ensemble de règles de qualité des données.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom unique de l'ensemble de règles de qualité des données.
- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description de l'ensemble de règles de qualité des données.
- `Ruleset` – Obligatoire : chaîne UTF-8, d'une longueur comprise entre 1 et 65536 octets.

  Ensemble de règles DQDL (Data Quality Definition Language). Pour plus d'informations, consultez le guide du AWS Glue développeur.
- `Tags` – Tableau de mappage de paires clé-valeur, avec 50 paires au maximum.

Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 256 octets.

Liste de balises appliquées à l'ensemble de règles de qualité des données.

- `TargetTable` – Un objet DataQualityTargetTable.

  Table cible associée à l'ensemble de règles de qualité des données.

- `RecommendationRunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID d'exécution unique pour l'exécution recommandée.

- `DataQualitySecurityConfiguration` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la configuration de sécurité créée avec l'option de chiffrement de la qualité des données.

- `ClientToken` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Utilisée pour l'idempotence ; il est recommandé de la définir sur un ID aléatoire (tel qu'un UUID) afin d'éviter de créer ou de démarrer plusieurs instances de la même ressource.

GetDataQualityRulesetResponse structure

Renvoie la réponse de l'ensemble de règles de qualité des données.

Champs

- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Le nom de l'ensemble de règles.

- `Description` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Description de l'ensemble de règles.

- `Ruleset` – Chaîne UTF-8, d'une longueur comprise entre 1 et 65536 octets.

Ensemble de règles DQDL (Data Quality Definition Language). Pour plus d'informations, consultez le guide du AWS Glue développeur.

- `TargetTable` – Un objet DataQualityTargetTable.

  Nom et nom de la base de données de la table cible.

- `CreatedOn` – Horodatage.

  Horodatage. Date et heure de création de cet ensemble de règles de qualité des données.

- `LastModifiedOn` – Horodatage.

  Horodatage. Dernier moment où cet ensemble de règles de qualité des données a été modifié.

- `RecommendationRunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Lors de la création d'un ensemble de règles à partir d'une exécution de recommandation, cet ID d'exécution est généré pour relier les deux.

- `DataQualitySecurityConfiguration` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la configuration de sécurité créée avec l'option de chiffrement de la qualité des données.

GetDataQualityResultResponse structure

La réponse pour le résultat en matière de qualité des données.

Champs

- `ResultId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID de résultat unique pour le résultat en matière de qualité des données.

- `ProfileId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  L'ID de profil pour le résultat en matière de qualité des données.

- `Score` – Nombre (double), au plus égal à 1,0.

Score de qualité des données agrégées. Représente le rapport entre le nombre de règles transmises et le nombre total de règles.

- `DataSource` – Un objet DataSource.

  Table associée au résultat sur la qualité des données, le cas échéant.

- `RulesetName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de l'ensemble de règles associé au résultat sur la qualité des données.

- `EvaluationContext` – Chaîne UTF-8.

  Dans le contexte d'une tâche dans AWS Glue Studio, chaque nœud du canevas se voit généralement attribuer un nom et les nœuds de qualité des données porteront un nom. Dans le cas de plusieurs nœuds, `evaluationContext` peut distinguer les nœuds.

- `StartedOn` – Horodatage.

  Date et heure du début de l'exécution de ce résultat en matière de qualité des données.

- `CompletedOn` – Horodatage.

  Date et heure de fin de l'exécution de ce résultat en matière de qualité des données.

- `JobName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Le nom de tâche associé au résultat sur la qualité des données, le cas échéant.

- `JobRunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID d'exécution de la tâche associé au résultat sur la qualité des données, le cas échéant.

- `RulesetEvaluationRunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID d'exécution unique associé à l'évaluation de l'ensemble de règles.

- `RuleResults` – Un tableau d'objets DataQualityRuleResult, 2 000 structures au maximum.

  Liste d'objets `DataQualityRuleResult` représentant les résultats de chaque règle.

- `AnalyzerResults` – Un tableau d'objets DataQualityAnalyzerResult, 2 000 structures au maximum.

Liste d'objets `DataQualityAnalyzerResult` représentant les résultats de chaque analyseur.

- `Observations` – Un tableau d'objets DataQualityObservation, 50 structures maximum.

  Liste d'objets `DataQualityObservation` représentant les observations générées après évaluation des règles et des analyseurs.

- `AggregatedMetrics` – Un objet DataQualityAggregatedMetrics.

  Récapitulatif des `DataQualityAggregatedMetrics` objets indiquant le nombre total de lignes et de règles traitées, y compris leurs pass/fail statistiques basées sur les résultats au niveau des lignes.

StartDataQualityRuleRecommendationRunRequest structure

La demande de recommandation d'une règle de qualité des données.

Champs

- `DataSource` – Obligatoire : un objet DataSource.

  La source de données (AWS Glue table) associée à cette exécution.

- `Role` – Obligatoire : chaîne UTF-8.

  IAM Rôle fourni pour chiffrer les résultats de l'exécution.

- `NumberOfWorkers` : nombre (entier).

  Nombre d'employés `G.1X` à utiliser dans l'exécution. La valeur par défaut est 5.

- `Timeout` – Nombre (entier), au moins égal à 1.

  Délai d'expiration d'une exécution en minutes. Durée maximale pendant laquelle une exécution peut consommer des ressources avant qu'elle ne se termine et n'entre dans l'état `TIMEOUT`. La valeur par défaut est de 2 880 minutes (48 heures).

- `CreatedRulesetName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de l'ensemble de règles.

- `DataQualitySecurityConfiguration` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

Nom de la configuration de sécurité créée avec l'option de chiffrement de la qualité des données.

- `ClientToken` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Utilisée pour l'idempotence ; il est recommandé de la définir sur un ID aléatoire (tel qu'un UUID) afin d'éviter de créer ou de démarrer plusieurs instances de la même ressource.

GetDataQualityRuleRecommendationRunResponse structure

La réponse à l'exécution de la recommandation d'une règle de qualité des données.

Champs

- `RunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Identifiant d'exécution unique associé à cette exécution.
- `DataSource` – Un objet DataSource.

  La source de données (une AWS Glue table) associée à cette exécution.
- `Role` – Chaîne UTF-8.

  IAM Rôle fourni pour chiffrer les résultats de l'exécution.
- `NumberOfWorkers` : nombre (entier).

  Nombre d'employés `G.1X` à utiliser dans l'exécution. La valeur par défaut est 5.
- `Timeout` – Nombre (entier), au moins égal à 1.

  Délai d'expiration d'une exécution en minutes. Durée maximale pendant laquelle une exécution peut consommer des ressources avant qu'elle ne se termine et n'entre dans l'état `TIMEOUT`. La valeur par défaut est de 2 880 minutes (48 heures).
- `Status` – Chaîne UTF-8 (valeurs valides : `RUNNING` | `FINISHED` | `FAILED` | `PENDING_EXECUTION` | `TIMED_OUT` | `CANCELING` | `CANCELED` | `RECEIVED_BY_TASKRUNNER`).

  Statut de cette exécution.
- `ErrorString` – Chaîne UTF-8.

  Chaînes d'erreur associées à l'exécution.

- `StartedOn` – Horodatage.

  Date et heure de lancement de cette exécution.

- `LastModifiedOn` – Horodatage.

  Horodatage. Dernier moment où cette exécution de recommandation de règle de qualité des données a été modifiée.

- `CompletedOn` – Horodatage.

  Date et heure de fin de cette exécution.

- `ExecutionTime` : nombre (entier).

  Durée (en secondes) pendant laquelle l'exécution a consommé des ressources.

- `RecommendedRuleset` – Chaîne UTF-8, d'une longueur comprise entre 1 et 65536 octets.

  À la fin de l'exécution d'une recommandation de règle de démarrage, un ensemble de règles recommandé est créé. Ce membre dispose de ces règles au format DQDL (Data Quality Definition Language).

- `CreatedRulesetName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de l'ensemble de règles créé par l'exécution.

- `DataQualitySecurityConfiguration` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la configuration de sécurité créée avec l'option de chiffrement de la qualité des données.

BatchPutDataQualityStatisticAnnotation action (Python : batch_put_data_quality_statistic_annotation)

Annotez les points de données au fil du temps pour une statistique de qualité des données spécifique. L'API nécessite à la fois ProfileID et StatisticID dans le cadre de l'entrée. InclusionAnnotation L'API ne fonctionne que pour un seul élément statisticId sur plusieurs profils.

Demande

- `InclusionAnnotations` – Obligatoire : un tableau d'objets DatapointInclusionAnnotation.

Une liste de `DatapointInclusionAnnotation`. Ils InclusionAnnotations doivent contenir un ProfileID et un StatisticID. S'il y en a plusieurs InclusionAnnotations, la liste doit faire référence à un seul StatisticID pour plusieurs ProfileID.

- `ClientToken` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Jeton client.

Réponse

- `FailedInclusionAnnotations` : un tableau d'objets AnnotationError.

  Une liste de `AnnotationError`.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `ResourceNumberLimitExceededException`

GetDataQualityModel action (Python : get_data_quality_model)

Récupérez l'état d'entraînement du modèle ainsi que des informations supplémentaires (CompletedOn, StartedOn, FailureReason).

Demande

- `StatisticId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  L'ID de statistique.

- `ProfileId` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  L'ID de profil.

Réponse

- `Status` – Chaîne UTF-8 (valeurs valides : `RUNNING` | `SUCCEEDED` | `FAILED`).

  Statut d'entraînement du modèle de qualité des données.

- `StartedOn` – Horodatage.

  Horodatage du début de l'entraînement des modèles de qualité des données.

- `CompletedOn` – Horodatage.

  Horodatage de l'achèvement de l'entraînement des modèles de qualité des données.

- `FailureReason` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Motif de l'échec de l'entraînement.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

GetDataQualityModelResult action (Python : get_data_quality_model_result)

Récupérez les prédictions d'une statistique pour un ID de profil donné.

Demande

- `StatisticId` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  L'ID de statistique.

- `ProfileId` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  L'ID de profil.

Réponse

- `CompletedOn` – Horodatage.

  Horodatage de l'achèvement de l'entraînement des modèles de qualité des données.

- `Model` : un tableau d'objets StatisticModelResult.

  Liste de `StatisticModelResult`.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

ListDataQualityStatisticAnnotations action (Python : list_data_quality_statistic_annotations)

Récupérez les annotations d'une statistique de qualité des données.

Demande

- `StatisticId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  L'ID de statistique.

- `ProfileId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  L'ID de profil.

- `TimestampFilter` – Un objet TimestampFilter.

  Un filtre d'horodatage.

- `MaxResults` – Nombre (entier), compris entre 1 et 1 000.

  Nombre maximal de résultats à renvoyer dans cette demande.

- `NextToken` – Chaîne UTF-8.

  Jeton de pagination permettant de récupérer l'ensemble suivant de résultats.

Réponse

- `Annotations` : un tableau d'objets StatisticAnnotation.

  Liste de `StatisticAnnotation` appliquées à la statistique
- `NextToken` – Chaîne UTF-8.

  Jeton de pagination permettant de récupérer l'ensemble suivant de résultats.

Erreurs

- `InvalidInputException`
- `InternalServiceException`

PutDataQualityProfileAnnotation action (Python : put_data_quality_profile_annotation)

Annotez tous les points de données d'un profil.

Demande

- `ProfileId` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID du profil de surveillance de la qualité des données à annoter.
- `InclusionAnnotation` – Obligatoire : Chaîne UTF-8 (valeurs valides : `INCLUDE` | `EXCLUDE`).

  Valeur d'annotation d'inclusion à appliquer au profil.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`

API Détection des données sensibles

L'API de détection des données sensibles décrit les APIs méthodes utilisées pour détecter les données sensibles dans les colonnes et les lignes de vos données structurées.

Types de données

- CustomEntityType structure

CustomEntityType structure

Objet représentant un modèle personnalisé permettant de détecter des données sensibles sur les colonnes et les lignes de vos données structurées.

Champs

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du modèle personnalisé qui permet de le récupérer ou de le supprimer ultérieurement. Ce nom doit être unique par AWS compte.
- `RegexString` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaîne d'expression régulière utilisée pour détecter des données sensibles dans un modèle personnalisé.
- `ContextWords` – Tableau de chaînes UTF-8, entre 1 et 20 chaînes.

  Une liste de mots contextuels. Si aucun de ces mots contextuels n'est trouvé à proximité de l'expression régulière, les données ne seront pas détectées en tant que données sensibles.

  Si aucun mot contextuel n'est transmis, seule une expression régulière est vérifiée.

Opérations

- CreateCustomEntityType action (Python : create_custom_entity_type)
- DeleteCustomEntityType action (Python : delete_custom_entity_type)
- GetCustomEntityType action (Python : get_custom_entity_type)
- BatchGetCustomEntityTypes action (Python : batch_get_custom_entity_types)
- ListCustomEntityTypes action (Python : list_custom_entity_types)

CreateCustomEntityType action (Python : create_custom_entity_type)

Crée un modèle personnalisé utilisé pour détecter les données sensibles sur les colonnes et les lignes de vos données structurées.

Chaque modèle personnalisé que vous créez spécifie une expression régulière et une liste facultative de mots contextuels. Si aucun mot contextuel n'est transmis, seule une expression régulière est vérifiée.

Demande

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du modèle personnalisé qui permet de le récupérer ou de le supprimer ultérieurement. Ce nom doit être unique par AWS compte.
- `RegexString` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaîne d'expression régulière utilisée pour détecter des données sensibles dans un modèle personnalisé.
- `ContextWords` – Tableau de chaînes UTF-8, entre 1 et 20 chaînes.

  Une liste de mots contextuels. Si aucun de ces mots contextuels n'est trouvé à proximité de l'expression régulière, les données ne seront pas détectées en tant que données sensibles.

  Si aucun mot contextuel n'est transmis, seule une expression régulière est vérifiée.
- `Tags` – Tableau de mappage de paires clé-valeur, avec 50 paires au maximum.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 256 octets.

Liste de balises appliquées à un type d'entité personnalisé.

Réponse

- Name – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom du modèle personnalisé que vous avez créé.

Erreurs

- `AccessDeniedException`
- `AlreadyExistsException`
- `IdempotentParameterMismatchException`
- `InternalServiceException`
- `InvalidInputException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`

DeleteCustomEntityType action (Python : delete_custom_entity_type)

Supprime un modèle personnalisé en spécifiant son nom.

Demande

- Name – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Le nom du modèle personnalisé que vous souhaitez supprimer.

Réponse

- Name – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

Nom du modèle personnalisé que vous avez supprimé.

Erreurs

- `EntityNotFoundException`
- `AccessDeniedException`
- `InternalServiceException`
- `InvalidInputException`
- `OperationTimeoutException`

GetCustomEntityType action (Python : get_custom_entity_type)

Récupère les détails d'un modèle personnalisé en spécifiant son nom.

Demande

- `Name` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Le nom du modèle personnalisé que vous souhaitez récupérer.

Réponse

- `Name` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Le nom du modèle personnalisé que vous avez récupéré.
- `RegexString` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Chaîne d'expression régulière utilisée pour détecter des données sensibles dans un modèle personnalisé.
- `ContextWords` – Tableau de chaînes UTF-8, entre 1 et 20 chaînes.

Liste de mots contextuels, s'il est spécifié lors de la création du modèle personnalisé. Si aucun de ces mots contextuels n'est trouvé à proximité de l'expression régulière, les données ne seront pas détectées en tant que données sensibles.

Erreurs

- `EntityNotFoundException`
- `AccessDeniedException`
- `InternalServiceException`
- `InvalidInputException`
- `OperationTimeoutException`

BatchGetCustomEntityTypes action (Python : batch_get_custom_entity_types)

Récupère les détails des modèles personnalisés spécifiés par une liste de noms.

Demande

- `Names` – Obligatoire : tableau de chaînes UTF-8, avec 1 chaîne minimum et 50 chaînes maximum.

  Liste des noms des modèles personnalisés que vous souhaitez récupérer.

Réponse

- `CustomEntityTypes` : un tableau d'objets CustomEntityType.

  Une liste d'objets `CustomEntityType` représentant les modèles personnalisés que vous avez créés.
- `CustomEntityTypesNotFound` – Tableau de chaînes UTF-8, entre 1 et 50 chaînes.

  Liste des noms de modèles personnalisés qui ont été introuvables.

Erreurs

- `InvalidInputException`

- `InternalServiceException`
- `OperationTimeoutException`

ListCustomEntityTypes action (Python : list_custom_entity_types)

Répertorie tous les modèles personnalisés qui ont été créés.

Demande

- `NextToken` – Chaîne UTF-8.

  Jeton de pagination pour décaler les résultats.
- `MaxResults` – Nombre (entier), compris entre 1 et 1 000.

  Nombre maximal de résultats à renvoyer.
- `Tags` – Tableau de mappage de paires clé-valeur, avec 50 paires au maximum.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 256 octets.

  Liste de balises de paire clé-valeur.

Réponse

- `CustomEntityTypes` : un tableau d'objets CustomEntityType.

  Une liste d'objets `CustomEntityType` représentant des motifs personnalisés.
- `NextToken` – Chaîne UTF-8.

  Un jeton de pagination, si d'autres résultats sont disponibles.

Erreurs

- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

Marquage APIs AWS Glue

Types de données

- Structure Tag

Structure Tag

L'`Tag`objet représente une étiquette que vous pouvez attribuer à une AWS ressource. Chaque balise est constituée d'une clé et d'une valeur facultative que vous définissez.

Pour plus d'informations sur les balises et le contrôle de l'accès aux ressources dans AWS Glue, voir AWS Tags in AWS Glue et Specifying AWS Glue Resource ARNs dans le guide du développeur.

Champs

- `key` – Chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Identification de balise. La clé est obligatoire lorsque vous créez une balise sur un objet. La clé est sensible à la casse et ne doit pas contenir le préfixe aws.
- `value` – Chaîne UTF-8, d'une longueur maximale de 256 octets.

  Valeur de balise. La valeur est facultative lorsque vous créez une balise sur un objet. La valeur est sensible à la casse et ne doit pas contenir le préfixe aws.

Opérations

- TagResource action (Python : tag_resource)
- UntagResource action (Python : untag_resource)
- GetTags action (Python : get_tags)

TagResource action (Python : tag_resource)

Ajoute des balises à une ressource. Une étiquette est une étiquette que vous pouvez attribuer à une AWS ressource. Dans AWS Glue, vous ne pouvez étiqueter que certaines ressources. Pour obtenir des informations sur les ressources que vous pouvez baliser, veuillez consulter la rubrique Balises AWS dans AWS Glue.

Outre les autorisations de balisage liées aux balises d'appel APIs, vous devez également avoir l'`glue:GetConnection`autorisation de baliser les appels APIs sur les connexions et l'`glue:GetDatabase`autorisation de baliser les appels sur les bases de données APIs .

Demande

- `ResourceArn` – Obligatoire : chaîne UTF-8, d'une longueur comprise entre 1 et 10 240 octets, correspondant au Custom string pattern #50.

  L'ARN de la AWS Glue ressource à laquelle ajouter les balises. Pour plus d'informations sur les AWS Glue ressources ARNs, consultez le modèle de chaîne AWS Glue ARN.
- `TagsToAdd` – Obligatoire : Tableau de mappage de paires clé-valeur, avec 50 paires au maximum.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 256 octets.

  Balises à ajouter à cette ressource.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `ResourceNotFoundException`

UntagResource action (Python : untag_resource)

Supprime les balises spécifiées d'une ressource d'intégration.

Demande

- `ResourceArn` – Obligatoire : chaîne UTF-8, d'une longueur comprise entre 1 et 10 240 octets, correspondant au Custom string pattern #50.

  Amazon Resource Name (ARN) de la ressource d'intégration.
- `TagsToRemove` – Obligatoire : Tableau de chaînes UTF-8, avec 50 chaînes maximum.

Liste des balises de métadonnés à supprimer de la ressource.

Réponse

- Paramètres d'absence de réponse.

Erreurs

- `ResourceNotFoundException`

GetTags action (Python : get_tags)

Récupère la liste des balises associées à une ressource.

Demande

- `ResourceArn` – Obligatoire : chaîne UTF-8, d'une longueur comprise entre 1 et 10 240 octets, correspondant au Custom string pattern #50.

  Amazon Resource Name (ARN) de la ressource pour laquelle récupérer les balises.

Réponse

- `Tags` – Tableau de mappage de paires clé-valeur, avec 50 paires au maximum.

  Chaque clé est une chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Chaque valeur est une chaîne UTF-8, d'une longueur maximale de 256 octets.

  Balises demandées.

Erreurs

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `EntityNotFoundException`

Types de données courants

Les types de données courants décrivent des types de données courants variés dans AWS Glue.

Structure de balise

L'`Tag`objet représente une étiquette que vous pouvez attribuer à une AWS ressource. Chaque balise est constituée d'une clé et d'une valeur facultative que vous définissez.

Pour plus d'informations sur les balises et le contrôle de l'accès aux ressources dans AWS Glue, voir AWS Tags in AWS Glue et Specifying AWS Glue Resource ARNs dans le guide du développeur.

Champs

- `key` – Chaîne UTF-8, d'une longueur comprise entre 1 et 128 octets.

  Identification de balise. La clé est obligatoire lorsque vous créez une balise sur un objet. La clé est sensible à la casse et ne doit pas contenir le préfixe aws.
- `value` – Chaîne UTF-8, d'une longueur maximale de 256 octets.

  Valeur de balise. La valeur est facultative lorsque vous créez une balise sur un objet. La valeur est sensible à la casse et ne doit pas contenir le préfixe aws.

DecimalNumber structure

Contient une valeur numérique au format décimal.

Champs

- `UnscaledValue` – obligatoire : blob.

  Valeur numérique non mise à l'échelle.
- `Scale` – obligatoire : nombre (entier).

  Échelle qui détermine où la virgule décimale tombe dans la valeur non mise à l'échelle.

ErrorDetail structure

Contient des informations détaillées sur une erreur.

Champs

- `ErrorCode` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Code associé à cette erreur.
- `ErrorMessage` – Chaîne de description, d'une longueur maximale de 2 048 octets, correspondant au URI address multi-line string pattern.

  Message décrivant l'erreur.

PropertyPredicate structure

Définit un prédicat de propriété.

Champs

- `Key` : chaîne de valeurs, d'une longueur comprise entre 1 et 1 024 octets.

  Clé de la propriété.
- `Value` : chaîne de valeurs, d'une longueur comprise entre 1 et 1 024 octets.

  Valeur de la propriété.
- `Comparator` – Chaîne UTF-8 (valeurs valides : `EQUALS` | `GREATER_THAN` | `LESS_THAN` | `GREATER_THAN_EQUALS` | `LESS_THAN_EQUALS`).

  Comparateur utilisé pour comparer cette propriété à d'autres.

ResourceUri structure

Les ressources URIs de la fonction for.

Champs

- `ResourceType` – Chaîne UTF-8 (valeurs valides : `JAR` | `FILE` | `ARCHIVE`).

  Type de la ressource.
- `Uri` – Identificateur de ressource uniforme (URI), d'une longueur comprise entre 1 et 1024 octets, correspondant au URI address multi-line string pattern.

URI pour accéder à la ressource.

ColumnStatistics structure

Représente les statistiques au niveau de la colonne générées pour une table ou une partition.

Champs

- `ColumnName` – Obligatoire : Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la colonne à laquelle appartiennent les statistiques.

- `ColumnType` – obligatoire : saisissez le nom, d'une longueur maximale de 20 000 octets, correspondant à Single-line string pattern.

  Type de données de la colonne.

- `AnalyzedTime` – obligatoire : horodatage.

  Horodatage du moment où les statistiques de colonne ont été générées.

- `StatisticsData` – Obligatoire : un objet ColumnStatisticsData.

  `ColumnStatisticData` qui contient les valeurs de données statistiques.

ColumnStatisticsError structure

Encapsule un objet `ColumnStatistics` qui a échoué et la raison de l'échec.

Champs

- `ColumnStatistics` – Un objet ColumnStatistics.

  `ColumnStatistics` de la colonne.

- `Error` : un objet ErrorDetail.

  Message d'erreur avec la raison de l'échec d'une opération.

ColumnError structure

Encapsule un nom de colonne qui a échoué et la raison de l'échec.

Champs

- `ColumnName` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  Nom de la colonne ayant échoué.
- `Error` : un objet ErrorDetail.

  Message d'erreur avec la raison de l'échec d'une opération.

ColumnStatisticsData structure

Contient chacun des types de données statistiques de colonne. Un seul objet de données doit être défini et indiqué par l'attribut `Type`.

Champs

- `Type` – obligatoire : chaîne UTF-8 (valeurs valides : `BOOLEAN` | `DATE` | `DECIMAL` | `DOUBLE` | `LONG` | `STRING` | `BINARY`).

  Type de données statistiques de colonne.
- `BooleanColumnStatisticsData` – Un objet BooleanColumnStatisticsData.

  Données statistiques de colonne booléennes.
- `DateColumnStatisticsData` – Un objet DateColumnStatisticsData.

  Données statistiques de colonne de date.
- `DecimalColumnStatisticsData` – Un objet DecimalColumnStatisticsData.

  Données statistiques sur les colonnes décimales. UnscaledValues à l'intérieur se trouvent des objets binaires codés en Base64 stockant des représentations complémentaires à deux de la valeur décimale non mise à l'échelle.
- `DoubleColumnStatisticsData` – Un objet DoubleColumnStatisticsData.

  Données statistiques à double colonne.

- `LongColumnStatisticsData` – Un objet LongColumnStatisticsData.

  Données statistiques de colonne longue.
- `StringColumnStatisticsData` – Un objet StringColumnStatisticsData.

  Données statistiques de colonne de chaîne.
- `BinaryColumnStatisticsData` – Un objet BinaryColumnStatisticsData.

  Données de statistiques de colonne binaire.

BooleanColumnStatisticsData structure

Définit les statistiques de colonne prises en charge pour les colonnes de données booléennes.

Champs

- `NumberOfTrues` – obligatoire : nombre (entier), pas plus que None (Aucun).

  Nombre de valeurs true dans la colonne.
- `NumberOfFalses` – obligatoire : nombre (entier), pas plus que None (Aucun).

  Nombre de valeurs false dans la colonne.
- `NumberOfNulls` – obligatoire : nombre (entier), pas plus que None (Aucun).

  Nombre de valeurs null dans la colonne.

DateColumnStatisticsData structure

Définit les statistiques de colonne prises en charge pour les colonnes de données d'horodatage.

Champs

- `MinimumValue` – Horodatage.

  Valeur la plus faible dans la colonne.
- `MaximumValue` – Horodatage.

  Valeur la plus élevée dans la colonne.
- `NumberOfNulls` – obligatoire : nombre (entier), pas plus que None (Aucun).

Nombre de valeurs null dans la colonne.

- `NumberOfDistinctValues` – obligatoire : nombre (entier), pas plus que None (Aucun).

  Nombre de valeurs distinctes dans une colonne.

DecimalColumnStatisticsData structure

Définit les statistiques de colonne prises en charge pour les colonnes de données à virgule fixe.

Champs

- `MinimumValue` – Un objet DecimalNumber.

  Valeur la plus faible dans la colonne.

- `MaximumValue` – Un objet DecimalNumber.

  Valeur la plus élevée dans la colonne.

- `NumberOfNulls` – obligatoire : nombre (entier), pas plus que None (Aucun).

  Nombre de valeurs null dans la colonne.

- `NumberOfDistinctValues` – obligatoire : nombre (entier), pas plus que None (Aucun).

  Nombre de valeurs distinctes dans une colonne.

DoubleColumnStatisticsData structure

Définit les statistiques de colonne prises en charge pour les colonnes de données à virgule flottante.

Champs

- `MinimumValue` – Nombre (double).

  Valeur la plus faible dans la colonne.

- `MaximumValue` – Nombre (double).

  Valeur la plus élevée dans la colonne.

- `NumberOfNulls` – obligatoire : nombre (entier), pas plus que None (Aucun).

Nombre de valeurs null dans la colonne.

- `NumberOfDistinctValues` – obligatoire : nombre (entier), pas plus que None (Aucun).

  Nombre de valeurs distinctes dans une colonne.

LongColumnStatisticsData structure

Définit les statistiques de colonne prises en charge pour les colonnes de données entières.

Champs

- `MinimumValue` – Nombre (long).

  Valeur la plus faible dans la colonne.
- `MaximumValue` – Nombre (long).

  Valeur la plus élevée dans la colonne.
- `NumberOfNulls` – obligatoire : nombre (entier), pas plus que None (Aucun).

  Nombre de valeurs null dans la colonne.
- `NumberOfDistinctValues` – obligatoire : nombre (entier), pas plus que None (Aucun).

  Nombre de valeurs distinctes dans une colonne.

StringColumnStatisticsData structure

Définit les statistiques de colonne prises en charge pour les valeurs de données de séquence de caractères.

Champs

- `MaximumLength` – obligatoire : nombre (entier), pas plus que None (Aucun).

  Taille de la chaîne la plus longue dans la colonne.
- `AverageLength` – obligatoire : nombre (double), pas plus que None (Aucun).

  Longueur moyenne de la chaîne dans la colonne.
- `NumberOfNulls` – obligatoire : nombre (entier), pas plus que None (Aucun).

Nombre de valeurs null dans la colonne.

- `NumberOfDistinctValues` – obligatoire : nombre (entier), pas plus que None (Aucun).

  Nombre de valeurs distinctes dans une colonne.

BinaryColumnStatisticsData structure

Définit les statistiques de colonne prises en charge pour les valeurs de données de séquence de bits.

Champs

- `MaximumLength` – obligatoire : nombre (entier), pas plus que None (Aucun).

  Taille de la séquence de bits la plus longue de la colonne.
- `AverageLength` – obligatoire : nombre (double), pas plus que None (Aucun).

  Longueur moyenne de la séquence de bits dans la colonne.
- `NumberOfNulls` – obligatoire : nombre (entier), pas plus que None (Aucun).

  Nombre de valeurs null dans la colonne.

Modèles de chaîne

L'API utilise les expressions régulières suivantes pour définir le contenu valide pour différents paramètres et membres de chaîne :

- Modèle de chaîne à ligne unique – `"[\u0020-\uD7FF\uE000-\uFFFD\uD800\uDC00-\uDBFF\uDFFF\t]*"`
- Modèle de chaîne à plusieurs lignes d'adresse URI – `"[\u0020-\uD7FF\uE000-\uFFFD\uD800\uDC00-\uDBFF\uDFFF\r\n\t]*"`
- Modèle de chaîne Logstash Grok – `"[\u0020-\uD7FF\uE000-\uFFFD\uD800\uDC00-\uDBFF\uDFFF\r\t]*"`
- Modèle de chaîne d'identifiant – `"[A-Za-z_][A-Za-z0-9_]*"`
- Modèle de chaîne ARN AWS IAM – `"arn:aws:iam::\d{12}:role/.*"`
- Modèle de chaîne de version – `"^[a-zA-Z0-9-_]+$"`
- Modèle de chaîne de groupe de journaux – `"[\.\-_/#A-Za-z0-9]+"`

- Modèle de chaîne de flux de journaux – `"[^:*]*"`
- Modèle de chaîne personnalisée #10 – `"[a-zA-Z0-9-_]+"`
- Modèle de chaîne personnalisée #11 – `"[-a-zA-Z0-9+=/:_]*"`
- Modèle de chaîne personnalisée #12 – `"[\S\s]*"`
- Modèle de chaîne personnalisée #13 – `".*\S.*"`
- Modèle de chaîne personnalisée #14 – `"[a-zA-Z0-9-=._/@]+"`
- Modèle de chaîne personnalisée #15 – `"[1-9][0-9]*|[1-9][0-9]*-[1-9][0-9]*"`
- Modèle de chaîne personnalisée #16 – `"[A-Z][A-Za-z\.]+"`
- Modèle de chaîne personnalisée #17 – `"[\S]*"`
- Modèle de chaîne personnalisée #18 – `"[\w]*"`
- Modèle de chaîne personnalisée #19 – `"arn:aws[a-z\-]*:iam::\d{12}:role/?[a-zA-Z_0-9+=,.@\-_/]+"`
- Modèle de chaîne personnalisée #20 – `"subnet-[a-z0-9]+"`
- Modèle de chaîne personnalisée #21 – `"\d{12}"`
- Modèle de chaîne personnalisée #22 – `"([a-z]+)-([a-z]+-)?([a-z]+)-[0-9]+[a-z]+"`
- Modèle de chaîne personnalisée #23 – `"[a-zA-Z0-9.-]*"`
- Modèle de chaîne personnalisée #24 – « `arn:aws[a-z0-9\-]*:lambda:[a-z0-9\-]+:\d{12}:function:([\w\-]{1,64})` »
- Modèle de chaîne personnalisée #25 – « `^(?!(.*[.\/\\]|aws:)).*$` »
- Modèle de chaîne personnalisée #26 – « `[^\r\n]` »
- Modèle de chaîne personnalisée #27 – « `^\w+\.\w+\.\w+$` »
- Modèle de chaîne personnalisée #28 – « `^\w+\.\w+$` »
- Modèle de chaîne personnalisée #29 – « `^$|arn:aws[a-z0-9-]*:kms:.*` »
- Modèle de chaîne personnalisée #30 – « `arn:aws[^:]*:iam::[0-9]*:role/.+` »
- Modèle de chaîne personnalisée #31 – « `[\.\-_A-Za-z0-9]+` »
- Modèle de chaîne personnalisée #32 – « `^s3://([^/]+)/([^/]+/)*([^/]+)$` »
- Modèle de chaîne personnalisée #33 – « `.*` »
- Modèle de chaîne personnalisée #34 – « `^(Sun|Mon|Tue|Wed|Thu|Fri|Sat):([01]?[0-9]|2[0-3])$` »
- Modèle de chaîne personnalisée n° 35 – « `[a-zA-Z0-9_.-]+` »

- Schéma de chaîne personnalisé #36 — « `^arn:aws(-(cn|us-gov|eusc|iso(-[bef])?))?:secretsmanager:.*$` »
- Modèle de chaîne personnalisée n° 37 : « `\S+` »
- Modèle de chaîne personnalisée n° 38 : « `^[\x20-\x7E]*$` »
- Modèle de chaîne personnalisée n° 39 : « `^([a-zA-Z0-9_=]+)\.([a-zA-Z0-9_=]+)\.([a-zA-Z0-9_\-\+\/=]*)` »
- Modèle de chaîne personnalisée n° 40 : « `^(https?://[-a-zA-Z0-9+&@#/%?=~_|!:,.;]*[-a-zA-Z0-9+&@#/%=~_|]` »
- Modèle de chaîne personnalisée n° 41 : « `^(https?):\/\/[^\s/$.?#].[^\s]*$` »
- Modèle de chaîne personnalisée n° 42 : « `arn:aws:kms:.*` »
- Modèle de chaîne personnalisée n° 43 : « `^subnet-[a-z0-9]+$` »
- Modèle de chaîne personnalisée n° 44 : « `[\p{L}\p{N}\p{P}]*` »
- Modèle de chaîne personnalisée n° 45 : « `[a-f0-9]{8}-[a-f0-9]{4}-[a-f0-9]{4}-[a-f0-9]{4}-[a-f0-9]{12}` »
- Modèle de chaîne personnalisée n° 46 : « `[a-zA-Z0-9-_$#.]+` »
- Modèle de chaîne personnalisée n° 47 : « `^\d{12}$` »
- Modèle de chaîne personnalisée n° 48 : « `^(\w+\.)+\w+$` »
- Modèle de chaîne personnalisée n° 49 : « `^([2-3]|3[.]9)$` »
- Modèle de chaîne personnalisée n° 50 : « `arn:aws(-(cn|us-gov|eusc|iso(-[bef])?))?:glue:.*` »
- Modèle de chaîne personnalisée n° 51 : « `[0-9a-fA-F]{8}-[0-9a-fA-F]{4}-[0-9a-fA-F]{4}-[0-9a-fA-F]{4}-[0-9a-fA-F]{12}` »
- Modèle de chaîne personnalisée n° 52 : « `(^arn:aws(-(cn|us-gov|eusc|iso(-[bef])?))?:iam::\w{12}:root)` »
- Modèle de chaîne personnalisée n° 53 : « `^arn:aws(-(cn|us-gov|eusc|iso(-[bef])?))?:iam::[0-9]{12}:role/.+` »
- Modèle de chaîne personnalisée n° 54 : « `[\s\S]*` »
- Modèle de chaîne personnalisée n° 55 : « `([\u0020-\uD7FF\uE000-\uFFFD\uD800\uDC00-\uDBFF\uDFFF]|[^\S\r\n"'= ;])*` »
- Modèle de chaîne personnalisée n° 56 : « `^[A-Z\_]+$` »
- Modèle de chaîne personnalisée n° 57 : « `^[A-Za-z0-9]+$` »
- Modèle de chaîne personnalisée n° 58 : « `[*A-Za-z0-9_-]*` »

- Modèle de chaîne personnalisée n° 59 : « `([\u0020-\u007E\r\s\n])*` »
- Modèle de chaîne personnalisée n° 60 : « `[A-Za-z0-9_-]*` »
- Modèle de chaîne personnalisée n° 61 : « `([\u0009\u000B\u000C\u0020-\uD7FF\uE000-\uFFFD\uD800\uDC00-\uDBFF\uDFFF])*` »
- Schéma de chaîne personnalisé #62 — « `([\u0020-\uD7FF\uE000-\uFFFD\uD800\uDC00-\uDBFF\uDFFF\s])*` »
- Schéma de chaîne personnalisé #63 — « `([^\r\n])*` »

Exceptions

Cette section décrit les AWS Glue exceptions que vous pouvez utiliser pour identifier la source des problèmes et les résoudre. Pour plus d'informations sur les chaînes et les codes d'erreur HTTP des exceptions liées au machine learning, consultez the section called "Exceptions AWS Glue liées au machine learning".

AccessDeniedException structure

L'accès à une ressource a été refusé.

Champs

- `Message` – Chaîne UTF-8.

  Un message décrivant le problème.

AlreadyExistsException structure

Une ressource à créer ou à ajouter existe déjà.

Champs

- `Message` – Chaîne UTF-8.

  Un message décrivant le problème.

ConcurrentModificationException structure

Deux processus tentent de modifier une ressource simultanément.

Champs

- `Message` – Chaîne UTF-8.

  Un message décrivant le problème.

ConcurrentRunsExceededException structure

Un trop grand nombre de tâches sont exécutées simultanément.

Champs

- `Message` – Chaîne UTF-8.

  Un message décrivant le problème.

CrawlerNotRunningException structure

L'crawler spécifié n'est pas en cours d'exécution.

Champs

- `Message` – Chaîne UTF-8.

  Un message décrivant le problème.

CrawlerRunningException structure

L'opération ne peut pas être effectuée car l'crawler est déjà en cours d'exécution.

Champs

- `Message` – Chaîne UTF-8.

  Un message décrivant le problème.

CrawlerStoppingException structure

L'crawler spécifié est en cours d'arrêt.

Champs

- `Message` – Chaîne UTF-8.

  Un message décrivant le problème.

EntityNotFoundException structure

Une entité spécifiée n'existe pas.

Champs

- `Message` – Chaîne UTF-8.

  Un message décrivant le problème.
- `FromFederationSource` : booléen.

  Indique si l'exception concerne une source fédérée.

FederationSourceException structure

Une source de fédération a échoué.

Champs

- `FederationSourceErrorCode`— Chaîne UTF-8 (valeurs valides : `AccessDeniedException` | | `EntityNotFoundException` | `InvalidCredentialsException` | `InvalidInputException` | `InvalidResponseException` | `OperationTimeoutException` | `OperationNotSupportedException` `InternalServiceException` `PartialFailureException` |`ThrottlingException`).

  Le code d'erreur du problème.
- `Message` – Chaîne UTF-8.

  Un message décrivant le problème.

FederationSourceRetryableException structure

Une source de fédération a échoué, mais l'opération peut être tentée à nouveau.

Champs

- `Message` – Chaîne UTF-8.

  Un message décrivant le problème.

GlueEncryptionException structure

Une opération de chiffrement a échoué.

Champs

- `Message` – Chaîne UTF-8.

  Un message décrivant le problème.

IdempotentParameterMismatchException structure

Le même identifiant unique a été associé à deux enregistrements différents.

Champs

- `Message` – Chaîne UTF-8.

  Un message décrivant le problème.

IllegalWorkflowStateException structure

Le flux de travail est dans un état non valide pour effectuer une opération demandée.

Champs

- `Message` – Chaîne UTF-8.

  Un message décrivant le problème.

InternalServiceException structure

Une erreur de service interne s'est produite.

Champs

- `Message` – Chaîne UTF-8.

  Un message décrivant le problème.

InvalidExecutionEngineException structure

Un moteur d'exécution inconnu ou non valide a été spécifié.

Champs

- `message` – Chaîne UTF-8.

  Un message décrivant le problème.

InvalidInputException structure

L'entrée fournie n'est pas valide.

Champs

- `Message` – Chaîne UTF-8.

  Un message décrivant le problème.
- `FromFederationSource` : booléen.

  Indique si l'exception concerne une source fédérée.

InvalidStateException structure

Une erreur indiquant que vos données sont dans un état non valide.

Champs

- `Message` – Chaîne UTF-8.

  Un message décrivant le problème.

InvalidTaskStatusTransitionException structure

Une transition correcte d'une tâche à une autre a échoué.

Champs

- `message` – Chaîne UTF-8.

  Un message décrivant le problème.

JobDefinitionErrorException structure

Une définition de tâche n'est pas valide.

Champs

- `message` – Chaîne UTF-8.

  Un message décrivant le problème.

JobRunInTerminalStateException structure

L'état de mise hors service d'une exécution de tâche signale une défaillance.

Champs

- `message` – Chaîne UTF-8.

  Un message décrivant le problème.

JobRunInvalidStateTransitionException structure

Une exécution de tâche a rencontré une transition non valide de l'état source vers l'état cible.

Champs

- `jobRunId` – Chaîne UTF-8, d'une longueur comprise entre 1 et 255 octets, correspondant au Single-line string pattern.

  ID de l'exécution de tâche en question.

- `message` – Chaîne UTF-8.

  Un message décrivant le problème.

- `sourceState`— Chaîne UTF-8 (valeurs valides : `STARTING` | | `RUNNING` | `STOPPING` | `STOPPED` | `SUCCEEDED` | `FAILED` | `TIMEOUT` `ERROR` `WAITING` |`EXPIRED`).

  État source.

- `targetState`— Chaîne UTF-8 (valeurs valides : `STARTING` | | `RUNNING` | `STOPPING` | `STOPPED` | `SUCCEEDED` | `FAILED` | `TIMEOUT` `ERROR` `WAITING` |`EXPIRED`).

  État cible.

JobRunNotInTerminalStateException structure

Une exécution de tâche n'est pas dans un état de mise hors service.

Champs

- `message` – Chaîne UTF-8.

  Un message décrivant le problème.

LateRunnerException structure

Un exécuteur de tâche est en retard.

Champs

- `Message` – Chaîne UTF-8.

  Un message décrivant le problème.

NoScheduleException structure

Il n'y a pas de calendrier applicable.

Champs

- `Message` – Chaîne UTF-8.

Un message décrivant le problème.

OperationTimeoutException structure

L'opération a expiré.

Champs

- `Message` – Chaîne UTF-8.

  Un message décrivant le problème.

ResourceNotReadyException structure

Une ressource n'était pas prête pour une transaction.

Champs

- `Message` – Chaîne UTF-8.

  Un message décrivant le problème.

ResourceNumberLimitExceededException structure

Une limite numérique de ressource a été dépassée.

Champs

- `Message` – Chaîne UTF-8.

  Un message décrivant le problème.

SchedulerNotRunningException structure

Le planificateur spécifié n'est pas en cours d'exécution.

Champs

- `Message` – Chaîne UTF-8.

Un message décrivant le problème.

SchedulerRunningException structure

Le planificateur spécifié est déjà en cours d'exécution.

Champs

- `Message` – Chaîne UTF-8.

  Un message décrivant le problème.

SchedulerTransitioningException structure

Le planificateur spécifié est en cours de conversion.

Champs

- `Message` – Chaîne UTF-8.

  Un message décrivant le problème.

UnrecognizedRunnerException structure

L' exécuteur de tâche na pas été reconnu.

Champs

- `Message` – Chaîne UTF-8.

  Un message décrivant le problème.

ValidationException structure

Une valeur n'a pas pu être validée.

Champs

- `Message` – Chaîne UTF-8.

Un message décrivant le problème.

VersionMismatchException structure

Un conflit de version s'est produit.

Champs

- `message` – Chaîne UTF-8.

  Un message décrivant le problème.

AWS Glue Exemples de code API utilisant AWS SDKs

Les exemples de code suivants montrent comment utiliser AWS Glue un kit de développement AWS logiciel (SDK).

Les principes de base sont des exemples de code qui vous montrent comment effectuer les opérations essentielles au sein d'un service.

Les actions sont des extraits de code de programmes plus larges et doivent être exécutées dans leur contexte. Alors que les actions vous indiquent comment appeler des fonctions de service individuelles, vous pouvez les voir en contexte dans leurs scénarios associés.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

AWS Glue Exemples de code API

- Exemples de base pour AWS Glue l'utilisation AWS SDKs
  - Bonjour AWS Glue
  - Apprenez les bases AWS Glue d'un AWS SDK
  - Actions d' AWS Glue utilisation AWS SDKs
    - Utilisation CreateCrawler avec un AWS SDK
    - Utilisation CreateJob avec un AWS SDK ou une CLI
    - Utilisation DeleteCrawler avec un AWS SDK
    - Utilisation DeleteDatabase avec un AWS SDK
    - Utilisation DeleteJob avec un AWS SDK ou une CLI
    - Utilisation DeleteTable avec un AWS SDK
    - Utilisation GetCrawler avec un AWS SDK
    - Utilisation GetDatabase avec un AWS SDK
    - Utilisation GetDatabases avec un AWS SDK ou une CLI
    - Utilisation GetJob avec un AWS SDK ou une CLI
    - Utilisation GetJobRun avec un AWS SDK ou une CLI
    - Utilisation GetJobRuns avec un AWS SDK ou une CLI
    - Utilisation GetTables avec un AWS SDK ou une CLI

- Utilisation ListJobs avec un AWS SDK
- Utilisation StartCrawler avec un AWS SDK ou une CLI
- Utilisation StartJobRun avec un AWS SDK ou une CLI

Exemples de base pour AWS Glue l'utilisation AWS SDKs

Les exemples de code suivants montrent comment utiliser les principes de base de AWS Glue with AWS SDKs.

Exemples

- Bonjour AWS Glue
- Apprenez les bases AWS Glue d'un AWS SDK
- Actions d' AWS Glue utilisation AWS SDKs
  - Utilisation CreateCrawler avec un AWS SDK
  - Utilisation CreateJob avec un AWS SDK ou une CLI
  - Utilisation DeleteCrawler avec un AWS SDK
  - Utilisation DeleteDatabase avec un AWS SDK
  - Utilisation DeleteJob avec un AWS SDK ou une CLI
  - Utilisation DeleteTable avec un AWS SDK
  - Utilisation GetCrawler avec un AWS SDK
  - Utilisation GetDatabase avec un AWS SDK
  - Utilisation GetDatabases avec un AWS SDK ou une CLI
  - Utilisation GetJob avec un AWS SDK ou une CLI
  - Utilisation GetJobRun avec un AWS SDK ou une CLI
  - Utilisation GetJobRuns avec un AWS SDK ou une CLI
  - Utilisation GetTables avec un AWS SDK ou une CLI
  - Utilisation ListJobs avec un AWS SDK
  - Utilisation StartCrawler avec un AWS SDK ou une CLI
  - Utilisation StartJobRun avec un AWS SDK ou une CLI

Bonjour AWS Glue

Les exemples de code suivants montrent comment démarrer avec AWS Glue.

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
namespace GlueActions;

public class HelloGlue
{
    private static ILogger logger = null!;

    static async Task Main(string[] args)
    {
        // Set up dependency injection for AWS Glue.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
 LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
 LogLevel.Trace))
            .ConfigureServices((_, services) =>
                services.AddAWSService<IAmazonGlue>()
                .AddTransient<GlueWrapper>()
            )
            .Build();

        logger = LoggerFactory.Create(builder => { builder.AddConsole(); })
            .CreateLogger<HelloGlue>();
        var glueClient = host.Services.GetRequiredService<IAmazonGlue>();

        var request = new ListJobsRequest();
```

```
        var jobNames = new List<string>();

        do
        {
            var response = await glueClient.ListJobsAsync(request);
            jobNames.AddRange(response.JobNames);
            request.NextToken = response.NextToken;
        }
        while (request.NextToken is not null);

        Console.Clear();
        Console.WriteLine("Hello, Glue. Let's list your existing Glue Jobs:");
        if (jobNames.Count == 0)
        {
            Console.WriteLine("You don't have any AWS Glue jobs.");
        }
        else
        {
            jobNames.ForEach(Console.WriteLine);
        }
    }
}
```

- Pour plus de détails sur l'API, reportez-vous ListJobsà la section Référence des AWS SDK pour .NET API.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Code pour le CMake fichier CMake Lists.txt.

```
# Set the minimum required version of CMake for this project.
cmake_minimum_required(VERSION 3.13)
```

```
# Set the AWS service components used by this project.
set(SERVICE_COMPONENTS glue)

# Set this project's name.
project("hello_glue")

# Set the C++ standard to use to build this target.
# At least C++ 11 is required for the AWS SDK for C++.
set(CMAKE_CXX_STANDARD 11)

# Use the MSVC variable to determine if this is a Windows build.
set(WINDOWS_BUILD ${MSVC})

if (WINDOWS_BUILD) # Set the location where CMake can find the installed
 libraries for the AWS SDK.
    string(REPLACE ";" "/aws-cpp-sdk-all;" SYSTEM_MODULE_PATH
 "${CMAKE_SYSTEM_PREFIX_PATH}/aws-cpp-sdk-all")
    list(APPEND CMAKE_PREFIX_PATH ${SYSTEM_MODULE_PATH})
endif ()

# Find the AWS SDK for C++ package.
find_package(AWSSDK REQUIRED COMPONENTS ${SERVICE_COMPONENTS})

if (WINDOWS_BUILD AND AWSSDK_INSTALL_AS_SHARED_LIBS)
     # Copy relevant AWS SDK for C++ libraries into the current binary directory
 for running and debugging.

     # set(BIN_SUB_DIR "/Debug") # if you are building from the command line you
 may need to uncomment this
                                     # and set the proper subdirectory to the
 executables' location.

     AWSSDK_CPY_DYN_LIBS(SERVICE_COMPONENTS ""
 ${CMAKE_CURRENT_BINARY_DIR}${BIN_SUB_DIR})
endif ()

add_executable(${PROJECT_NAME}
        hello_glue.cpp)

target_link_libraries(${PROJECT_NAME}
        ${AWSSDK_LINK_LIBRARIES})
```

Code pour le fichier source hello_glue.cpp.

```
#include <aws/core/Aws.h>
#include <aws/glue/GlueClient.h>
#include <aws/glue/model/ListJobsRequest.h>
#include <iostream>

/*
 *  A "Hello Glue" starter application which initializes an AWS Glue client and
 lists the
 *  AWS Glue job definitions.
 *
 *  main function
 *
 *  Usage: 'hello_glue'
 *
 */

int main(int argc, char **argv) {
    Aws::SDKOptions options;
    // Optionally change the log level for debugging.
//   options.loggingOptions.logLevel = Utils::Logging::LogLevel::Debug;
    Aws::InitAPI(options); // Should only be called once.
    int result = 0;
    {
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

        Aws::Glue::GlueClient glueClient(clientConfig);

        std::vector<Aws::String> jobs;

        Aws::String nextToken;  // Used for pagination.
        do {
            Aws::Glue::Model::ListJobsRequest listJobsRequest;
            if (!nextToken.empty()) {
                listJobsRequest.SetNextToken(nextToken);
            }

            Aws::Glue::Model::ListJobsOutcome listRunsOutcome =
 glueClient.ListJobs(
                    listJobsRequest);
```

```
            if (listRunsOutcome.IsSuccess()) {
                const std::vector<Aws::String> &jobNames =
 listRunsOutcome.GetResult().GetJobNames();
                jobs.insert(jobs.end(), jobNames.begin(), jobNames.end());

                nextToken = listRunsOutcome.GetResult().GetNextToken();
            } else {
                std::cerr << "Error listing jobs. "
                          << listRunsOutcome.GetError().GetMessage()
                          << std::endl;
                result = 1;
                break;
            }
        } while (!nextToken.empty());

        std::cout << "Your account has " << jobs.size() << " jobs."
                  << std::endl;
        for (size_t i = 0; i < jobs.size(); ++i) {
            std::cout << "   " << i + 1 << ". " << jobs[i] << std::endl;
        }
    }
    Aws::ShutdownAPI(options); // Should only be called once.
    return result;
}
```

- Pour plus de détails sur l'API, reportez-vous ListJobsà la section Référence des AWS SDK pour C++ API.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
package com.example.glue;
```

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.glue.GlueClient;
import software.amazon.awssdk.services.glue.model.ListJobsRequest;
import software.amazon.awssdk.services.glue.model.ListJobsResponse;
import java.util.List;

public class HelloGlue {
    public static void main(String[] args) {
        GlueClient glueClient = GlueClient.builder()
                .region(Region.US_EAST_1)
                .build();

        listJobs(glueClient);
    }

    public static void listJobs(GlueClient glueClient) {
        ListJobsRequest request = ListJobsRequest.builder()
                .maxResults(10)
                .build();
        ListJobsResponse response = glueClient.listJobs(request);
        List<String> jobList = response.jobNames();
        jobList.forEach(job -> {
            System.out.println("Job Name: " + job);
        });
    }
}
```

- Pour plus de détails sur l'API, reportez-vous ListJobsà la section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import { ListJobsCommand, GlueClient } from "@aws-sdk/client-glue";

const client = new GlueClient({});

export const main = async () => {
  const command = new ListJobsCommand({});

  const { JobNames } = await client.send(command);
  const formattedJobNames = JobNames.join("\n");
  console.log("Job names: ");
  console.log(formattedJobNames);
  return JobNames;
};
```

- Pour plus de détails sur l'API, reportez-vous ListJobsà la section Référence des AWS SDK pour JavaScript API.

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import boto3
from botocore.exceptions import ClientError


def hello_glue():
    """
    Lists the job definitions in your AWS Glue account, using the AWS SDK for
 Python (Boto3).
    """
    try:
        # Create the Glue client
        glue = boto3.client("glue")
```

```
        # List the jobs, limiting the results to 10 per page
        paginator = glue.get_paginator("get_jobs")
        response_iterator = paginator.paginate(
            PaginationConfig={"MaxItems": 10, "PageSize": 10}
        )

        # Print the job names
        print("Here are the jobs in your account:")
        for page in response_iterator:
            for job in page["Jobs"]:
                print(f"\t{job['Name']}")

    except ClientError as e:
        print(f"Error: {e}")


if __name__ == "__main__":
    hello_glue()
```

- Pour plus de détails sur l'API, consultez ListJobsle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
require 'aws-sdk-glue'
require 'logger'

# GlueManager is a class responsible for managing AWS Glue operations
# such as listing all Glue jobs in the current AWS account.
```

```
class GlueManager
  def initialize(client)
    @client = client
    @logger = Logger.new($stdout)
  end

  # Lists and prints all Glue jobs in the current AWS account.
  def list_jobs
    @logger.info('Here are the Glue jobs in your account:')

    paginator = @client.get_jobs(max_results: 10)
    jobs = []

    paginator.each_page do |page|
      jobs.concat(page.jobs)
    end

    if jobs.empty?
      @logger.info("You don't have any Glue jobs.")
    else
      jobs.each do |job|
        @logger.info("- #{job.name}")
      end
    end
  end
end

if $PROGRAM_NAME == __FILE__
  glue_client = Aws::Glue::Client.new
  manager = GlueManager.new(glue_client)
  manager.list_jobs
end
```

- Pour plus de détails sur l'API, reportez-vous ListJobsà la section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
        let mut list_jobs = glue.list_jobs().into_paginator().send();
        while let Some(list_jobs_output) = list_jobs.next().await {
            match list_jobs_output {
                Ok(list_jobs) => {
                    let names = list_jobs.job_names();
                    info!(?names, "Found these jobs")
                }
                Err(err) => return Err(GlueMvpError::from_glue_sdk(err)),
            }
        }
```

- Pour plus de détails sur l'API, voir ListJobsla section de référence de l'API AWS SDK for Rust.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Apprenez les bases AWS Glue d'un AWS SDK

Les exemples de code suivants montrent comment :

- Créez un Crawler qui indexe un compartiment Amazon S3 public et génère une base de données de métadonnées au format CSV.
- Répertoriez les informations relatives aux bases de données et aux tables de votre AWS Glue Data Catalog.
- Créez une tâche pour extraire les données CSV du compartiment S3, transformer les données et charger la sortie au format JSON dans un autre compartiment S3.

- Répertoriez les informations relatives aux exécutions de tâches, visualisez les données transformées et nettoyez les ressources.

Pour plus d'informations, consultez Tutoriel : prise en main de AWS Glue Studio.

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez une classe qui englobe les AWS Glue fonctions utilisées dans le scénario.

```
using System.Net;

namespace GlueActions;

public class GlueWrapper
{
    private readonly IAmazonGlue _amazonGlue;

    /// <summary>
    /// Constructor for the AWS Glue actions wrapper.
    /// </summary>
    /// <param name="amazonGlue"></param>
    public GlueWrapper(IAmazonGlue amazonGlue)
    {
        _amazonGlue = amazonGlue;
    }

    /// <summary>
    /// Create an AWS Glue crawler.
    /// </summary>
    /// <param name="crawlerName">The name for the crawler.</param>
    /// <param name="crawlerDescription">A description of the crawler.</param>
    /// <param name="role">The AWS Identity and Access Management (IAM) role to
    /// be assumed by the crawler.</param>
```

```
   /// <param name="schedule">The schedule on which the crawler will be
executed.</param>
   /// <param name="s3Path">The path to the Amazon Simple Storage Service
(Amazon S3)
   /// bucket where the Python script has been stored.</param>
   /// <param name="dbName">The name to use for the database that will be
   /// created by the crawler.</param>
   /// <returns>A Boolean value indicating the success of the action.</returns>
   public async Task<bool> CreateCrawlerAsync(
       string crawlerName,
       string crawlerDescription,
       string role,
       string schedule,
       string s3Path,
       string dbName)
   {
       var s3Target = new S3Target
       {
           Path = s3Path,
       };

       var targetList = new List<S3Target>
       {
           s3Target,
       };

       var targets = new CrawlerTargets
       {
           S3Targets = targetList,
       };

       var crawlerRequest = new CreateCrawlerRequest
       {
           DatabaseName = dbName,
           Name = crawlerName,
           Description = crawlerDescription,
           Targets = targets,
           Role = role,
           Schedule = schedule,
       };

       var response = await _amazonGlue.CreateCrawlerAsync(crawlerRequest);
       return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
   }
```

```
   /// <summary>
   /// Create an AWS Glue job.
   /// </summary>
   /// <param name="jobName">The name of the job.</param>
   /// <param name="roleName">The name of the IAM role to be assumed by
   /// the job.</param>
   /// <param name="description">A description of the job.</param>
   /// <param name="scriptUrl">The URL to the script.</param>
   /// <returns>A Boolean value indicating the success of the action.</returns>
   public async Task<bool> CreateJobAsync(string dbName, string tableName,
string bucketUrl, string jobName, string roleName, string description, string
scriptUrl)
   {
       var command = new JobCommand
       {
           PythonVersion = "3",
           Name = "glueetl",
           ScriptLocation = scriptUrl,
       };

       var arguments = new Dictionary<string, string>
       {
           { "--input_database", dbName },
           { "--input_table", tableName },
           { "--output_bucket_url", bucketUrl }
       };

       var request = new CreateJobRequest
       {
           Command = command,
           DefaultArguments = arguments,
           Description = description,
           GlueVersion = "3.0",
           Name = jobName,
           NumberOfWorkers = 10,
           Role = roleName,
           WorkerType = "G.1X"
       };

       var response = await _amazonGlue.CreateJobAsync(request);
       return response.HttpStatusCode == HttpStatusCode.OK;
   }
```

```
    /// <summary>
    /// Delete an AWS Glue crawler.
    /// </summary>
    /// <param name="crawlerName">The name of the crawler.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteCrawlerAsync(string crawlerName)
    {
        var response = await _amazonGlue.DeleteCrawlerAsync(new
DeleteCrawlerRequest { Name = crawlerName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete the AWS Glue database.
    /// </summary>
    /// <param name="dbName">The name of the database.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteDatabaseAsync(string dbName)
    {
        var response = await _amazonGlue.DeleteDatabaseAsync(new
DeleteDatabaseRequest { Name = dbName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an AWS Glue job.
    /// </summary>
    /// <param name="jobName">The name of the job.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteJobAsync(string jobName)
    {
        var response = await _amazonGlue.DeleteJobAsync(new DeleteJobRequest
{ JobName = jobName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete a table from an AWS Glue database.
    /// </summary>
```

```
    /// <param name="tableName">The table to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteTableAsync(string dbName, string tableName)
    {
        var response = await _amazonGlue.DeleteTableAsync(new DeleteTableRequest
{ Name = tableName, DatabaseName = dbName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Get information about an AWS Glue crawler.
    /// </summary>
    /// <param name="crawlerName">The name of the crawler.</param>
    /// <returns>A Crawler object describing the crawler.</returns>
    public async Task<Crawler?> GetCrawlerAsync(string crawlerName)
    {
        var crawlerRequest = new GetCrawlerRequest
        {
            Name = crawlerName,
        };

        var response = await _amazonGlue.GetCrawlerAsync(crawlerRequest);
        if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
        {
            var databaseName = response.Crawler.DatabaseName;
            Console.WriteLine($"{crawlerName} has the database {databaseName}");
            return response.Crawler;
        }

        Console.WriteLine($"No information regarding {crawlerName} could be
found.");
        return null;
    }


    /// <summary>
    /// Get information about the state of an AWS Glue crawler.
    /// </summary>
    /// <param name="crawlerName">The name of the crawler.</param>
    /// <returns>A value describing the state of the crawler.</returns>
    public async Task<CrawlerState> GetCrawlerStateAsync(string crawlerName)
    {
        var response = await _amazonGlue.GetCrawlerAsync(
```

```
            new GetCrawlerRequest { Name = crawlerName });
        return response.Crawler.State;
    }


    /// <summary>
    /// Get information about an AWS Glue database.
    /// </summary>
    /// <param name="dbName">The name of the database.</param>
    /// <returns>A Database object containing information about the database.</
returns>
    public async Task<Database> GetDatabaseAsync(string dbName)
    {
        var databasesRequest = new GetDatabaseRequest
        {
            Name = dbName,
        };

        var response = await _amazonGlue.GetDatabaseAsync(databasesRequest);
        return response.Database;
    }


    /// <summary>
    /// Get information about a specific AWS Glue job run.
    /// </summary>
    /// <param name="jobName">The name of the job.</param>
    /// <param name="jobRunId">The Id of the job run.</param>
    /// <returns>A JobRun object with information about the job run.</returns>
    public async Task<JobRun> GetJobRunAsync(string jobName, string jobRunId)
    {
        var response = await _amazonGlue.GetJobRunAsync(new GetJobRunRequest
 { JobName = jobName, RunId = jobRunId });
        return response.JobRun;
    }


    /// <summary>
    /// Get information about all AWS Glue runs of a specific job.
    /// </summary>
    /// <param name="jobName">The name of the job.</param>
    /// <returns>A list of JobRun objects.</returns>
    public async Task<List<JobRun>> GetJobRunsAsync(string jobName)
    {
```

```
        var jobRuns = new List<JobRun>();

        var request = new GetJobRunsRequest
        {
            JobName = jobName,
        };

        // No need to loop to get all the log groups--the SDK does it for us
behind the scenes
        var paginatorForJobRuns =
            _amazonGlue.Paginators.GetJobRuns(request);

        await foreach (var response in paginatorForJobRuns.Responses)
        {
            response.JobRuns.ForEach(jobRun =>
            {
                jobRuns.Add(jobRun);
            });
        }

        return jobRuns;
    }


    /// <summary>
    /// Get a list of tables for an AWS Glue database.
    /// </summary>
    /// <param name="dbName">The name of the database.</param>
    /// <returns>A list of Table objects.</returns>
    public async Task<List<Table>> GetTablesAsync(string dbName)
    {
        var request = new GetTablesRequest { DatabaseName = dbName };
        var tables = new List<Table>();

        // Get a paginator for listing the tables.
        var tablePaginator = _amazonGlue.Paginators.GetTables(request);

        await foreach (var response in tablePaginator.Responses)
        {
            tables.AddRange(response.TableList);
        }

        return tables;
    }
```

```
    /// <summary>
    /// List AWS Glue jobs using a paginator.
    /// </summary>
    /// <returns>A list of AWS Glue job names.</returns>
    public async Task<List<string>> ListJobsAsync()
    {
        var jobNames = new List<string>();

        var listJobsPaginator = _amazonGlue.Paginators.ListJobs(new
ListJobsRequest { MaxResults = 10 });
        await foreach (var response in listJobsPaginator.Responses)
        {
            jobNames.AddRange(response.JobNames);
        }

        return jobNames;
    }


    /// <summary>
    /// Start an AWS Glue crawler.
    /// </summary>
    /// <param name="crawlerName">The name of the crawler.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> StartCrawlerAsync(string crawlerName)
    {
        var crawlerRequest = new StartCrawlerRequest
        {
            Name = crawlerName,
        };

        var response = await _amazonGlue.StartCrawlerAsync(crawlerRequest);

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Start an AWS Glue job run.
    /// </summary>
    /// <param name="jobName">The name of the job.</param>
    /// <returns>A string representing the job run Id.</returns>
```

```
    public async Task<string> StartJobRunAsync(
        string jobName,
        string inputDatabase,
        string inputTable,
        string bucketName)
    {
        var request = new StartJobRunRequest
        {
            JobName = jobName,
            Arguments = new Dictionary<string, string>
            {
                {"--input_database", inputDatabase},
                {"--input_table", inputTable},
                {"--output_bucket_url", $"s3://{bucketName}/"}
            }
        };

        var response = await _amazonGlue.StartJobRunAsync(request);
        return response.JobRunId;
    }

}
```

Créez une classe qui exécute le scénario.

```
global using Amazon.Glue;
global using GlueActions;
global using Microsoft.Extensions.Configuration;
global using Microsoft.Extensions.DependencyInjection;
global using Microsoft.Extensions.Hosting;
global using Microsoft.Extensions.Logging;
global using Microsoft.Extensions.Logging.Console;
global using Microsoft.Extensions.Logging.Debug;



using Amazon.Glue.Model;
using Amazon.S3;
using Amazon.S3.Model;

namespace GlueBasics;
```

```
public class GlueBasics
{
    private static ILogger logger = null!;
    private static IConfiguration _configuration = null!;

    static async Task Main(string[] args)
    {
        // Set up dependency injection for AWS Glue.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
 LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
 LogLevel.Trace))
            .ConfigureServices((_, services) =>
            services.AddAWSService<IAmazonGlue>()
            .AddTransient<GlueWrapper>()
            .AddTransient<UiWrapper>()
            )
            .Build();

        logger = LoggerFactory.Create(builder => { builder.AddConsole(); })
        .CreateLogger<GlueBasics>();

        _configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load settings from .json file.
            .AddJsonFile("settings.local.json",
                true) // Optionally load local settings.
            .Build();

        // These values are stored in settings.json
        // Once you have run the CDK script to deploy the resources,
        // edit the file to set "BucketName", "RoleName", and "ScriptURL"
        // to the appropriate values. Also set "CrawlerName" to the name
        // you want to give the crawler when it is created.
        string bucketName = _configuration["BucketName"]!;
        string bucketUrl = _configuration["BucketUrl"]!;
        string crawlerName = _configuration["CrawlerName"]!;
        string roleName = _configuration["RoleName"]!;
        string sourceData = _configuration["SourceData"]!;
        string dbName = _configuration["DbName"]!;
```

```
        string cron = _configuration["Cron"]!;
        string scriptUrl = _configuration["ScriptURL"]!;
        string jobName = _configuration["JobName"]!;

        var wrapper = host.Services.GetRequiredService<GlueWrapper>();
        var uiWrapper = host.Services.GetRequiredService<UiWrapper>();

        uiWrapper.DisplayOverview();
        uiWrapper.PressEnter();

        // Create the crawler and wait for it to be ready.
        uiWrapper.DisplayTitle("Create AWS Glue crawler");
        Console.WriteLine("Let's begin by creating the AWS Glue crawler.");

        var crawlerDescription = "Crawler created for the AWS Glue Basics
scenario.";
        var crawlerCreated = await wrapper.CreateCrawlerAsync(crawlerName,
crawlerDescription, roleName, cron, sourceData, dbName);
        if (crawlerCreated)
        {
            Console.WriteLine($"The crawler: {crawlerName} has been created. Now
let's wait until it's ready.");
            CrawlerState crawlerState;
            do
            {
                crawlerState = await wrapper.GetCrawlerStateAsync(crawlerName);
            }
            while (crawlerState != "READY");
            Console.WriteLine($"The crawler {crawlerName} is now ready for
use.");
        }
        else
        {
            Console.WriteLine($"Couldn't create crawler {crawlerName}.");
            return; // Exit the application.
        }

        uiWrapper.DisplayTitle("Start AWS Glue crawler");
        Console.WriteLine("Now let's wait until the crawler has successfully
started.");
        var crawlerStarted = await wrapper.StartCrawlerAsync(crawlerName);
        if (crawlerStarted)
        {
            CrawlerState crawlerState;
```

```
            do
            {
                crawlerState = await wrapper.GetCrawlerStateAsync(crawlerName);
            }
            while (crawlerState != "READY");
            Console.WriteLine($"The crawler {crawlerName} is now ready for
use.");
        }
        else
        {
            Console.WriteLine($"Couldn't start the crawler {crawlerName}.");
            return; // Exit the application.
        }

        uiWrapper.PressEnter();

        Console.WriteLine($"\nLet's take a look at the database: {dbName}");
        var database = await wrapper.GetDatabaseAsync(dbName);

        if (database != null)
        {
            uiWrapper.DisplayTitle($"{database.Name} Details");
            Console.WriteLine($"{database.Name} created on
{database.CreateTime}");
            Console.WriteLine(database.Description);
        }

        uiWrapper.PressEnter();

        var tables = await wrapper.GetTablesAsync(dbName);
        if (tables.Count > 0)
        {
            tables.ForEach(table =>
            {
                Console.WriteLine($"{table.Name}\tCreated:
{table.CreateTime}\tUpdated: {table.UpdateTime}");
            });
        }

        uiWrapper.PressEnter();

        uiWrapper.DisplayTitle("Create AWS Glue job");
        Console.WriteLine("Creating a new AWS Glue job.");
        var description = "An AWS Glue job created using the AWS SDK for .NET";
```

```
        await wrapper.CreateJobAsync(dbName, tables[0].Name, bucketUrl, jobName,
roleName, description, scriptUrl);

        uiWrapper.PressEnter();

        uiWrapper.DisplayTitle("Starting AWS Glue job");
        Console.WriteLine("Starting the new AWS Glue job...");
        var jobRunId = await wrapper.StartJobRunAsync(jobName, dbName,
tables[0].Name, bucketName);
        var jobRunComplete = false;
        var jobRun = new JobRun();
        do
        {
            jobRun = await wrapper.GetJobRunAsync(jobName, jobRunId);
            if (jobRun.JobRunState == "SUCCEEDED" || jobRun.JobRunState ==
"STOPPED" ||
                jobRun.JobRunState == "FAILED" || jobRun.JobRunState ==
"TIMEOUT")
            {
                jobRunComplete = true;
            }
        } while (!jobRunComplete);

        uiWrapper.DisplayTitle($"Data in {bucketName}");

        // Get the list of data stored in the S3 bucket.
        var s3Client = new AmazonS3Client();

        var response = await s3Client.ListObjectsAsync(new ListObjectsRequest
{ BucketName = bucketName });
        response.S3Objects.ForEach(s3Object =>
        {
            Console.WriteLine(s3Object.Key);
        });

        uiWrapper.DisplayTitle("AWS Glue jobs");
        var jobNames = await wrapper.ListJobsAsync();
        jobNames.ForEach(jobName =>
        {
            Console.WriteLine(jobName);
        });

        uiWrapper.PressEnter();
```

```
        uiWrapper.DisplayTitle("Get AWS Glue job run information");
        Console.WriteLine("Getting information about the AWS Glue job.");
        var jobRuns = await wrapper.GetJobRunsAsync(jobName);

        jobRuns.ForEach(jobRun =>
        {

 Console.WriteLine($"{jobRun.JobName}\t{jobRun.JobRunState}\t{jobRun.CompletedOn}");
        });

        uiWrapper.PressEnter();

        uiWrapper.DisplayTitle("Deleting resources");
        Console.WriteLine("Deleting the AWS Glue job used by the example.");
        await wrapper.DeleteJobAsync(jobName);

        Console.WriteLine("Deleting the tables from the database.");
        tables.ForEach(async table =>
        {
            await wrapper.DeleteTableAsync(dbName, table.Name);
        });

        Console.WriteLine("Deleting the database.");
        await wrapper.DeleteDatabaseAsync(dbName);

        Console.WriteLine("Deleting the AWS Glue crawler.");
        await wrapper.DeleteCrawlerAsync(crawlerName);

        Console.WriteLine("The AWS Glue scenario has completed.");
        uiWrapper.PressEnter();
    }
}


namespace GlueBasics;

public class UiWrapper
{
    public readonly string SepBar = new string('-', Console.WindowWidth);

    /// <summary>
    /// Show information about the scenario.
    /// </summary>
    public void DisplayOverview()
```

```
    {
        Console.Clear();
        DisplayTitle("Amazon Glue: get started with crawlers and jobs");

        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t 1. Create a crawler, pass it the IAM role and the
URL to the public S3 bucket that contains the source data");
        Console.WriteLine("\t 2. Start the crawler.");
        Console.WriteLine("\t 3. Get the database created by the crawler and the
tables in the database.");
        Console.WriteLine("\t 4. Create a job.");
        Console.WriteLine("\t 5. Start a job run.");
        Console.WriteLine("\t 6. Wait for the job run to complete.");
        Console.WriteLine("\t 7. Show the data stored in the bucket.");
        Console.WriteLine("\t 8. List jobs for the account.");
        Console.WriteLine("\t 9. Get job run details for the job that was run.");
        Console.WriteLine("\t10. Delete the demo job.");
        Console.WriteLine("\t11. Delete the database and tables created for the
demo.");
        Console.WriteLine("\t12. Delete the crawler.");
    }

    /// <summary>
    /// Display a message and wait until the user presses enter.
    /// </summary>
    public void PressEnter()
    {
        Console.Write("\nPlease press <Enter> to continue. ");
        _ = Console.ReadLine();
    }

    /// <summary>
    /// Pad a string with spaces to center it on the console display.
    /// </summary>
    /// <param name="strToCenter">The string to center on the screen.</param>
    /// <returns>The string padded to make it center on the screen.</returns>
    public string CenterString(string strToCenter)
    {
        var padAmount = (Console.WindowWidth - strToCenter.Length) / 2;
        var leftPad = new string(' ', padAmount);
        return $"{leftPad}{strToCenter}";
    }

    /// <summary>
```

```
    /// Display a line of hyphens, the centered text of the title and another
    /// line of hyphens.
    /// </summary>
    /// <param name="strTitle">The string to be displayed.</param>
    public void DisplayTitle(string strTitle)
    {
        Console.WriteLine(SepBar);
        Console.WriteLine(CenterString(strTitle));
        Console.WriteLine(SepBar);
    }
}
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des API du kit AWS SDK pour .NET .
  - CreateCrawler
  - CreateJob
  - DeleteCrawler
  - DeleteDatabase
  - DeleteJob
  - DeleteTable
  - GetCrawler
  - GetDatabase
  - GetDatabases
  - GetJob
  - GetJobRun
  - GetJobRuns
  - GetTables
  - ListJobs
  - StartCrawler
  - StartJobRun

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
//! Scenario which demonstrates using AWS Glue to add a crawler and run a job.
/*!
 \\sa runGettingStartedWithGlueScenario()
 \param bucketName: An S3 bucket created in the setup.
 \param roleName: An AWS Identity and Access Management (IAM) role created in the
 setup.
 \param clientConfig: AWS client configuration.
 \return bool: Successful completion.
 */

bool AwsDoc::Glue::runGettingStartedWithGlueScenario(const Aws::String
 &bucketName,
                                                      const Aws::String &roleName,
                                                      const
 Aws::Client::ClientConfiguration &clientConfig) {
    Aws::Glue::GlueClient client(clientConfig);

    Aws::String roleArn;
    if (!getRoleArn(roleName, roleArn, clientConfig)) {
        std::cerr << "Error getting role ARN for role." << std::endl;
        return false;
    }

    // 1. Upload the job script to the S3 bucket.
    {
        std::cout << "Uploading the job script '"
                  << AwsDoc::Glue::PYTHON_SCRIPT
                  << "'." << std::endl;

        if (!AwsDoc::Glue::uploadFile(bucketName,
                                      AwsDoc::Glue::PYTHON_SCRIPT_PATH,
                                      AwsDoc::Glue::PYTHON_SCRIPT,
```

```
                                        clientConfig)) {
            std::cerr << "Error uploading the job file." << std::endl;
            return false;
        }
    }

    // 2. Create a crawler.
    {
        Aws::Glue::Model::S3Target s3Target;
        s3Target.SetPath("s3://crawler-public-us-east-1/flight/2016/csv");
        Aws::Glue::Model::CrawlerTargets crawlerTargets;
        crawlerTargets.AddS3Targets(s3Target);

        Aws::Glue::Model::CreateCrawlerRequest request;
        request.SetTargets(crawlerTargets);
        request.SetName(CRAWLER_NAME);
        request.SetDatabaseName(CRAWLER_DATABASE_NAME);
        request.SetTablePrefix(CRAWLER_DATABASE_PREFIX);
        request.SetRole(roleArn);

        Aws::Glue::Model::CreateCrawlerOutcome outcome =
client.CreateCrawler(request);

        if (outcome.IsSuccess()) {
            std::cout << "Successfully created the crawler." << std::endl;
        }
        else {
            std::cerr << "Error creating a crawler. " <<
outcome.GetError().GetMessage()
                      << std::endl;
            deleteAssets("", CRAWLER_DATABASE_NAME, "", bucketName,
clientConfig);
            return false;
        }
    }

    // 3. Get a crawler.
    {
        Aws::Glue::Model::GetCrawlerRequest request;
        request.SetName(CRAWLER_NAME);

        Aws::Glue::Model::GetCrawlerOutcome outcome = client.GetCrawler(request);

        if (outcome.IsSuccess()) {
```

```
            Aws::Glue::Model::CrawlerState crawlerState =
outcome.GetResult().GetCrawler().GetState();
            std::cout << "Retrieved crawler with state " <<

Aws::Glue::Model::CrawlerStateMapper::GetNameForCrawlerState(
                              crawlerState)
                      << "." << std::endl;
        }
        else {
            std::cerr << "Error retrieving a crawler.  "
                      << outcome.GetError().GetMessage() << std::endl;
            deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, "", bucketName,
                         clientConfig);
            return false;
        }
    }

    // 4. Start a crawler.
    {
        Aws::Glue::Model::StartCrawlerRequest request;
        request.SetName(CRAWLER_NAME);

        Aws::Glue::Model::StartCrawlerOutcome outcome =
client.StartCrawler(request);


        if (outcome.IsSuccess() || (Aws::Glue::GlueErrors::CRAWLER_RUNNING ==
                                    outcome.GetError().GetErrorType())) {
            if (!outcome.IsSuccess()) {
                std::cout << "Crawler was already started." << std::endl;
            }
            else {
                std::cout << "Successfully started crawler." << std::endl;
            }

            std::cout << "This may take a while to run." << std::endl;

            Aws::Glue::Model::CrawlerState crawlerState =
Aws::Glue::Model::CrawlerState::NOT_SET;
            int iterations = 0;
            while (Aws::Glue::Model::CrawlerState::READY != crawlerState) {
                std::this_thread::sleep_for(std::chrono::seconds(1));
                ++iterations;
                if ((iterations % 10) == 0) { // Log status every 10 seconds.
```

```
                    std::cout << "Crawler status " <<

Aws::Glue::Model::CrawlerStateMapper::GetNameForCrawlerState(
                                       crawlerState)
                              << ". After " << iterations
                              << " seconds elapsed."
                              << std::endl;
                }
                Aws::Glue::Model::GetCrawlerRequest getCrawlerRequest;
                getCrawlerRequest.SetName(CRAWLER_NAME);

                Aws::Glue::Model::GetCrawlerOutcome getCrawlerOutcome =
client.GetCrawler(
                        getCrawlerRequest);

                if (getCrawlerOutcome.IsSuccess()) {
                    crawlerState =
getCrawlerOutcome.GetResult().GetCrawler().GetState();
                }
                else {
                    std::cerr << "Error getting crawler.  "
                              << getCrawlerOutcome.GetError().GetMessage() <<
std::endl;
                    break;
                }
            }

            if (Aws::Glue::Model::CrawlerState::READY == crawlerState) {
                std::cout << "Crawler finished running after " << iterations
                          << " seconds."
                          << std::endl;
            }
        }
        else {
            std::cerr << "Error starting a crawler.  "
                      << outcome.GetError().GetMessage()
                      << std::endl;

            deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, "", bucketName,
                         clientConfig);
            return false;
        }
    }
```

```
   // 5. Get a database.
   {
       Aws::Glue::Model::GetDatabaseRequest request;
       request.SetName(CRAWLER_DATABASE_NAME);

       Aws::Glue::Model::GetDatabaseOutcome outcome =
client.GetDatabase(request);

       if (outcome.IsSuccess()) {
           const Aws::Glue::Model::Database &database =
outcome.GetResult().GetDatabase();

           std::cout << "Successfully retrieve the database\n" <<
                     database.Jsonize().View().WriteReadable() << "'." <<
std::endl;
       }
       else {
           std::cerr << "Error getting the database.  "
                     << outcome.GetError().GetMessage() << std::endl;
           deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, "", bucketName,
                        clientConfig);
           return false;
       }
   }

   // 6. Get tables.
   Aws::String tableName;
   {
       Aws::Glue::Model::GetTablesRequest request;
       request.SetDatabaseName(CRAWLER_DATABASE_NAME);
       std::vector<Aws::Glue::Model::Table> all_tables;
       Aws::String nextToken; // Used for pagination.
       do {
           Aws::Glue::Model::GetTablesOutcome outcome =
client.GetTables(request);

           if (outcome.IsSuccess()) {
               const std::vector<Aws::Glue::Model::Table> &tables =
outcome.GetResult().GetTableList();
               all_tables.insert(all_tables.end(), tables.begin(),
tables.end());
               nextToken = outcome.GetResult().GetNextToken();
           }
           else {
```

```
                std::cerr << "Error getting the tables. "
                          << outcome.GetError().GetMessage()
                          << std::endl;
                deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, "", bucketName,
                             clientConfig);
                return false;
            }
        } while (!nextToken.empty());

        std::cout << "The database contains " << all_tables.size()
                  << (all_tables.size() == 1 ?
                      " table." : "tables.") << std::endl;
        std::cout << "Here is a list of the tables in the database.";
        for (size_t index = 0; index < all_tables.size(); ++index) {
            std::cout << "    " << index + 1 << ":  " <<
all_tables[index].GetName()
                      << std::endl;
        }

        if (!all_tables.empty()) {
            int tableIndex = askQuestionForIntRange(
                    "Enter an index to display the database detail ",
                    1, static_cast<int>(all_tables.size()));
            std::cout << all_tables[tableIndex -
1].Jsonize().View().WriteReadable()
                      << std::endl;

            tableName = all_tables[tableIndex - 1].GetName();
        }
    }

    // 7. Create a job.
    {
        Aws::Glue::Model::CreateJobRequest request;
        request.SetName(JOB_NAME);
        request.SetRole(roleArn);
        request.SetGlueVersion(GLUE_VERSION);

        Aws::Glue::Model::JobCommand command;
        command.SetName(JOB_COMMAND_NAME);
        command.SetPythonVersion(JOB_PYTHON_VERSION);
        command.SetScriptLocation(
                Aws::String("s3://") + bucketName + "/" + PYTHON_SCRIPT);
        request.SetCommand(command);
```

```
        Aws::Glue::Model::CreateJobOutcome outcome = client.CreateJob(request);

        if (outcome.IsSuccess()) {
            std::cout << "Successfully created the job." << std::endl;
        }
        else {
            std::cerr << "Error creating the job. " <<
outcome.GetError().GetMessage()
                      << std::endl;
            deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, "", bucketName,
                         clientConfig);
            return false;
        }
    }

    // 8. Start a job run.
    {
        Aws::Glue::Model::StartJobRunRequest request;
        request.SetJobName(JOB_NAME);

        Aws::Map<Aws::String, Aws::String> arguments;
        arguments["--input_database"] = CRAWLER_DATABASE_NAME;
        arguments["--input_table"] = tableName;
        arguments["--output_bucket_url"] = Aws::String("s3://") + bucketName +
"/";
        request.SetArguments(arguments);

        Aws::Glue::Model::StartJobRunOutcome outcome =
client.StartJobRun(request);

        if (outcome.IsSuccess()) {
            std::cout << "Successfully started the job." << std::endl;

            Aws::String jobRunId = outcome.GetResult().GetJobRunId();

            int iterator = 0;
            bool done = false;
            while (!done) {
                ++iterator;
                std::this_thread::sleep_for(std::chrono::seconds(1));
                Aws::Glue::Model::GetJobRunRequest jobRunRequest;
                jobRunRequest.SetJobName(JOB_NAME);
                jobRunRequest.SetRunId(jobRunId);
```

```
                Aws::Glue::Model::GetJobRunOutcome jobRunOutcome =
client.GetJobRun(
                        jobRunRequest);

                if (jobRunOutcome.IsSuccess()) {
                    const Aws::Glue::Model::JobRun &jobRun =
jobRunOutcome.GetResult().GetJobRun();
                    Aws::Glue::Model::JobRunState jobRunState =
jobRun.GetJobRunState();

                    if ((jobRunState == Aws::Glue::Model::JobRunState::STOPPED)
||
                        (jobRunState == Aws::Glue::Model::JobRunState::FAILED) ||
                        (jobRunState == Aws::Glue::Model::JobRunState::TIMEOUT))
{
                        std::cerr << "Error running job. "
                                  << jobRun.GetErrorMessage()
                                  << std::endl;
                        deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME,
JOB_NAME,
                                     bucketName,
                                     clientConfig);
                        return false;
                    }
                    else if (jobRunState ==
                             Aws::Glue::Model::JobRunState::SUCCEEDED) {
                        std::cout << "Job run succeeded after  " << iterator <<
                                  " seconds elapsed." << std::endl;
                        done = true;
                    }
                    else if ((iterator % 10) == 0) { // Log status every 10
seconds.
                        std::cout << "Job run status " <<

Aws::Glue::Model::JobRunStateMapper::GetNameForJobRunState(
                                         jobRunState) <<
                                  ". " << iterator <<
                                  " seconds elapsed." << std::endl;
                    }
                }
                else {
                    std::cerr << "Error retrieving job run state. "
                              << jobRunOutcome.GetError().GetMessage()
```

```
                            << std::endl;
                    deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, JOB_NAME,
                                 bucketName, clientConfig);
                    return false;
                }
            }
        }
        else {
            std::cerr << "Error starting a job. " <<
outcome.GetError().GetMessage()
                      << std::endl;
            deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, JOB_NAME,
bucketName,
                         clientConfig);
            return false;
        }
    }

    // 9. List the output data stored in the S3 bucket.
    {
        Aws::S3::S3Client s3Client;
        Aws::S3::Model::ListObjectsV2Request request;
        request.SetBucket(bucketName);
        request.SetPrefix(OUTPUT_FILE_PREFIX);

        Aws::String continuationToken; // Used for pagination.
        std::vector<Aws::S3::Model::Object> allObjects;
        do {
            if (!continuationToken.empty()) {
                request.SetContinuationToken(continuationToken);
            }
            Aws::S3::Model::ListObjectsV2Outcome outcome =
s3Client.ListObjectsV2(
                    request);

            if (outcome.IsSuccess()) {
                const std::vector<Aws::S3::Model::Object> &objects =
                        outcome.GetResult().GetContents();
                allObjects.insert(allObjects.end(), objects.begin(),
objects.end());
                continuationToken =
outcome.GetResult().GetNextContinuationToken();
            }
            else {
```

```
                std::cerr << "Error listing objects. "
                          << outcome.GetError().GetMessage()
                          << std::endl;
                break;
            }
        } while (!continuationToken.empty());

        std::cout << "Data from your job is in " << allObjects.size() <<
                  " files in the S3 bucket, " << bucketName << "." << std::endl;

        for (size_t i = 0; i < allObjects.size(); ++i) {
            std::cout << "    " << i + 1 << ". " << allObjects[i].GetKey()
                      << std::endl;
        }

        int objectIndex = askQuestionForIntRange(
                std::string(
                        "Enter the number of a block to download it and see the
 first ") +
                std::to_string(LINES_OF_RUN_FILE_TO_DISPLAY) +
                " lines of JSON output in the block: ", 1,
                static_cast<int>(allObjects.size()));

        Aws::String objectKey = allObjects[objectIndex - 1].GetKey();

        std::stringstream stringStream;
        if (getObjectFromBucket(bucketName, objectKey, stringStream,
                                clientConfig)) {
            for (int i = 0; i < LINES_OF_RUN_FILE_TO_DISPLAY && stringStream; +
+i) {
                std::string line;
                std::getline(stringStream, line);
                std::cout << "    " << line << std::endl;
            }
        }
        else {
            deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, JOB_NAME,
 bucketName,
                         clientConfig);
            return false;
        }
    }

    // 10. List all the jobs.
```

```
    Aws::String jobName;
    {
        Aws::Glue::Model::ListJobsRequest listJobsRequest;

        Aws::String nextToken;
        std::vector<Aws::String> allJobNames;

        do {
            if (!nextToken.empty()) {
                listJobsRequest.SetNextToken(nextToken);
            }
            Aws::Glue::Model::ListJobsOutcome listRunsOutcome = client.ListJobs(
                    listJobsRequest);

            if (listRunsOutcome.IsSuccess()) {
                const std::vector<Aws::String> &jobNames =
listRunsOutcome.GetResult().GetJobNames();
                allJobNames.insert(allJobNames.end(), jobNames.begin(),
jobNames.end());
                nextToken = listRunsOutcome.GetResult().GetNextToken();
            }
            else {
                std::cerr << "Error listing jobs. "
                          << listRunsOutcome.GetError().GetMessage()
                          << std::endl;
            }
        } while (!nextToken.empty());
        std::cout << "Your account has " << allJobNames.size() << " jobs."
                  << std::endl;
        for (size_t i = 0; i < allJobNames.size(); ++i) {
            std::cout << "   " << i + 1 << ". " << allJobNames[i] << std::endl;
        }
        int jobIndex = askQuestionForIntRange(
                Aws::String("Enter a number between 1 and ") +
                std::to_string(allJobNames.size()) +
                " to see the list of runs for a job: ",
                1, static_cast<int>(allJobNames.size()));

        jobName = allJobNames[jobIndex - 1];
    }

    // 11. Get the job runs for a job.
    Aws::String jobRunID;
    if (!jobName.empty()) {
```

```
        Aws::Glue::Model::GetJobRunsRequest getJobRunsRequest;
        getJobRunsRequest.SetJobName(jobName);

        Aws::String nextToken; // Used for pagination.
        std::vector<Aws::Glue::Model::JobRun> allJobRuns;
        do {
            if (!nextToken.empty()) {
                getJobRunsRequest.SetNextToken(nextToken);
            }
            Aws::Glue::Model::GetJobRunsOutcome jobRunsOutcome =
client.GetJobRuns(
                    getJobRunsRequest);

            if (jobRunsOutcome.IsSuccess()) {
                const std::vector<Aws::Glue::Model::JobRun> &jobRuns =
jobRunsOutcome.GetResult().GetJobRuns();
                allJobRuns.insert(allJobRuns.end(), jobRuns.begin(),
jobRuns.end());

                nextToken = jobRunsOutcome.GetResult().GetNextToken();
            }
            else {
                std::cerr << "Error getting job runs. "
                          << jobRunsOutcome.GetError().GetMessage()
                          << std::endl;
                break;
            }
        } while (!nextToken.empty());

        std::cout << "There are " << allJobRuns.size() << " runs in the job '"
                  <<
                  jobName << "'." << std::endl;

        for (size_t i = 0; i < allJobRuns.size(); ++i) {
            std::cout << "   " << i + 1 << ". " << allJobRuns[i].GetJobName()
                      << std::endl;
        }

        int runIndex = askQuestionForIntRange(
                Aws::String("Enter a number between 1 and ") +
                std::to_string(allJobRuns.size()) +
                " to see details for a run: ",
                1, static_cast<int>(allJobRuns.size()));
        jobRunID = allJobRuns[runIndex - 1].GetId();
```

```
    }

    // 12. Get a single job run.
    if (!jobRunID.empty()) {
        Aws::Glue::Model::GetJobRunRequest jobRunRequest;
        jobRunRequest.SetJobName(jobName);
        jobRunRequest.SetRunId(jobRunID);

        Aws::Glue::Model::GetJobRunOutcome jobRunOutcome = client.GetJobRun(
                jobRunRequest);

        if (jobRunOutcome.IsSuccess()) {
            std::cout << "Displaying the job run JSON description." << std::endl;
            std::cout
                    <<
 jobRunOutcome.GetResult().GetJobRun().Jsonize().View().WriteReadable()
                    << std::endl;
        }
        else {
            std::cerr << "Error get a job run. "
                      << jobRunOutcome.GetError().GetMessage()
                      << std::endl;
        }
    }

    return deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, JOB_NAME,
 bucketName,
                        clientConfig);
}

//! Cleanup routine to delete created assets.
/*!
 \\sa deleteAssets()
 \param crawler: Name of an AWS Glue crawler.
 \param database: The name of an AWS Glue database.
 \param job: The name of an AWS Glue job.
 \param bucketName: The name of an S3 bucket.
 \param clientConfig: AWS client configuration.
 \return bool: Successful completion.
 */
bool AwsDoc::Glue::deleteAssets(const Aws::String &crawler, const Aws::String
 &database,
                                const Aws::String &job, const Aws::String
 &bucketName,
```

```
                                const Aws::Client::ClientConfiguration
&clientConfig) {
   const Aws::Glue::GlueClient client(clientConfig);
   bool result = true;

   // 13. Delete a job.
   if (!job.empty()) {
       Aws::Glue::Model::DeleteJobRequest request;
       request.SetJobName(job);

       Aws::Glue::Model::DeleteJobOutcome outcome = client.DeleteJob(request);


       if (outcome.IsSuccess()) {
           std::cout << "Successfully deleted the job." << std::endl;
       }
       else {
           std::cerr << "Error deleting the job. " <<
outcome.GetError().GetMessage()
                     << std::endl;
           result = false;
       }
   }

   // 14. Delete a database.
   if (!database.empty()) {
       Aws::Glue::Model::DeleteDatabaseRequest request;
       request.SetName(database);

       Aws::Glue::Model::DeleteDatabaseOutcome outcome = client.DeleteDatabase(
               request);

       if (outcome.IsSuccess()) {
           std::cout << "Successfully deleted the database." << std::endl;
       }
       else {
           std::cerr << "Error deleting database. " <<
outcome.GetError().GetMessage()
                     << std::endl;
           result = false;
       }
   }

   // 15. Delete a crawler.
```

```
    if (!crawler.empty()) {
        Aws::Glue::Model::DeleteCrawlerRequest request;
        request.SetName(crawler);

        Aws::Glue::Model::DeleteCrawlerOutcome outcome =
 client.DeleteCrawler(request);

        if (outcome.IsSuccess()) {
            std::cout << "Successfully deleted the crawler." << std::endl;
        }
        else {
            std::cerr << "Error deleting the crawler. "
                      << outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
    }

    // 16. Delete the job script and run data from the S3 bucket.
    result &= AwsDoc::Glue::deleteAllObjectsInS3Bucket(bucketName,
                                                       clientConfig);
    return result;
}

//! Routine which uploads a file to an S3 bucket.
/*!
 \\sa uploadFile()
 \param bucketName: An S3 bucket created in the setup.
 \param filePath: The path of the file to upload.
 \param fileName The name for the uploaded file.
 \param clientConfig: AWS client configuration.
 \return bool: Successful completion.
 */
bool
AwsDoc::Glue::uploadFile(const Aws::String &bucketName,
                         const Aws::String &filePath,
                         const Aws::String &fileName,
                         const Aws::Client::ClientConfiguration &clientConfig) {
    Aws::S3::S3Client s3_client(clientConfig);

    Aws::S3::Model::PutObjectRequest request;
    request.SetBucket(bucketName);
    request.SetKey(fileName);

    std::shared_ptr<Aws::IOStream> inputData =
```

```
            Aws::MakeShared<Aws::FStream>("SampleAllocationTag",
                                          filePath.c_str(),
                                          std::ios_base::in |
 std::ios_base::binary);

    if (!*inputData) {
        std::cerr << "Error unable to read file " << filePath << std::endl;
        return false;
    }

    request.SetBody(inputData);

    Aws::S3::Model::PutObjectOutcome outcome =
            s3_client.PutObject(request);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error: PutObject: " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Added object '" << filePath << "' to bucket '"
                  << bucketName << "'." << std::endl;
    }

    return outcome.IsSuccess();
}

//! Routine which deletes all objects in an S3 bucket.
/*!
 \\sa deleteAllObjectsInS3Bucket()
 \param bucketName: The S3 bucket name.
 \param clientConfig: AWS client configuration.
 \return bool: Successful completion.
 */
bool AwsDoc::Glue::deleteAllObjectsInS3Bucket(const Aws::String &bucketName,
                                              const
 Aws::Client::ClientConfiguration &clientConfig) {
    Aws::S3::S3Client client(clientConfig);
    Aws::S3::Model::ListObjectsV2Request listObjectsRequest;
    listObjectsRequest.SetBucket(bucketName);

    Aws::String continuationToken; // Used for pagination.
    bool result = true;
    do {
```

```
        if (!continuationToken.empty()) {
            listObjectsRequest.SetContinuationToken(continuationToken);
        }

        Aws::S3::Model::ListObjectsV2Outcome listObjectsOutcome =
client.ListObjectsV2(
                listObjectsRequest);

        if (listObjectsOutcome.IsSuccess()) {
            const std::vector<Aws::S3::Model::Object> &objects =
listObjectsOutcome.GetResult().GetContents();
            if (!objects.empty()) {
                Aws::S3::Model::DeleteObjectsRequest deleteObjectsRequest;
                deleteObjectsRequest.SetBucket(bucketName);

                std::vector<Aws::S3::Model::ObjectIdentifier> objectIdentifiers;
                for (const Aws::S3::Model::Object &object: objects) {
                    objectIdentifiers.push_back(
                            Aws::S3::Model::ObjectIdentifier().WithKey(
                                    object.GetKey()));
                }
                Aws::S3::Model::Delete objectsDelete;
                objectsDelete.SetObjects(objectIdentifiers);
                objectsDelete.SetQuiet(true);
                deleteObjectsRequest.SetDelete(objectsDelete);

                Aws::S3::Model::DeleteObjectsOutcome deleteObjectsOutcome =
                        client.DeleteObjects(deleteObjectsRequest);

                if (!deleteObjectsOutcome.IsSuccess()) {
                    std::cerr << "Error deleting objects. " <<
                              deleteObjectsOutcome.GetError().GetMessage() <<
std::endl;
                    result = false;
                    break;
                }
                else {
                    std::cout << "Successfully deleted the objects." <<
std::endl;

                }
            }
            else {
                std::cout << "No objects to delete in '" << bucketName << "'."
```

```
                          << std::endl;
            }

            continuationToken =
 listObjectsOutcome.GetResult().GetNextContinuationToken();
        }
        else {
            std::cerr << "Error listing objects. "
                      << listObjectsOutcome.GetError().GetMessage() << std::endl;
            result = false;
            break;
        }
    } while (!continuationToken.empty());

    return result;
}

//! Routine which retrieves an object from an S3 bucket.
/*!
 \\sa getObjectFromBucket()
 \param bucketName: The S3 bucket name.
 \param objectKey: The object's name.
 \param objectStream: A stream to receive the retrieved data.
 \param clientConfig: AWS client configuration.
 \return bool: Successful completion.
 */
bool AwsDoc::Glue::getObjectFromBucket(const Aws::String &bucketName,
                                       const Aws::String &objectKey,
                                       std::ostream &objectStream,
                                       const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::S3::S3Client client(clientConfig);
    Aws::S3::Model::GetObjectRequest request;
    request.SetBucket(bucketName);
    request.SetKey(objectKey);

    Aws::S3::Model::GetObjectOutcome outcome = client.GetObject(request);


    if (outcome.IsSuccess()) {
        std::cout << "Successfully retrieved '" << objectKey << "'." <<
 std::endl;
        auto &body = outcome.GetResult().GetBody();
        objectStream << body.rdbuf();
```

```
    }
    else {
        std::cerr << "Error retrieving object. " <<
 outcome.GetError().GetMessage()
                  << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des API du kit AWS SDK pour C++ .
  - CreateCrawler
  - CreateJob
  - DeleteCrawler
  - DeleteDatabase
  - DeleteJob
  - DeleteTable
  - GetCrawler
  - GetDatabase
  - GetDatabases
  - GetJob
  - GetJobRun
  - GetJobRuns
  - GetTables
  - ListJobs
  - StartCrawler
  - StartJobRun

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 * <p>
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 *
 * To set up the resources, see this documentation topic:
 *
 * https://docs.aws.amazon.com/glue/latest/ug/tutorial-add-crawler.html
 *
 * This example performs the following tasks:
 *
 * 1. Create a database.
 * 2. Create a crawler.
 * 3. Get a crawler.
 * 4. Start a crawler.
 * 5. Get a database.
 * 6. Get tables.
 * 7. Create a job.
 * 8. Start a job run.
 * 9. List all jobs.
 * 10. Get job runs.
 * 11. Delete a job.
 * 12. Delete a database.
 * 13. Delete a crawler.
 */

public class GlueScenario {
```

```
   public static final String DASHES = new String(new char[80]).replace("\0",
"-");

   public static void main(String[] args) throws InterruptedException {
       final String usage = """

           Usage:
               <iam> <s3Path> <cron> <dbName> <crawlerName> <jobName>
<scriptLocation> <locationUri> <bucketNameSc>\s

           Where:
               iam - The ARN of the IAM role that has AWS Glue and S3
permissions.\s
               s3Path - The Amazon Simple Storage Service (Amazon S3) target
that contains data (for example, s3://<bucket name>/read).
               cron - A cron expression used to specify the schedule  (i.e.,
cron(15 12 * * ? *).
               dbName - The database name.\s
               crawlerName - The name of the crawler.\s
               jobName - The name you assign to this job definition.
               scriptLocation - The Amazon S3 path to a script that runs a job.
               locationUri - The location of the database (you can find this
file in resources folder).
               bucketNameSc - The Amazon S3 bucket name used when creating a job
               """;

       if (args.length != 9) {
           System.out.println(usage);
           return;
       }
       Scanner scanner = new Scanner(System.in);
       String iam = args[0];
       String s3Path = args[1];
       String cron = args[2];
       String dbName = args[3];
       String crawlerName = args[4];
       String jobName = args[5];
       String scriptLocation = args[6];
       String locationUri = args[7];
       String bucketNameSc = args[8];

       Region region = Region.US_EAST_1;
       GlueClient glueClient = GlueClient.builder()
           .region(region)
```

```
            .build();
        System.out.println(DASHES);
        System.out.println("Welcome to the AWS Glue scenario.");
        System.out.println("""
            AWS Glue is a fully managed extract, transform, and load (ETL)
service provided by Amazon
            Web Services (AWS). It is designed to simplify the process of
building, running, and maintaining
            ETL pipelines, which are essential for data integration and data
warehousing tasks.

            One of the key features of AWS Glue is its ability to automatically
discover and catalog data
            stored in various sources, such as Amazon S3, Amazon RDS, Amazon
Redshift, and other databases.
            This cataloging process creates a central metadata repository, known
as the AWS Glue Data Catalog,
            which provides a unified view of an organization's data assets. This
metadata can then be used to
            create ETL jobs, which can be scheduled and run on-demand or on a
regular basis.

            Lets get started.

            """);
        waitForInputToContinue(scanner);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("1. Create a database.");
        try {
            createDatabase(glueClient, dbName, locationUri);
        } catch (GlueException e) {
            if (e.awsErrorDetails().errorMessage().equals("Database already
exists.")) {
                System.out.println("Database " + dbName + " already exists.
Skipping creation.");
            } else {
                System.err.println(e.awsErrorDetails().errorMessage());
                return;
            }
        }

        waitForInputToContinue(scanner);
```

```
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("2. Create a crawler.");
        try {
            createGlueCrawler(glueClient, iam, s3Path, cron, dbName,
crawlerName);
        } catch (GlueException e) {
            if (e.awsErrorDetails().errorMessage().contains("already exists")) {
                System.out.println("Crawler " + crawlerName + " already exists.
Skipping creation.");
            } else {
                System.err.println(e.awsErrorDetails().errorMessage());
                System.exit(1);
            }
        }
        waitForInputToContinue(scanner);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("3. Get a crawler.");
        try {
            getSpecificCrawler(glueClient, crawlerName);
        } catch (GlueException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            return;
        }
        waitForInputToContinue(scanner);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("4. Start a crawler.");
        try {
            startSpecificCrawler(glueClient, crawlerName);
        } catch (GlueException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            return;
        }
        waitForInputToContinue(scanner);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("5. Get a database.");
        try {
```

```
            getSpecificDatabase(glueClient, dbName);
        } catch (GlueException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            return;
        }
        waitForInputToContinue(scanner);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("*** Wait 5 min for the tables to become available");
        TimeUnit.MINUTES.sleep(5);
        System.out.println("6. Get tables.");
        String myTableName;
        try {
            myTableName = getGlueTables(glueClient, dbName);
        } catch (GlueException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            return;
        }
        waitForInputToContinue(scanner);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("7. Create a job.");
        try {
            createJob(glueClient, jobName, iam, scriptLocation);
        } catch (GlueException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            return;
        }
        waitForInputToContinue(scanner);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("8. Start a Job run.");
        try {
            startJob(glueClient, jobName, dbName, myTableName, bucketNameSc);
        } catch (GlueException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            return;
        }
        waitForInputToContinue(scanner);
        System.out.println(DASHES);
```

```
        System.out.println(DASHES);
        System.out.println("9. List all jobs.");
        try {
            getAllJobs(glueClient);
        } catch (GlueException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            return;
        }
        waitForInputToContinue(scanner);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("10. Get job runs.");
        try {
            getJobRuns(glueClient, jobName);
        } catch (GlueException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            return;
        }
        waitForInputToContinue(scanner);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("11. Delete a job.");
        try {
            deleteJob(glueClient, jobName);
        } catch (GlueException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            return;
        }
        System.out.println("*** Wait 5 MIN for the " + crawlerName + " to stop");
        TimeUnit.MINUTES.sleep(5);
        waitForInputToContinue(scanner);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("12. Delete a database.");
        try {
            deleteDatabase(glueClient, dbName);
        } catch (GlueException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            return;
        }
        waitForInputToContinue(scanner);
```

```
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("Delete a crawler.");
        try {
            deleteSpecificCrawler(glueClient, crawlerName);
        } catch (GlueException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            return;
        }
        waitForInputToContinue(scanner);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("Successfully completed the AWS Glue Scenario");
        System.out.println(DASHES);
    }


    /**
     * Creates a Glue database with the specified name and location URI.
     *
     * @param glueClient  The Glue client to use for the database creation.
     * @param dbName      The name of the database to create.
     * @param locationUri The location URI for the database.
     */
    public static void createDatabase(GlueClient glueClient, String dbName,
String locationUri) {
        try {
            DatabaseInput input = DatabaseInput.builder()
                .description("Built with the AWS SDK for Java V2")
                .name(dbName)
                .locationUri(locationUri)
                .build();

            CreateDatabaseRequest request = CreateDatabaseRequest.builder()
                .databaseInput(input)
                .build();

            glueClient.createDatabase(request);
            System.out.println(dbName + " was successfully created");

        } catch (GlueException e) {
            throw e;
```

```
        }
    }


    /**
     * Creates a new AWS Glue crawler using the AWS Glue Java API.
     *
     * @param glueClient  the AWS Glue client used to interact with the AWS Glue
service
     * @param iam         the IAM role that the crawler will use to access the
data source
     * @param s3Path      the S3 path that the crawler will scan for data
     * @param cron        the cron expression that defines the crawler's schedule
     * @param dbName      the name of the AWS Glue database where the crawler
will store the metadata
     * @param crawlerName the name of the crawler to be created
     */
    public static void createGlueCrawler(GlueClient glueClient,
                                         String iam,
                                         String s3Path,
                                         String cron,
                                         String dbName,
                                         String crawlerName) {

        try {
            S3Target s3Target = S3Target.builder()
                .path(s3Path)
                .build();

            List<S3Target> targetList = new ArrayList<>();
            targetList.add(s3Target);
            CrawlerTargets targets = CrawlerTargets.builder()
                .s3Targets(targetList)
                .build();

            CreateCrawlerRequest crawlerRequest = CreateCrawlerRequest.builder()
                .databaseName(dbName)
                .name(crawlerName)
                .description("Created by the AWS Glue Java API")
                .targets(targets)
                .role(iam)
                .schedule(cron)
                .build();
```

```
            glueClient.createCrawler(crawlerRequest);
            System.out.println(crawlerName + " was successfully created");

        } catch (GlueException e) {
            throw e;
        }
    }

    /**
     * Retrieves a specific crawler from the AWS Glue service and waits for it to
be in the "READY" state.
     *
     * @param glueClient  the AWS Glue client used to interact with the Glue
service
     * @param crawlerName the name of the crawler to be retrieved
     */
    public static void getSpecificCrawler(GlueClient glueClient, String
crawlerName) throws InterruptedException {
        try {
            GetCrawlerRequest crawlerRequest = GetCrawlerRequest.builder()
                .name(crawlerName)
                .build();

            boolean ready = false;
            while (!ready) {
                GetCrawlerResponse response =
glueClient.getCrawler(crawlerRequest);
                String status = response.crawler().stateAsString();
                if (status.compareTo("READY") == 0) {
                    ready = true;
                }
                Thread.sleep(3000);
            }

            System.out.println("The crawler is now ready");

        } catch (GlueException | InterruptedException e) {
            throw e;
        }
    }

    /**
     * Starts a specific AWS Glue crawler.
     *
```

```
    * @param glueClient  the AWS Glue client to use for the crawler operation
    * @param crawlerName the name of the crawler to start
    * @throws GlueException if there is an error starting the crawler
    */
   public static void startSpecificCrawler(GlueClient glueClient, String
crawlerName) {
       try {
           StartCrawlerRequest crawlerRequest = StartCrawlerRequest.builder()
               .name(crawlerName)
               .build();

           glueClient.startCrawler(crawlerRequest);
           System.out.println(crawlerName + " was successfully started!");

       } catch (GlueException e) {
           throw e;
       }
   }

   /**
    * Retrieves the specific database from the AWS Glue service.
    *
    * @param glueClient   an instance of the AWS Glue client used to interact
with the service
    * @param databaseName the name of the database to retrieve
    * @throws GlueException if there is an error retrieving the database from
the AWS Glue service
    */
   public static void getSpecificDatabase(GlueClient glueClient, String
databaseName) {
       try {
           GetDatabaseRequest databasesRequest = GetDatabaseRequest.builder()
               .name(databaseName)
               .build();

           GetDatabaseResponse response =
glueClient.getDatabase(databasesRequest);
           Instant createDate = response.database().createTime();

           // Convert the Instant to readable date.
           DateTimeFormatter formatter =
DateTimeFormatter.ofLocalizedDateTime(FormatStyle.SHORT)
               .withLocale(Locale.US)
               .withZone(ZoneId.systemDefault());
```

```
            formatter.format(createDate);
            System.out.println("The create date of the database is " +
createDate);

        } catch (GlueException e) {
            throw e;
        }
    }


    /**
     * Retrieves the names of the tables in the specified Glue database.
     *
     * @param glueClient the Glue client to use for the operation
     * @param dbName     the name of the Glue database to retrieve the table
names from
     * @return the name of the first table retrieved, or an empty string if no
tables were found
     */
    public static String getGlueTables(GlueClient glueClient, String dbName) {
        String myTableName = "";
        try {
            GetTablesRequest tableRequest = GetTablesRequest.builder()
                .databaseName(dbName)
                .build();

            GetTablesResponse response = glueClient.getTables(tableRequest);
            List<Table> tables = response.tableList();
            if (tables.isEmpty()) {
                System.out.println("No tables were returned");
            } else {
                for (Table table : tables) {
                    myTableName = table.name();
                    System.out.println("Table name is: " + myTableName);
                }
            }

        } catch (GlueException e) {
            throw e;
        }
        return myTableName;
    }
```

```
   /**
    * Starts a job run in AWS Glue.
    *
    * @param glueClient    the AWS Glue client to use for the job run
    * @param jobName       the name of the Glue job to run
    * @param inputDatabase the name of the input database
    * @param inputTable    the name of the input table
    * @param outBucket     the URL of the output S3 bucket
    * @throws GlueException if there is an error starting the job run
    */
   public static void startJob(GlueClient glueClient, String jobName, String
inputDatabase, String inputTable,
                               String outBucket) {
       try {
           Map<String, String> myMap = new HashMap<>();
           myMap.put("--input_database", inputDatabase);
           myMap.put("--input_table", inputTable);
           myMap.put("--output_bucket_url", outBucket);

           StartJobRunRequest runRequest = StartJobRunRequest.builder()
               .workerType(WorkerType.G_1_X)
               .numberOfWorkers(10)
               .arguments(myMap)
               .jobName(jobName)
               .build();

           StartJobRunResponse response = glueClient.startJobRun(runRequest);
           System.out.println("The request Id of the job is " +
response.responseMetadata().requestId());

       } catch (GlueException e) {
           throw e;
       }
   }


   /**
    * Creates a new AWS Glue job.
    *
    * @param glueClient     the AWS Glue client to use for the operation
    * @param jobName        the name of the job to create
    * @param iam            the IAM role to associate with the job
    * @param scriptLocation the location of the script to be used by the job
```

```
     * @throws GlueException if there is an error creating the job
     */
    public static void createJob(GlueClient glueClient, String jobName, String
iam, String scriptLocation) {
        try {
            JobCommand command = JobCommand.builder()
                .pythonVersion("3")
                .name("glueetl")
                .scriptLocation(scriptLocation)
                .build();

            CreateJobRequest jobRequest = CreateJobRequest.builder()
                .description("A Job created by using the AWS SDK for Java V2")
                .glueVersion("2.0")
                .workerType(WorkerType.G_1_X)
                .numberOfWorkers(10)
                .name(jobName)
                .role(iam)
                .command(command)
                .build();

            glueClient.createJob(jobRequest);
            System.out.println(jobName + " was successfully created.");

        } catch (GlueException e) {
            throw e;
        }
    }


    /**
     * Retrieves and prints information about all the jobs in the Glue data
catalog.
     *
     * @param glueClient the Glue client used to interact with the AWS Glue
service
     */
    public static void getAllJobs(GlueClient glueClient) {
        try {
            GetJobsRequest jobsRequest = GetJobsRequest.builder()
                .maxResults(10)
                .build();

            GetJobsResponse jobsResponse = glueClient.getJobs(jobsRequest);
```

```
            List<Job> jobs = jobsResponse.jobs();
            for (Job job : jobs) {
                System.out.println("Job name is : " + job.name());
                System.out.println("The job worker type is : " +
job.workerType().name());
            }

        } catch (GlueException e) {
            throw e;
        }
    }

    /**
     * Retrieves the job runs for a given Glue job and prints the status of the
job runs.
     *
     * @param glueClient the Glue client used to make API calls
     * @param jobName    the name of the Glue job to retrieve the job runs for
     */
    public static void getJobRuns(GlueClient glueClient, String jobName) {
        try {
            GetJobRunsRequest runsRequest = GetJobRunsRequest.builder()
                .jobName(jobName)
                .maxResults(20)
                .build();

            boolean jobDone = false;
            while (!jobDone) {
                GetJobRunsResponse response = glueClient.getJobRuns(runsRequest);
                List<JobRun> jobRuns = response.jobRuns();
                for (JobRun jobRun : jobRuns) {
                    String jobState = jobRun.jobRunState().name();
                    if (jobState.compareTo("SUCCEEDED") == 0) {
                        System.out.println(jobName + " has succeeded");
                        jobDone = true;

                    } else if (jobState.compareTo("STOPPED") == 0) {
                        System.out.println("Job run has stopped");
                        jobDone = true;

                    } else if (jobState.compareTo("FAILED") == 0) {
                        System.out.println("Job run has failed");
                        jobDone = true;
```

```
                    } else if (jobState.compareTo("TIMEOUT") == 0) {
                        System.out.println("Job run has timed out");
                        jobDone = true;

                    } else {
                        System.out.println("*** Job run state is " +
jobRun.jobRunState().name());
                        System.out.println("Job run Id is " + jobRun.id());
                        System.out.println("The Glue version is " +
jobRun.glueVersion());
                    }
                    TimeUnit.SECONDS.sleep(5);
                }
            }

        } catch (GlueException e) {
            throw e;
        } catch (InterruptedException e) {
            throw new RuntimeException(e);
        }
    }


    /**
     * Deletes a Glue job.
     *
     * @param glueClient the Glue client to use for the operation
     * @param jobName    the name of the job to be deleted
     * @throws GlueException if there is an error deleting the job
     */
    public static void deleteJob(GlueClient glueClient, String jobName) {
        try {
            DeleteJobRequest jobRequest = DeleteJobRequest.builder()
                .jobName(jobName)
                .build();

            glueClient.deleteJob(jobRequest);
            System.out.println(jobName + " was successfully deleted");

        } catch (GlueException e) {
            throw e;
        }
    }
```

```
   /**
    * Deletes a AWS Glue Database.
    *
    * @param glueClient   An instance of the AWS Glue client used to interact
with the AWS Glue service.
    * @param databaseName The name of the database to be deleted.
    * @throws GlueException If an error occurs while deleting the database.
    */
   public static void deleteDatabase(GlueClient glueClient, String databaseName)
{
       try {
           DeleteDatabaseRequest request = DeleteDatabaseRequest.builder()
               .name(databaseName)
               .build();

           glueClient.deleteDatabase(request);
           System.out.println(databaseName + " was successfully deleted");

       } catch (GlueException e) {
           throw e;
       }
   }


   /**
    * Deletes a specific AWS Glue crawler.
    *
    * @param glueClient  the AWS Glue client object
    * @param crawlerName the name of the crawler to be deleted
    * @throws GlueException if an error occurs during the deletion process
    */
   public static void deleteSpecificCrawler(GlueClient glueClient, String
crawlerName) {
       try {
           DeleteCrawlerRequest deleteCrawlerRequest =
DeleteCrawlerRequest.builder()
               .name(crawlerName)
               .build();

           glueClient.deleteCrawler(deleteCrawlerRequest);
           System.out.println(crawlerName + " was deleted");

       } catch (GlueException e) {
           throw e;
```

```
        }
    }

    private static void waitForInputToContinue(Scanner scanner) {
        while (true) {
            System.out.println("");
            System.out.println("Enter 'c' followed by <ENTER> to continue:");
            String input = scanner.nextLine();

            if (input.trim().equalsIgnoreCase("c")) {
                System.out.println("Continuing with the program...");
                System.out.println("");
                break;
            } else {
                // Handle invalid input.
                System.out.println("Invalid input. Please try again.");
            }
        }
    }
}
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des API du kit AWS SDK for Java 2.x .
  - CreateCrawler
  - CreateJob
  - DeleteCrawler
  - DeleteDatabase
  - DeleteJob
  - DeleteTable
  - GetCrawler
  - GetDatabase
  - GetDatabases
  - GetJob
  - GetJobRun
  - GetJobRuns
  - GetTables

- ListJobs
- StartCrawler
- StartJobRun

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez et exécutez un crawler qui analyse un compartiment Amazon Simple Storage Service (Amazon S3) public et génère une base de données de métadonnées qui décrit les données au format CSV trouvées.

```
const createCrawler = (name, role, dbName, tablePrefix, s3TargetPath) => {
  const client = new GlueClient({});

  const command = new CreateCrawlerCommand({
    Name: name,
    Role: role,
    DatabaseName: dbName,
    TablePrefix: tablePrefix,
    Targets: {
      S3Targets: [{ Path: s3TargetPath }],
    },
  });

  return client.send(command);
};

const getCrawler = (name) => {
  const client = new GlueClient({});

  const command = new GetCrawlerCommand({
    Name: name,
  });
```

```
  return client.send(command);
};

const startCrawler = (name) => {
  const client = new GlueClient({});

  const command = new StartCrawlerCommand({
    Name: name,
  });

  return client.send(command);
};

const crawlerExists = async ({ getCrawler }, crawlerName) => {
  try {
    await getCrawler(crawlerName);
    return true;
  } catch {
    return false;
  }
};

/**
 * @param {{ createCrawler: import('../../../actions/create-
crawler.js').createCrawler}} actions
 */
const makeCreateCrawlerStep = (actions) => async (context) => {
  if (await crawlerExists(actions, process.env.CRAWLER_NAME)) {
    log("Crawler already exists. Skipping creation.");
  } else {
    await actions.createCrawler(
      process.env.CRAWLER_NAME,
      process.env.ROLE_NAME,
      process.env.DATABASE_NAME,
      process.env.TABLE_PREFIX,
      process.env.S3_TARGET_PATH,
    );

    log("Crawler created successfully.", { type: "success" });
  }

  return { ...context };
};
```

```
/**
 * @param {(name: string) => Promise<import('@aws-sdk/client-
glue').GetCrawlerCommandOutput>} getCrawler
 * @param {string} crawlerName
 */
const waitForCrawler = async (getCrawler, crawlerName) => {
  const waitTimeInSeconds = 30;
  const { Crawler } = await getCrawler(crawlerName);

  if (!Crawler) {
    throw new Error(`Crawler with name ${crawlerName} not found.`);
  }

  if (Crawler.State === "READY") {
    return;
  }

  log(`Crawler is ${Crawler.State}. Waiting ${waitTimeInSeconds} seconds...`);
  await wait(waitTimeInSeconds);
  return waitForCrawler(getCrawler, crawlerName);
};

const makeStartCrawlerStep =
  ({ startCrawler, getCrawler }) =>
  async (context) => {
    log("Starting crawler.");
    await startCrawler(process.env.CRAWLER_NAME);
    log("Crawler started.", { type: "success" });

    log("Waiting for crawler to finish running. This can take a while.");
    await waitForCrawler(getCrawler, process.env.CRAWLER_NAME);
    log("Crawler ready.", { type: "success" });

    return { ...context };
  };
```

Répertoriez les informations relatives aux bases de données et aux tables de votre AWS Glue Data Catalog.

```
const getDatabase = (name) => {
  const client = new GlueClient({});
```

```
  const command = new GetDatabaseCommand({
    Name: name,
  });

  return client.send(command);
};

const getTables = (databaseName) => {
  const client = new GlueClient({});

  const command = new GetTablesCommand({
    DatabaseName: databaseName,
  });

  return client.send(command);
};

const makeGetDatabaseStep =
  ({ getDatabase }) =>
  async (context) => {
    const {
      Database: { Name },
    } = await getDatabase(process.env.DATABASE_NAME);
    log(`Database: ${Name}`);
    return { ...context };
  };

/**
 * @param {{ getTables: () => Promise<import('@aws-sdk/client-
glue').GetTablesCommandOutput}} config
 */
const makeGetTablesStep =
  ({ getTables }) =>
  async (context) => {
    const { TableList } = await getTables(process.env.DATABASE_NAME);
    log("Tables:");
    log(TableList.map((table) => `  • ${table.Name}\n`));
    return { ...context };
  };
```

Créez et exécutez une tâche qui extrait les données CSV du compartiment Amazon S3 source, les transforme en supprimant et en renommant des champs, et charge la sortie au format JSON dans un autre compartiment Amazon S3.

```
const createJob = (name, role, scriptBucketName, scriptKey) => {
  const client = new GlueClient({});

  const command = new CreateJobCommand({
    Name: name,
    Role: role,
    Command: {
      Name: "glueetl",
      PythonVersion: "3",
      ScriptLocation: `s3://${scriptBucketName}/${scriptKey}`,
    },
    GlueVersion: "3.0",
  });

  return client.send(command);
};

const startJobRun = (jobName, dbName, tableName, bucketName) => {
  const client = new GlueClient({});

  const command = new StartJobRunCommand({
    JobName: jobName,
    Arguments: {
      "--input_database": dbName,
      "--input_table": tableName,
      "--output_bucket_url": `s3://${bucketName}/`,
    },
  });

  return client.send(command);
};

const makeCreateJobStep =
  ({ createJob }) =>
  async (context) => {
    log("Creating Job.");
    await createJob(
      process.env.JOB_NAME,
      process.env.ROLE_NAME,
```

```
      process.env.BUCKET_NAME,
      process.env.PYTHON_SCRIPT_KEY,
    );
    log("Job created.", { type: "success" });

    return { ...context };
  };

/**
 * @param {(name: string, runId: string) => Promise<import('@aws-sdk/client-
glue').GetJobRunCommandOutput> }  getJobRun
 * @param {string} jobName
 * @param {string} jobRunId
 */
const waitForJobRun = async (getJobRun, jobName, jobRunId) => {
  const waitTimeInSeconds = 30;
  const { JobRun } = await getJobRun(jobName, jobRunId);

  if (!JobRun) {
    throw new Error(`Job run with id ${jobRunId} not found.`);
  }

  switch (JobRun.JobRunState) {
    case "FAILED":
    case "TIMEOUT":
    case "STOPPED":
    case "ERROR":
      throw new Error(
        `Job ${JobRun.JobRunState}. Error: ${JobRun.ErrorMessage}`,
      );
    case "SUCCEEDED":
      return;
    default:
      break;
  }

  log(
    `Job ${JobRun.JobRunState}. Waiting ${waitTimeInSeconds} more seconds...`,
  );
  await wait(waitTimeInSeconds);
  return waitForJobRun(getJobRun, jobName, jobRunId);
};

/**
```

```
 * @param {{ prompter: { prompt: () => Promise<{ shouldOpen: boolean }>} }}
 context
 */
const promptToOpen = async (context) => {
  const { shouldOpen } = await context.prompter.prompt({
    name: "shouldOpen",
    type: "confirm",
    message: "Open the output bucket in your browser?",
  });

  if (shouldOpen) {
    return open(
      `https://s3.console.aws.amazon.com/s3/buckets/${process.env.BUCKET_NAME} to
 view the output.`,
    );
  }
};

const makeStartJobRunStep =
  ({ startJobRun, getJobRun }) =>
  async (context) => {
    log("Starting job.");
    const { JobRunId } = await startJobRun(
      process.env.JOB_NAME,
      process.env.DATABASE_NAME,
      process.env.TABLE_NAME,
      process.env.BUCKET_NAME,
    );
    log("Job started.", { type: "success" });

    log("Waiting for job to finish running. This can take a while.");
    await waitForJobRun(getJobRun, process.env.JOB_NAME, JobRunId);
    log("Job run succeeded.", { type: "success" });

    await promptToOpen(context);

    return { ...context };
  };
```

Répertoriez les informations relatives aux exécutions de tâches et affichez certaines des données transformées.

```
const getJobRuns = (jobName) => {
  const client = new GlueClient({});
  const command = new GetJobRunsCommand({
    JobName: jobName,
  });

  return client.send(command);
};

const getJobRun = (jobName, jobRunId) => {
  const client = new GlueClient({});
  const command = new GetJobRunCommand({
    JobName: jobName,
    RunId: jobRunId,
  });

  return client.send(command);
};

/**
 * @typedef {{ prompter: { prompt: () => Promise<{jobName: string}> } }} Context
 */

/**
 * @typedef {() => Promise<import('@aws-sdk/client-
glue').GetJobRunCommandOutput>} getJobRun
 */

/**
 * @typedef {() => Promise<import('@aws-sdk/client-
glue').GetJobRunsCommandOutput} getJobRuns
 */

/**
 *
 * @param {getJobRun} getJobRun
 * @param {string} jobName
 * @param {string} jobRunId
 */
const logJobRunDetails = async (getJobRun, jobName, jobRunId) => {
  const { JobRun } = await getJobRun(jobName, jobRunId);
  log(JobRun, { type: "object" });
};
```

```
/**
 *
 * @param {{getJobRuns: getJobRuns, getJobRun: getJobRun }} funcs
 */
const makePickJobRunStep =
  ({ getJobRuns, getJobRun }) =>
  async (/** @type { Context } */ context) => {
    if (context.selectedJobName) {
      const { JobRuns } = await getJobRuns(context.selectedJobName);

      const { jobRunId } = await context.prompter.prompt({
        name: "jobRunId",
        type: "list",
        message: "Select a job run to see details.",
        choices: JobRuns.map((run) => run.Id),
      });

      logJobRunDetails(getJobRun, context.selectedJobName, jobRunId);
    }

    return { ...context };
  };
```

Supprimez toutes les ressources créées par la démonstration.

```
const deleteJob = (jobName) => {
  const client = new GlueClient({});

  const command = new DeleteJobCommand({
    JobName: jobName,
  });

  return client.send(command);
};

const deleteTable = (databaseName, tableName) => {
  const client = new GlueClient({});

  const command = new DeleteTableCommand({
    DatabaseName: databaseName,
    Name: tableName,
```

```
  });

  return client.send(command);
};

const deleteDatabase = (databaseName) => {
  const client = new GlueClient({});

  const command = new DeleteDatabaseCommand({
    Name: databaseName,
  });

  return client.send(command);
};

const deleteCrawler = (crawlerName) => {
  const client = new GlueClient({});

  const command = new DeleteCrawlerCommand({
    Name: crawlerName,
  });

  return client.send(command);
};

/**
 *
 * @param {import('../../../actions/delete-job.js').deleteJob} deleteJobFn
 * @param {string[]} jobNames
 * @param {{ prompter: { prompt: () => Promise<any> }}} context
 */
const handleDeleteJobs = async (deleteJobFn, jobNames, context) => {
  /**
   * @type {{ selectedJobNames: string[] }}
   */
  const { selectedJobNames } = await context.prompter.prompt({
    name: "selectedJobNames",
    type: "checkbox",
    message: "Let's clean up jobs. Select jobs to delete.",
    choices: jobNames,
  });

  if (selectedJobNames.length === 0) {
    log("No jobs selected.");
```

```
  } else {
    log("Deleting jobs.");
    await Promise.all(
      selectedJobNames.map((n) => deleteJobFn(n).catch(console.error)),
    );
    log("Jobs deleted.", { type: "success" });
  }
};

/**
 * @param {{
 *   listJobs: import('../../../actions/list-jobs.js').listJobs,
 *   deleteJob: import('../../../actions/delete-job.js').deleteJob
 * }} config
 */
const makeCleanUpJobsStep =
  ({ listJobs, deleteJob }) =>
  async (context) => {
    const { JobNames } = await listJobs();
    if (JobNames.length > 0) {
      await handleDeleteJobs(deleteJob, JobNames, context);
    }

    return { ...context };
  };

/**
 * @param {import('../../../actions/delete-table.js').deleteTable} deleteTable
 * @param {string} databaseName
 * @param {string[]} tableNames
 */
const deleteTables = (deleteTable, databaseName, tableNames) =>
  Promise.all(
    tableNames.map((tableName) =>
      deleteTable(databaseName, tableName).catch(console.error),
    ),
  );

/**
 * @param {{
 *  getTables: import('../../../actions/get-tables.js').getTables,
 *  deleteTable: import('../../../actions/delete-table.js').deleteTable
 * }} config
 */
```

```
const makeCleanUpTablesStep =
  ({ getTables, deleteTable }) =>
  /**
   * @param {{ prompter: { prompt: () => Promise<any>}}} context
   */
  async (context) => {
    const { TableList } = await getTables(process.env.DATABASE_NAME).catch(
      () => ({ TableList: null }),
    );

    if (TableList && TableList.length > 0) {
      /**
       * @type {{ tableNames: string[] }}
       */
      const { tableNames } = await context.prompter.prompt({
        name: "tableNames",
        type: "checkbox",
        message: "Let's clean up tables. Select tables to delete.",
        choices: TableList.map((t) => t.Name),
      });

      if (tableNames.length === 0) {
        log("No tables selected.");
      } else {
        log("Deleting tables.");
        await deleteTables(deleteTable, process.env.DATABASE_NAME, tableNames);
        log("Tables deleted.", { type: "success" });
      }
    }

    return { ...context };
  };

/**
 * @param {import('../../../actions/delete-database.js').deleteDatabase}
 deleteDatabase
 * @param {string[]} databaseNames
 */
const deleteDatabases = (deleteDatabase, databaseNames) =>
  Promise.all(
    databaseNames.map((dbName) => deleteDatabase(dbName).catch(console.error)),
  );

/**
```

```
 * @param {{
 *   getDatabases: import('../../../actions/get-databases.js').getDatabases
 *   deleteDatabase: import('../../../actions/delete-database.js').deleteDatabase
 * }} config
 */
const makeCleanUpDatabasesStep =
  ({ getDatabases, deleteDatabase }) =>
  /**
   * @param {{ prompter: { prompt: () => Promise<any>}} context
   */
  async (context) => {
    const { DatabaseList } = await getDatabases();

    if (DatabaseList.length > 0) {
      /** @type {{ dbNames: string[] }} */
      const { dbNames } = await context.prompter.prompt({
        name: "dbNames",
        type: "checkbox",
        message: "Let's clean up databases. Select databases to delete.",
        choices: DatabaseList.map((db) => db.Name),
      });

      if (dbNames.length === 0) {
        log("No databases selected.");
      } else {
        log("Deleting databases.");
        await deleteDatabases(deleteDatabase, dbNames);
        log("Databases deleted.", { type: "success" });
      }
    }

    return { ...context };
  };

const cleanUpCrawlerStep = async (context) => {
  log("Deleting crawler.");

  try {
    await deleteCrawler(process.env.CRAWLER_NAME);
    log("Crawler deleted.", { type: "success" });
  } catch (err) {
    if (err.name === "EntityNotFoundException") {
      log("Crawler is already deleted.");
    } else {
```

```
      throw err;
    }
  }

  return { ...context };
};
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des API du kit AWS SDK pour JavaScript .
  - CreateCrawler
  - CreateJob
  - DeleteCrawler
  - DeleteDatabase
  - DeleteJob
  - DeleteTable
  - GetCrawler
  - GetDatabase
  - GetDatabases
  - GetJob
  - GetJobRun
  - GetJobRuns
  - GetTables
  - ListJobs
  - StartCrawler
  - StartJobRun

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun main(args: Array<String>) {
    val usage = """
        Usage:
            <iam> <s3Path> <cron> <dbName> <crawlerName> <jobName>
 <scriptLocation> <locationUri>

        Where:
            iam - The Amazon Resource Name (ARN) of the AWS Identity and Access
 Management (IAM) role that has AWS Glue and Amazon Simple Storage Service
 (Amazon S3) permissions.
            s3Path - The Amazon Simple Storage Service (Amazon S3) target that
 contains data (for example, CSV data).
            cron - A cron expression used to specify the schedule (for example,
 cron(15 12 * * ? *).
            dbName - The database name.
            crawlerName - The name of the crawler.
            jobName - The name you assign to this job definition.
            scriptLocation - Specifies the Amazon S3 path to a script that runs a
 job.
            locationUri - Specifies the location of the database
        """

    if (args.size != 8) {
        println(usage)
        exitProcess(1)
    }

    val iam = args[0]
    val s3Path = args[1]
    val cron = args[2]
    val dbName = args[3]
    val crawlerName = args[4]
```

```
    val jobName = args[5]
    val scriptLocation = args[6]
    val locationUri = args[7]

    println("About to start the AWS Glue Scenario")
    createDatabase(dbName, locationUri)
    createCrawler(iam, s3Path, cron, dbName, crawlerName)
    getCrawler(crawlerName)
    startCrawler(crawlerName)
    getDatabase(dbName)
    getGlueTables(dbName)
    createJob(jobName, iam, scriptLocation)
    startJob(jobName)
    getJobs()
    getJobRuns(jobName)
    deleteJob(jobName)
    println("*** Wait for 5 MIN so the $crawlerName is ready to be deleted")
    TimeUnit.MINUTES.sleep(5)
    deleteMyDatabase(dbName)
    deleteCrawler(crawlerName)
}

suspend fun createDatabase(
    dbName: String?,
    locationUriVal: String?,
) {
    val input =
        DatabaseInput {
            description = "Built with the AWS SDK for Kotlin"
            name = dbName
            locationUri = locationUriVal
        }

    val request =
        CreateDatabaseRequest {
            databaseInput = input
        }

    GlueClient.fromEnvironment { region = "us-east-1" }.use { glueClient ->
        glueClient.createDatabase(request)
        println("The database was successfully created")
    }
}
```

```
suspend fun createCrawler(
    iam: String?,
    s3Path: String?,
    cron: String?,
    dbName: String?,
    crawlerName: String,
) {
    val s3Target =
        S3Target {
            path = s3Path
        }

    val targetList = ArrayList<S3Target>()
    targetList.add(s3Target)

    val targetOb =
        CrawlerTargets {
            s3Targets = targetList
        }

    val crawlerRequest =
        CreateCrawlerRequest {
            databaseName = dbName
            name = crawlerName
            description = "Created by the AWS Glue Java API"
            targets = targetOb
            role = iam
            schedule = cron
        }

    GlueClient.fromEnvironment { region = "us-east-1" }.use { glueClient ->
        glueClient.createCrawler(crawlerRequest)
        println("$crawlerName was successfully created")
    }
}

suspend fun getCrawler(crawlerName: String?) {
    val request =
        GetCrawlerRequest {
            name = crawlerName
        }

    GlueClient.fromEnvironment { region = "us-east-1" }.use { glueClient ->
        val response = glueClient.getCrawler(request)
```

```
        val role = response.crawler?.role
        println("The role associated with this crawler is $role")
    }
}

suspend fun startCrawler(crawlerName: String) {
    val crawlerRequest =
        StartCrawlerRequest {
            name = crawlerName
        }

    GlueClient.fromEnvironment { region = "us-east-1" }.use { glueClient ->
        glueClient.startCrawler(crawlerRequest)
        println("$crawlerName was successfully started.")
    }
}

suspend fun getDatabase(databaseName: String?) {
    val request =
        GetDatabaseRequest {
            name = databaseName
        }

    GlueClient.fromEnvironment { region = "us-east-1" }.use { glueClient ->
        val response = glueClient.getDatabase(request)
        val dbDesc = response.database?.description
        println("The database description is $dbDesc")
    }
}

suspend fun getGlueTables(dbName: String?) {
    val tableRequest =
        GetTablesRequest {
            databaseName = dbName
        }

    GlueClient.fromEnvironment { region = "us-east-1" }.use { glueClient ->
        val response = glueClient.getTables(tableRequest)
        response.tableList?.forEach { tableName ->
            println("Table name is ${tableName.name}")
        }
    }
}
```

```
suspend fun startJob(jobNameVal: String?) {
    val runRequest =
        StartJobRunRequest {
            workerType = WorkerType.G1X
            numberOfWorkers = 10
            jobName = jobNameVal
        }

    GlueClient.fromEnvironment { region = "us-east-1" }.use { glueClient ->
        val response = glueClient.startJobRun(runRequest)
        println("The job run Id is ${response.jobRunId}")
    }
}

suspend fun createJob(
    jobName: String,
    iam: String?,
    scriptLocationVal: String?,
) {
    val commandOb =
        JobCommand {
            pythonVersion = "3"
            name = "MyJob1"
            scriptLocation = scriptLocationVal
        }

    val jobRequest =
        CreateJobRequest {
            description = "A Job created by using the AWS SDK for Java V2"
            glueVersion = "2.0"
            workerType = WorkerType.G1X
            numberOfWorkers = 10
            name = jobName
            role = iam
            command = commandOb
        }

    GlueClient.fromEnvironment { region = "us-east-1" }.use { glueClient ->
        glueClient.createJob(jobRequest)
        println("$jobName was successfully created.")
    }
}

suspend fun getJobs() {
```

```
    val request =
        GetJobsRequest {
            maxResults = 10
        }

    GlueClient.fromEnvironment { region = "us-east-1" }.use { glueClient ->
        val response = glueClient.getJobs(request)
        response.jobs?.forEach { job ->
            println("Job name is ${job.name}")
        }
    }
}

suspend fun getJobRuns(jobNameVal: String?) {
    val request =
        GetJobRunsRequest {
            jobName = jobNameVal
        }

    GlueClient.fromEnvironment { region = "us-east-1" }.use { glueClient ->
        val response = glueClient.getJobRuns(request)
        response.jobRuns?.forEach { job ->
            println("Job name is ${job.jobName}")
        }
    }
}

suspend fun deleteJob(jobNameVal: String) {
    val jobRequest =
        DeleteJobRequest {
            jobName = jobNameVal
        }

    GlueClient.fromEnvironment { region = "us-east-1" }.use { glueClient ->
        glueClient.deleteJob(jobRequest)
        println("$jobNameVal was successfully deleted")
    }
}

suspend fun deleteMyDatabase(databaseName: String) {
    val request =
        DeleteDatabaseRequest {
            name = databaseName
        }
```

```
    GlueClient.fromEnvironment { region = "us-east-1" }.use { glueClient ->
        glueClient.deleteDatabase(request)
        println("$databaseName was successfully deleted")
    }
}

suspend fun deleteCrawler(crawlerName: String) {
    val request =
        DeleteCrawlerRequest {
            name = crawlerName
        }
    GlueClient.fromEnvironment { region = "us-east-1" }.use { glueClient ->
        glueClient.deleteCrawler(request)
        println("$crawlerName was deleted")
    }
}
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des API du kit AWS SDK pour Kotlin.
  - CreateCrawler
  - CreateJob
  - DeleteCrawler
  - DeleteDatabase
  - DeleteJob
  - DeleteTable
  - GetCrawler
  - GetDatabase
  - GetDatabases
  - GetJob
  - GetJobRun
  - GetJobRuns
  - GetTables
  - ListJobs
  - StartCrawler

- StartJobRun

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
namespace Glue;

use Aws\Glue\GlueClient;
use Aws\S3\S3Client;
use AwsUtilities\AWSServiceClass;
use GuzzleHttp\Psr7\Stream;
use Iam\IAMService;

class GettingStartedWithGlue
{
    public function run()
    {
        echo("\n");
        echo("--------------------------------------\n");
        print("Welcome to the AWS Glue getting started demo using PHP!\n");
        echo("--------------------------------------\n");

        $clientArgs = [
            'region' => 'us-west-2',
            'version' => 'latest',
            'profile' => 'default',
        ];
        $uniqid = uniqid();

        $glueClient = new GlueClient($clientArgs);
        $glueService = new GlueService($glueClient);
        $iamService = new IAMService();
        $crawlerName = "example-crawler-test-" . $uniqid;
```

```
        AWSServiceClass::$waitTime = 5;
        AWSServiceClass::$maxWaitAttempts = 20;

        $role = $iamService->getRole("AWSGlueServiceRole-DocExample");

        $databaseName = "doc-example-database-$uniqid";
        $path = 's3://crawler-public-us-east-1/flight/2016/csv';
        $glueService->createCrawler($crawlerName, $role['Role']['Arn'],
$databaseName, $path);
        $glueService->startCrawler($crawlerName);

        echo "Waiting for crawler";
        do {
            $crawler = $glueService->getCrawler($crawlerName);
            echo ".";
            sleep(10);
        } while ($crawler['Crawler']['State'] != "READY");
        echo "\n";

        $database = $glueService->getDatabase($databaseName);
        echo "Found a database named " . $database['Database']['Name'] . "\n";

        //Upload job script
        $s3client = new S3Client($clientArgs);
        $bucketName = "test-glue-bucket-" . $uniqid;
        $s3client->createBucket([
            'Bucket' => $bucketName,
            'CreateBucketConfiguration' => ['LocationConstraint' => 'us-west-2'],
        ]);

        $s3client->putObject([
            'Bucket' => $bucketName,
            'Key' => 'run_job.py',
            'SourceFile' => __DIR__ . '/flight_etl_job_script.py'
        ]);
        $s3client->putObject([
            'Bucket' => $bucketName,
            'Key' => 'setup_scenario_getting_started.yaml',
            'SourceFile' => __DIR__ . '/setup_scenario_getting_started.yaml'
        ]);

        $tables = $glueService->getTables($databaseName);

        $jobName = 'test-job-' . $uniqid;
```

```
        $scriptLocation = "s3://$bucketName/run_job.py";
        $job = $glueService->createJob($jobName, $role['Role']['Arn'],
$scriptLocation);

        $outputBucketUrl = "s3://$bucketName";
        $runId = $glueService->startJobRun($jobName, $databaseName, $tables,
$outputBucketUrl)['JobRunId'];

        echo "waiting for job";
        do {
            $jobRun = $glueService->getJobRun($jobName, $runId);
            echo ".";
            sleep(10);
        } while (!array_intersect([$jobRun['JobRun']['JobRunState']],
['SUCCEEDED', 'STOPPED', 'FAILED', 'TIMEOUT']));
        echo "\n";

        $jobRuns = $glueService->getJobRuns($jobName);

        $objects = $s3client->listObjects([
            'Bucket' => $bucketName,
        ])['Contents'];

        foreach ($objects as $object) {
            echo $object['Key'] . "\n";
        }

        echo "Downloading " . $objects[1]['Key'] . "\n";
        /** @var Stream $downloadObject */
        $downloadObject = $s3client->getObject([
            'Bucket' => $bucketName,
            'Key' => $objects[1]['Key'],
        ])['Body']->getContents();
        echo "Here is the first 1000 characters in the object.";
        echo substr($downloadObject, 0, 1000);

        $jobs = $glueService->listJobs();
        echo "Current jobs:\n";
        foreach ($jobs['JobNames'] as $jobsName) {
            echo "{$jobsName}\n";
        }

        echo "Delete the job.\n";
        $glueClient->deleteJob([
```

```
            'JobName' => $job['Name'],
        ]);

        echo "Delete the tables.\n";
        foreach ($tables['TableList'] as $table) {
            $glueService->deleteTable($table['Name'], $databaseName);
        }

        echo "Delete the databases.\n";
        $glueClient->deleteDatabase([
            'Name' => $databaseName,
        ]);

        echo "Delete the crawler.\n";
        $glueClient->deleteCrawler([
            'Name' => $crawlerName,
        ]);

        $deleteObjects = $s3client->listObjectsV2([
            'Bucket' => $bucketName,
        ]);
        echo "Delete all objects in the bucket.\n";
        $deleteObjects = $s3client->deleteObjects([
            'Bucket' => $bucketName,
            'Delete' => [
                'Objects' => $deleteObjects['Contents'],
            ]
        ]);
        echo "Delete the bucket.\n";
        $s3client->deleteBucket(['Bucket' => $bucketName]);

        echo "This job was brought to you by the number $uniqid\n";
    }
}

namespace Glue;

use Aws\Glue\GlueClient;
use Aws\Result;

use function PHPUnit\Framework\isEmpty;

class GlueService extends \AwsUtilities\AWSServiceClass
{
```

```
    protected GlueClient $glueClient;

    public function __construct($glueClient)
    {
        $this->glueClient = $glueClient;
    }

    public function getCrawler($crawlerName)
    {
        return $this->customWaiter(function () use ($crawlerName) {
            return $this->glueClient->getCrawler([
                'Name' => $crawlerName,
            ]);
        });
    }

    public function createCrawler($crawlerName, $role, $databaseName, $path):
Result
    {
        return $this->customWaiter(function () use ($crawlerName, $role,
$databaseName, $path) {
            return $this->glueClient->createCrawler([
                'Name' => $crawlerName,
                'Role' => $role,
                'DatabaseName' => $databaseName,
                'Targets' => [
                    'S3Targets' =>
                        [[
                            'Path' => $path,
                        ]]
                ],
            ]);
        });
    }

    public function startCrawler($crawlerName): Result
    {
        return $this->glueClient->startCrawler([
            'Name' => $crawlerName,
        ]);
    }

    public function getDatabase(string $databaseName): Result
    {
```

```
        return $this->customWaiter(function () use ($databaseName) {
            return $this->glueClient->getDatabase([
                'Name' => $databaseName,
            ]);
        });
    }

    public function getTables($databaseName): Result
    {
        return $this->glueClient->getTables([
            'DatabaseName' => $databaseName,
        ]);
    }

    public function createJob($jobName, $role, $scriptLocation, $pythonVersion =
'3', $glueVersion = '3.0'): Result
    {
        return $this->glueClient->createJob([
            'Name' => $jobName,
            'Role' => $role,
            'Command' => [
                'Name' => 'glueetl',
                'ScriptLocation' => $scriptLocation,
                'PythonVersion' => $pythonVersion,
            ],
            'GlueVersion' => $glueVersion,
        ]);
    }

    public function startJobRun($jobName, $databaseName, $tables,
$outputBucketUrl): Result
    {
        return $this->glueClient->startJobRun([
            'JobName' => $jobName,
            'Arguments' => [
                'input_database' => $databaseName,
                'input_table' => $tables['TableList'][0]['Name'],
                'output_bucket_url' => $outputBucketUrl,
                '--input_database' => $databaseName,
                '--input_table' => $tables['TableList'][0]['Name'],
                '--output_bucket_url' => $outputBucketUrl,
            ],
        ]);
    }
```

```
    public function listJobs($maxResults = null, $nextToken = null, $tags = []):
Result
    {
        $arguments = [];
        if ($maxResults) {
            $arguments['MaxResults'] = $maxResults;
        }
        if ($nextToken) {
            $arguments['NextToken'] = $nextToken;
        }
        if (!empty($tags)) {
            $arguments['Tags'] = $tags;
        }
        return $this->glueClient->listJobs($arguments);
    }

    public function getJobRuns($jobName, $maxResults = 0, $nextToken = ''):
Result
    {
        $arguments = ['JobName' => $jobName];
        if ($maxResults) {
            $arguments['MaxResults'] = $maxResults;
        }
        if ($nextToken) {
            $arguments['NextToken'] = $nextToken;
        }
        return $this->glueClient->getJobRuns($arguments);
    }

    public function getJobRun($jobName, $runId, $predecessorsIncluded = false):
Result
    {
        return $this->glueClient->getJobRun([
            'JobName' => $jobName,
            'RunId' => $runId,
            'PredecessorsIncluded' => $predecessorsIncluded,
        ]);
    }

    public function deleteJob($jobName)
    {
        return $this->glueClient->deleteJob([
            'JobName' => $jobName,
```

```
        ]);
    }

    public function deleteTable($tableName, $databaseName)
    {
        return $this->glueClient->deleteTable([
            'DatabaseName' => $databaseName,
            'Name' => $tableName,
        ]);
    }

    public function deleteDatabase($databaseName)
    {
        return $this->glueClient->deleteDatabase([
            'Name' => $databaseName,
        ]);
    }

    public function deleteCrawler($crawlerName)
    {
        return $this->glueClient->deleteCrawler([
            'Name' => $crawlerName,
        ]);
    }
}
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des API du kit AWS SDK pour PHP .
  - CreateCrawler
  - CreateJob
  - DeleteCrawler
  - DeleteDatabase
  - DeleteJob
  - DeleteTable
  - GetCrawler
  - GetDatabase
  - GetDatabases
  - GetJob

- GetJobRun
- GetJobRuns
- GetTables
- ListJobs
- StartCrawler
- StartJobRun

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez une classe qui englobe les AWS Glue fonctions utilisées dans le scénario.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client


    def get_crawler(self, name):
        """
        Gets information about a crawler.

        :param name: The name of the crawler to look up.
        :return: Data about the crawler.
        """
        crawler = None
        try:
            response = self.glue_client.get_crawler(Name=name)
```

```
            crawler = response["Crawler"]
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityNotFoundException":
                logger.info("Crawler %s doesn't exist.", name)
            else:
                logger.error(
                    "Couldn't get crawler %s. Here's why: %s: %s",
                    name,
                    err.response["Error"]["Code"],
                    err.response["Error"]["Message"],
                )
                raise
        return crawler


    def create_crawler(self, name, role_arn, db_name, db_prefix, s3_target):
        """
        Creates a crawler that can crawl the specified target and populate a
        database in your AWS Glue Data Catalog with metadata that describes the
data
        in the target.

        :param name: The name of the crawler.
        :param role_arn: The Amazon Resource Name (ARN) of an AWS Identity and
Access
                         Management (IAM) role that grants permission to let AWS
Glue
                         access the resources it needs.
        :param db_name: The name to give the database that is created by the
crawler.
        :param db_prefix: The prefix to give any database tables that are created
by
                          the crawler.
        :param s3_target: The URL to an S3 bucket that contains data that is
                          the target of the crawler.
        """
        try:
            self.glue_client.create_crawler(
                Name=name,
                Role=role_arn,
                DatabaseName=db_name,
                TablePrefix=db_prefix,
                Targets={"S3Targets": [{"Path": s3_target}]},
            )
```

```
        except ClientError as err:
            logger.error(
                "Couldn't create crawler. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def start_crawler(self, name):
        """
        Starts a crawler. The crawler crawls its configured target and creates
        metadata that describes the data it finds in the target data source.

        :param name: The name of the crawler to start.
        """
        try:
            self.glue_client.start_crawler(Name=name)
        except ClientError as err:
            logger.error(
                "Couldn't start crawler %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def get_database(self, name):
        """
        Gets information about a database in your Data Catalog.

        :param name: The name of the database to look up.
        :return: Information about the database.
        """
        try:
            response = self.glue_client.get_database(Name=name)
        except ClientError as err:
            logger.error(
                "Couldn't get database %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
```

```
            raise
        else:
            return response["Database"]


    def get_tables(self, db_name):
        """
        Gets a list of tables in a Data Catalog database.

        :param db_name: The name of the database to query.
        :return: The list of tables in the database.
        """
        try:
            response = self.glue_client.get_tables(DatabaseName=db_name)
        except ClientError as err:
            logger.error(
                "Couldn't get tables %s. Here's why: %s: %s",
                db_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response["TableList"]


    def create_job(self, name, description, role_arn, script_location):
        """
        Creates a job definition for an extract, transform, and load (ETL) job
that can
        be run by AWS Glue.

        :param name: The name of the job definition.
        :param description: The description of the job definition.
        :param role_arn: The ARN of an IAM role that grants AWS Glue the
permissions
                         it requires to run the job.
        :param script_location: The Amazon S3 URL of a Python ETL script that is
run as
                                part of the job. The script defines how the data
is
                                transformed.
        """
        try:
```

```
            self.glue_client.create_job(
                Name=name,
                Description=description,
                Role=role_arn,
                Command={
                    "Name": "glueetl",
                    "ScriptLocation": script_location,
                    "PythonVersion": "3",
                },
                GlueVersion="3.0",
            )
        except ClientError as err:
            logger.error(
                "Couldn't create job %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def start_job_run(self, name, input_database, input_table,
output_bucket_name):
        """
        Starts a job run. A job run extracts data from the source, transforms it,
        and loads it to the output bucket.

        :param name: The name of the job definition.
        :param input_database: The name of the metadata database that contains
tables
                               that describe the source data. This is typically
created
                               by a crawler.
        :param input_table: The name of the table in the metadata database that
                            describes the source data.
        :param output_bucket_name: The S3 bucket where the output is written.
        :return: The ID of the job run.
        """
        try:
            # The custom Arguments that are passed to this function are used by
the
            # Python ETL script to determine the location of input and output
data.
            response = self.glue_client.start_job_run(
```

```
                JobName=name,
                Arguments={
                    "--input_database": input_database,
                    "--input_table": input_table,
                    "--output_bucket_url": f"s3://{output_bucket_name}/",
                },
            )
        except ClientError as err:
            logger.error(
                "Couldn't start job run %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response["JobRunId"]


    def list_jobs(self):
        """
        Lists the names of job definitions in your account.

        :return: The list of job definition names.
        """
        try:
            response = self.glue_client.list_jobs()
        except ClientError as err:
            logger.error(
                "Couldn't list jobs. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response["JobNames"]


    def get_job_runs(self, job_name):
        """
        Gets information about runs that have been performed for a specific job
        definition.

        :param job_name: The name of the job definition to look up.
```

```
        :return: The list of job runs.
        """
        try:
            response = self.glue_client.get_job_runs(JobName=job_name)
        except ClientError as err:
            logger.error(
                "Couldn't get job runs for %s. Here's why: %s: %s",
                job_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response["JobRuns"]


    def get_job_run(self, name, run_id):
        """
        Gets information about a single job run.

        :param name: The name of the job definition for the run.
        :param run_id: The ID of the run.
        :return: Information about the run.
        """
        try:
            response = self.glue_client.get_job_run(JobName=name, RunId=run_id)
        except ClientError as err:
            logger.error(
                "Couldn't get job run %s/%s. Here's why: %s: %s",
                name,
                run_id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response["JobRun"]


    def delete_job(self, job_name):
        """
        Deletes a job definition. This also deletes data about all runs that are
        associated with this job definition.
```

```
        :param job_name: The name of the job definition to delete.
        """
        try:
            self.glue_client.delete_job(JobName=job_name)
        except ClientError as err:
            logger.error(
                "Couldn't delete job %s. Here's why: %s: %s",
                job_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def delete_table(self, db_name, table_name):
        """
        Deletes a table from a metadata database.

        :param db_name: The name of the database that contains the table.
        :param table_name: The name of the table to delete.
        """
        try:
            self.glue_client.delete_table(DatabaseName=db_name, Name=table_name)
        except ClientError as err:
            logger.error(
                "Couldn't delete table %s. Here's why: %s: %s",
                table_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def delete_database(self, name):
        """
        Deletes a metadata database from your Data Catalog.

        :param name: The name of the database to delete.
        """
        try:
            self.glue_client.delete_database(Name=name)
        except ClientError as err:
            logger.error(
                "Couldn't delete database %s. Here's why: %s: %s",
```

```
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def delete_crawler(self, name):
        """
        Deletes a crawler.

        :param name: The name of the crawler to delete.
        """
        try:
            self.glue_client.delete_crawler(Name=name)
        except ClientError as err:
            logger.error(
                "Couldn't delete crawler %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

Créez une classe qui exécute le scénario.

```
class GlueCrawlerJobScenario:
    """
    Encapsulates a scenario that shows how to create an AWS Glue crawler and job
 and use
    them to transform data from CSV to JSON format.
    """


    def __init__(self, glue_client, glue_service_role, glue_bucket):
        """
        :param glue_client: A Boto3 AWS Glue client.
        :param glue_service_role: An AWS Identity and Access Management (IAM)
 role
                                  that AWS Glue can assume to gain access to the
```

```
                                    resources it requires.
        :param glue_bucket: An S3 bucket that can hold a job script and output
data
                            from AWS Glue job runs.
        """
        self.glue_client = glue_client
        self.glue_service_role = glue_service_role
        self.glue_bucket = glue_bucket

    @staticmethod
    def wait(seconds, tick=12):
        """
        Waits for a specified number of seconds, while also displaying an
animated
        spinner.

        :param seconds: The number of seconds to wait.
        :param tick: The number of frames per second used to animate the spinner.
        """
        progress = "|/-\\"
        waited = 0
        while waited < seconds:
            for frame in range(tick):
                sys.stdout.write(f"\r{progress[frame % len(progress)]}")
                sys.stdout.flush()
                time.sleep(1 / tick)
            waited += 1

    def upload_job_script(self, job_script):
        """
        Uploads a Python ETL script to an S3 bucket. The script is used by the
AWS Glue
        job to transform data.

        :param job_script: The relative path to the job script.
        """
        try:
            self.glue_bucket.upload_file(Filename=job_script, Key=job_script)
            print(f"Uploaded job script '{job_script}' to the example bucket.")
        except S3UploadFailedError as err:
            logger.error("Couldn't upload job script. Here's why: %s", err)
            raise
```

```
   def run(self, crawler_name, db_name, db_prefix, data_source, job_script,
job_name):
       """
       Runs the scenario. This is an interactive experience that runs at a
command
       prompt and asks you for input throughout.

       :param crawler_name: The name of the crawler used in the scenario. If the
                            crawler does not exist, it is created.
       :param db_name: The name to give the metadata database created by the
crawler.
       :param db_prefix: The prefix to give tables added to the database by the
                         crawler.
       :param data_source: The location of the data source that is targeted by
the
                           crawler and extracted during job runs.
       :param job_script: The job script that is used to transform data during
job
                          runs.
       :param job_name: The name to give the job definition that is created
during the
                        scenario.
       """
       wrapper = GlueWrapper(self.glue_client)
       print(f"Checking for crawler {crawler_name}.")
       crawler = wrapper.get_crawler(crawler_name)
       if crawler is None:
           print(f"Creating crawler {crawler_name}.")
           wrapper.create_crawler(
               crawler_name,
               self.glue_service_role.arn,
               db_name,
               db_prefix,
               data_source,
           )
           print(f"Created crawler {crawler_name}.")
           crawler = wrapper.get_crawler(crawler_name)
       pprint(crawler)
       print("-" * 88)

       print(
           f"When you run the crawler, it crawls data stored in {data_source}
and "
```

```
            f"creates a metadata database in the AWS Glue Data Catalog that
describes "
            f"the data in the data source."
        )
        print("In this example, the source data is in CSV format.")
        ready = False
        while not ready:
            ready = Question.ask_question(
                "Ready to start the crawler? (y/n) ", Question.is_yesno
            )
        wrapper.start_crawler(crawler_name)
        print("Let's wait for the crawler to run. This typically takes a few
minutes.")
        crawler_state = None
        while crawler_state != "READY":
            self.wait(10)
            crawler = wrapper.get_crawler(crawler_name)
            crawler_state = crawler["State"]
            print(f"Crawler is {crawler['State']}.")
        print("-" * 88)

        database = wrapper.get_database(db_name)
        print(f"The crawler created database {db_name}:")
        pprint(database)
        print(f"The database contains these tables:")
        tables = wrapper.get_tables(db_name)
        for index, table in enumerate(tables):
            print(f"\t{index + 1}. {table['Name']}")
        table_index = Question.ask_question(
            f"Enter the number of a table to see more detail: ",
            Question.is_int,
            Question.in_range(1, len(tables)),
        )
        pprint(tables[table_index - 1])
        print("-" * 88)

        print(f"Creating job definition {job_name}.")
        wrapper.create_job(
            job_name,
            "Getting started example job.",
            self.glue_service_role.arn,
            f"s3://{self.glue_bucket.name}/{job_script}",
        )
        print("Created job definition.")
```

```
        print(
            f"When you run the job, it extracts data from {data_source},
transforms it "
            f"by using the {job_script} script, and loads the output into "
            f"S3 bucket {self.glue_bucket.name}."
        )
        print(
            "In this example, the data is transformed from CSV to JSON, and only
a few "
            "fields are included in the output."
        )
        job_run_status = None
        if Question.ask_question(f"Ready to run? (y/n) ", Question.is_yesno):
            job_run_id = wrapper.start_job_run(
                job_name, db_name, tables[0]["Name"], self.glue_bucket.name
            )
            print(f"Job {job_name} started. Let's wait for it to run.")
            while job_run_status not in ["SUCCEEDED", "STOPPED", "FAILED",
"TIMEOUT"]:
                self.wait(10)
                job_run = wrapper.get_job_run(job_name, job_run_id)
                job_run_status = job_run["JobRunState"]
                print(f"Job {job_name}/{job_run_id} is {job_run_status}.")
        print("-" * 88)

        if job_run_status == "SUCCEEDED":
            print(
                f"Data from your job run is stored in your S3 bucket
'{self.glue_bucket.name}':"
            )
            try:
                keys = [
                    obj.key for obj in
self.glue_bucket.objects.filter(Prefix="run-")
                ]
                for index, key in enumerate(keys):
                    print(f"\t{index + 1}: {key}")
                lines = 4
                key_index = Question.ask_question(
                    f"Enter the number of a block to download it and see the
first {lines} "
                    f"lines of JSON output in the block: ",
                    Question.is_int,
                    Question.in_range(1, len(keys)),
```

```
                )
                job_data = io.BytesIO()
                self.glue_bucket.download_fileobj(keys[key_index - 1], job_data)
                job_data.seek(0)
                for _ in range(lines):
                    print(job_data.readline().decode("utf-8"))
            except ClientError as err:
                logger.error(
                    "Couldn't get job run data. Here's why: %s: %s",
                    err.response["Error"]["Code"],
                    err.response["Error"]["Message"],
                )
                raise
            print("-" * 88)

        job_names = wrapper.list_jobs()
        if job_names:
            print(f"Your account has {len(job_names)} jobs defined:")
            for index, job_name in enumerate(job_names):
                print(f"\t{index + 1}. {job_name}")
            job_index = Question.ask_question(
                f"Enter a number between 1 and {len(job_names)} to see the list
of runs for "
                f"a job: ",
                Question.is_int,
                Question.in_range(1, len(job_names)),
            )
            job_runs = wrapper.get_job_runs(job_names[job_index - 1])
            if job_runs:
                print(f"Found {len(job_runs)} runs for job {job_names[job_index -
1]}:")
                for index, job_run in enumerate(job_runs):
                    print(
                        f"\t{index + 1}. {job_run['JobRunState']} on "
                        f"{job_run['CompletedOn']:%Y-%m-%d %H:%M:%S}"
                    )
                run_index = Question.ask_question(
                    f"Enter a number between 1 and {len(job_runs)} to see details
for a run: ",
                    Question.is_int,
                    Question.in_range(1, len(job_runs)),
                )
                pprint(job_runs[run_index - 1])
            else:
```

```
                print(f"No runs found for job {job_names[job_index - 1]}")
        else:
            print("Your account doesn't have any jobs defined.")
        print("-" * 88)

        print(
            f"Let's clean up. During this example we created job definition
 '{job_name}'."
        )
        if Question.ask_question(
            "Do you want to delete the definition and all runs? (y/n) ",
            Question.is_yesno,
        ):
            wrapper.delete_job(job_name)
            print(f"Job definition '{job_name}' deleted.")
        tables = wrapper.get_tables(db_name)
        print(f"We also created database '{db_name}' that contains these
 tables:")
        for table in tables:
            print(f"\t{table['Name']}")
        if Question.ask_question(
            "Do you want to delete the tables and the database? (y/n) ",
            Question.is_yesno,
        ):
            for table in tables:
                wrapper.delete_table(db_name, table["Name"])
                print(f"Deleted table {table['Name']}.")
            wrapper.delete_database(db_name)
            print(f"Deleted database {db_name}.")
        print(f"We also created crawler '{crawler_name}'.")
        if Question.ask_question(
            "Do you want to delete the crawler? (y/n) ", Question.is_yesno
        ):
            wrapper.delete_crawler(crawler_name)
            print(f"Deleted crawler {crawler_name}.")
        print("-" * 88)


def parse_args(args):
    """
    Parse command line arguments.

    :param args: The command line arguments.
    :return: The parsed arguments.
```

```
    """
    parser = argparse.ArgumentParser(
        description="Runs the AWS Glue getting started with crawlers and jobs
 scenario. "
        "Before you run this scenario, set up scaffold resources by running "
        "'python scaffold.py deploy'."
    )
    parser.add_argument(
        "role_name",
        help="The name of an IAM role that AWS Glue can assume. This role must
 grant access "
        "to Amazon S3 and to the permissions granted by the AWSGlueServiceRole "
        "managed policy.",
    )
    parser.add_argument(
        "bucket_name",
        help="The name of an S3 bucket that AWS Glue can access to get the job
 script and "
        "put job results.",
    )
    parser.add_argument(
        "--job_script",
        default="flight_etl_job_script.py",
        help="The name of the job script file that is used in the scenario.",
    )
    return parser.parse_args(args)


def main():
    args = parse_args(sys.argv[1:])
    try:
        print("-" * 88)
        print(
            "Welcome to the AWS Glue getting started with crawlers and jobs
 scenario."
        )
        print("-" * 88)
        scenario = GlueCrawlerJobScenario(
            boto3.client("glue"),
            boto3.resource("iam").Role(args.role_name),
            boto3.resource("s3").Bucket(args.bucket_name),
        )
        scenario.upload_job_script(args.job_script)
        scenario.run(
```

```
            "doc-example-crawler",
            "doc-example-database",
            "doc-example-",
            "s3://crawler-public-us-east-1/flight/2016/csv",
            args.job_script,
            "doc-example-job",
        )
        print("-" * 88)
        print(
            "To destroy scaffold resources, including the IAM role and S3 bucket
 "
            "used in this scenario, run 'python scaffold.py destroy'."
        )
        print("\nThanks for watching!")
        print("-" * 88)
    except Exception:
        logging.exception("Something went wrong with the example.")
```

Créez un script ETL utilisé AWS Glue pour extraire, transformer et charger des données lors de l'exécution des tâches.

```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job

"""
These custom arguments must be passed as Arguments to the StartJobRun request.
    --input_database    The name of a metadata database that is contained in
 your
                        AWS Glue Data Catalog and that contains tables that
 describe
                        the data to be processed.
    --input_table       The name of a table in the database that describes the
 data to
                        be processed.
    --output_bucket_url An S3 bucket that receives the transformed output data.
"""
```

```
args = getResolvedOptions(
    sys.argv, ["JOB_NAME", "input_database", "input_table", "output_bucket_url"]
)
sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
job = Job(glueContext)
job.init(args["JOB_NAME"], args)

# Script generated for node S3 Flight Data.
S3FlightData_node1 = glueContext.create_dynamic_frame.from_catalog(
    database=args["input_database"],
    table_name=args["input_table"],
    transformation_ctx="S3FlightData_node1",
)

# This mapping performs two main functions:
# 1. It simplifies the output by removing most of the fields from the data.
# 2. It renames some fields. For example, `fl_date` is renamed to `flight_date`.
ApplyMapping_node2 = ApplyMapping.apply(
    frame=S3FlightData_node1,
    mappings=[
        ("year", "long", "year", "long"),
        ("month", "long", "month", "tinyint"),
        ("day_of_month", "long", "day", "tinyint"),
        ("fl_date", "string", "flight_date", "string"),
        ("carrier", "string", "carrier", "string"),
        ("fl_num", "long", "flight_num", "long"),
        ("origin_city_name", "string", "origin_city_name", "string"),
        ("origin_state_abr", "string", "origin_state_abr", "string"),
        ("dest_city_name", "string", "dest_city_name", "string"),
        ("dest_state_abr", "string", "dest_state_abr", "string"),
        ("dep_time", "long", "departure_time", "long"),
        ("wheels_off", "long", "wheels_off", "long"),
        ("wheels_on", "long", "wheels_on", "long"),
        ("arr_time", "long", "arrival_time", "long"),
        ("mon", "string", "mon", "string"),
    ],
    transformation_ctx="ApplyMapping_node2",
)

# Script generated for node Revised Flight Data.
RevisedFlightData_node3 = glueContext.write_dynamic_frame.from_options(
    frame=ApplyMapping_node2,
```

```
    connection_type="s3",
    format="json",
    connection_options={"path": args["output_bucket_url"], "partitionKeys": []},
    transformation_ctx="RevisedFlightData_node3",
)

job.commit()
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des API du kit AWS SDK for Python (Boto3).
  - CreateCrawler
  - CreateJob
  - DeleteCrawler
  - DeleteDatabase
  - DeleteJob
  - DeleteTable
  - GetCrawler
  - GetDatabase
  - GetDatabases
  - GetJob
  - GetJobRun
  - GetJobRuns
  - GetTables
  - ListJobs
  - StartCrawler
  - StartJobRun

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez une classe qui englobe les AWS Glue fonctions utilisées dans le scénario.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Retrieves information about a specific crawler.
  #
  # @param name [String] The name of the crawler to retrieve information about.
  # @return [Aws::Glue::Types::Crawler, nil] The crawler object if found, or nil
 if not found.
  def get_crawler(name)
    @glue_client.get_crawler(name: name)
  rescue Aws::Glue::Errors::EntityNotFoundException
    @logger.info("Crawler #{name} doesn't exist.")
    false
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not get crawler #{name}: \n#{e.message}")
    raise
  end

  # Creates a new crawler with the specified configuration.
  #
  # @param name [String] The name of the crawler.
```

```
 # @param role_arn [String] The ARN of the IAM role to be used by the crawler.
 # @param db_name [String] The name of the database where the crawler stores its
metadata.
 # @param db_prefix [String] The prefix to be added to the names of tables that
the crawler creates.
 # @param s3_target [String] The S3 path that the crawler will crawl.
 # @return [void]
 def create_crawler(name, role_arn, db_name, _db_prefix, s3_target)
   @glue_client.create_crawler(
     name: name,
     role: role_arn,
     database_name: db_name,
     targets: {
       s3_targets: [
         {
           path: s3_target
         }
       ]
     }
   )
 rescue Aws::Glue::Errors::GlueException => e
   @logger.error("Glue could not create crawler: \n#{e.message}")
   raise
 end

 # Starts a crawler with the specified name.
 #
 # @param name [String] The name of the crawler to start.
 # @return [void]
 def start_crawler(name)
   @glue_client.start_crawler(name: name)
 rescue Aws::Glue::Errors::ServiceError => e
   @logger.error("Glue could not start crawler #{name}: \n#{e.message}")
   raise
 end

 # Deletes a crawler with the specified name.
 #
 # @param name [String] The name of the crawler to delete.
 # @return [void]
 def delete_crawler(name)
   @glue_client.delete_crawler(name: name)
 rescue Aws::Glue::Errors::ServiceError => e
   @logger.error("Glue could not delete crawler #{name}: \n#{e.message}")
```

```
    raise
  end

  # Retrieves information about a specific database.
  #
  # @param name [String] The name of the database to retrieve information about.
  # @return [Aws::Glue::Types::Database, nil] The database object if found, or
nil if not found.
  def get_database(name)
    response = @glue_client.get_database(name: name)
    response.database
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not get database #{name}: \n#{e.message}")
    raise
  end

  # Retrieves a list of tables in the specified database.
  #
  # @param db_name [String] The name of the database to retrieve tables from.
  # @return [Array<Aws::Glue::Types::Table>]
  def get_tables(db_name)
    response = @glue_client.get_tables(database_name: db_name)
    response.table_list
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not get tables #{db_name}: \n#{e.message}")
    raise
  end

  # Creates a new job with the specified configuration.
  #
  # @param name [String] The name of the job.
  # @param description [String] The description of the job.
  # @param role_arn [String] The ARN of the IAM role to be used by the job.
  # @param script_location [String] The location of the ETL script for the job.
  # @return [void]
  def create_job(name, description, role_arn, script_location)
    @glue_client.create_job(
      name: name,
      description: description,
      role: role_arn,
      command: {
        name: 'glueetl',
        script_location: script_location,
        python_version: '3'
```

```
      },
      glue_version: '3.0'
    )
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not create job #{name}: \n#{e.message}")
    raise
  end

  # Starts a job run for the specified job.
  #
  # @param name [String] The name of the job to start the run for.
  # @param input_database [String] The name of the input database for the job.
  # @param input_table [String] The name of the input table for the job.
  # @param output_bucket_name [String] The name of the output S3 bucket for the
job.
  # @return [String] The ID of the started job run.
  def start_job_run(name, input_database, input_table, output_bucket_name)
    response = @glue_client.start_job_run(
      job_name: name,
      arguments: {
        '--input_database': input_database,
        '--input_table': input_table,
        '--output_bucket_url': "s3://#{output_bucket_name}/"
      }
    )
    response.job_run_id
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not start job run #{name}: \n#{e.message}")
    raise
  end

  # Retrieves a list of jobs in AWS Glue.
  #
  # @return [Aws::Glue::Types::ListJobsResponse]
  def list_jobs
    @glue_client.list_jobs
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not list jobs: \n#{e.message}")
    raise
  end

  # Retrieves a list of job runs for the specified job.
  #
  # @param job_name [String] The name of the job to retrieve job runs for.
```

```
  # @return [Array<Aws::Glue::Types::JobRun>]
  def get_job_runs(job_name)
    response = @glue_client.get_job_runs(job_name: job_name)
    response.job_runs
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not get job runs: \n#{e.message}")
  end

  # Retrieves data for a specific job run.
  #
  # @param job_name [String] The name of the job run to retrieve data for.
  # @return [Glue::Types::GetJobRunResponse]
  def get_job_run(job_name, run_id)
    @glue_client.get_job_run(job_name: job_name, run_id: run_id)
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not get job runs: \n#{e.message}")
  end

  # Deletes a job with the specified name.
  #
  # @param job_name [String] The name of the job to delete.
  # @return [void]
  def delete_job(job_name)
    @glue_client.delete_job(job_name: job_name)
  rescue Aws::Glue::Errors::ServiceError => e
    @logger.error("Glue could not delete job: \n#{e.message}")
  end

  # Deletes a table with the specified name.
  #
  # @param database_name [String] The name of the catalog database in which the
table resides.
  # @param table_name [String] The name of the table to be deleted.
  # @return [void]
  def delete_table(database_name, table_name)
    @glue_client.delete_table(database_name: database_name, name: table_name)
  rescue Aws::Glue::Errors::ServiceError => e
    @logger.error("Glue could not delete job: \n#{e.message}")
  end

  # Removes a specified database from a Data Catalog.
  #
  # @param database_name [String] The name of the database to delete.
  # @return [void]
```

```
  def delete_database(database_name)
    @glue_client.delete_database(name: database_name)
  rescue Aws::Glue::Errors::ServiceError => e
    @logger.error("Glue could not delete database: \n#{e.message}")
  end

  # Uploads a job script file to an S3 bucket.
  #
  # @param file_path [String] The local path of the job script file.
  # @param bucket_resource [Aws::S3::Bucket] The S3 bucket resource to upload the
 file to.
  # @return [void]
  def upload_job_script(file_path, bucket_resource)
    File.open(file_path) do |file|
      bucket_resource.client.put_object({
                                          body: file,
                                          bucket: bucket_resource.name,
                                          key: file_path
                                        })
    end
  rescue Aws::S3::Errors::S3UploadFailedError => e
    @logger.error("S3 could not upload job script: \n#{e.message}")
    raise
  end
end
```

Créez une classe qui exécute le scénario.

```
class GlueCrawlerJobScenario
  def initialize(glue_client, glue_service_role, glue_bucket, logger)
    @glue_client = glue_client
    @glue_service_role = glue_service_role
    @glue_bucket = glue_bucket
    @logger = logger
  end

  def run(crawler_name, db_name, db_prefix, data_source, job_script, job_name)
    wrapper = GlueWrapper.new(@glue_client, @logger)
    setup_crawler(wrapper, crawler_name, db_name, db_prefix, data_source)
    query_database(wrapper, crawler_name, db_name)
    create_and_run_job(wrapper, job_script, job_name, db_name)
  end
```

```
  private

  def setup_crawler(wrapper, crawler_name, db_name, db_prefix, data_source)
    new_step(1, 'Create a crawler')
    crawler = wrapper.get_crawler(crawler_name)
    unless crawler
      puts "Creating crawler #{crawler_name}."
      wrapper.create_crawler(crawler_name, @glue_service_role.arn, db_name,
 db_prefix, data_source)
      puts "Successfully created #{crawler_name}."
    end
    wrapper.start_crawler(crawler_name)
    monitor_crawler(wrapper, crawler_name)
  end

  def monitor_crawler(wrapper, crawler_name)
    new_step(2, 'Monitor Crawler')
    crawler_state = nil
    until crawler_state == 'READY'
      custom_wait(15)
      crawler = wrapper.get_crawler(crawler_name)
      crawler_state = crawler[0]['state']
      print "Crawler status: #{crawler_state}".yellow
    end
  end

  def query_database(wrapper, _crawler_name, db_name)
    new_step(3, 'Query the database.')
    wrapper.get_database(db_name)
    puts "The crawler created database #{db_name}:"
    puts "Database contains tables: #{wrapper.get_tables(db_name).map { |t|
 t['name'] }}"
  end

  def create_and_run_job(wrapper, job_script, job_name, db_name)
    new_step(4, 'Create and run job.')
    wrapper.upload_job_script(job_script, @glue_bucket)
    wrapper.create_job(job_name, 'ETL Job', @glue_service_role.arn, "s3://
#{@glue_bucket.name}/#{job_script}")
    run_job(wrapper, job_name, db_name)
  end

  def run_job(wrapper, job_name, db_name)
```

```
    new_step(5, 'Run the job.')
    wrapper.start_job_run(job_name, db_name, wrapper.get_tables(db_name)[0]
['name'], @glue_bucket.name)
    job_run_status = nil
    until %w[SUCCEEDED FAILED STOPPED].include?(job_run_status)
      custom_wait(10)
      job_run = wrapper.get_job_runs(job_name)
      job_run_status = job_run[0]['job_run_state']
      print "Job #{job_name} status: #{job_run_status}".yellow
    end
  end
end

def main
  banner('../../helpers/banner.txt')
  puts 'Starting AWS Glue demo...'

  # Load resource names from YAML.
  resource_names = YAML.load_file('resource_names.yaml')

  # Setup services and resources.
  iam_role = Aws::IAM::Resource.new(region: 'us-
east-1').role(resource_names['glue_service_role'])
  s3_bucket = Aws::S3::Resource.new(region: 'us-
east-1').bucket(resource_names['glue_bucket'])

  # Instantiate scenario and run.
  scenario = GlueCrawlerJobScenario.new(Aws::Glue::Client.new(region: 'us-
east-1'), iam_role, s3_bucket, @logger)
  random_suffix = rand(10**4)
  scenario.run("crawler-#{random_suffix}", "db-#{random_suffix}", "prefix-
#{random_suffix}-", 's3://data_source',
               'job_script.py', "job-#{random_suffix}")

  puts 'Demo complete.'
end
```

Créez un script ETL utilisé AWS Glue pour extraire, transformer et charger des données lors de l'exécution des tâches.

```
import sys
from awsglue.transforms import *
```

```
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job

"""
These custom arguments must be passed as Arguments to the StartJobRun request.
    --input_database    The name of a metadata database that is contained in
 your
                        AWS Glue Data Catalog and that contains tables that
 describe
                        the data to be processed.
    --input_table       The name of a table in the database that describes the
 data to
                        be processed.
    --output_bucket_url An S3 bucket that receives the transformed output data.
"""
args = getResolvedOptions(
    sys.argv, ["JOB_NAME", "input_database", "input_table", "output_bucket_url"]
)
sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
job = Job(glueContext)
job.init(args["JOB_NAME"], args)

# Script generated for node S3 Flight Data.
S3FlightData_node1 = glueContext.create_dynamic_frame.from_catalog(
    database=args["input_database"],
    table_name=args["input_table"],
    transformation_ctx="S3FlightData_node1",
)

# This mapping performs two main functions:
# 1. It simplifies the output by removing most of the fields from the data.
# 2. It renames some fields. For example, `fl_date` is renamed to `flight_date`.
ApplyMapping_node2 = ApplyMapping.apply(
    frame=S3FlightData_node1,
    mappings=[
        ("year", "long", "year", "long"),
        ("month", "long", "month", "tinyint"),
        ("day_of_month", "long", "day", "tinyint"),
        ("fl_date", "string", "flight_date", "string"),
        ("carrier", "string", "carrier", "string"),
```

```
        ("fl_num", "long", "flight_num", "long"),
        ("origin_city_name", "string", "origin_city_name", "string"),
        ("origin_state_abr", "string", "origin_state_abr", "string"),
        ("dest_city_name", "string", "dest_city_name", "string"),
        ("dest_state_abr", "string", "dest_state_abr", "string"),
        ("dep_time", "long", "departure_time", "long"),
        ("wheels_off", "long", "wheels_off", "long"),
        ("wheels_on", "long", "wheels_on", "long"),
        ("arr_time", "long", "arrival_time", "long"),
        ("mon", "string", "mon", "string"),
    ],
    transformation_ctx="ApplyMapping_node2",
)

# Script generated for node Revised Flight Data.
RevisedFlightData_node3 = glueContext.write_dynamic_frame.from_options(
    frame=ApplyMapping_node2,
    connection_type="s3",
    format="json",
    connection_options={"path": args["output_bucket_url"], "partitionKeys": []},
    transformation_ctx="RevisedFlightData_node3",
)

job.commit()
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des API du kit AWS SDK pour Ruby .
  - CreateCrawler
  - CreateJob
  - DeleteCrawler
  - DeleteDatabase
  - DeleteJob
  - DeleteTable
  - GetCrawler
  - GetDatabase
  - GetDatabases
  - GetJob
  - GetJobRun

- GetJobRuns
- GetTables
- ListJobs
- StartCrawler
- StartJobRun

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez et exécutez un crawler qui analyse un compartiment Amazon Simple Storage Service (Amazon S3) public et génère une base de données de métadonnées qui décrit les données au format CSV trouvées.

```
        let create_crawler = glue
            .create_crawler()
            .name(self.crawler())
            .database_name(self.database())
            .role(self.iam_role.expose_secret())
            .targets(
                CrawlerTargets::builder()
                    .s3_targets(S3Target::builder().path(CRAWLER_TARGET).build())
                    .build(),
            )
            .send()
            .await;

        match create_crawler {
            Err(err) => {
                let glue_err: aws_sdk_glue::Error = err.into();
                match glue_err {
                    aws_sdk_glue::Error::AlreadyExistsException(_) => {
                        info!("Using existing crawler");
```

```
                    Ok(())
                }
                _ => Err(GlueMvpError::GlueSdk(glue_err)),
            }
        }
        Ok(_) => Ok(()),
    }?;

    let start_crawler =
glue.start_crawler().name(self.crawler()).send().await;

    match start_crawler {
        Ok(_) => Ok(()),
        Err(err) => {
            let glue_err: aws_sdk_glue::Error = err.into();
            match glue_err {
                aws_sdk_glue::Error::CrawlerRunningException(_) => Ok(()),
                _ => Err(GlueMvpError::GlueSdk(glue_err)),
            }
        }
    }?;
```

Répertoriez les informations relatives aux bases de données et aux tables de votre AWS Glue Data Catalog.

```
    let database = glue
        .get_database()
        .name(self.database())
        .send()
        .await
        .map_err(GlueMvpError::from_glue_sdk)?
        .to_owned();
    let database = database
        .database()
        .ok_or_else(|| GlueMvpError::Unknown("Could not find
database".into()))?;

    let tables = glue
        .get_tables()
        .database_name(self.database())
        .send()
        .await
```

```
            .map_err(GlueMvpError::from_glue_sdk)?;

        let tables = tables.table_list();
```

Créez et exécutez une tâche qui extrait les données CSV du compartiment Amazon S3 source, les transforme en supprimant et en renommant des champs, et charge la sortie au format JSON dans un autre compartiment Amazon S3.

```
        let create_job = glue
            .create_job()
            .name(self.job())
            .role(self.iam_role.expose_secret())
            .command(
                JobCommand::builder()
                    .name("glueetl")
                    .python_version("3")
                    .script_location(format!("s3://{}/job.py", self.bucket()))
                    .build(),
            )
            .glue_version("3.0")
            .send()
            .await
            .map_err(GlueMvpError::from_glue_sdk)?;

        let job_name = create_job.name().ok_or_else(|| {
            GlueMvpError::Unknown("Did not get job name after creating
job".into())
        })?;

        let job_run_output = glue
            .start_job_run()
            .job_name(self.job())
            .arguments("--input_database", self.database())
            .arguments(
                "--input_table",
                self.tables
                    .first()
                    .ok_or_else(|| GlueMvpError::Unknown("Missing crawler
table".into()))?
                    .name(),
            )
            .arguments("--output_bucket_url", self.bucket())
```

```
            .send()
            .await
            .map_err(GlueMvpError::from_glue_sdk)?;

        let job = job_run_output
            .job_run_id()
            .ok_or_else(|| GlueMvpError::Unknown("Missing run id from just
started job".into()))?
            .to_string();
```

Supprimez toutes les ressources créées par la démonstration.

```
        glue.delete_job()
            .job_name(self.job())
            .send()
            .await
            .map_err(GlueMvpError::from_glue_sdk)?;

        for t in &self.tables {
            glue.delete_table()
                .name(t.name())
                .database_name(self.database())
                .send()
                .await
                .map_err(GlueMvpError::from_glue_sdk)?;
        }

        glue.delete_database()
            .name(self.database())
            .send()
            .await
            .map_err(GlueMvpError::from_glue_sdk)?;

        glue.delete_crawler()
            .name(self.crawler())
            .send()
            .await
            .map_err(GlueMvpError::from_glue_sdk)?;
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des API du kit AWS SDK pour Rust.

- CreateCrawler
- CreateJob
- DeleteCrawler
- DeleteDatabase
- DeleteJob
- DeleteTable
- GetCrawler
- GetDatabase
- GetDatabases
- GetJob
- GetJobRun
- GetJobRuns
- GetTables
- ListJobs
- StartCrawler
- StartJobRun

Swift

Kit SDK pour Swift

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Fichier `Package.swift`.

```
// swift-tools-version: 5.9
//
// The swift-tools-version declares the minimum version of Swift required to
// build this package.

import PackageDescription
```

```
let package = Package(
    name: "glue-scenario",
    // Let Xcode know the minimum Apple platforms supported.
    platforms: [
        .macOS(.v13),
        .iOS(.v15)
    ],
    dependencies: [
        // Dependencies declare other packages that this package depends on.
        .package(
            url: "https://github.com/awslabs/aws-sdk-swift",
            from: "1.0.0"),
        .package(
            url: "https://github.com/apple/swift-argument-parser.git",
            branch: "main"
        )
    ],
    targets: [
        // Targets are the basic building blocks of a package, defining a module
 or a test suite.
        // Targets can depend on other targets in this package and products
        // from dependencies.
        .executableTarget(
            name: "glue-scenario",
            dependencies: [
                .product(name: "AWSGlue", package: "aws-sdk-swift"),
                .product(name: "AWSS3", package: "aws-sdk-swift"),
                .product(name: "ArgumentParser", package: "swift-argument-
parser")
            ],
            path: "Sources")

    ]
)
```

Le fichier de code Swift, `entry.swift`.

```
// An example that shows how to use the AWS SDK for Swift to demonstrate
// creating and using crawlers and jobs using AWS Glue.
//
// 0. Upload the Python job script to Amazon S3 so it can be used when
```

```
//    calling `startJobRun()` later.
// 1. Create a crawler, pass it the IAM role and the URL of the public Amazon
//    S3 bucket that contains the source data:
//    s3://crawler-public-us-east-1/flight/2016/csv.
// 2. Start the crawler. This takes time, so after starting it, use a loop
//    that calls `getCrawler()` until the state is "READY".
// 3. Get the database created by the crawler, and the tables in the
//    database. Display them to the user.
// 4. Create a job. Pass it the IAM role and the URL to a Python ETL script
//    previously uploaded to the user's S3 bucket.
// 5. Start a job run, passing the following custom arguments. These are
//    expected by the ETL script, so must exactly match.
//    * `--input_database: <name of the database created by the crawler>`
//    * `--input_table: <name of the table created by the crawler>`
//    * `--output_bucket_url: <URL to the scaffold bucket created for the
//      user>`
// 6. Loop and get the job run until it returns one of the following states:
//    "SUCCEEDED", "STOPPED", "FAILED", or "TIMEOUT".
// 7. Output data is stored in a group of files in the user's S3 bucket.
//    Either direct the user to their location or download a file and display
//    the results inline.
// 8. List the jobs for the user's account.
// 9. Get job run details for a job run.
// 10. Delete the demo job.
// 11. Delete the database and tables created by the example.
// 12. Delete the crawler created by the example.

import ArgumentParser
import AWSS3
import Foundation
import Smithy

import AWSClientRuntime
import AWSGlue

struct ExampleCommand: ParsableCommand {
    @Option(help: "The AWS IAM role to use for AWS Glue calls.")
    var role: String

    @Option(help: "The Amazon S3 bucket to use for this example.")
    var bucket: String

    @Option(help: "The Amazon S3 URL of the data to crawl.")
    var s3url: String = "s3://crawler-public-us-east-1/flight/2016/csv"
```

```
    @Option(help: "The Python script to run as a job with AWS Glue.")
    var script: String = "./flight_etl_job_script.py"

    @Option(help: "The AWS Region to run AWS API calls in.")
    var awsRegion = "us-east-1"

    @Option(help: "A prefix string to use when naming tables.")
    var tablePrefix = "swift-glue-basics-table"

    @Option(
        help: ArgumentHelp("The level of logging for the Swift SDK to perform."),
        completion: .list([
            "critical",
            "debug",
            "error",
            "info",
            "notice",
            "trace",
            "warning"
        ])
    )
    var logLevel: String = "error"

    static var configuration = CommandConfiguration(
        commandName: "glue-scenario",
        abstract: """
        Demonstrates various features of AWS Glue.
        """,
        discussion: """
        An example showing how to use AWS Glue to create, run, and monitor
        crawlers and jobs.
        """
    )

    /// Generate and return a unique file name that begins with the specified
    /// string.
    ///
    /// - Parameters:
    ///   - prefix: Text to use at the beginning of the returned name.
    ///
    /// - Returns: A string containing a unique filename that begins with the
    ///   specified `prefix`.
    ///
```

```
    /// The returned name uses a random number between 1 million and 1 billion to
    /// provide reasonable certainty of uniqueness for the purposes of this
    /// example.
    func tempName(prefix: String) -> String {
        return "\(prefix)-\(Int.random(in: 1000000..<1000000000))"
    }

    /// Upload a file to an Amazon S3 bucket.
    ///
    /// - Parameters:
    ///   - s3Client: The S3 client to use when uploading the file.
    ///   - path: The local path of the source file to upload.
    ///   - toBucket: The name of the S3 bucket into which to upload the file.
    ///   - key: The key (name) to give the file in the S3 bucket.
    ///
    /// - Returns: `true` if the file is uploaded successfully, otherwise
`false`.
    func uploadFile(s3Client: S3Client, path: String, toBucket: String, key:
String) async -> Bool {
        do {
            let fileData: Data = try Data(contentsOf: URL(fileURLWithPath: path))
            let dataStream = ByteStream.data(fileData)
            _ = try await s3Client.putObject(
                input: PutObjectInput(
                    body: dataStream,
                    bucket: toBucket,
                    key: key
                )
            )
        } catch {
            print("*** An unexpected error occurred uploading the script to the
Amazon S3 bucket \"\(bucket)\".")
            return false
        }

        return true
    }

    /// Create a new AWS Glue crawler.
    ///
    /// - Parameters:
    ///   - glueClient: An AWS Glue client to use for the crawler.
    ///   - crawlerName: A name for the new crawler.
    ///   - iamRole: The name of an Amazon IAM role for the crawler to use.
```

```
    ///   - s3Path: The path of an Amazon S3 folder to use as a target location.
    ///   - cronSchedule: A `cron` schedule indicating when to run the crawler.
    ///   - databaseName: The name of an AWS Glue database to operate on.
    ///
    /// - Returns: `true` if the crawler is created successfully, otherwise
`false`.
    func createCrawler(glueClient: GlueClient, crawlerName: String, iamRole:
String,
                       s3Path: String, cronSchedule: String, databaseName:
String) async -> Bool {
        let s3Target = GlueClientTypes.S3Target(path: s3url)
        let targetList = GlueClientTypes.CrawlerTargets(s3Targets: [s3Target])

        do {
            _ = try await glueClient.createCrawler(
                input: CreateCrawlerInput(
                    databaseName: databaseName,
                    description: "Created by the AWS SDK for Swift Scenario
Example for AWS Glue.",
                    name: crawlerName,
                    role: iamRole,
                    schedule: cronSchedule,
                    tablePrefix: tablePrefix,
                    targets: targetList
                )
            )
        } catch _ as AlreadyExistsException {
            print("*** A crawler named \"\(crawlerName)\" already exists.")
            return false
        } catch _ as OperationTimeoutException {
            print("*** The attempt to create the AWS Glue crawler timed out.")
            return false
        } catch {
            print("*** An unexpected error occurred creating the AWS Glue
crawler: \(error.localizedDescription)")
            return false
        }

        return true
    }

    /// Delete an AWS Glue crawler.
    ///
    /// - Parameters:
```

```
    ///   - glueClient: The AWS Glue client to use.
    ///   - name: The name of the crawler to delete.
    ///
    /// - Returns: `true` if successful, otherwise `false`.
    func deleteCrawler(glueClient: GlueClient, name: String) async -> Bool {
        do {
            _ = try await glueClient.deleteCrawler(
                input: DeleteCrawlerInput(name: name)
            )
        } catch {
            return false
        }
        return true
    }

    /// Start running an AWS Glue crawler.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use when starting the crawler.
    ///   - name: The name of the crawler to start running.
    ///
    /// - Returns: `true` if the crawler is started successfully, otherwise
`false`.
    func startCrawler(glueClient: GlueClient, name: String) async -> Bool {
        do {
            _ = try await glueClient.startCrawler(
                input: StartCrawlerInput(name: name)
            )
        } catch {
            print("*** An unexpected error occurred starting the crawler.")
            return false
        }

        return true
    }

    /// Get the state of the specified AWS Glue crawler.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - name: The name of the crawler whose state should be returned.
    ///
    /// - Returns: A `GlueClientTypes.CrawlerState` value describing the
    ///   state of the crawler.
```

```
   func getCrawlerState(glueClient: GlueClient, name: String) async ->
GlueClientTypes.CrawlerState {
       do {
           let output = try await glueClient.getCrawler(
               input: GetCrawlerInput(name: name)
           )

           // If the crawler or its state is `nil`, report that the crawler
           // is stopping. This may not be what you want for your
           // application but it works for this one!

           guard let crawler = output.crawler else {
               return GlueClientTypes.CrawlerState.stopping
           }
           guard let state = crawler.state else {
               return GlueClientTypes.CrawlerState.stopping
           }
           return state
       } catch {
           return GlueClientTypes.CrawlerState.stopping
       }
   }

   /// Wait until the specified crawler is ready to run.
   ///
   /// - Parameters:
   ///   - glueClient: The AWS Glue client to use.
   ///   - name: The name of the crawler to wait for.
   ///
   /// - Returns: `true` if the crawler is ready, `false` if the client is
   ///   stopping (and will therefore never be ready).
   func waitUntilCrawlerReady(glueClient: GlueClient, name: String) async ->
Bool {
       while true {
           let state = await getCrawlerState(glueClient: glueClient, name: name)

           if state == .ready {
               return true
           } else if state == .stopping {
               return false
           }

           // Wait four seconds before trying again.
```

```
            do {
                try await Task.sleep(for: .seconds(4))
            } catch {
                print("*** Error pausing the task.")
            }
        }
    }

    /// Create a new AWS Glue job.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - jobName: The name to give the new job.
    ///   - role: The IAM role for the job to use when accessing AWS services.
    ///   - scriptLocation: The AWS S3 URI of the script to be run by the job.
    ///
    /// - Returns: `true` if the job is created successfully, otherwise `false`.
    func createJob(glueClient: GlueClient, name jobName: String, role: String,
                   scriptLocation: String) async -> Bool {
        let command = GlueClientTypes.JobCommand(
            name: "glueetl",
            pythonVersion: "3",
            scriptLocation: scriptLocation
        )

        do {
            _ = try await glueClient.createJob(
                input: CreateJobInput(
                    command: command,
                    description: "Created by the AWS SDK for Swift Glue basic
scenario example.",
                    glueVersion: "3.0",
                    name: jobName,
                    numberOfWorkers: 10,
                    role: role,
                    workerType: .g1x
                )
            )
        } catch {
            return false
        }
        return true
    }
```

```
    /// Return a list of the AWS Glue jobs listed on the user's account.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - maxJobs: The maximum number of jobs to return (default: 100).
    ///
    /// - Returns: An array of strings listing the names of all available AWS
    ///   Glue jobs.
    func listJobs(glueClient: GlueClient, maxJobs: Int = 100) async -> [String] {
        var jobList: [String] = []
        var nextToken: String?

        repeat {
            do {
                let output = try await glueClient.listJobs(
                    input: ListJobsInput(
                        maxResults: maxJobs,
                        nextToken: nextToken
                    )
                )

                guard let jobs = output.jobNames else {
                    return jobList
                }

                jobList = jobList + jobs
                nextToken = output.nextToken
            } catch {
                return jobList
            }
        } while (nextToken != nil)

        return jobList
    }

    /// Delete an AWS Glue job.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - jobName: The name of the job to delete.
    ///
    /// - Returns: `true` if the job is successfully deleted, otherwise `false`.
    func deleteJob(glueClient: GlueClient, name jobName: String) async -> Bool {
        do {
```

```
            _ = try await glueClient.deleteJob(
                input: DeleteJobInput(jobName: jobName)
            )
        } catch {
            return false
        }
        return true
    }

    /// Create an AWS Glue database.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - databaseName: The name to give the new database.
    ///   - location: The URL of the source data to use with AWS Glue.
    ///
    /// - Returns: `true` if the database is created successfully, otherwise
`false`.
    func createDatabase(glueClient: GlueClient, name databaseName: String,
location: String) async -> Bool {
        let databaseInput = GlueClientTypes.DatabaseInput(
            description: "Created by the AWS SDK for Swift Glue basic scenario
example.",
            locationUri: location,
            name: databaseName
        )

        do {
            _ = try await glueClient.createDatabase(
                input: CreateDatabaseInput(
                    databaseInput: databaseInput
                )
            )
        } catch {
            return false
        }

        return true
    }

    /// Get the AWS Glue database with the specified name.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
```

```
    ///   - name: The name of the database to return.
    ///
    /// - Returns: The `GlueClientTypes.Database` object describing the
    ///   specified database, or `nil` if an error occurs or the database
    ///   isn't found.
    func getDatabase(glueClient: GlueClient, name: String) async ->
GlueClientTypes.Database? {
        do {
            let output = try await glueClient.getDatabase(
                input: GetDatabaseInput(name: name)
            )

            return output.database
        } catch {
            return nil
        }
    }

    /// Returns a list of the tables in the specified database.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - databaseName: The name of the database whose tables are to be
    ///     returned.
    ///
    /// - Returns: An array of `GlueClientTypes.Table` objects, each
    ///   describing one table in the named database. An empty array indicates
    ///   that there are either no tables in the database, or an error
    ///   occurred before any tables could be found.
    func getTablesInDatabase(glueClient: GlueClient, databaseName: String) async
-> [GlueClientTypes.Table] {
        var tables: [GlueClientTypes.Table] = []
        var nextToken: String?

        repeat {
            do {
                let output = try await glueClient.getTables(
                    input: GetTablesInput(
                        databaseName: databaseName,
                        nextToken: nextToken
                    )
                )

                guard let tableList = output.tableList else {
```

```
                    return tables
                }

                tables = tables + tableList
                nextToken = output.nextToken
            } catch {
                return tables
            }
        } while nextToken != nil

        return tables
    }

    /// Delete the specified database.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - databaseName: The name of the database to delete.
    ///   - deleteTables: A Bool indicating whether or not to delete the
    ///     tables in the database before attempting to delete the database.
    ///
    /// - Returns: `true` if the database (and optionally its tables) are
    ///   deleted, otherwise `false`.
    func deleteDatabase(glueClient: GlueClient, name databaseName: String,
                        withTables deleteTables: Bool = false) async -> Bool {
        if deleteTables {
            var tableNames: [String] = []

            // Get a list of the names of all of the tables in the database.

            let tableList = await self.getTablesInDatabase(glueClient:
glueClient, databaseName: databaseName)
            for table in tableList {
                guard let name = table.name else {
                    continue
                }
                tableNames.append(name)
            }

            // Delete the tables. If there's only one table, use
            // `deleteTable()`, otherwise, use `batchDeleteTable()`. You can
            // use `batchDeleteTable()` for a single table, but this
            // demonstrates the use of `deleteTable()`.
```

```
            if tableNames.count == 1 {
                do {
                    print("    Deleting table...")
                    _ = try await glueClient.deleteTable(
                        input: DeleteTableInput(
                            databaseName: databaseName,
                            name: tableNames[0]
                        )
                    )
                } catch {
                    print("*** Unable to delete the table.")
                }
            } else {
                do {
                    print("    Deleting tables...")
                    _ = try await glueClient.batchDeleteTable(
                        input: BatchDeleteTableInput(
                            databaseName: databaseName,
                            tablesToDelete: tableNames
                        )
                    )
                } catch {
                    print("*** Unable to delete the tables.")
                }
            }
        }

        // Delete the database itself.

        do {
            print("    Deleting the database itself...")
            _ = try await glueClient.deleteDatabase(
                input: DeleteDatabaseInput(name: databaseName)
            )
        } catch {
            print("*** Unable to delete the database.")
            return false
        }
        return true
    }

    /// Start an AWS Glue job run.
    ///
    /// - Parameters:
```

```
    ///   - glueClient: The AWS Glue client to use.
    ///   - jobName: The name of the job to run.
    ///   - databaseName: The name of the AWS Glue database to run the job
against.
    ///   - tableName: The name of the table in the database to run the job
against.
    ///   - outputURL: The AWS S3 URI of the bucket location into which to
    ///     write the resulting output.
    ///
    /// - Returns: `true` if the job run is started successfully, otherwise
`false`.
    func startJobRun(glueClient: GlueClient, name jobName: String, databaseName:
String,
                     tableName: String, outputURL: String) async -> String? {
        do {
            let output = try await glueClient.startJobRun(
                input: StartJobRunInput(
                    arguments: [
                        "--input_database": databaseName,
                        "--input_table": tableName,
                        "--output_bucket_url": outputURL
                    ],
                    jobName: jobName,
                    numberOfWorkers: 10,
                    workerType: .g1x
                )
            )

            guard let id = output.jobRunId else {
                return nil
            }

            return id
        } catch {
            return nil
        }
    }

    /// Return a list of the job runs for the specified job.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - jobName: The name of the job for which to return its job runs.
    ///   - maxResults: The maximum number of job runs to return (default:
```

```
    ///     1000).
    ///
    /// - Returns: An array of `GlueClientTypes.JobRun` objects describing
    ///   each job run.
    func getJobRuns(glueClient: GlueClient, name jobName: String, maxResults:
Int? = nil) async -> [GlueClientTypes.JobRun] {
        do {
            let output = try await glueClient.getJobRuns(
                input: GetJobRunsInput(
                    jobName: jobName,
                    maxResults: maxResults
                )
            )

            guard let jobRuns = output.jobRuns else {
                print("*** No job runs found.")
                return []
            }

            return jobRuns
        } catch is EntityNotFoundException {
            print("*** The specified job name, \(jobName), doesn't exist.")
            return []
        } catch {
            print("*** Unexpected error getting job runs:")
            dump(error)
            return []
        }
    }

    /// Get information about a specific AWS Glue job run.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - jobName: The name of the job to return job run data for.
    ///   - id: The run ID of the specific job run to return.
    ///
    /// - Returns: A `GlueClientTypes.JobRun` object describing the state of
    ///   the job run, or `nil` if an error occurs.
    func getJobRun(glueClient: GlueClient, name jobName: String, id: String)
async -> GlueClientTypes.JobRun? {
        do {
            let output = try await glueClient.getJobRun(
                input: GetJobRunInput(
```

```
                    jobName: jobName,
                    runId: id
                )
            )

            return output.jobRun
        } catch {
            return nil
        }
    }

    /// Called by ``main()`` to run the bulk of the example.
    func runAsync() async throws {
        // A name to give the Python script upon upload to the Amazon S3
        // bucket.
        let scriptName = "jobscript.py"

        // Schedule string in `cron` format, as described here:
        // https://docs.aws.amazon.com/glue/latest/dg/monitor-data-warehouse-
schedule.html
        let cron = "cron(15 12 * * ? *)"

        let glueConfig = try await GlueClient.GlueClientConfiguration(region:
 awsRegion)
        let glueClient = GlueClient(config: glueConfig)

        let s3Config = try await S3Client.S3ClientConfiguration(region:
 awsRegion)
        let s3Client = S3Client(config: s3Config)

        // Create random names for things that need them.

        let crawlerName = tempName(prefix: "swift-glue-basics-crawler")
        let databaseName = tempName(prefix: "swift-glue-basics-db")

        // Create a name for the AWS Glue job.

        let jobName = tempName(prefix: "scenario-job")

        // The URL of the Python script on S3.

        let scriptURL = "s3://\(bucket)/\(scriptName)"

        print("Welcome to the AWS SDK for Swift basic scenario for AWS Glue!")
```

```
        //====================================================================
        // 0. Upload the Python script to the target bucket so it's available
        //    for use by the Amazon Glue service.
        //====================================================================

        print("Uploading the Python script: \(script) as key \(scriptName)")
        print("Destination bucket: \(bucket)")
        if !(await uploadFile(s3Client: s3Client, path: script, toBucket: bucket,
key: scriptName)) {
            return
        }

        //====================================================================
        // 1. Create the database and crawler using the randomized names
        //    generated previously.
        //====================================================================

        print("Creating database \"\(databaseName)\"...")
        if !(await createDatabase(glueClient: glueClient, name: databaseName,
location: s3url)) {
            print("*** Unable to create the database.")
            return
        }

        print("Creating crawler \"\(crawlerName)\"...")
        if !(await createCrawler(glueClient: glueClient, crawlerName:
crawlerName,
                                 iamRole: role, s3Path: s3url, cronSchedule:
cron,
                                 databaseName: databaseName)) {
            return
        }

        //====================================================================
        // 2. Start the crawler, then wait for it to be ready.
        //====================================================================

        print("Starting the crawler and waiting until it's ready...")
        if !(await startCrawler(glueClient: glueClient, name: crawlerName)) {
            _ = await deleteCrawler(glueClient: glueClient, name: crawlerName)
            return
        }
```

```
        if !(await waitUntilCrawlerReady(glueClient: glueClient, name:
crawlerName)) {
            _ = await deleteCrawler(glueClient: glueClient, name: crawlerName)
        }

        //========================================================================
        // 3. Get the database and table created by the crawler.
        //========================================================================

        print("Getting the crawler's database...")
        let database = await getDatabase(glueClient: glueClient, name:
databaseName)

        guard let database else {
            print("*** Unable to get the database.")
            return
        }
        print("Database URI: \(database.locationUri ?? "<unknown>")")

        let tableList = await getTablesInDatabase(glueClient: glueClient,
databaseName: databaseName)

        print("Found \(tableList.count) table(s):")
        for table in tableList {
            print("  \(table.name ?? "<unnamed>")")
        }

        if tableList.count != 1 {
            print("*** Incorrect number of tables found. There should only be
one.")
            _ = await deleteDatabase(glueClient: glueClient, name: databaseName,
withTables: true)
            _ = await deleteCrawler(glueClient: glueClient, name: crawlerName)
            return
        }

        guard let tableName = tableList[0].name else {
            print("*** Table is unnamed.")
            _ = await deleteDatabase(glueClient: glueClient, name: databaseName,
withTables: true)
            _ = await deleteCrawler(glueClient: glueClient, name: crawlerName)
            return
        }
```

```
        //===================================================================
        // 4. Create a job.
        //===================================================================

        print("Creating a job...")
        if !(await createJob(glueClient: glueClient, name: jobName, role: role,
                             scriptLocation: scriptURL)) {
            _ = await deleteDatabase(glueClient: glueClient, name: databaseName,
withTables: true)
            _ = await deleteCrawler(glueClient: glueClient, name: crawlerName)
            return
        }

        //===================================================================
        // 5. Start a job run.
        //===================================================================

        print("Starting the job...")

        // Construct the Amazon S3 URL for the job run's output. This is in
        // the bucket specified on the command line, with a folder name that's
        // unique for this job run.

        let timeStamp = Date().timeIntervalSince1970
        let jobPath = "\(jobName)-\(Int(timeStamp))"
        let outputURL = "s3://\(bucket)/\(jobPath)"

        // Start the job run.

        let jobRunID = await startJobRun(glueClient: glueClient, name: jobName,
                                         databaseName: databaseName,
                                         tableName: tableName,
                                         outputURL: outputURL)

        guard let jobRunID else {
            print("*** Job run ID is invalid.")
            _ = await deleteJob(glueClient: glueClient, name: jobName)
            _ = await deleteDatabase(glueClient: glueClient, name: databaseName,
withTables: true)
            _ = await deleteCrawler(glueClient: glueClient, name: crawlerName)
            return
        }

        //===================================================================
```

```
        // 6. Wait for the job run to indicate that the run is complete.
        //=====================================================================

        print("Waiting for job run to end...")

        var jobRunFinished = false
        var jobRunState: GlueClientTypes.JobRunState

        repeat {
            let jobRun = await getJobRun(glueClient: glueClient, name: jobName,
 id: jobRunID)
            guard let jobRun else {
                print("*** Unable to get the job run.")
                _ = await deleteJob(glueClient: glueClient, name: jobName)
                _ = await deleteDatabase(glueClient: glueClient, name:
 databaseName, withTables: true)
                _ = await deleteCrawler(glueClient: glueClient, name:
 crawlerName)
                return
            }
            jobRunState = jobRun.jobRunState ?? .failed

            //
=====================================================================
            // 7. Output where to find the data if the job run was successful.
            //    If the job run failed for any reason, output an appropriate
            //    error message.
            //
=====================================================================

            switch jobRunState {
                case .succeeded:
                    print("Job run succeeded. JSON files are in the Amazon S3
 path:")
                    print("    \(outputURL)")
                    jobRunFinished = true
                case .stopped:
                    jobRunFinished = true
                case .error:
                    print("*** Error: Job run ended in an error.
 \(jobRun.errorMessage ?? "")")
                    jobRunFinished = true
                case .failed:
```

```
                    print("*** Error: Job run failed. \(jobRun.errorMessage ??
""")")
                    jobRunFinished = true
                case .timeout:
                    print("*** Warning: Job run timed out.")
                    jobRunFinished = true
                default:
                    do {
                        try await Task.sleep(for: .milliseconds(250))
                    } catch {
                        print("*** Error pausing the task.")
                    }
            }
        } while jobRunFinished != true

        //=========================================================================
        // 7.5. List the job runs for this job, showing each job run's ID and
        // its execution time.
        //=========================================================================

        print("Getting all job runs for the job \(jobName):")
        let jobRuns = await getJobRuns(glueClient: glueClient, name: jobName)

        if jobRuns.count == 0 {
            print("    <no job runs found>")
        } else {
            print("Found \(jobRuns.count) job runs... listing execution times:")
            for jobRun in jobRuns {
                print("    \(jobRun.id ?? "<unnamed>"): \(jobRun.executionTime)
seconds")
            }
        }

        //=========================================================================
        // 8. List the jobs for the user's account.
        //=========================================================================

        print("\nThe account has the following jobs:")
        let jobs = await listJobs(glueClient: glueClient)

        if jobs.count == 0 {
            print("    <no jobs found>")
        } else {
            for job in jobs {
```

```
                print("    \(job)")
            }
        }

        //===========================================================================
        // 9. Get the job run details for a job run.
        //===========================================================================

        print("Information about the job run:")
        let jobRun = await getJobRun(glueClient: glueClient, name: jobName, id:
jobRunID)

        guard let jobRun else {
            print("*** Unable to retrieve the job run.")
            _ = await deleteJob(glueClient: glueClient, name: jobName)
            _ = await deleteDatabase(glueClient: glueClient, name: databaseName,
withTables: true)
            _ = await deleteCrawler(glueClient: glueClient, name: crawlerName)
            return
        }

        let startDate = jobRun.startedOn ?? Date(timeIntervalSince1970: 0)
        let endDate = jobRun.completedOn ?? Date(timeIntervalSince1970: 0)
        let dateFormatter: DateFormatter = DateFormatter()
        dateFormatter.dateStyle = .long
        dateFormatter.timeStyle = .long

        print("    Started at: \(dateFormatter.string(from: startDate))")
        print("  Completed at: \(dateFormatter.string(from: endDate))")

        //===========================================================================
        // 10. Delete the job.
        //===========================================================================

        print("\nDeleting the job...")
        _ = await deleteJob(glueClient: glueClient, name: jobName)

        //===========================================================================
        // 11. Delete the database and tables created by this example.
        //===========================================================================

        print("Deleting the database...")
        _ = await deleteDatabase(glueClient: glueClient, name: databaseName,
withTables: true)
```

```
        //====================================================================
        // 12. Delete the crawler.
        //====================================================================

        print("Deleting the crawler...")
        if !(await deleteCrawler(glueClient: glueClient, name: crawlerName)) {
            return
        }
    }
}

/// The program's asynchronous entry point.
@main
struct Main {
    static func main() async {
        let args = Array(CommandLine.arguments.dropFirst())

        do {
            let command = try ExampleCommand.parse(args)
            try await command.runAsync()
        } catch {
            ExampleCommand.exit(withError: error)
        }
    }
}
```

- Pour plus de détails sur l'API, consultez les rubriques suivantes dans la Référence des API du kit AWS SDK pour Swift.
  - CreateCrawler
  - CreateJob
  - DeleteCrawler
  - DeleteDatabase
  - DeleteJob
  - DeleteTable
  - GetCrawler
  - GetDatabase
  - GetDatabases

- GetJob
- GetJobRun
- GetJobRuns
- GetTables
- ListJobs
- StartCrawler
- StartJobRun

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Actions d' AWS Glue utilisation AWS SDKs

Les exemples de code suivants montrent comment effectuer des AWS Glue actions individuelles avec AWS SDKs. Chaque exemple inclut un lien vers GitHub, où vous pouvez trouver des instructions pour configurer et exécuter le code.

Les exemples suivants incluent uniquement les actions les plus couramment utilisées. Pour obtenir la liste complète, consultez la Référence des API AWS Glue.

Exemples

- Utilisation CreateCrawler avec un AWS SDK
- Utilisation CreateJob avec un AWS SDK ou une CLI
- Utilisation DeleteCrawler avec un AWS SDK
- Utilisation DeleteDatabase avec un AWS SDK
- Utilisation DeleteJob avec un AWS SDK ou une CLI
- Utilisation DeleteTable avec un AWS SDK
- Utilisation GetCrawler avec un AWS SDK
- Utilisation GetDatabase avec un AWS SDK
- Utilisation GetDatabases avec un AWS SDK ou une CLI
- Utilisation GetJob avec un AWS SDK ou une CLI
- Utilisation GetJobRun avec un AWS SDK ou une CLI
- Utilisation GetJobRuns avec un AWS SDK ou une CLI

- Utilisation GetTables avec un AWS SDK ou une CLI
- Utilisation ListJobs avec un AWS SDK
- Utilisation StartCrawler avec un AWS SDK ou une CLI
- Utilisation StartJobRun avec un AWS SDK ou une CLI

Utilisation **`CreateCrawler`** avec un AWS SDK

Les exemples de code suivants illustrent comment utiliser `CreateCrawler`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Principes de base

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Create an AWS Glue crawler.
    /// </summary>
    /// <param name="crawlerName">The name for the crawler.</param>
    /// <param name="crawlerDescription">A description of the crawler.</param>
    /// <param name="role">The AWS Identity and Access Management (IAM) role to
    /// be assumed by the crawler.</param>
    /// <param name="schedule">The schedule on which the crawler will be
executed.</param>
    /// <param name="s3Path">The path to the Amazon Simple Storage Service
(Amazon S3)
    /// bucket where the Python script has been stored.</param>
    /// <param name="dbName">The name to use for the database that will be
    /// created by the crawler.</param>
```

```
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> CreateCrawlerAsync(
        string crawlerName,
        string crawlerDescription,
        string role,
        string schedule,
        string s3Path,
        string dbName)
    {
        var s3Target = new S3Target
        {
            Path = s3Path,
        };

        var targetList = new List<S3Target>
        {
            s3Target,
        };

        var targets = new CrawlerTargets
        {
            S3Targets = targetList,
        };

        var crawlerRequest = new CreateCrawlerRequest
        {
            DatabaseName = dbName,
            Name = crawlerName,
            Description = crawlerDescription,
            Targets = targets,
            Role = role,
            Schedule = schedule,
        };

        var response = await _amazonGlue.CreateCrawlerAsync(crawlerRequest);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Pour plus de détails sur l'API, reportez-vous CreateCrawlerà la section Référence des AWS SDK pour .NET API.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region in which the bucket was created
(overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::Glue::GlueClient client(clientConfig);

        Aws::Glue::Model::S3Target s3Target;
        s3Target.SetPath("s3://crawler-public-us-east-1/flight/2016/csv");
        Aws::Glue::Model::CrawlerTargets crawlerTargets;
        crawlerTargets.AddS3Targets(s3Target);

        Aws::Glue::Model::CreateCrawlerRequest request;
        request.SetTargets(crawlerTargets);
        request.SetName(CRAWLER_NAME);
        request.SetDatabaseName(CRAWLER_DATABASE_NAME);
        request.SetTablePrefix(CRAWLER_DATABASE_PREFIX);
        request.SetRole(roleArn);

        Aws::Glue::Model::CreateCrawlerOutcome outcome =
client.CreateCrawler(request);

        if (outcome.IsSuccess()) {
            std::cout << "Successfully created the crawler." << std::endl;
        }
        else {
            std::cerr << "Error creating a crawler. " <<
outcome.GetError().GetMessage()
                      << std::endl;
            deleteAssets("", CRAWLER_DATABASE_NAME, "", bucketName,
clientConfig);
            return false;
```

```
        }
```

- Pour plus de détails sur l'API, reportez-vous CreateCrawlerà la section Référence des AWS SDK pour C++ API.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
   /**
    * Creates a new AWS Glue crawler using the AWS Glue Java API.
    *
    * @param glueClient  the AWS Glue client used to interact with the AWS Glue
service
    * @param iam         the IAM role that the crawler will use to access the
data source
    * @param s3Path      the S3 path that the crawler will scan for data
    * @param cron        the cron expression that defines the crawler's schedule
    * @param dbName      the name of the AWS Glue database where the crawler
will store the metadata
    * @param crawlerName the name of the crawler to be created
    */
   public static void createGlueCrawler(GlueClient glueClient,
                                         String iam,
                                         String s3Path,
                                         String cron,
                                         String dbName,
                                         String crawlerName) {

       try {
           S3Target s3Target = S3Target.builder()
               .path(s3Path)
               .build();
```

```
            List<S3Target> targetList = new ArrayList<>();
            targetList.add(s3Target);
            CrawlerTargets targets = CrawlerTargets.builder()
                .s3Targets(targetList)
                .build();

            CreateCrawlerRequest crawlerRequest = CreateCrawlerRequest.builder()
                .databaseName(dbName)
                .name(crawlerName)
                .description("Created by the AWS Glue Java API")
                .targets(targets)
                .role(iam)
                .schedule(cron)
                .build();

            glueClient.createCrawler(crawlerRequest);
            System.out.println(crawlerName + " was successfully created");

        } catch (GlueException e) {
            throw e;
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous CreateCrawlerà la section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
const createCrawler = (name, role, dbName, tablePrefix, s3TargetPath) => {
  const client = new GlueClient({});
```

```
  const command = new CreateCrawlerCommand({
    Name: name,
    Role: role,
    DatabaseName: dbName,
    TablePrefix: tablePrefix,
    Targets: {
      S3Targets: [{ Path: s3TargetPath }],
    },
  });

  return client.send(command);
};
```

- Pour plus de détails sur l'API, reportez-vous CreateCrawlerà la section Référence des AWS SDK pour JavaScript API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun createGlueCrawler(
    iam: String?,
    s3Path: String?,
    cron: String?,
    dbName: String?,
    crawlerName: String,
) {
    val s3Target =
        S3Target {
            path = s3Path
        }

    // Add the S3Target to a list.
    val targetList = mutableListOf<S3Target>()
```

```
    targetList.add(s3Target)

    val targetOb =
        CrawlerTargets {
            s3Targets = targetList
        }

    val request =
        CreateCrawlerRequest {
            databaseName = dbName
            name = crawlerName
            description = "Created by the AWS Glue Kotlin API"
            targets = targetOb
            role = iam
            schedule = cron
        }

    GlueClient.fromEnvironment { region = "us-west-2" }.use { glueClient ->
        glueClient.createCrawler(request)
        println("$crawlerName was successfully created")
    }
}
```

- Pour plus de détails sur l'API, consultez CreateCrawlerla section AWS SDK pour la référence de l'API Kotlin.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
        $crawlerName = "example-crawler-test-" . $uniqid;

        $role = $iamService->getRole("AWSGlueServiceRole-DocExample");
```

```
        $path = 's3://crawler-public-us-east-1/flight/2016/csv';
        $glueService->createCrawler($crawlerName, $role['Role']['Arn'],
$databaseName, $path);

    public function createCrawler($crawlerName, $role, $databaseName, $path):
Result
    {
        return $this->customWaiter(function () use ($crawlerName, $role,
$databaseName, $path) {
            return $this->glueClient->createCrawler([
                'Name' => $crawlerName,
                'Role' => $role,
                'DatabaseName' => $databaseName,
                'Targets' => [
                    'S3Targets' =>
                        [[
                            'Path' => $path,
                        ]]
                ],
            ]);
        });
    }
```

- Pour plus de détails sur l'API, reportez-vous CreateCrawlerà la section Référence des AWS SDK pour PHP API.

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
```

```
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client


    def create_crawler(self, name, role_arn, db_name, db_prefix, s3_target):
        """
        Creates a crawler that can crawl the specified target and populate a
        database in your AWS Glue Data Catalog with metadata that describes the
data
        in the target.

        :param name: The name of the crawler.
        :param role_arn: The Amazon Resource Name (ARN) of an AWS Identity and
Access
                         Management (IAM) role that grants permission to let AWS
Glue
                         access the resources it needs.
        :param db_name: The name to give the database that is created by the
crawler.
        :param db_prefix: The prefix to give any database tables that are created
by
                          the crawler.
        :param s3_target: The URL to an S3 bucket that contains data that is
                          the target of the crawler.
        """
        try:
            self.glue_client.create_crawler(
                Name=name,
                Role=role_arn,
                DatabaseName=db_name,
                TablePrefix=db_prefix,
                Targets={"S3Targets": [{"Path": s3_target}]},
            )
        except ClientError as err:
            logger.error(
                "Couldn't create crawler. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- Pour plus de détails sur l'API, consultez CreateCrawlerle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Creates a new crawler with the specified configuration.
  #
  # @param name [String] The name of the crawler.
  # @param role_arn [String] The ARN of the IAM role to be used by the crawler.
  # @param db_name [String] The name of the database where the crawler stores its
 metadata.
  # @param db_prefix [String] The prefix to be added to the names of tables that
 the crawler creates.
  # @param s3_target [String] The S3 path that the crawler will crawl.
  # @return [void]
  def create_crawler(name, role_arn, db_name, _db_prefix, s3_target)
    @glue_client.create_crawler(
      name: name,
```

```
      role: role_arn,
      database_name: db_name,
      targets: {
        s3_targets: [
          {
            path: s3_target
          }
        ]
      }
    )
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not create crawler: \n#{e.message}")
    raise
  end
```

- Pour plus de détails sur l'API, reportez-vous CreateCrawlerà la section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
        let create_crawler = glue
            .create_crawler()
            .name(self.crawler())
            .database_name(self.database())
            .role(self.iam_role.expose_secret())
            .targets(
                CrawlerTargets::builder()
                    .s3_targets(S3Target::builder().path(CRAWLER_TARGET).build())
                    .build(),
            )
            .send()
            .await;
```

```
        match create_crawler {
            Err(err) => {
                let glue_err: aws_sdk_glue::Error = err.into();
                match glue_err {
                    aws_sdk_glue::Error::AlreadyExistsException(_) => {
                        info!("Using existing crawler");
                        Ok(())
                    }
                    _ => Err(GlueMvpError::GlueSdk(glue_err)),
                }
            }
            Ok(_) => Ok(()),
        }?;
```

- Pour plus de détails sur l'API, voir CreateCrawlerla section de référence de l'API AWS SDK for Rust.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    TRY.
        " iv_crawler_name = 'my-crawler'
        " iv_role_arn = 'arn:aws:iam::123456789012:role/AWSGlueServiceRole-Test'
        " iv_database_name = 'my-database'
        " iv_table_prefix = 'test_'
        " iv_s3_target = 's3://example-bucket/data/'

        DATA(lt_s3_targets) = VALUE /aws1/cl_glus3target=>tt_s3targetlist(
          ( NEW /aws1/cl_glus3target( iv_path = iv_s3_target ) ) ).

        DATA(lo_targets) = NEW /aws1/cl_glucrawlertargets(
          it_s3targets = lt_s3_targets ).
```

```
        lo_glu->createcrawler(
          iv_name = iv_crawler_name
          iv_role = iv_role_arn
          iv_databasename = iv_database_name
          iv_tableprefix = iv_table_prefix
          io_targets = lo_targets ).
        MESSAGE 'Crawler created successfully.' TYPE 'I'.
      CATCH /aws1/cx_glualreadyexistsex.
        MESSAGE 'Crawler already exists.' TYPE 'E'.
      CATCH /aws1/cx_gluinvalidinputex INTO DATA(lo_invalid_ex).
        DATA(lv_invalid_error) = lo_invalid_ex->if_message~get_longtext( ).
        MESSAGE lv_invalid_error TYPE 'E'.
      CATCH /aws1/cx_gluoperationtimeoutex INTO DATA(lo_timeout_ex).
        DATA(lv_timeout_error) = lo_timeout_ex->if_message~get_longtext( ).
        MESSAGE lv_timeout_error TYPE 'E'.
      CATCH /aws1/cx_gluresrcnumlmtexcdex INTO DATA(lo_limit_ex).
        DATA(lv_limit_error) = lo_limit_ex->if_message~get_longtext( ).
        MESSAGE lv_limit_error TYPE 'E'.
    ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous CreateCrawlerà la section de référence du AWS SDK pour l'API SAP ABAP.

Swift

Kit SDK pour Swift

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import AWSClientRuntime
import AWSGlue

    /// Create a new AWS Glue crawler.
    ///
    /// - Parameters:
```

```
    ///   - glueClient: An AWS Glue client to use for the crawler.
    ///   - crawlerName: A name for the new crawler.
    ///   - iamRole: The name of an Amazon IAM role for the crawler to use.
    ///   - s3Path: The path of an Amazon S3 folder to use as a target location.
    ///   - cronSchedule: A `cron` schedule indicating when to run the crawler.
    ///   - databaseName: The name of an AWS Glue database to operate on.
    ///
    /// - Returns: `true` if the crawler is created successfully, otherwise
`false`.
    func createCrawler(glueClient: GlueClient, crawlerName: String, iamRole:
String,
                       s3Path: String, cronSchedule: String, databaseName:
String) async -> Bool {
        let s3Target = GlueClientTypes.S3Target(path: s3url)
        let targetList = GlueClientTypes.CrawlerTargets(s3Targets: [s3Target])

        do {
            _ = try await glueClient.createCrawler(
                input: CreateCrawlerInput(
                    databaseName: databaseName,
                    description: "Created by the AWS SDK for Swift Scenario
Example for AWS Glue.",
                    name: crawlerName,
                    role: iamRole,
                    schedule: cronSchedule,
                    tablePrefix: tablePrefix,
                    targets: targetList
                )
            )
        } catch _ as AlreadyExistsException {
            print("*** A crawler named \"\(crawlerName)\" already exists.")
            return false
        } catch _ as OperationTimeoutException {
            print("*** The attempt to create the AWS Glue crawler timed out.")
            return false
        } catch {
            print("*** An unexpected error occurred creating the AWS Glue
crawler: \(error.localizedDescription)")
            return false
        }

        return true
    }
```

- Pour plus de détails sur l'API, reportez-vous CreateCrawlerà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation **`CreateJob`** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `CreateJob`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Principes de base

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
/// <summary>
/// Create an AWS Glue job.
/// </summary>
/// <param name="jobName">The name of the job.</param>
/// <param name="roleName">The name of the IAM role to be assumed by
/// the job.</param>
/// <param name="description">A description of the job.</param>
/// <param name="scriptUrl">The URL to the script.</param>
/// <returns>A Boolean value indicating the success of the action.</returns>
```

```
   public async Task<bool> CreateJobAsync(string dbName, string tableName,
string bucketUrl, string jobName, string roleName, string description, string
scriptUrl)
   {
       var command = new JobCommand
       {
           PythonVersion = "3",
           Name = "glueetl",
           ScriptLocation = scriptUrl,
       };

       var arguments = new Dictionary<string, string>
       {
           { "--input_database", dbName },
           { "--input_table", tableName },
           { "--output_bucket_url", bucketUrl }
       };

       var request = new CreateJobRequest
       {
           Command = command,
           DefaultArguments = arguments,
           Description = description,
           GlueVersion = "3.0",
           Name = jobName,
           NumberOfWorkers = 10,
           Role = roleName,
           WorkerType = "G.1X"
       };

       var response = await _amazonGlue.CreateJobAsync(request);
       return response.HttpStatusCode == HttpStatusCode.OK;
   }
```

- Pour plus de détails sur l'API, reportez-vous CreateJobà la section Référence des AWS SDK pour .NET API.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region in which the bucket was created
(overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::Glue::GlueClient client(clientConfig);

        Aws::Glue::Model::CreateJobRequest request;
        request.SetName(JOB_NAME);
        request.SetRole(roleArn);
        request.SetGlueVersion(GLUE_VERSION);

        Aws::Glue::Model::JobCommand command;
        command.SetName(JOB_COMMAND_NAME);
        command.SetPythonVersion(JOB_PYTHON_VERSION);
        command.SetScriptLocation(
                Aws::String("s3://") + bucketName + "/" + PYTHON_SCRIPT);
        request.SetCommand(command);

        Aws::Glue::Model::CreateJobOutcome outcome = client.CreateJob(request);

        if (outcome.IsSuccess()) {
            std::cout << "Successfully created the job." << std::endl;
        }
        else {
            std::cerr << "Error creating the job. " <<
outcome.GetError().GetMessage()
                      << std::endl;
            deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, "", bucketName,
                         clientConfig);
            return false;
        }
```

- Pour plus de détails sur l'API, reportez-vous CreateJobà la section Référence des AWS SDK pour C++ API.

CLI

AWS CLI

Pour créer une tâche afin de transformer des données

L'exemple `create-job` suivant crée une tâche de streaming qui exécute un script stocké dans S3.

```
aws glue create-job \
    --name my-testing-job \
    --role AWSGlueServiceRoleDefault \
    --command '{ \
        "Name": "gluestreaming", \
        "ScriptLocation": "s3://amzn-s3-demo-bucket/folder/" \
    }' \
    --region us-east-1 \
    --output json \
    --default-arguments '{ \
        "--job-language":"scala", \
        "--class":"GlueApp" \
    }' \
    --profile my-profile \
    --endpoint https://glue.us-east-1.amazonaws.com
```

Contenu de `test_script.scala` :

```
import com.amazonaws.services.glue.ChoiceOption
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.MappingSpec
import com.amazonaws.services.glue.ResolveSpec
import com.amazonaws.services.glue.errors.CallSite
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._
```

```
object GlueApp {
    def main(sysArgs: Array[String]) {
        val spark: SparkContext = new SparkContext()
        val glueContext: GlueContext = new GlueContext(spark)
        // @params: [JOB_NAME]
        val args = GlueArgParser.getResolvedOptions(sysArgs,
 Seq("JOB_NAME").toArray)
        Job.init(args("JOB_NAME"), glueContext, args.asJava)
        // @type: DataSource
        // @args: [database = "tempdb", table_name = "s3-source",
 transformation_ctx = "datasource0"]
        // @return: datasource0
        // @inputs: []
        val datasource0 = glueContext.getCatalogSource(database = "tempdb",
 tableName = "s3-source", redshiftTmpDir = "", transformationContext =
 "datasource0").getDynamicFrame()
        // @type: ApplyMapping
        // @args: [mapping = [("sensorid", "int", "sensorid", "int"),
 ("currenttemperature", "int", "currenttemperature", "int"), ("status", "string",
 "status", "string")], transformation_ctx = "applymapping1"]
        // @return: applymapping1
        // @inputs: [frame = datasource0]
        val applymapping1 = datasource0.applyMapping(mappings = Seq(("sensorid",
 "int", "sensorid", "int"), ("currenttemperature", "int", "currenttemperature",
 "int"), ("status", "string", "status", "string")), caseSensitive = false,
 transformationContext = "applymapping1")
        // @type: SelectFields
        // @args: [paths = ["sensorid", "currenttemperature", "status"],
 transformation_ctx = "selectfields2"]
        // @return: selectfields2
        // @inputs: [frame = applymapping1]
        val selectfields2 = applymapping1.selectFields(paths = Seq("sensorid",
 "currenttemperature", "status"), transformationContext = "selectfields2")
        // @type: ResolveChoice
        // @args: [choice = "MATCH_CATALOG", database = "tempdb", table_name =
 "my-s3-sink", transformation_ctx = "resolvechoice3"]
        // @return: resolvechoice3
        // @inputs: [frame = selectfields2]
        val resolvechoice3 = selectfields2.resolveChoice(choiceOption =
 Some(ChoiceOption("MATCH_CATALOG")), database = Some("tempdb"), tableName =
 Some("my-s3-sink"), transformationContext = "resolvechoice3")
        // @type: DataSink
```

```
        // @args: [database = "tempdb", table_name = "my-s3-sink",
 transformation_ctx = "datasink4"]
        // @return: datasink4
        // @inputs: [frame = resolvechoice3]
        val datasink4 = glueContext.getCatalogSink(database = "tempdb",
 tableName = "my-s3-sink", redshiftTmpDir = "", transformationContext =
 "datasink4").writeDynamicFrame(resolvechoice3)
        Job.commit()
    }
}
```

Sortie :

```
{
    "Name": "my-testing-job"
}
```

Pour plus d'informations, consultez la section Création de jobs dans AWS Glue du AWS Glue Developer Guide.

- Pour plus de détails sur l'API, reportez-vous CreateJobà la section Référence des AWS CLI commandes.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /**
     * Creates a new AWS Glue job.
     *
     * @param glueClient     the AWS Glue client to use for the operation
     * @param jobName        the name of the job to create
     * @param iam            the IAM role to associate with the job
     * @param scriptLocation the location of the script to be used by the job
```

```
     * @throws GlueException if there is an error creating the job
     */
    public static void createJob(GlueClient glueClient, String jobName, String
iam, String scriptLocation) {
        try {
            JobCommand command = JobCommand.builder()
                .pythonVersion("3")
                .name("glueetl")
                .scriptLocation(scriptLocation)
                .build();

            CreateJobRequest jobRequest = CreateJobRequest.builder()
                .description("A Job created by using the AWS SDK for Java V2")
                .glueVersion("2.0")
                .workerType(WorkerType.G_1_X)
                .numberOfWorkers(10)
                .name(jobName)
                .role(iam)
                .command(command)
                .build();

            glueClient.createJob(jobRequest);
            System.out.println(jobName + " was successfully created.");

        } catch (GlueException e) {
            throw e;
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous CreateJobà la section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
const createJob = (name, role, scriptBucketName, scriptKey) => {
  const client = new GlueClient({});

  const command = new CreateJobCommand({
    Name: name,
    Role: role,
    Command: {
      Name: "glueetl",
      PythonVersion: "3",
      ScriptLocation: `s3://${scriptBucketName}/${scriptKey}`,
    },
    GlueVersion: "3.0",
  });

  return client.send(command);
};
```

- Pour plus de détails sur l'API, reportez-vous CreateJobà la section Référence des AWS SDK pour JavaScript API.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
        $role = $iamService->getRole("AWSGlueServiceRole-DocExample");

        $jobName = 'test-job-' . $uniqid;

        $scriptLocation = "s3://$bucketName/run_job.py";
        $job = $glueService->createJob($jobName, $role['Role']['Arn'],
 $scriptLocation);

    public function createJob($jobName, $role, $scriptLocation, $pythonVersion =
 '3', $glueVersion = '3.0'): Result
```

```
    {
        return $this->glueClient->createJob([
            'Name' => $jobName,
            'Role' => $role,
            'Command' => [
                'Name' => 'glueetl',
                'ScriptLocation' => $scriptLocation,
                'PythonVersion' => $pythonVersion,
            ],
            'GlueVersion' => $glueVersion,
        ]);
    }
```

- Pour plus de détails sur l'API, reportez-vous CreateJobà la section Référence des AWS SDK pour PHP API.

PowerShell

Outils pour PowerShell V4

Exemple 1 : Cet exemple crée une nouvelle tâche dans AWS Glue. La valeur du nom de commande est toujours**`glueetl`**. AWS Glue permet d'exécuter des scripts de travail écrits en Python ou Scala. Dans cet exemple, le script de tâche (MyTestGlueJob.py) est écrit en Python. Les paramètres Python sont spécifiés dans la **`$DefArgs`** variable, puis transmis à la PowerShell commande dans le **`DefaultArguments`** paramètre, qui accepte une table de hachage. Les paramètres de la **`$JobParams`** variable proviennent de l' CreateJob API, documentés dans la rubrique Jobs (https://docs.aws.amazon.com/glue/latest/dg/aws- glue-api-jobs-job .html) de la référence de l'API AWS Glue.

```
$Command = New-Object Amazon.Glue.Model.JobCommand
$Command.Name = 'glueetl'
$Command.ScriptLocation = 's3://amzn-s3-demo-source-bucket/admin/
MyTestGlueJob.py'
$Command

$Source = "source_test_table"
$Target = "target_test_table"
$Connections = $Source, $Target

$DefArgs = @{
```

```
     '--TempDir' = 's3://amzn-s3-demo-bucket/admin'
     '--job-bookmark-option' = 'job-bookmark-disable'
     '--job-language' = 'python'
     }
$DefArgs

$ExecutionProp = New-Object Amazon.Glue.Model.ExecutionProperty
$ExecutionProp.MaxConcurrentRuns = 1
$ExecutionProp

$JobParams = @{
    "AllocatedCapacity"    = "5"
    "Command"              = $Command
    "Connections_Connection" = $Connections
    "DefaultArguments"  = $DefArgs
    "Description"       = "This is a test"
    "ExecutionProperty" = $ExecutionProp
    "MaxRetries"        = "1"
    "Name"              = "MyOregonTestGlueJob"
    "Role"              = "Amazon-GlueServiceRoleForSSM"
    "Timeout"           = "20"
     }

New-GlueJob @JobParams
```

- Pour plus de détails sur l'API, reportez-vous CreateJobà la section Référence des Outils AWS pour PowerShell applets de commande (V4).

Outils pour PowerShell V5

Exemple 1 : Cet exemple crée une nouvelle tâche dans AWS Glue. La valeur du nom de commande est toujours**`glueetl`**. AWS Glue permet d'exécuter des scripts de travail écrits en Python ou Scala. Dans cet exemple, le script de tâche (MyTestGlueJob.py) est écrit en Python. Les paramètres Python sont spécifiés dans la **`$DefArgs`** variable, puis transmis à la PowerShell commande dans le **`DefaultArguments`** paramètre, qui accepte une table de hachage. Les paramètres de la **`$JobParams`** variable proviennent de l' CreateJob API, documentés dans la rubrique Jobs (https://docs.aws.amazon.com/glue/latest/dg/aws- glue-api-jobs-job .html) de la référence de l'API AWS Glue.

```
$Command = New-Object Amazon.Glue.Model.JobCommand
$Command.Name = 'glueetl'
$Command.ScriptLocation = 's3://amzn-s3-demo-source-bucket/admin/
MyTestGlueJob.py'
```

```
$Command

$Source = "source_test_table"
$Target = "target_test_table"
$Connections = $Source, $Target

$DefArgs = @{
     '--TempDir' = 's3://amzn-s3-demo-bucket/admin'
     '--job-bookmark-option' = 'job-bookmark-disable'
     '--job-language' = 'python'
     }
$DefArgs

$ExecutionProp = New-Object Amazon.Glue.Model.ExecutionProperty
$ExecutionProp.MaxConcurrentRuns = 1
$ExecutionProp

$JobParams = @{
    "AllocatedCapacity"    = "5"
    "Command"              = $Command
    "Connections_Connection" = $Connections
    "DefaultArguments"  = $DefArgs
    "Description"       = "This is a test"
    "ExecutionProperty" = $ExecutionProp
    "MaxRetries"        = "1"
    "Name"              = "MyOregonTestGlueJob"
    "Role"              = "Amazon-GlueServiceRoleForSSM"
    "Timeout"           = "20"
     }

New-GlueJob @JobParams
```

- Pour plus de détails sur l'API, reportez-vous CreateJobà la section Référence des Outils AWS pour PowerShell applets de commande (V5).

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client


    def create_job(self, name, description, role_arn, script_location):
        """
        Creates a job definition for an extract, transform, and load (ETL) job
 that can
        be run by AWS Glue.

        :param name: The name of the job definition.
        :param description: The description of the job definition.
        :param role_arn: The ARN of an IAM role that grants AWS Glue the
 permissions
                         it requires to run the job.
        :param script_location: The Amazon S3 URL of a Python ETL script that is
 run as
                                part of the job. The script defines how the data
 is
                                transformed.
        """
        try:
            self.glue_client.create_job(
                Name=name,
                Description=description,
                Role=role_arn,
```

```
                Command={
                    "Name": "glueetl",
                    "ScriptLocation": script_location,
                    "PythonVersion": "3",
                },
                GlueVersion="3.0",
            )
        except ClientError as err:
            logger.error(
                "Couldn't create job %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- Pour plus de détails sur l'API, consultez CreateJoble AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
```

```
    @logger = logger
  end

  # Creates a new job with the specified configuration.
  #
  # @param name [String] The name of the job.
  # @param description [String] The description of the job.
  # @param role_arn [String] The ARN of the IAM role to be used by the job.
  # @param script_location [String] The location of the ETL script for the job.
  # @return [void]
  def create_job(name, description, role_arn, script_location)
    @glue_client.create_job(
      name: name,
      description: description,
      role: role_arn,
      command: {
        name: 'glueetl',
        script_location: script_location,
        python_version: '3'
      },
      glue_version: '3.0'
    )
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not create job #{name}: \n#{e.message}")
    raise
  end
```

- Pour plus de détails sur l'API, reportez-vous CreateJobà la section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
        let create_job = glue
            .create_job()
            .name(self.job())
            .role(self.iam_role.expose_secret())
            .command(
                JobCommand::builder()
                    .name("glueetl")
                    .python_version("3")
                    .script_location(format!("s3://{}/job.py", self.bucket()))
                    .build(),
            )
            .glue_version("3.0")
            .send()
            .await
            .map_err(GlueMvpError::from_glue_sdk)?;

        let job_name = create_job.name().ok_or_else(|| {
            GlueMvpError::Unknown("Did not get job name after creating
job".into())
        })?;
```

- Pour plus de détails sur l'API, voir CreateJobla section de référence de l'API AWS SDK for Rust.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    TRY.
        " iv_job_name = 'my-etl-job'
        " iv_description = 'ETL job for data transformation'
        " iv_role_arn = 'arn:aws:iam::123456789012:role/AWSGlueServiceRole-Test'
        " iv_script_location = 's3://example-bucket/scripts/my-script.py'
```

```
        DATA(lo_command) = NEW /aws1/cl_glujobcommand(
          iv_name = 'glueetl'
          iv_scriptlocation = iv_script_location
          iv_pythonversion = '3' ).

        lo_glu->createjob(
          iv_name = iv_job_name
          iv_description = iv_description
          iv_role = iv_role_arn
          io_command = lo_command
          iv_glueversion = '3.0' ).
        MESSAGE 'Job created successfully.' TYPE 'I'.
      CATCH /aws1/cx_glualreadyexistsex.
        MESSAGE 'Job already exists.' TYPE 'E'.
      CATCH /aws1/cx_gluinvalidinputex INTO DATA(lo_invalid_ex).
        DATA(lv_invalid_error) = lo_invalid_ex->if_message~get_longtext( ).
        MESSAGE lv_invalid_error TYPE 'E'.
      CATCH /aws1/cx_gluinternalserviceex INTO DATA(lo_internal_ex).
        DATA(lv_internal_error) = lo_internal_ex->if_message~get_longtext( ).
        MESSAGE lv_internal_error TYPE 'E'.
      CATCH /aws1/cx_gluoperationtimeoutex INTO DATA(lo_timeout_ex).
        DATA(lv_timeout_error) = lo_timeout_ex->if_message~get_longtext( ).
        MESSAGE lv_timeout_error TYPE 'E'.
      CATCH /aws1/cx_gluresrcnumlmtexcdex INTO DATA(lo_limit_ex).
        DATA(lv_limit_error) = lo_limit_ex->if_message~get_longtext( ).
        MESSAGE lv_limit_error TYPE 'E'.
    ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous CreateJobà la section de référence du AWS SDK pour l'API SAP ABAP.

Swift

Kit SDK pour Swift

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import AWSClientRuntime
import AWSGlue

    /// Create a new AWS Glue job.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - jobName: The name to give the new job.
    ///   - role: The IAM role for the job to use when accessing AWS services.
    ///   - scriptLocation: The AWS S3 URI of the script to be run by the job.
    ///
    /// - Returns: `true` if the job is created successfully, otherwise `false`.
    func createJob(glueClient: GlueClient, name jobName: String, role: String,
                   scriptLocation: String) async -> Bool {
        let command = GlueClientTypes.JobCommand(
            name: "glueetl",
            pythonVersion: "3",
            scriptLocation: scriptLocation
        )

        do {
            _ = try await glueClient.createJob(
                input: CreateJobInput(
                    command: command,
                    description: "Created by the AWS SDK for Swift Glue basic
 scenario example.",
                    glueVersion: "3.0",
                    name: jobName,
                    numberOfWorkers: 10,
                    role: role,
                    workerType: .g1x
                )
            )
        } catch {
            return false
        }
        return true
    }
```

- Pour plus de détails sur l'API, reportez-vous CreateJobà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation **`DeleteCrawler`** avec un AWS SDK

Les exemples de code suivants illustrent comment utiliser `DeleteCrawler`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Principes de base

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Delete an AWS Glue crawler.
    /// </summary>
    /// <param name="crawlerName">The name of the crawler.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteCrawlerAsync(string crawlerName)
    {
        var response = await _amazonGlue.DeleteCrawlerAsync(new
DeleteCrawlerRequest { Name = crawlerName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Pour plus de détails sur l'API, reportez-vous DeleteCrawlerà la section Référence des AWS SDK pour .NET API.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region in which the bucket was created
(overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::Glue::GlueClient client(clientConfig);

        Aws::Glue::Model::DeleteCrawlerRequest request;
        request.SetName(crawler);

        Aws::Glue::Model::DeleteCrawlerOutcome outcome =
client.DeleteCrawler(request);

        if (outcome.IsSuccess()) {
            std::cout << "Successfully deleted the crawler." << std::endl;
        }
        else {
            std::cerr << "Error deleting the crawler. "
                      << outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
```

- Pour plus de détails sur l'API, reportez-vous DeleteCrawlerà la section Référence des AWS SDK pour C++ API.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
   /**
    * Deletes a specific AWS Glue crawler.
    *
    * @param glueClient  the AWS Glue client object
    * @param crawlerName the name of the crawler to be deleted
    * @throws GlueException if an error occurs during the deletion process
    */
   public static void deleteSpecificCrawler(GlueClient glueClient, String
crawlerName) {
       try {
           DeleteCrawlerRequest deleteCrawlerRequest =
DeleteCrawlerRequest.builder()
               .name(crawlerName)
               .build();

           glueClient.deleteCrawler(deleteCrawlerRequest);
           System.out.println(crawlerName + " was deleted");

       } catch (GlueException e) {
           throw e;
       }
   }
```

- Pour plus de détails sur l'API, reportez-vous DeleteCrawlerà la section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
const deleteCrawler = (crawlerName) => {
  const client = new GlueClient({});

  const command = new DeleteCrawlerCommand({
    Name: crawlerName,
  });

  return client.send(command);
};
```

- Pour plus de détails sur l'API, reportez-vous DeleteCrawlerà la section Référence des AWS SDK pour JavaScript API.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
        echo "Delete the crawler.\n";
        $glueClient->deleteCrawler([
            'Name' => $crawlerName,
        ]);

    public function deleteCrawler($crawlerName)
```

```
    {
        return $this->glueClient->deleteCrawler([
            'Name' => $crawlerName,
        ]);
    }
```

- Pour plus de détails sur l'API, reportez-vous DeleteCrawlerà la section Référence des AWS SDK pour PHP API.

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client


    def delete_crawler(self, name):
        """
        Deletes a crawler.

        :param name: The name of the crawler to delete.
        """
        try:
            self.glue_client.delete_crawler(Name=name)
        except ClientError as err:
            logger.error(
                "Couldn't delete crawler %s. Here's why: %s: %s",
```

```
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- Pour plus de détails sur l'API, consultez DeleteCrawlerle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Deletes a crawler with the specified name.
  #
  # @param name [String] The name of the crawler to delete.
  # @return [void]
  def delete_crawler(name)
    @glue_client.delete_crawler(name: name)
  rescue Aws::Glue::Errors::ServiceError => e
```

```
    @logger.error("Glue could not delete crawler #{name}: \n#{e.message}")
    raise
  end
```

- Pour plus de détails sur l'API, reportez-vous DeleteCrawlerà la section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
        glue.delete_crawler()
            .name(self.crawler())
            .send()
            .await
            .map_err(GlueMvpError::from_glue_sdk)?;
```

- Pour plus de détails sur l'API, voir DeleteCrawlerla section de référence de l'API AWS SDK for Rust.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    TRY.
```

```
        " iv_crawler_name = 'my-crawler'
        lo_glu->deletecrawler( iv_name = iv_crawler_name ).
        MESSAGE 'Crawler deleted successfully.' TYPE 'I'.
      CATCH /aws1/cx_glucrawlerrunningex.
        MESSAGE 'Crawler is currently running.' TYPE 'E'.
      CATCH /aws1/cx_gluentitynotfoundex.
        MESSAGE 'Crawler does not exist.' TYPE 'E'.
      CATCH /aws1/cx_gluoperationtimeoutex INTO DATA(lo_timeout_ex).
        DATA(lv_timeout_error) = lo_timeout_ex->if_message~get_longtext( ).
        MESSAGE lv_timeout_error TYPE 'E'.
      CATCH /aws1/cx_gluschdrtransingex.
        MESSAGE 'Scheduler is transitioning.' TYPE 'E'.
    ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous DeleteCrawlerà la section de référence du AWS SDK pour l'API SAP ABAP.

Swift

Kit SDK pour Swift

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import AWSClientRuntime
import AWSGlue

    /// Delete an AWS Glue crawler.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - name: The name of the crawler to delete.
    ///
    /// - Returns: `true` if successful, otherwise `false`.
    func deleteCrawler(glueClient: GlueClient, name: String) async -> Bool {
        do {
            _ = try await glueClient.deleteCrawler(
```

```
                input: DeleteCrawlerInput(name: name)
            )
        } catch {
            return false
        }
        return true
    }
```

- Pour plus de détails sur l'API, reportez-vous DeleteCrawlerà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation **`DeleteDatabase`** avec un AWS SDK

Les exemples de code suivants illustrent comment utiliser `DeleteDatabase`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Principes de base

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
/// <summary>
/// Delete the AWS Glue database.
/// </summary>
```

```
    /// <param name="dbName">The name of the database.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteDatabaseAsync(string dbName)
    {
        var response = await _amazonGlue.DeleteDatabaseAsync(new
DeleteDatabaseRequest { Name = dbName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Pour plus de détails sur l'API, reportez-vous DeleteDatabaseà la section Référence des AWS SDK pour .NET API.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region in which the bucket was created
(overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::Glue::GlueClient client(clientConfig);

        Aws::Glue::Model::DeleteDatabaseRequest request;
        request.SetName(database);

        Aws::Glue::Model::DeleteDatabaseOutcome outcome = client.DeleteDatabase(
                request);

        if (outcome.IsSuccess()) {
            std::cout << "Successfully deleted the database." << std::endl;
        }
        else {
```

```
            std::cerr << "Error deleting database. " <<
outcome.GetError().GetMessage()
                      << std::endl;
            result = false;
        }
```

- Pour plus de détails sur l'API, reportez-vous DeleteDatabaseà la section Référence des AWS SDK pour C++ API.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /**
     * Deletes a AWS Glue Database.
     *
     * @param glueClient   An instance of the AWS Glue client used to interact
with the AWS Glue service.
     * @param databaseName The name of the database to be deleted.
     * @throws GlueException If an error occurs while deleting the database.
     */
    public static void deleteDatabase(GlueClient glueClient, String databaseName)
{
        try {
            DeleteDatabaseRequest request = DeleteDatabaseRequest.builder()
                .name(databaseName)
                .build();

            glueClient.deleteDatabase(request);
            System.out.println(databaseName + " was successfully deleted");

        } catch (GlueException e) {
            throw e;
        }
```

```
    }
```

- Pour plus de détails sur l'API, reportez-vous DeleteDatabaseà la section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
const deleteDatabase = (databaseName) => {
  const client = new GlueClient({});

  const command = new DeleteDatabaseCommand({
    Name: databaseName,
  });

  return client.send(command);
};
```

- Pour plus de détails sur l'API, reportez-vous DeleteDatabaseà la section Référence des AWS SDK pour JavaScript API.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
        echo "Delete the databases.\n";
        $glueClient->deleteDatabase([
            'Name' => $databaseName,
        ]);

    public function deleteDatabase($databaseName)
    {
        return $this->glueClient->deleteDatabase([
            'Name' => $databaseName,
        ]);
    }
```

- Pour plus de détails sur l'API, reportez-vous DeleteDatabaseà la section Référence des AWS SDK pour PHP API.

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client


    def delete_database(self, name):
        """
        Deletes a metadata database from your Data Catalog.

        :param name: The name of the database to delete.
```

```
        """
        try:
            self.glue_client.delete_database(Name=name)
        except ClientError as err:
            logger.error(
                "Couldn't delete database %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- Pour plus de détails sur l'API, consultez DeleteDatabasele AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Removes a specified database from a Data Catalog.
```

```
#
# @param database_name [String] The name of the database to delete.
# @return [void]
def delete_database(database_name)
  @glue_client.delete_database(name: database_name)
rescue Aws::Glue::Errors::ServiceError => e
  @logger.error("Glue could not delete database: \n#{e.message}")
end
```

- Pour plus de détails sur l'API, reportez-vous DeleteDatabaseà la section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
        glue.delete_database()
            .name(self.database())
            .send()
            .await
            .map_err(GlueMvpError::from_glue_sdk)?;
```

- Pour plus de détails sur l'API, voir DeleteDatabasela section de référence de l'API AWS SDK for Rust.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
TRY.
    " iv_database_name = 'my-database'
    lo_glu->deletedatabase( iv_name = iv_database_name ).
    MESSAGE 'Database deleted successfully.' TYPE 'I'.
  CATCH /aws1/cx_gluentitynotfoundex.
    MESSAGE 'Database does not exist.' TYPE 'E'.
  CATCH /aws1/cx_gluinvalidinputex INTO DATA(lo_invalid_ex).
    DATA(lv_invalid_error) = lo_invalid_ex->if_message~get_longtext( ).
    MESSAGE lv_invalid_error TYPE 'E'.
  CATCH /aws1/cx_gluinternalserviceex INTO DATA(lo_internal_ex).
    DATA(lv_internal_error) = lo_internal_ex->if_message~get_longtext( ).
    MESSAGE lv_internal_error TYPE 'E'.
  CATCH /aws1/cx_gluoperationtimeoutex INTO DATA(lo_timeout_ex).
    DATA(lv_timeout_error) = lo_timeout_ex->if_message~get_longtext( ).
    MESSAGE lv_timeout_error TYPE 'E'.
ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous DeleteDatabaseà la section de référence du AWS SDK pour l'API SAP ABAP.

Swift

Kit SDK pour Swift

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import AWSClientRuntime
import AWSGlue

    /// Delete the specified database.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - databaseName: The name of the database to delete.
    ///   - deleteTables: A Bool indicating whether or not to delete the
    ///     tables in the database before attempting to delete the database.
    ///
    /// - Returns: `true` if the database (and optionally its tables) are
    ///   deleted, otherwise `false`.
    func deleteDatabase(glueClient: GlueClient, name databaseName: String,
                        withTables deleteTables: Bool = false) async -> Bool {
        if deleteTables {
            var tableNames: [String] = []

            // Get a list of the names of all of the tables in the database.

            let tableList = await self.getTablesInDatabase(glueClient:
 glueClient, databaseName: databaseName)
            for table in tableList {
                guard let name = table.name else {
                    continue
                }
                tableNames.append(name)
            }

            // Delete the tables. If there's only one table, use
            // `deleteTable()`, otherwise, use `batchDeleteTable()`. You can
            // use `batchDeleteTable()` for a single table, but this
            // demonstrates the use of `deleteTable()`.

            if tableNames.count == 1 {
                do {
                    print("    Deleting table...")
                    _ = try await glueClient.deleteTable(
                        input: DeleteTableInput(
                            databaseName: databaseName,
                            name: tableNames[0]
                        )
                    )
```

```
                } catch {
                    print("*** Unable to delete the table.")
                }
            } else {
                do {
                    print("    Deleting tables...")
                    _ = try await glueClient.batchDeleteTable(
                        input: BatchDeleteTableInput(
                            databaseName: databaseName,
                            tablesToDelete: tableNames
                        )
                    )
                } catch {
                    print("*** Unable to delete the tables.")
                }
            }
        }

        // Delete the database itself.

        do {
            print("    Deleting the database itself...")
            _ = try await glueClient.deleteDatabase(
                input: DeleteDatabaseInput(name: databaseName)
            )
        } catch {
            print("*** Unable to delete the database.")
            return false
        }
        return true
    }
```

- Pour plus de détails sur l'API, reportez-vous DeleteDatabaseà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation **`DeleteJob`** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `DeleteJob`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Principes de base

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Delete an AWS Glue job.
    /// </summary>
    /// <param name="jobName">The name of the job.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteJobAsync(string jobName)
    {
        var response = await _amazonGlue.DeleteJobAsync(new DeleteJobRequest
{ JobName = jobName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Pour plus de détails sur l'API, voir DeleteJobla section Référence des AWS SDK pour .NET API.

C++

SDK pour C++

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region in which the bucket was created
(overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::Glue::GlueClient client(clientConfig);

        Aws::Glue::Model::DeleteJobRequest request;
        request.SetJobName(job);

        Aws::Glue::Model::DeleteJobOutcome outcome = client.DeleteJob(request);


        if (outcome.IsSuccess()) {
            std::cout << "Successfully deleted the job." << std::endl;
        }
        else {
            std::cerr << "Error deleting the job. " <<
outcome.GetError().GetMessage()
                      << std::endl;
            result = false;
        }
```

- Pour plus de détails sur l'API, voir DeleteJobla section Référence des AWS SDK pour C++ API.

CLI

AWS CLI

Pour supprimer une tâche

L'exemple `delete-job` suivant supprime une tâche qui n'est plus nécessaire.

```
aws glue delete-job \
    --job-name my-testing-job
```

Sortie :

```
{
    "JobName": "my-testing-job"
}
```

Pour plus d'informations, consultez Working with Jobs on the AWS Glue Console dans le AWS Glue Developer Guide.

- Pour plus de détails sur l'API, voir DeleteJobla section Référence des AWS CLI commandes.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /**
     * Deletes a Glue job.
     *
     * @param glueClient the Glue client to use for the operation
     * @param jobName    the name of the job to be deleted
     * @throws GlueException if there is an error deleting the job
     */
```

```
    public static void deleteJob(GlueClient glueClient, String jobName) {
        try {
            DeleteJobRequest jobRequest = DeleteJobRequest.builder()
                .jobName(jobName)
                .build();

            glueClient.deleteJob(jobRequest);
            System.out.println(jobName + " was successfully deleted");

        } catch (GlueException e) {
            throw e;
        }
    }
```

- Pour plus de détails sur l'API, voir DeleteJobla section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
const deleteJob = (jobName) => {
  const client = new GlueClient({});

  const command = new DeleteJobCommand({
    JobName: jobName,
  });

  return client.send(command);
};
```

- Pour plus de détails sur l'API, voir DeleteJobla section Référence des AWS SDK pour JavaScript API.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
        echo "Delete the job.\n";
        $glueClient->deleteJob([
            'JobName' => $job['Name'],
        ]);

    public function deleteJob($jobName)
    {
        return $this->glueClient->deleteJob([
            'JobName' => $jobName,
        ]);
    }
```

- Pour plus de détails sur l'API, voir DeleteJobla section Référence des AWS SDK pour PHP API.

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""
```

```
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client


    def delete_job(self, job_name):
        """
        Deletes a job definition. This also deletes data about all runs that are
        associated with this job definition.

        :param job_name: The name of the job definition to delete.
        """
        try:
            self.glue_client.delete_job(JobName=job_name)
        except ClientError as err:
            logger.error(
                "Couldn't delete job %s. Here's why: %s: %s",
                job_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- Pour plus de détails sur l'API, consultez DeleteJoble AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Deletes a job with the specified name.
  #
  # @param job_name [String] The name of the job to delete.
  # @return [void]
  def delete_job(job_name)
    @glue_client.delete_job(job_name: job_name)
  rescue Aws::Glue::Errors::ServiceError => e
    @logger.error("Glue could not delete job: \n#{e.message}")
  end
```

- Pour plus de détails sur l'API, voir DeleteJobla section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
        glue.delete_job()
            .job_name(self.job())
            .send()
            .await
            .map_err(GlueMvpError::from_glue_sdk)?;
```

- Pour plus de détails sur l'API, voir DeleteJobla section de référence de l'API AWS SDK for Rust.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
TRY.
    " iv_job_name = 'my-etl-job'
    lo_glu->deletejob( iv_jobname = iv_job_name ).
    MESSAGE 'Job deleted successfully.' TYPE 'I'.
  CATCH /aws1/cx_gluinvalidinputex INTO DATA(lo_invalid_ex).
    DATA(lv_invalid_error) = lo_invalid_ex->if_message~get_longtext( ).
    MESSAGE lv_invalid_error TYPE 'E'.
  CATCH /aws1/cx_gluinternalserviceex INTO DATA(lo_internal_ex).
    DATA(lv_internal_error) = lo_internal_ex->if_message~get_longtext( ).
    MESSAGE lv_internal_error TYPE 'E'.
  CATCH /aws1/cx_gluoperationtimeoutex INTO DATA(lo_timeout_ex).
    DATA(lv_timeout_error) = lo_timeout_ex->if_message~get_longtext( ).
    MESSAGE lv_timeout_error TYPE 'E'.
ENDTRY.
```

- Pour plus de détails sur l'API, consultez DeleteJobla section de référence du AWS SDK pour l'API SAP ABAP.

Swift

Kit SDK pour Swift

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import AWSClientRuntime
import AWSGlue

    /// Delete an AWS Glue job.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - jobName: The name of the job to delete.
    ///
    /// - Returns: `true` if the job is successfully deleted, otherwise `false`.
    func deleteJob(glueClient: GlueClient, name jobName: String) async -> Bool {
        do {
            _ = try await glueClient.deleteJob(
                input: DeleteJobInput(jobName: jobName)
            )
        } catch {
            return false
        }
        return true
    }
```

- Pour plus de détails sur l'API, reportez-vous DeleteJobà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation **`DeleteTable`** avec un AWS SDK

Les exemples de code suivants illustrent comment utiliser `DeleteTable`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Principes de base

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Delete a table from an AWS Glue database.
    /// </summary>
    /// <param name="tableName">The table to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteTableAsync(string dbName, string tableName)
    {
        var response = await _amazonGlue.DeleteTableAsync(new DeleteTableRequest
{ Name = tableName, DatabaseName = dbName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Pour plus de détails sur l'API, reportez-vous DeleteTableà la section Référence des AWS SDK pour .NET API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
const deleteTable = (databaseName, tableName) => {
  const client = new GlueClient({});

  const command = new DeleteTableCommand({
    DatabaseName: databaseName,
    Name: tableName,
  });

  return client.send(command);
};
```

- Pour plus de détails sur l'API, reportez-vous DeleteTableà la section Référence des AWS SDK pour JavaScript API.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
        echo "Delete the tables.\n";
        foreach ($tables['TableList'] as $table) {
            $glueService->deleteTable($table['Name'], $databaseName);
        }
```

```
    public function deleteTable($tableName, $databaseName)
    {
        return $this->glueClient->deleteTable([
            'DatabaseName' => $databaseName,
            'Name' => $tableName,
        ]);
    }
```

- Pour plus de détails sur l'API, reportez-vous DeleteTableà la section Référence des AWS SDK pour PHP API.

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client


    def delete_table(self, db_name, table_name):
        """
        Deletes a table from a metadata database.

        :param db_name: The name of the database that contains the table.
        :param table_name: The name of the table to delete.
        """
        try:
            self.glue_client.delete_table(DatabaseName=db_name, Name=table_name)
```

```
        except ClientError as err:
            logger.error(
                "Couldn't delete table %s. Here's why: %s: %s",
                table_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- Pour plus de détails sur l'API, consultez DeleteTablele AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Deletes a table with the specified name.
  #
  # @param database_name [String] The name of the catalog database in which the
 table resides.
```

```
  # @param table_name [String] The name of the table to be deleted.
  # @return [void]
  def delete_table(database_name, table_name)
    @glue_client.delete_table(database_name: database_name, name: table_name)
  rescue Aws::Glue::Errors::ServiceError => e
    @logger.error("Glue could not delete job: \n#{e.message}")
  end
```

- Pour plus de détails sur l'API, reportez-vous DeleteTableà la section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
        for t in &self.tables {
            glue.delete_table()
                .name(t.name())
                .database_name(self.database())
                .send()
                .await
                .map_err(GlueMvpError::from_glue_sdk)?;
        }
```

- Pour plus de détails sur l'API, voir DeleteTablela section de référence de l'API AWS SDK for Rust.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
TRY.
    " iv_database_name = 'my-database'
    " iv_table_name = 'my-table'
    lo_glu->deletetable(
      iv_databasename = iv_database_name
      iv_name = iv_table_name ).
    MESSAGE 'Table deleted successfully.' TYPE 'I'.
  CATCH /aws1/cx_gluentitynotfoundex.
    MESSAGE 'Table or database does not exist.' TYPE 'E'.
  CATCH /aws1/cx_gluinvalidinputex INTO DATA(lo_invalid_ex).
    DATA(lv_invalid_error) = lo_invalid_ex->if_message~get_longtext( ).
    MESSAGE lv_invalid_error TYPE 'E'.
  CATCH /aws1/cx_gluinternalserviceex INTO DATA(lo_internal_ex).
    DATA(lv_internal_error) = lo_internal_ex->if_message~get_longtext( ).
    MESSAGE lv_internal_error TYPE 'E'.
  CATCH /aws1/cx_gluoperationtimeoutex INTO DATA(lo_timeout_ex).
    DATA(lv_timeout_error) = lo_timeout_ex->if_message~get_longtext( ).
    MESSAGE lv_timeout_error TYPE 'E'.
ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous DeleteTableà la section de référence du AWS SDK pour l'API SAP ABAP.

Swift

Kit SDK pour Swift

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import AWSClientRuntime
import AWSGlue

                do {
                    print("    Deleting table...")
                    _ = try await glueClient.deleteTable(
                        input: DeleteTableInput(
                            databaseName: databaseName,
                            name: tableNames[0]
                        )
                    )
                } catch {
                    print("*** Unable to delete the table.")
                }
```

- Pour plus de détails sur l'API, reportez-vous DeleteTableà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation **`GetCrawler`** avec un AWS SDK

Les exemples de code suivants illustrent comment utiliser `GetCrawler`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Principes de base

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Get information about an AWS Glue crawler.
    /// </summary>
    /// <param name="crawlerName">The name of the crawler.</param>
    /// <returns>A Crawler object describing the crawler.</returns>
    public async Task<Crawler?> GetCrawlerAsync(string crawlerName)
    {
        var crawlerRequest = new GetCrawlerRequest
        {
            Name = crawlerName,
        };

        var response = await _amazonGlue.GetCrawlerAsync(crawlerRequest);
        if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
        {
            var databaseName = response.Crawler.DatabaseName;
            Console.WriteLine($"{crawlerName} has the database {databaseName}");
            return response.Crawler;
        }

        Console.WriteLine($"No information regarding {crawlerName} could be
found.");
        return null;
    }
```

- Pour plus de détails sur l'API, reportez-vous GetCrawlerà la section Référence des AWS SDK pour .NET API.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
       Aws::Client::ClientConfiguration clientConfig;
       // Optional: Set to the AWS Region in which the bucket was created
(overrides config file).
       // clientConfig.region = "us-east-1";

   Aws::Glue::GlueClient client(clientConfig);

       Aws::Glue::Model::GetCrawlerRequest request;
       request.SetName(CRAWLER_NAME);

       Aws::Glue::Model::GetCrawlerOutcome outcome = client.GetCrawler(request);

       if (outcome.IsSuccess()) {
           Aws::Glue::Model::CrawlerState crawlerState =
outcome.GetResult().GetCrawler().GetState();
           std::cout << "Retrieved crawler with state " <<

Aws::Glue::Model::CrawlerStateMapper::GetNameForCrawlerState(
                               crawlerState)
                   << "." << std::endl;
       }
       else {
           std::cerr << "Error retrieving a crawler.  "
                     << outcome.GetError().GetMessage() << std::endl;
           deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, "", bucketName,
                        clientConfig);
           return false;
       }
```

- Pour plus de détails sur l'API, reportez-vous GetCrawlerà la section Référence des AWS SDK pour C++ API.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
   /**
    * Retrieves a specific crawler from the AWS Glue service and waits for it to
be in the "READY" state.
    *
    * @param glueClient  the AWS Glue client used to interact with the Glue
service
    * @param crawlerName the name of the crawler to be retrieved
    */
   public static void getSpecificCrawler(GlueClient glueClient, String
crawlerName) throws InterruptedException {
       try {
           GetCrawlerRequest crawlerRequest = GetCrawlerRequest.builder()
               .name(crawlerName)
               .build();

           boolean ready = false;
           while (!ready) {
               GetCrawlerResponse response =
glueClient.getCrawler(crawlerRequest);
               String status = response.crawler().stateAsString();
               if (status.compareTo("READY") == 0) {
                   ready = true;
               }
               Thread.sleep(3000);
           }

           System.out.println("The crawler is now ready");

       } catch (GlueException | InterruptedException e) {
           throw e;
       }
   }
```

- Pour plus de détails sur l'API, reportez-vous GetCrawlerà la section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
const getCrawler = (name) => {
  const client = new GlueClient({});

  const command = new GetCrawlerCommand({
    Name: name,
  });

  return client.send(command);
};
```

- Pour plus de détails sur l'API, reportez-vous GetCrawlerà la section Référence des AWS SDK pour JavaScript API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun getSpecificCrawler(crawlerName: String?) {
    val request =
        GetCrawlerRequest {
            name = crawlerName
        }
    GlueClient.fromEnvironment { region = "us-east-1" }.use { glueClient ->
        val response = glueClient.getCrawler(request)
        val role = response.crawler?.role
        println("The role associated with this crawler is $role")
    }
}
```

- Pour plus de détails sur l'API, consultez GetCrawlerla section AWS SDK pour la référence de l'API Kotlin.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
        echo "Waiting for crawler";
        do {
            $crawler = $glueService->getCrawler($crawlerName);
            echo ".";
            sleep(10);
        } while ($crawler['Crawler']['State'] != "READY");
        echo "\n";

    public function getCrawler($crawlerName)
    {
        return $this->customWaiter(function () use ($crawlerName) {
            return $this->glueClient->getCrawler([
                'Name' => $crawlerName,
            ]);
        });
```

```
    }
```

- Pour plus de détails sur l'API, reportez-vous GetCrawlerà la section Référence des AWS SDK pour PHP API.

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client


    def get_crawler(self, name):
        """
        Gets information about a crawler.

        :param name: The name of the crawler to look up.
        :return: Data about the crawler.
        """
        crawler = None
        try:
            response = self.glue_client.get_crawler(Name=name)
            crawler = response["Crawler"]
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityNotFoundException":
                logger.info("Crawler %s doesn't exist.", name)
            else:
```

```
                logger.error(
                    "Couldn't get crawler %s. Here's why: %s: %s",
                    name,
                    err.response["Error"]["Code"],
                    err.response["Error"]["Message"],
                )
                raise
        return crawler
```

- Pour plus de détails sur l'API, consultez GetCrawlerle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Retrieves information about a specific crawler.
  #
  # @param name [String] The name of the crawler to retrieve information about.
```

```
  # @return [Aws::Glue::Types::Crawler, nil] The crawler object if found, or nil
if not found.
 def get_crawler(name)
   @glue_client.get_crawler(name: name)
 rescue Aws::Glue::Errors::EntityNotFoundException
   @logger.info("Crawler #{name} doesn't exist.")
   false
 rescue Aws::Glue::Errors::GlueException => e
   @logger.error("Glue could not get crawler #{name}: \n#{e.message}")
   raise
 end
```

- Pour plus de détails sur l'API, reportez-vous GetCrawlerà la section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
            let tmp_crawler = glue
                .get_crawler()
                .name(self.crawler())
                .send()
                .await
                .map_err(GlueMvpError::from_glue_sdk)?;
```

- Pour plus de détails sur l'API, voir GetCrawlerla section de référence de l'API AWS SDK for Rust.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    TRY.
        " iv_crawler_name = 'my-crawler'
        oo_result = lo_glu->getcrawler( iv_name = iv_crawler_name ).
        DATA(lo_crawler) = oo_result->get_crawler( ).
        MESSAGE 'Crawler information retrieved.' TYPE 'I'.
      CATCH /aws1/cx_gluentitynotfoundex.
        MESSAGE 'Crawler does not exist.' TYPE 'I'.
      CATCH /aws1/cx_gluoperationtimeoutex INTO DATA(lo_timeout_ex).
        DATA(lv_timeout_error) = lo_timeout_ex->if_message~get_longtext( ).
        MESSAGE lv_timeout_error TYPE 'E'.
    ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous GetCrawlerà la section de référence du AWS SDK pour l'API SAP ABAP.

Swift

Kit SDK pour Swift

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import AWSClientRuntime
import AWSGlue
```

```
    /// Get the state of the specified AWS Glue crawler.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - name: The name of the crawler whose state should be returned.
    ///
    /// - Returns: A `GlueClientTypes.CrawlerState` value describing the
    ///   state of the crawler.
    func getCrawlerState(glueClient: GlueClient, name: String) async ->
GlueClientTypes.CrawlerState {
        do {
            let output = try await glueClient.getCrawler(
                input: GetCrawlerInput(name: name)
            )

            // If the crawler or its state is `nil`, report that the crawler
            // is stopping. This may not be what you want for your
            // application but it works for this one!

            guard let crawler = output.crawler else {
                return GlueClientTypes.CrawlerState.stopping
            }
            guard let state = crawler.state else {
                return GlueClientTypes.CrawlerState.stopping
            }
            return state
        } catch {
            return GlueClientTypes.CrawlerState.stopping
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous GetCrawlerà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation **`GetDatabase`** avec un AWS SDK

Les exemples de code suivants illustrent comment utiliser `GetDatabase`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Principes de base

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Get information about an AWS Glue database.
    /// </summary>
    /// <param name="dbName">The name of the database.</param>
    /// <returns>A Database object containing information about the database.</
returns>
    public async Task<Database> GetDatabaseAsync(string dbName)
    {
        var databasesRequest = new GetDatabaseRequest
        {
            Name = dbName,
        };

        var response = await _amazonGlue.GetDatabaseAsync(databasesRequest);
        return response.Database;
    }
```

- Pour plus de détails sur l'API, reportez-vous GetDatabaseà la section Référence des AWS SDK pour .NET API.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region in which the bucket was created
(overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::Glue::GlueClient client(clientConfig);

        Aws::Glue::Model::GetDatabaseRequest request;
        request.SetName(CRAWLER_DATABASE_NAME);

        Aws::Glue::Model::GetDatabaseOutcome outcome =
client.GetDatabase(request);

        if (outcome.IsSuccess()) {
            const Aws::Glue::Model::Database &database =
outcome.GetResult().GetDatabase();

            std::cout << "Successfully retrieve the database\n" <<
                      database.Jsonize().View().WriteReadable() << "'." <<
std::endl;
        }
        else {
            std::cerr << "Error getting the database.  "
                      << outcome.GetError().GetMessage() << std::endl;
            deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, "", bucketName,
                         clientConfig);
            return false;
        }
```

- Pour plus de détails sur l'API, reportez-vous GetDatabaseà la section Référence des AWS SDK pour C++ API.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
   /**
    * Retrieves the specific database from the AWS Glue service.
    *
    * @param glueClient   an instance of the AWS Glue client used to interact
with the service
    * @param databaseName the name of the database to retrieve
    * @throws GlueException if there is an error retrieving the database from
the AWS Glue service
    */
   public static void getSpecificDatabase(GlueClient glueClient, String
databaseName) {
       try {
           GetDatabaseRequest databasesRequest = GetDatabaseRequest.builder()
               .name(databaseName)
               .build();

           GetDatabaseResponse response =
glueClient.getDatabase(databasesRequest);
           Instant createDate = response.database().createTime();

           // Convert the Instant to readable date.
           DateTimeFormatter formatter =
DateTimeFormatter.ofLocalizedDateTime(FormatStyle.SHORT)
               .withLocale(Locale.US)
               .withZone(ZoneId.systemDefault());

           formatter.format(createDate);
           System.out.println("The create date of the database is " +
createDate);

       } catch (GlueException e) {
           throw e;
```

```
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous GetDatabaseà la section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
const getDatabase = (name) => {
  const client = new GlueClient({});

  const command = new GetDatabaseCommand({
    Name: name,
  });

  return client.send(command);
};
```

- Pour plus de détails sur l'API, reportez-vous GetDatabaseà la section Référence des AWS SDK pour JavaScript API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun getSpecificDatabase(databaseName: String?) {
    val request =
        GetDatabaseRequest {
            name = databaseName
        }

    GlueClient.fromEnvironment { region = "us-east-1" }.use { glueClient ->
        val response = glueClient.getDatabase(request)
        val dbDesc = response.database?.description
        println("The database description is $dbDesc")
    }
}
```

- Pour plus de détails sur l'API, consultez GetDatabasela section AWS SDK pour la référence de l'API Kotlin.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
        $databaseName = "doc-example-database-$uniqid";

        $database = $glueService->getDatabase($databaseName);
        echo "Found a database named " . $database['Database']['Name'] . "\n";

    public function getDatabase(string $databaseName): Result
    {
        return $this->customWaiter(function () use ($databaseName) {
            return $this->glueClient->getDatabase([
                'Name' => $databaseName,
            ]);
        });
    }
```

- Pour plus de détails sur l'API, reportez-vous GetDatabaseà la section Référence des AWS SDK pour PHP API.

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client


    def get_database(self, name):
        """
        Gets information about a database in your Data Catalog.

        :param name: The name of the database to look up.
        :return: Information about the database.
        """
        try:
            response = self.glue_client.get_database(Name=name)
        except ClientError as err:
            logger.error(
                "Couldn't get database %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
```

```
            raise
        else:
            return response["Database"]
```

- Pour plus de détails sur l'API, consultez GetDatabasele AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Retrieves information about a specific database.
  #
  # @param name [String] The name of the database to retrieve information about.
  # @return [Aws::Glue::Types::Database, nil] The database object if found, or
 nil if not found.
  def get_database(name)
    response = @glue_client.get_database(name: name)
    response.database
  rescue Aws::Glue::Errors::GlueException => e
```

```
    @logger.error("Glue could not get database #{name}: \n#{e.message}")
    raise
  end
```

- Pour plus de détails sur l'API, reportez-vous GetDatabaseà la section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
        let database = glue
            .get_database()
            .name(self.database())
            .send()
            .await
            .map_err(GlueMvpError::from_glue_sdk)?
            .to_owned();
        let database = database
            .database()
            .ok_or_else(|| GlueMvpError::Unknown("Could not find
database".into()))?;
```

- Pour plus de détails sur l'API, voir GetDatabasela section de référence de l'API AWS SDK for Rust.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
TRY.
    " iv_database_name = 'my-database'
    oo_result = lo_glu->getdatabase( iv_name = iv_database_name ).
    DATA(lo_database) = oo_result->get_database( ).
    MESSAGE 'Database information retrieved.' TYPE 'I'.
  CATCH /aws1/cx_gluentitynotfoundex.
    MESSAGE 'Database does not exist.' TYPE 'E'.
  CATCH /aws1/cx_gluinvalidinputex INTO DATA(lo_invalid_ex).
    DATA(lv_invalid_error) = lo_invalid_ex->if_message~get_longtext( ).
    MESSAGE lv_invalid_error TYPE 'E'.
  CATCH /aws1/cx_gluinternalserviceex INTO DATA(lo_internal_ex).
    DATA(lv_internal_error) = lo_internal_ex->if_message~get_longtext( ).
    MESSAGE lv_internal_error TYPE 'E'.
  CATCH /aws1/cx_gluoperationtimeoutex INTO DATA(lo_timeout_ex).
    DATA(lv_timeout_error) = lo_timeout_ex->if_message~get_longtext( ).
    MESSAGE lv_timeout_error TYPE 'E'.
ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous GetDatabaseà la section de référence du AWS SDK pour l'API SAP ABAP.

Swift

Kit SDK pour Swift

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import AWSClientRuntime
import AWSGlue

    /// Get the AWS Glue database with the specified name.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - name: The name of the database to return.
    ///
    /// - Returns: The `GlueClientTypes.Database` object describing the
    ///   specified database, or `nil` if an error occurs or the database
    ///   isn't found.
    func getDatabase(glueClient: GlueClient, name: String) async ->
 GlueClientTypes.Database? {
        do {
            let output = try await glueClient.getDatabase(
                input: GetDatabaseInput(name: name)
            )

            return output.database
        } catch {
            return nil
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous GetDatabaseà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation **`GetDatabases`** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `GetDatabases`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Principes de base

CLI

AWS CLI

Pour répertorier les définitions de certaines ou de toutes les bases de données du AWS Glue Data Catalog

L'exemple `get-databases` suivant renvoie les informations sur les bases de données du catalogue de données.

```
aws glue get-databases
```

Sortie :

```
{
    "DatabaseList": [
        {
            "Name": "default",
            "Description": "Default Hive database",
            "LocationUri": "file:/spark-warehouse",
            "CreateTime": 1602084052.0,
            "CreateTableDefaultPermissions": [
                {
                    "Principal": {
                        "DataLakePrincipalIdentifier": "IAM_ALLOWED_PRINCIPALS"
                    },
                    "Permissions": [
                        "ALL"
                    ]
                }
            ],
            "CatalogId": "111122223333"
        },
        {
            "Name": "flights-db",
            "CreateTime": 1587072847.0,
            "CreateTableDefaultPermissions": [
                {
                    "Principal": {
                        "DataLakePrincipalIdentifier": "IAM_ALLOWED_PRINCIPALS"
```

```
                    },
                    "Permissions": [
                        "ALL"
                    ]
                }
            ],
            "CatalogId": "111122223333"
        },
        {
            "Name": "legislators",
            "CreateTime": 1601415625.0,
            "CreateTableDefaultPermissions": [
                {
                    "Principal": {
                        "DataLakePrincipalIdentifier": "IAM_ALLOWED_PRINCIPALS"
                    },
                    "Permissions": [
                        "ALL"
                    ]
                }
            ],
            "CatalogId": "111122223333"
        },
        {
            "Name": "tempdb",
            "CreateTime": 1601498566.0,
            "CreateTableDefaultPermissions": [
                {
                    "Principal": {
                        "DataLakePrincipalIdentifier": "IAM_ALLOWED_PRINCIPALS"
                    },
                    "Permissions": [
                        "ALL"
                    ]
                }
            ],
            "CatalogId": "111122223333"
        }
    ]
}
```

Pour plus d'informations, consultez Définition d'une base de données dans votre catalogue de données dans le Guide du développeur AWS Glue.

- Pour plus de détails sur l'API, voir GetDatabasesla section Référence des AWS CLI commandes.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
const getDatabases = () => {
  const client = new GlueClient({});

  const command = new GetDatabasesCommand({});

  return client.send(command);
};
```

- Pour plus de détails sur l'API, voir GetDatabasesla section Référence des AWS SDK pour JavaScript API.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation **`GetJob`** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `GetJob`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Principes de base

CLI

AWS CLI

Pour récupérer des informations sur une tâche

L'exemple `get-job` suivant récupère les informations sur une tâche.

```
aws glue get-job \
    --job-name my-testing-job
```

Sortie :

```
{
    "Job": {
        "Name": "my-testing-job",
        "Role": "Glue_DefaultRole",
        "CreatedOn": 1602805698.167,
        "LastModifiedOn": 1602805698.167,
        "ExecutionProperty": {
            "MaxConcurrentRuns": 1
        },
        "Command": {
            "Name": "gluestreaming",
            "ScriptLocation": "s3://janetst-bucket-01/Scripts/test_script.scala",
            "PythonVersion": "2"
        },
        "DefaultArguments": {
            "--class": "GlueApp",
            "--job-language": "scala"
        },
        "MaxRetries": 0,
        "AllocatedCapacity": 10,
        "MaxCapacity": 10.0,
        "GlueVersion": "1.0"
    }
}
```

Pour plus d'informations, consultez Tâches dans le Guide du développeur AWS Glue.

- Pour plus de détails sur l'API, reportez-vous GetJobà la section Référence des AWS CLI commandes.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
const getJob = (jobName) => {
  const client = new GlueClient({});

  const command = new GetJobCommand({
    JobName: jobName,
  });

  return client.send(command);
};
```

- Pour plus de détails sur l'API, reportez-vous GetJobà la section Référence des AWS SDK pour JavaScript API.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation **`GetJobRun`** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `GetJobRun`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Principes de base

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Get information about a specific AWS Glue job run.
    /// </summary>
    /// <param name="jobName">The name of the job.</param>
    /// <param name="jobRunId">The Id of the job run.</param>
    /// <returns>A JobRun object with information about the job run.</returns>
    public async Task<JobRun> GetJobRunAsync(string jobName, string jobRunId)
    {
        var response = await _amazonGlue.GetJobRunAsync(new GetJobRunRequest
{ JobName = jobName, RunId = jobRunId });
        return response.JobRun;
    }
```

- Pour plus de détails sur l'API, reportez-vous GetJobRunà la section Référence des AWS SDK pour .NET API.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
        Aws::Client::ClientConfiguration clientConfig;
```

```
        // Optional: Set to the AWS Region in which the bucket was created
(overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::Glue::GlueClient client(clientConfig);

        Aws::Glue::Model::GetJobRunRequest jobRunRequest;
        jobRunRequest.SetJobName(jobName);
        jobRunRequest.SetRunId(jobRunID);

        Aws::Glue::Model::GetJobRunOutcome jobRunOutcome = client.GetJobRun(
                jobRunRequest);

        if (jobRunOutcome.IsSuccess()) {
            std::cout << "Displaying the job run JSON description." << std::endl;
            std::cout
                    <<
jobRunOutcome.GetResult().GetJobRun().Jsonize().View().WriteReadable()
                    << std::endl;
        }
        else {
            std::cerr << "Error get a job run. "
                      << jobRunOutcome.GetError().GetMessage()
                      << std::endl;
        }
```

- Pour plus de détails sur l'API, reportez-vous GetJobRunà la section Référence des AWS SDK pour C++ API.

CLI

AWS CLI

Pour obtenir les informations sur l'exécution d'une tâche

L'exemple `get-job-run` suivant récupère les informations sur l'exécution d'une tâche.

```
aws glue get-job-run \
    --job-name "Combine legistators data" \
    --run-id jr_012e176506505074d94d761755e5c62538ee1aad6f17d39f527e9140cf0c9a5e
```

Sortie :

```
{
    "JobRun": {
        "Id":
 "jr_012e176506505074d94d761755e5c62538ee1aad6f17d39f527e9140cf0c9a5e",
        "Attempt": 0,
        "JobName": "Combine legistators data",
        "StartedOn": 1602873931.255,
        "LastModifiedOn": 1602874075.985,
        "CompletedOn": 1602874075.985,
        "JobRunState": "SUCCEEDED",
        "Arguments": {
            "--enable-continuous-cloudwatch-log": "true",
            "--enable-metrics": "",
            "--enable-spark-ui": "true",
            "--job-bookmark-option": "job-bookmark-enable",
            "--spark-event-logs-path": "s3://aws-glue-assets-111122223333-us-
east-1/sparkHistoryLogs/"
        },
        "PredecessorRuns": [],
        "AllocatedCapacity": 10,
        "ExecutionTime": 117,
        "Timeout": 2880,
        "MaxCapacity": 10.0,
        "WorkerType": "G.1X",
        "NumberOfWorkers": 10,
        "LogGroupName": "/aws-glue/jobs",
        "GlueVersion": "2.0"
    }
}
```

Pour plus d'informations, consultez Exécutions de tâches dans le Guide du développeur AWS Glue.

- Pour plus de détails sur l'API, reportez-vous GetJobRunà la section Référence des AWS CLI commandes.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
const getJobRun = (jobName, jobRunId) => {
  const client = new GlueClient({});
  const command = new GetJobRunCommand({
    JobName: jobName,
    RunId: jobRunId,
  });

  return client.send(command);
};
```

- Pour plus de détails sur l'API, reportez-vous GetJobRunà la section Référence des AWS SDK pour JavaScript API.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
        $jobName = 'test-job-' . $uniqid;

        $outputBucketUrl = "s3://$bucketName";
        $runId = $glueService->startJobRun($jobName, $databaseName, $tables,
 $outputBucketUrl)['JobRunId'];
```

```
        echo "waiting for job";
        do {
            $jobRun = $glueService->getJobRun($jobName, $runId);
            echo ".";
            sleep(10);
        } while (!array_intersect([$jobRun['JobRun']['JobRunState']],
['SUCCEEDED', 'STOPPED', 'FAILED', 'TIMEOUT']));
        echo "\n";

    public function getJobRun($jobName, $runId, $predecessorsIncluded = false):
Result
    {
        return $this->glueClient->getJobRun([
            'JobName' => $jobName,
            'RunId' => $runId,
            'PredecessorsIncluded' => $predecessorsIncluded,
        ]);
    }
```

- Pour plus de détails sur l'API, reportez-vous GetJobRunà la section Référence des AWS SDK pour PHP API.

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
```

```
        self.glue_client = glue_client


    def get_job_run(self, name, run_id):
        """
        Gets information about a single job run.

        :param name: The name of the job definition for the run.
        :param run_id: The ID of the run.
        :return: Information about the run.
        """
        try:
            response = self.glue_client.get_job_run(JobName=name, RunId=run_id)
        except ClientError as err:
            logger.error(
                "Couldn't get job run %s/%s. Here's why: %s: %s",
                name,
                run_id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response["JobRun"]
```

- Pour plus de détails sur l'API, consultez GetJobRunle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Retrieves data for a specific job run.
  #
  # @param job_name [String] The name of the job run to retrieve data for.
  # @return [Glue::Types::GetJobRunResponse]
  def get_job_run(job_name, run_id)
    @glue_client.get_job_run(job_name: job_name, run_id: run_id)
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not get job runs: \n#{e.message}")
  end
```

- Pour plus de détails sur l'API, reportez-vous GetJobRunà la section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
        let get_job_run = || async {
            Ok::<JobRun, GlueMvpError>(
                glue.get_job_run()
                    .job_name(self.job())
                    .run_id(job_run_id.to_string())
```

```
                    .send()
                    .await
                    .map_err(GlueMvpError::from_glue_sdk)?
                    .job_run()
                    .ok_or_else(|| GlueMvpError::Unknown("Failed to get
job_run".into()))?
                    .to_owned(),
            )
        };

        let mut job_run = get_job_run().await?;
        let mut state =
job_run.job_run_state().unwrap_or(&unknown_state).to_owned();

        while matches!(
            state,
            JobRunState::Starting | JobRunState::Stopping | JobRunState::Running
        ) {
            info!(?state, "Waiting for job to finish");
            tokio::time::sleep(self.wait_delay).await;

            job_run = get_job_run().await?;
            state = job_run.job_run_state().unwrap_or(&unknown_state).to_owned();
        }
```

- Pour plus de détails sur l'API, voir GetJobRunla section de référence de l'API AWS SDK for Rust.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    TRY.
        " iv_job_name = 'my-etl-job'
```

```
        " iv_run_id = 'jr_abcd1234567890abcdef1234567890abcdef12345678'
        oo_result = lo_glu->getjobrun(
          iv_jobname = iv_job_name
          iv_runid = iv_run_id ).
        DATA(lo_job_run) = oo_result->get_jobrun( ).
        MESSAGE 'Job run information retrieved.' TYPE 'I'.
      CATCH /aws1/cx_gluentitynotfoundex.
        MESSAGE 'Job or job run does not exist.' TYPE 'E'.
      CATCH /aws1/cx_gluinvalidinputex INTO DATA(lo_invalid_ex).
        DATA(lv_invalid_error) = lo_invalid_ex->if_message~get_longtext( ).
        MESSAGE lv_invalid_error TYPE 'E'.
      CATCH /aws1/cx_gluinternalserviceex INTO DATA(lo_internal_ex).
        DATA(lv_internal_error) = lo_internal_ex->if_message~get_longtext( ).
        MESSAGE lv_internal_error TYPE 'E'.
      CATCH /aws1/cx_gluoperationtimeoutex INTO DATA(lo_timeout_ex).
        DATA(lv_timeout_error) = lo_timeout_ex->if_message~get_longtext( ).
        MESSAGE lv_timeout_error TYPE 'E'.
    ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous GetJobRunà la section de référence du AWS SDK pour l'API SAP ABAP.

Swift

Kit SDK pour Swift

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import AWSClientRuntime
import AWSGlue

    /// Get information about a specific AWS Glue job run.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - jobName: The name of the job to return job run data for.
```

```
    ///   - id: The run ID of the specific job run to return.
    ///
    /// - Returns: A `GlueClientTypes.JobRun` object describing the state of
    ///   the job run, or `nil` if an error occurs.
    func getJobRun(glueClient: GlueClient, name jobName: String, id: String)
async -> GlueClientTypes.JobRun? {
        do {
            let output = try await glueClient.getJobRun(
                input: GetJobRunInput(
                    jobName: jobName,
                    runId: id
                )
            )

            return output.jobRun
        } catch {
            return nil
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous GetJobRunà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation **`GetJobRuns`** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `GetJobRuns`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Principes de base

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Get information about all AWS Glue runs of a specific job.
    /// </summary>
    /// <param name="jobName">The name of the job.</param>
    /// <returns>A list of JobRun objects.</returns>
    public async Task<List<JobRun>> GetJobRunsAsync(string jobName)
    {
        var jobRuns = new List<JobRun>();

        var request = new GetJobRunsRequest
        {
            JobName = jobName,
        };

        // No need to loop to get all the log groups--the SDK does it for us
behind the scenes
        var paginatorForJobRuns =
            _amazonGlue.Paginators.GetJobRuns(request);

        await foreach (var response in paginatorForJobRuns.Responses)
        {
            response.JobRuns.ForEach(jobRun =>
            {
                jobRuns.Add(jobRun);
            });
        }

        return jobRuns;
    }
```

- Pour plus de détails sur l'API, reportez-vous GetJobRunsà la section Référence des AWS SDK pour .NET API.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region in which the bucket was created
(overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::Glue::GlueClient client(clientConfig);

        Aws::Glue::Model::GetJobRunsRequest getJobRunsRequest;
        getJobRunsRequest.SetJobName(jobName);

        Aws::String nextToken; // Used for pagination.
        std::vector<Aws::Glue::Model::JobRun> allJobRuns;
        do {
            if (!nextToken.empty()) {
                getJobRunsRequest.SetNextToken(nextToken);
            }
            Aws::Glue::Model::GetJobRunsOutcome jobRunsOutcome =
client.GetJobRuns(
                    getJobRunsRequest);

            if (jobRunsOutcome.IsSuccess()) {
                const std::vector<Aws::Glue::Model::JobRun> &jobRuns =
jobRunsOutcome.GetResult().GetJobRuns();
                allJobRuns.insert(allJobRuns.end(), jobRuns.begin(),
jobRuns.end());

                nextToken = jobRunsOutcome.GetResult().GetNextToken();
            }
```

```
            else {
                std::cerr << "Error getting job runs. "
                          << jobRunsOutcome.GetError().GetMessage()
                          << std::endl;
                break;
            }
        } while (!nextToken.empty());
```

- Pour plus de détails sur l'API, reportez-vous GetJobRunsà la section Référence des AWS SDK pour C++ API.

CLI

AWS CLI

Pour obtenir les informations sur toutes les exécutions d'une tâche

L'exemple `get-job-runs` suivant récupère les informations sur toutes les exécutions d'une tâche.

```
aws glue get-job-runs \
    --job-name "my-testing-job"
```

Sortie :

```
{
    "JobRuns": [
        {
            "Id":
 "jr_012e176506505074d94d761755e5c62538ee1aad6f17d39f527e9140cf0c9a5e",
            "Attempt": 0,
            "JobName": "my-testing-job",
            "StartedOn": 1602873931.255,
            "LastModifiedOn": 1602874075.985,
            "CompletedOn": 1602874075.985,
            "JobRunState": "SUCCEEDED",
            "Arguments": {
                "--enable-continuous-cloudwatch-log": "true",
                "--enable-metrics": "",
                "--enable-spark-ui": "true",
                "--job-bookmark-option": "job-bookmark-enable",
```

```
                "--spark-event-logs-path": "s3://aws-glue-assets-111122223333-us-
east-1/sparkHistoryLogs/"
            },
            "PredecessorRuns": [],
            "AllocatedCapacity": 10,
            "ExecutionTime": 117,
            "Timeout": 2880,
            "MaxCapacity": 10.0,
            "WorkerType": "G.1X",
            "NumberOfWorkers": 10,
            "LogGroupName": "/aws-glue/jobs",
            "GlueVersion": "2.0"
        },
        {
            "Id":
 "jr_03cc19ddab11c4e244d3f735567de74ff93b0b3ef468a713ffe73e53d1aec08f_attempt_2",
            "Attempt": 2,
            "PreviousRunId":
 "jr_03cc19ddab11c4e244d3f735567de74ff93b0b3ef468a713ffe73e53d1aec08f_attempt_1",
            "JobName": "my-testing-job",
            "StartedOn": 1602811168.496,
            "LastModifiedOn": 1602811282.39,
            "CompletedOn": 1602811282.39,
            "JobRunState": "FAILED",
            "ErrorMessage": "An error occurred while calling
 o122.pyWriteDynamicFrame.
                Access Denied (Service: Amazon S3; Status Code: 403; Error Code:
 AccessDenied;
                Request ID: 021AAB703DB20A2D;
                S3 Extended Request ID: teZk24Y09TkXzBvMPG502L5VJBhe9DJuWA9/
TXtuGOqfByajkfL/Tlqt5JBGdEGpigAqzdMDM/U=)",
            "PredecessorRuns": [],
            "AllocatedCapacity": 10,
            "ExecutionTime": 110,
            "Timeout": 2880,
            "MaxCapacity": 10.0,
            "WorkerType": "G.1X",
            "NumberOfWorkers": 10,
            "LogGroupName": "/aws-glue/jobs",
            "GlueVersion": "2.0"
        },
        {
            "Id":
 "jr_03cc19ddab11c4e244d3f735567de74ff93b0b3ef468a713ffe73e53d1aec08f_attempt_1",
```

```
            "Attempt": 1,
            "PreviousRunId":
 "jr_03cc19ddab11c4e244d3f735567de74ff93b0b3ef468a713ffe73e53d1aec08f",
            "JobName": "my-testing-job",
            "StartedOn": 1602811020.518,
            "LastModifiedOn": 1602811138.364,
            "CompletedOn": 1602811138.364,
            "JobRunState": "FAILED",
            "ErrorMessage": "An error occurred while calling
 o122.pyWriteDynamicFrame.
                Access Denied (Service: Amazon S3; Status Code: 403; Error Code:
 AccessDenied;
                Request ID: 2671D37856AE7ABB;
                S3 Extended Request ID: RLJCJw20brV
+PpC6GpORahyF2fp9flB5SSb2bTGPnUSPVizLXRl1PN3QZldb+v1o9qRVktNYbW8=)",
            "PredecessorRuns": [],
            "AllocatedCapacity": 10,
            "ExecutionTime": 113,
            "Timeout": 2880,
            "MaxCapacity": 10.0,
            "WorkerType": "G.1X",
            "NumberOfWorkers": 10,
            "LogGroupName": "/aws-glue/jobs",
            "GlueVersion": "2.0"
        }
    ]
}
```

Pour plus d'informations, consultez Exécutions de tâches dans le Guide du développeur AWS Glue.

- Pour plus de détails sur l'API, reportez-vous GetJobRunsà la section Référence des AWS CLI commandes.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
   /**
    * Retrieves the job runs for a given Glue job and prints the status of the
job runs.
    *
    * @param glueClient the Glue client used to make API calls
    * @param jobName    the name of the Glue job to retrieve the job runs for
    */
   public static void getJobRuns(GlueClient glueClient, String jobName) {
       try {
           GetJobRunsRequest runsRequest = GetJobRunsRequest.builder()
               .jobName(jobName)
               .maxResults(20)
               .build();

           boolean jobDone = false;
           while (!jobDone) {
               GetJobRunsResponse response = glueClient.getJobRuns(runsRequest);
               List<JobRun> jobRuns = response.jobRuns();
               for (JobRun jobRun : jobRuns) {
                   String jobState = jobRun.jobRunState().name();
                   if (jobState.compareTo("SUCCEEDED") == 0) {
                       System.out.println(jobName + " has succeeded");
                       jobDone = true;

                   } else if (jobState.compareTo("STOPPED") == 0) {
                       System.out.println("Job run has stopped");
                       jobDone = true;

                   } else if (jobState.compareTo("FAILED") == 0) {
                       System.out.println("Job run has failed");
                       jobDone = true;

                   } else if (jobState.compareTo("TIMEOUT") == 0) {
                       System.out.println("Job run has timed out");
                       jobDone = true;

                   } else {
                       System.out.println("*** Job run state is " +
jobRun.jobRunState().name());
                       System.out.println("Job run Id is " + jobRun.id());
                       System.out.println("The Glue version is " +
jobRun.glueVersion());
                   }
```

```
                    TimeUnit.SECONDS.sleep(5);
                }
            }

        } catch (GlueException e) {
            throw e;
        } catch (InterruptedException e) {
            throw new RuntimeException(e);
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous GetJobRunsà la section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
const getJobRuns = (jobName) => {
  const client = new GlueClient({});
  const command = new GetJobRunsCommand({
    JobName: jobName,
  });

  return client.send(command);
};
```

- Pour plus de détails sur l'API, reportez-vous GetJobRunsà la section Référence des AWS SDK pour JavaScript API.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
        $jobName = 'test-job-' . $uniqid;

        $jobRuns = $glueService->getJobRuns($jobName);

    public function getJobRuns($jobName, $maxResults = 0, $nextToken = ''):
Result
    {
        $arguments = ['JobName' => $jobName];
        if ($maxResults) {
            $arguments['MaxResults'] = $maxResults;
        }
        if ($nextToken) {
            $arguments['NextToken'] = $nextToken;
        }
        return $this->glueClient->getJobRuns($arguments);
    }
```

- Pour plus de détails sur l'API, reportez-vous GetJobRunsà la section Référence des AWS SDK pour PHP API.

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client


    def get_job_runs(self, job_name):
        """
        Gets information about runs that have been performed for a specific job
        definition.

        :param job_name: The name of the job definition to look up.
        :return: The list of job runs.
        """
        try:
            response = self.glue_client.get_job_runs(JobName=job_name)
        except ClientError as err:
            logger.error(
                "Couldn't get job runs for %s. Here's why: %s: %s",
                job_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response["JobRuns"]
```

- Pour plus de détails sur l'API, consultez GetJobRunsle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Retrieves a list of job runs for the specified job.
  #
  # @param job_name [String] The name of the job to retrieve job runs for.
  # @return [Array<Aws::Glue::Types::JobRun>]
  def get_job_runs(job_name)
    response = @glue_client.get_job_runs(job_name: job_name)
    response.job_runs
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not get job runs: \n#{e.message}")
  end
```

- Pour plus de détails sur l'API, reportez-vous GetJobRunsà la section Référence des AWS SDK pour Ruby API.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
TRY.
    " iv_job_name = 'my-etl-job'
    oo_result = lo_glu->getjobruns( iv_jobname = iv_job_name ).
    DATA(lt_job_runs) = oo_result->get_jobruns( ).
    MESSAGE 'Job runs retrieved successfully.' TYPE 'I'.
  CATCH /aws1/cx_gluentitynotfoundex.
    MESSAGE 'Job does not exist.' TYPE 'E'.
  CATCH /aws1/cx_gluinvalidinputex INTO DATA(lo_invalid_ex).
    DATA(lv_invalid_error) = lo_invalid_ex->if_message~get_longtext( ).
    MESSAGE lv_invalid_error TYPE 'E'.
  CATCH /aws1/cx_gluinternalserviceex INTO DATA(lo_internal_ex).
    DATA(lv_internal_error) = lo_internal_ex->if_message~get_longtext( ).
    MESSAGE lv_internal_error TYPE 'E'.
  CATCH /aws1/cx_gluoperationtimeoutex INTO DATA(lo_timeout_ex).
    DATA(lv_timeout_error) = lo_timeout_ex->if_message~get_longtext( ).
    MESSAGE lv_timeout_error TYPE 'E'.
ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous GetJobRunsà la section de référence du AWS SDK pour l'API SAP ABAP.

Swift

Kit SDK pour Swift

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import AWSClientRuntime
import AWSGlue

    /// Return a list of the job runs for the specified job.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - jobName: The name of the job for which to return its job runs.
    ///   - maxResults: The maximum number of job runs to return (default:
    ///     1000).
    ///
    /// - Returns: An array of `GlueClientTypes.JobRun` objects describing
    ///   each job run.
    func getJobRuns(glueClient: GlueClient, name jobName: String, maxResults:
 Int? = nil) async -> [GlueClientTypes.JobRun] {
        do {
            let output = try await glueClient.getJobRuns(
                input: GetJobRunsInput(
                    jobName: jobName,
                    maxResults: maxResults
                )
            )

            guard let jobRuns = output.jobRuns else {
                print("*** No job runs found.")
                return []
            }

            return jobRuns
        } catch is EntityNotFoundException {
            print("*** The specified job name, \(jobName), doesn't exist.")
            return []
        } catch {
            print("*** Unexpected error getting job runs:")
            dump(error)
            return []
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous GetJobRunsà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation **`GetTables`** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `GetTables`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Principes de base

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
/// <summary>
/// Get a list of tables for an AWS Glue database.
/// </summary>
/// <param name="dbName">The name of the database.</param>
/// <returns>A list of Table objects.</returns>
public async Task<List<Table>> GetTablesAsync(string dbName)
{
    var request = new GetTablesRequest { DatabaseName = dbName };
    var tables = new List<Table>();

    // Get a paginator for listing the tables.
    var tablePaginator = _amazonGlue.Paginators.GetTables(request);

    await foreach (var response in tablePaginator.Responses)
    {
        tables.AddRange(response.TableList);
    }
```

```
        return tables;
    }
```

- Pour plus de détails sur l'API, reportez-vous GetTablesà la section Référence des AWS SDK pour .NET API.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region in which the bucket was created
(overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::Glue::GlueClient client(clientConfig);

        Aws::Glue::Model::GetTablesRequest request;
        request.SetDatabaseName(CRAWLER_DATABASE_NAME);
        std::vector<Aws::Glue::Model::Table> all_tables;
        Aws::String nextToken; // Used for pagination.
        do {
            Aws::Glue::Model::GetTablesOutcome outcome =
client.GetTables(request);

            if (outcome.IsSuccess()) {
                const std::vector<Aws::Glue::Model::Table> &tables =
outcome.GetResult().GetTableList();
                all_tables.insert(all_tables.end(), tables.begin(),
tables.end());
                nextToken = outcome.GetResult().GetNextToken();
            }
```

```
            else {
                std::cerr << "Error getting the tables. "
                          << outcome.GetError().GetMessage()
                          << std::endl;
                deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, "", bucketName,
                             clientConfig);
                return false;
            }
        } while (!nextToken.empty());

        std::cout << "The database contains " << all_tables.size()
                  << (all_tables.size() == 1 ?
                      " table." : "tables.") << std::endl;
        std::cout << "Here is a list of the tables in the database.";
        for (size_t index = 0; index < all_tables.size(); ++index) {
            std::cout << "    " << index + 1 << ":  " <<
all_tables[index].GetName()
                      << std::endl;
        }

        if (!all_tables.empty()) {
            int tableIndex = askQuestionForIntRange(
                    "Enter an index to display the database detail ",
                    1, static_cast<int>(all_tables.size()));
            std::cout << all_tables[tableIndex -
1].Jsonize().View().WriteReadable()
                      << std::endl;

            tableName = all_tables[tableIndex - 1].GetName();
        }
```

- Pour plus de détails sur l'API, reportez-vous GetTablesà la section Référence des AWS SDK pour C++ API.

CLI

AWS CLI

Pour répertorier les définitions de tout ou partie des tables dans la base de données spécifiée

L'exemple `get-tables` suivant renvoie les informations sur les tables de la base de données spécifiée.

```
aws glue get-tables --database-name 'tempdb'
```

Sortie :


```
{
    "TableList": [
        {
            "Name": "my-s3-sink",
            "DatabaseName": "tempdb",
            "CreateTime": 1602730539.0,
            "UpdateTime": 1602730539.0,
            "Retention": 0,
            "StorageDescriptor": {
                "Columns": [
                    {
                        "Name": "sensorid",
                        "Type": "int"
                    },
                    {
                        "Name": "currenttemperature",
                        "Type": "int"
                    },
                    {
                        "Name": "status",
                        "Type": "string"
                    }
                ],
                "Location": "s3://janetst-bucket-01/test-s3-output/",
                "Compressed": false,
                "NumberOfBuckets": 0,
                "SerdeInfo": {
                    "SerializationLibrary": "org.openx.data.jsonserde.JsonSerDe"
                },
                "SortColumns": [],
                "StoredAsSubDirectories": false
            },
            "Parameters": {
                "classification": "json"
            },
            "CreatedBy": "arn:aws:iam::007436865787:user/JRSTERN",
            "IsRegisteredWithLakeFormation": false,
            "CatalogId": "007436865787"
        },
```

```
        {
            "Name": "s3-source",
            "DatabaseName": "tempdb",
            "CreateTime": 1602730658.0,
            "UpdateTime": 1602730658.0,
            "Retention": 0,
            "StorageDescriptor": {
                "Columns": [
                    {
                        "Name": "sensorid",
                        "Type": "int"
                    },
                    {
                        "Name": "currenttemperature",
                        "Type": "int"
                    },
                    {
                        "Name": "status",
                        "Type": "string"
                    }
                ],
                "Location": "s3://janetst-bucket-01/",
                "Compressed": false,
                "NumberOfBuckets": 0,
                "SortColumns": [],
                "StoredAsSubDirectories": false
            },
            "Parameters": {
                "classification": "json"
            },
            "CreatedBy": "arn:aws:iam::007436865787:user/JRSTERN",
            "IsRegisteredWithLakeFormation": false,
            "CatalogId": "007436865787"
        },
        {
            "Name": "test-kinesis-input",
            "DatabaseName": "tempdb",
            "CreateTime": 1601507001.0,
            "UpdateTime": 1601507001.0,
            "Retention": 0,
            "StorageDescriptor": {
                "Columns": [
                    {
                        "Name": "sensorid",
```

```
                        "Type": "int"
                    },
                    {
                        "Name": "currenttemperature",
                        "Type": "int"
                    },
                    {
                        "Name": "status",
                        "Type": "string"
                    }
                ],
                "Location": "my-testing-stream",
                "Compressed": false,
                "NumberOfBuckets": 0,
                "SerdeInfo": {
                    "SerializationLibrary": "org.openx.data.jsonserde.JsonSerDe"
                },
                "SortColumns": [],
                "Parameters": {
                    "kinesisUrl": "https://kinesis.us-east-1.amazonaws.com",
                    "streamName": "my-testing-stream",
                    "typeOfData": "kinesis"
                },
                "StoredAsSubDirectories": false
            },
            "Parameters": {
                "classification": "json"
            },
            "CreatedBy": "arn:aws:iam::007436865787:user/JRSTERN",
            "IsRegisteredWithLakeFormation": false,
            "CatalogId": "007436865787"
        }
    ]
}
```

Pour plus d'informations, consultez la section Définition des tables dans le catalogue de données AWS Glue du AWS Glue Developer Guide.

- Pour plus de détails sur l'API, reportez-vous GetTablesà la section Référence des AWS CLI commandes.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
   /**
    * Retrieves the names of the tables in the specified Glue database.
    *
    * @param glueClient the Glue client to use for the operation
    * @param dbName     the name of the Glue database to retrieve the table
names from
    * @return the name of the first table retrieved, or an empty string if no
tables were found
    */
   public static String getGlueTables(GlueClient glueClient, String dbName) {
       String myTableName = "";
       try {
           GetTablesRequest tableRequest = GetTablesRequest.builder()
               .databaseName(dbName)
               .build();

           GetTablesResponse response = glueClient.getTables(tableRequest);
           List<Table> tables = response.tableList();
           if (tables.isEmpty()) {
               System.out.println("No tables were returned");
           } else {
               for (Table table : tables) {
                   myTableName = table.name();
                   System.out.println("Table name is: " + myTableName);
               }
           }

       } catch (GlueException e) {
           throw e;
       }
       return myTableName;
```

```
    }
```

- Pour plus de détails sur l'API, reportez-vous GetTablesà la section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
const getTables = (databaseName) => {
  const client = new GlueClient({});

  const command = new GetTablesCommand({
    DatabaseName: databaseName,
  });

  return client.send(command);
};
```

- Pour plus de détails sur l'API, reportez-vous GetTablesà la section Référence des AWS SDK pour JavaScript API.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
        $databaseName = "doc-example-database-$uniqid";

        $tables = $glueService->getTables($databaseName);

    public function getTables($databaseName): Result
    {
        return $this->glueClient->getTables([
            'DatabaseName' => $databaseName,
        ]);
    }
```

- Pour plus de détails sur l'API, reportez-vous GetTablesà la section Référence des AWS SDK pour PHP API.

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client


    def get_tables(self, db_name):
        """
        Gets a list of tables in a Data Catalog database.

        :param db_name: The name of the database to query.
        :return: The list of tables in the database.
```

```
        """
        try:
            response = self.glue_client.get_tables(DatabaseName=db_name)
        except ClientError as err:
            logger.error(
                "Couldn't get tables %s. Here's why: %s: %s",
                db_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response["TableList"]
```

- Pour plus de détails sur l'API, consultez GetTablesle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end
```

```
  # Retrieves a list of tables in the specified database.
  #
  # @param db_name [String] The name of the database to retrieve tables from.
  # @return [Array<Aws::Glue::Types::Table>]
  def get_tables(db_name)
    response = @glue_client.get_tables(database_name: db_name)
    response.table_list
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not get tables #{db_name}: \n#{e.message}")
    raise
  end
```

- Pour plus de détails sur l'API, reportez-vous GetTablesà la section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
        let tables = glue
            .get_tables()
            .database_name(self.database())
            .send()
            .await
            .map_err(GlueMvpError::from_glue_sdk)?;

        let tables = tables.table_list();
```

- Pour plus de détails sur l'API, voir GetTablesla section de référence de l'API AWS SDK for Rust.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
TRY.
    " iv_database_name = 'my-database'
    oo_result = lo_glu->gettables( iv_databasename = iv_database_name ).
    DATA(lt_tables) = oo_result->get_tablelist( ).
    MESSAGE 'Tables retrieved successfully.' TYPE 'I'.
  CATCH /aws1/cx_gluentitynotfoundex.
    MESSAGE 'Database does not exist.' TYPE 'E'.
  CATCH /aws1/cx_gluinvalidinputex INTO DATA(lo_invalid_ex).
    DATA(lv_invalid_error) = lo_invalid_ex->if_message~get_longtext( ).
    MESSAGE lv_invalid_error TYPE 'E'.
  CATCH /aws1/cx_gluinternalserviceex INTO DATA(lo_internal_ex).
    DATA(lv_internal_error) = lo_internal_ex->if_message~get_longtext( ).
    MESSAGE lv_internal_error TYPE 'E'.
  CATCH /aws1/cx_gluoperationtimeoutex INTO DATA(lo_timeout_ex).
    DATA(lv_timeout_error) = lo_timeout_ex->if_message~get_longtext( ).
    MESSAGE lv_timeout_error TYPE 'E'.
ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous GetTablesà la section de référence du AWS SDK pour l'API SAP ABAP.

Swift

Kit SDK pour Swift

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import AWSClientRuntime
import AWSGlue

    /// Returns a list of the tables in the specified database.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - databaseName: The name of the database whose tables are to be
    ///     returned.
    ///
    /// - Returns: An array of `GlueClientTypes.Table` objects, each
    ///   describing one table in the named database. An empty array indicates
    ///   that there are either no tables in the database, or an error
    ///   occurred before any tables could be found.
    func getTablesInDatabase(glueClient: GlueClient, databaseName: String) async
 -> [GlueClientTypes.Table] {
        var tables: [GlueClientTypes.Table] = []
        var nextToken: String?

        repeat {
            do {
                let output = try await glueClient.getTables(
                    input: GetTablesInput(
                        databaseName: databaseName,
                        nextToken: nextToken
                    )
                )

                guard let tableList = output.tableList else {
                    return tables
                }

                tables = tables + tableList
                nextToken = output.nextToken
            } catch {
                return tables
            }
        } while nextToken != nil

        return tables
    }
```

- Pour plus de détails sur l'API, reportez-vous GetTablesà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation **`ListJobs`** avec un AWS SDK

Les exemples de code suivants illustrent comment utiliser `ListJobs`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Principes de base

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// List AWS Glue jobs using a paginator.
    /// </summary>
    /// <returns>A list of AWS Glue job names.</returns>
    public async Task<List<string>> ListJobsAsync()
    {
        var jobNames = new List<string>();

        var listJobsPaginator = _amazonGlue.Paginators.ListJobs(new
ListJobsRequest { MaxResults = 10 });
        await foreach (var response in listJobsPaginator.Responses)
        {
```

```
            jobNames.AddRange(response.JobNames);
        }

        return jobNames;
    }
```

- Pour plus de détails sur l'API, reportez-vous ListJobsà la section Référence des AWS SDK pour .NET API.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
       Aws::Client::ClientConfiguration clientConfig;
       // Optional: Set to the AWS Region in which the bucket was created
(overrides config file).
       // clientConfig.region = "us-east-1";

   Aws::Glue::GlueClient client(clientConfig);

       Aws::Glue::Model::ListJobsRequest listJobsRequest;

       Aws::String nextToken;
       std::vector<Aws::String> allJobNames;

       do {
           if (!nextToken.empty()) {
               listJobsRequest.SetNextToken(nextToken);
           }
           Aws::Glue::Model::ListJobsOutcome listRunsOutcome = client.ListJobs(
                   listJobsRequest);

           if (listRunsOutcome.IsSuccess()) {
```

```
                const std::vector<Aws::String> &jobNames =
 listRunsOutcome.GetResult().GetJobNames();
                allJobNames.insert(allJobNames.end(), jobNames.begin(),
 jobNames.end());
                nextToken = listRunsOutcome.GetResult().GetNextToken();
            }
            else {
                std::cerr << "Error listing jobs. "
                          << listRunsOutcome.GetError().GetMessage()
                          << std::endl;
            }
        } while (!nextToken.empty());
```

- Pour plus de détails sur l'API, reportez-vous ListJobsà la section Référence des AWS SDK pour C++ API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
const listJobs = () => {
  const client = new GlueClient({});

  const command = new ListJobsCommand({});

  return client.send(command);
};
```

- Pour plus de détails sur l'API, reportez-vous ListJobsà la section Référence des AWS SDK pour JavaScript API.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
        $jobs = $glueService->listJobs();
        echo "Current jobs:\n";
        foreach ($jobs['JobNames'] as $jobsName) {
            echo "{$jobsName}\n";
        }

    public function listJobs($maxResults = null, $nextToken = null, $tags = []):
Result
    {
        $arguments = [];
        if ($maxResults) {
            $arguments['MaxResults'] = $maxResults;
        }
        if ($nextToken) {
            $arguments['NextToken'] = $nextToken;
        }
        if (!empty($tags)) {
            $arguments['Tags'] = $tags;
        }
        return $this->glueClient->listJobs($arguments);
    }
```

- Pour plus de détails sur l'API, reportez-vous ListJobsà la section Référence des AWS SDK pour PHP API.

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client


    def list_jobs(self):
        """
        Lists the names of job definitions in your account.

        :return: The list of job definition names.
        """
        try:
            response = self.glue_client.list_jobs()
        except ClientError as err:
            logger.error(
                "Couldn't list jobs. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response["JobNames"]
```

- Pour plus de détails sur l'API, consultez ListJobsle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Retrieves a list of jobs in AWS Glue.
  #
  # @return [Aws::Glue::Types::ListJobsResponse]
  def list_jobs
    @glue_client.list_jobs
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not list jobs: \n#{e.message}")
    raise
  end
```

- Pour plus de détails sur l'API, reportez-vous ListJobsà la section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
        let mut list_jobs = glue.list_jobs().into_paginator().send();
        while let Some(list_jobs_output) = list_jobs.next().await {
            match list_jobs_output {
                Ok(list_jobs) => {
                    let names = list_jobs.job_names();
                    info!(?names, "Found these jobs")
                }
                Err(err) => return Err(GlueMvpError::from_glue_sdk(err)),
            }
        }
```

- Pour plus de détails sur l'API, voir ListJobsla section de référence de l'API AWS SDK for Rust.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    TRY.
        oo_result = lo_glu->listjobs( ).
        DATA(lt_job_names) = oo_result->get_jobnames( ).
        MESSAGE 'Job list retrieved successfully.' TYPE 'I'.
      CATCH /aws1/cx_gluentitynotfoundex.
```

```
        MESSAGE 'No jobs found.' TYPE 'I'.
      CATCH /aws1/cx_gluinvalidinputex INTO DATA(lo_invalid_ex).
        DATA(lv_invalid_error) = lo_invalid_ex->if_message~get_longtext( ).
        MESSAGE lv_invalid_error TYPE 'E'.
      CATCH /aws1/cx_gluinternalserviceex INTO DATA(lo_internal_ex).
        DATA(lv_internal_error) = lo_internal_ex->if_message~get_longtext( ).
        MESSAGE lv_internal_error TYPE 'E'.
      CATCH /aws1/cx_gluoperationtimeoutex INTO DATA(lo_timeout_ex).
        DATA(lv_timeout_error) = lo_timeout_ex->if_message~get_longtext( ).
        MESSAGE lv_timeout_error TYPE 'E'.
    ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous ListJobsà la section de référence du AWS SDK pour l'API SAP ABAP.

Swift

Kit SDK pour Swift

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import AWSClientRuntime
import AWSGlue

    /// Return a list of the AWS Glue jobs listed on the user's account.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - maxJobs: The maximum number of jobs to return (default: 100).
    ///
    /// - Returns: An array of strings listing the names of all available AWS
    ///   Glue jobs.
    func listJobs(glueClient: GlueClient, maxJobs: Int = 100) async -> [String] {
        var jobList: [String] = []
        var nextToken: String?
```

```
        repeat {
            do {
                let output = try await glueClient.listJobs(
                    input: ListJobsInput(
                        maxResults: maxJobs,
                        nextToken: nextToken
                    )
                )

                guard let jobs = output.jobNames else {
                    return jobList
                }

                jobList = jobList + jobs
                nextToken = output.nextToken
            } catch {
                return jobList
            }
        } while (nextToken != nil)

        return jobList
    }
```

- Pour plus de détails sur l'API, reportez-vous ListJobsà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation **`StartCrawler`** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `StartCrawler`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Principes de base

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
/// <summary>
/// Start an AWS Glue crawler.
/// </summary>
/// <param name="crawlerName">The name of the crawler.</param>
/// <returns>A Boolean value indicating the success of the action.</returns>
public async Task<bool> StartCrawlerAsync(string crawlerName)
{
    var crawlerRequest = new StartCrawlerRequest
    {
        Name = crawlerName,
    };

    var response = await _amazonGlue.StartCrawlerAsync(crawlerRequest);

    return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
}
```

- Pour plus de détails sur l'API, reportez-vous StartCrawlerà la section Référence des AWS SDK pour .NET API.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
       Aws::Client::ClientConfiguration clientConfig;
       // Optional: Set to the AWS Region in which the bucket was created
(overrides config file).
       // clientConfig.region = "us-east-1";

   Aws::Glue::GlueClient client(clientConfig);

       Aws::Glue::Model::StartCrawlerRequest request;
       request.SetName(CRAWLER_NAME);

       Aws::Glue::Model::StartCrawlerOutcome outcome =
client.StartCrawler(request);


       if (outcome.IsSuccess() || (Aws::Glue::GlueErrors::CRAWLER_RUNNING ==
                                   outcome.GetError().GetErrorType())) {
           if (!outcome.IsSuccess()) {
               std::cout << "Crawler was already started." << std::endl;
           }
           else {
               std::cout << "Successfully started crawler." << std::endl;
           }

           std::cout << "This may take a while to run." << std::endl;

           Aws::Glue::Model::CrawlerState crawlerState =
Aws::Glue::Model::CrawlerState::NOT_SET;
           int iterations = 0;
           while (Aws::Glue::Model::CrawlerState::READY != crawlerState) {
               std::this_thread::sleep_for(std::chrono::seconds(1));
               ++iterations;
               if ((iterations % 10) == 0) { // Log status every 10 seconds.
                   std::cout << "Crawler status " <<

Aws::Glue::Model::CrawlerStateMapper::GetNameForCrawlerState(
                                       crawlerState)
                             << ". After " << iterations
                             << " seconds elapsed."
                             << std::endl;
               }
               Aws::Glue::Model::GetCrawlerRequest getCrawlerRequest;
               getCrawlerRequest.SetName(CRAWLER_NAME);
```

```
                Aws::Glue::Model::GetCrawlerOutcome getCrawlerOutcome =
client.GetCrawler(
                        getCrawlerRequest);

                if (getCrawlerOutcome.IsSuccess()) {
                    crawlerState =
getCrawlerOutcome.GetResult().GetCrawler().GetState();
                }
                else {
                    std::cerr << "Error getting crawler.  "
                              << getCrawlerOutcome.GetError().GetMessage() <<
std::endl;
                    break;
                }
            }

            if (Aws::Glue::Model::CrawlerState::READY == crawlerState) {
                std::cout << "Crawler finished running after " << iterations
                          << " seconds."
                          << std::endl;
            }
        }
        else {
            std::cerr << "Error starting a crawler.  "
                      << outcome.GetError().GetMessage()
                      << std::endl;

            deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, "", bucketName,
                         clientConfig);
            return false;
        }
```

- Pour plus de détails sur l'API, reportez-vous StartCrawlerà la section Référence des AWS SDK pour C++ API.

CLI

AWS CLI

Pour démarrer un robot

L'exemple `start-crawler` suivant démarre un crawler.

```
aws glue start-crawler --name my-crawler
```

Sortie :

```
None
```

Pour plus d'informations, consultez Définition des robots dans le Guide du développeur AWS Glue.

- Pour plus de détails sur l'API, reportez-vous StartCrawlerà la section Référence des AWS CLI commandes.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /**
     * Starts a specific AWS Glue crawler.
     *
     * @param glueClient  the AWS Glue client to use for the crawler operation
     * @param crawlerName the name of the crawler to start
     * @throws GlueException if there is an error starting the crawler
     */
    public static void startSpecificCrawler(GlueClient glueClient, String
crawlerName) {
        try {
            StartCrawlerRequest crawlerRequest = StartCrawlerRequest.builder()
                .name(crawlerName)
                .build();

            glueClient.startCrawler(crawlerRequest);
            System.out.println(crawlerName + " was successfully started!");

        } catch (GlueException e) {
            throw e;
```

```
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous StartCrawlerà la section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
const startCrawler = (name) => {
  const client = new GlueClient({});

  const command = new StartCrawlerCommand({
    Name: name,
  });

  return client.send(command);
};
```

- Pour plus de détails sur l'API, reportez-vous StartCrawlerà la section Référence des AWS SDK pour JavaScript API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun startSpecificCrawler(crawlerName: String?) {
    val request =
        StartCrawlerRequest {
            name = crawlerName
        }

    GlueClient.fromEnvironment { region = "us-west-2" }.use { glueClient ->
        glueClient.startCrawler(request)
        println("$crawlerName was successfully started.")
    }
}
```

- Pour plus de détails sur l'API, consultez StartCrawlerla section AWS SDK pour la référence de l'API Kotlin.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
        $crawlerName = "example-crawler-test-" . $uniqid;

        $databaseName = "doc-example-database-$uniqid";

        $glueService->startCrawler($crawlerName);

    public function startCrawler($crawlerName): Result
    {
        return $this->glueClient->startCrawler([
            'Name' => $crawlerName,
        ]);
    }
```

- Pour plus de détails sur l'API, reportez-vous StartCrawlerà la section Référence des AWS SDK pour PHP API.

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client


    def start_crawler(self, name):
        """
        Starts a crawler. The crawler crawls its configured target and creates
        metadata that describes the data it finds in the target data source.

        :param name: The name of the crawler to start.
        """
        try:
            self.glue_client.start_crawler(Name=name)
        except ClientError as err:
            logger.error(
                "Couldn't start crawler %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- Pour plus de détails sur l'API, consultez StartCrawlerle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Starts a crawler with the specified name.
  #
  # @param name [String] The name of the crawler to start.
  # @return [void]
  def start_crawler(name)
    @glue_client.start_crawler(name: name)
  rescue Aws::Glue::Errors::ServiceError => e
    @logger.error("Glue could not start crawler #{name}: \n#{e.message}")
    raise
  end
```

- Pour plus de détails sur l'API, reportez-vous StartCrawlerà la section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
        let start_crawler =
glue.start_crawler().name(self.crawler()).send().await;

        match start_crawler {
            Ok(_) => Ok(()),
            Err(err) => {
                let glue_err: aws_sdk_glue::Error = err.into();
                match glue_err {
                    aws_sdk_glue::Error::CrawlerRunningException(_) => Ok(()),
                    _ => Err(GlueMvpError::GlueSdk(glue_err)),
                }
            }
        }?;
```

- Pour plus de détails sur l'API, voir StartCrawlerla section de référence de l'API AWS SDK for Rust.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    TRY.
        " iv_crawler_name = 'my-crawler'
        lo_glu->startcrawler( iv_name = iv_crawler_name ).
        MESSAGE 'Crawler started successfully.' TYPE 'I'.
      CATCH /aws1/cx_glucrawlerrunningex.
        MESSAGE 'Crawler is already running.' TYPE 'I'.
      CATCH /aws1/cx_gluentitynotfoundex.
        MESSAGE 'Crawler does not exist.' TYPE 'E'.
      CATCH /aws1/cx_gluoperationtimeoutex INTO DATA(lo_timeout_ex).
        DATA(lv_timeout_error) = lo_timeout_ex->if_message~get_longtext( ).
        MESSAGE lv_timeout_error TYPE 'E'.
    ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous StartCrawlerà la section de référence du AWS SDK pour l'API SAP ABAP.

Swift

Kit SDK pour Swift

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import AWSClientRuntime
import AWSGlue

    /// Start running an AWS Glue crawler.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use when starting the crawler.
    ///   - name: The name of the crawler to start running.
    ///
    /// - Returns: `true` if the crawler is started successfully, otherwise
 `false`.
    func startCrawler(glueClient: GlueClient, name: String) async -> Bool {
        do {
            _ = try await glueClient.startCrawler(
```

```
                input: StartCrawlerInput(name: name)
            )
        } catch {
            print("*** An unexpected error occurred starting the crawler.")
            return false
        }

        return true
    }
```

- Pour plus de détails sur l'API, reportez-vous StartCrawlerà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Utilisation **`StartJobRun`** avec un AWS SDK ou une CLI

Les exemples de code suivants illustrent comment utiliser `StartJobRun`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Principes de base

.NET

SDK pour .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
```

```
    /// Start an AWS Glue job run.
    /// </summary>
    /// <param name="jobName">The name of the job.</param>
    /// <returns>A string representing the job run Id.</returns>
    public async Task<string> StartJobRunAsync(
        string jobName,
        string inputDatabase,
        string inputTable,
        string bucketName)
    {
        var request = new StartJobRunRequest
        {
            JobName = jobName,
            Arguments = new Dictionary<string, string>
            {
                {"--input_database", inputDatabase},
                {"--input_table", inputTable},
                {"--output_bucket_url", $"s3://{bucketName}/"}
            }
        };

        var response = await _amazonGlue.StartJobRunAsync(request);
        return response.JobRunId;
    }
```

- Pour plus de détails sur l'API, reportez-vous StartJobRunà la section Référence des AWS SDK pour .NET API.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
        Aws::Client::ClientConfiguration clientConfig;
```

```
       // Optional: Set to the AWS Region in which the bucket was created
(overrides config file).
       // clientConfig.region = "us-east-1";

   Aws::Glue::GlueClient client(clientConfig);

       Aws::Glue::Model::StartJobRunRequest request;
       request.SetJobName(JOB_NAME);

       Aws::Map<Aws::String, Aws::String> arguments;
       arguments["--input_database"] = CRAWLER_DATABASE_NAME;
       arguments["--input_table"] = tableName;
       arguments["--output_bucket_url"] = Aws::String("s3://") + bucketName +
"/";
       request.SetArguments(arguments);

       Aws::Glue::Model::StartJobRunOutcome outcome =
client.StartJobRun(request);

       if (outcome.IsSuccess()) {
           std::cout << "Successfully started the job." << std::endl;

           Aws::String jobRunId = outcome.GetResult().GetJobRunId();

           int iterator = 0;
           bool done = false;
           while (!done) {
               ++iterator;
               std::this_thread::sleep_for(std::chrono::seconds(1));
               Aws::Glue::Model::GetJobRunRequest jobRunRequest;
               jobRunRequest.SetJobName(JOB_NAME);
               jobRunRequest.SetRunId(jobRunId);

               Aws::Glue::Model::GetJobRunOutcome jobRunOutcome =
client.GetJobRun(
                       jobRunRequest);

               if (jobRunOutcome.IsSuccess()) {
                   const Aws::Glue::Model::JobRun &jobRun =
jobRunOutcome.GetResult().GetJobRun();
                   Aws::Glue::Model::JobRunState jobRunState =
jobRun.GetJobRunState();
```

```
                    if ((jobRunState == Aws::Glue::Model::JobRunState::STOPPED)
||
                        (jobRunState == Aws::Glue::Model::JobRunState::FAILED) ||
                        (jobRunState == Aws::Glue::Model::JobRunState::TIMEOUT))
{
                        std::cerr << "Error running job. "
                                  << jobRun.GetErrorMessage()
                                  << std::endl;
                        deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME,
JOB_NAME,
                                     bucketName,
                                     clientConfig);
                        return false;
                    }
                    else if (jobRunState ==
                             Aws::Glue::Model::JobRunState::SUCCEEDED) {
                        std::cout << "Job run succeeded after  " << iterator <<
                                  " seconds elapsed." << std::endl;
                        done = true;
                    }
                    else if ((iterator % 10) == 0) { // Log status every 10
seconds.
                        std::cout << "Job run status " <<

Aws::Glue::Model::JobRunStateMapper::GetNameForJobRunState(
                                          jobRunState) <<
                                  ". " << iterator <<
                                  " seconds elapsed." << std::endl;
                    }
                }
                else {
                    std::cerr << "Error retrieving job run state. "
                              << jobRunOutcome.GetError().GetMessage()
                              << std::endl;
                    deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, JOB_NAME,
                                 bucketName, clientConfig);
                    return false;
                }
            }
        }
        else {
            std::cerr << "Error starting a job. " <<
outcome.GetError().GetMessage()
                      << std::endl;
```

```
            deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, JOB_NAME,
bucketName,
                         clientConfig);
            return false;
        }
```

- Pour plus de détails sur l'API, reportez-vous StartJobRunà la section Référence des AWS SDK pour C++ API.

CLI

AWS CLI

Pour commencer l'exécution d'une tâche

L'exemple `start-job-run` suivant démarre une tâche.

```
aws glue start-job-run \
    --job-name my-job
```

Sortie :

```
{
    "JobRunId":
 "jr_22208b1f44eb5376a60569d4b21dd20fcb8621e1a366b4e7b2494af764b82ded"
}
```

Pour plus d'informations, consultez Création de tâches dans le Guide du développeur AWS Glue.

- Pour plus de détails sur l'API, reportez-vous StartJobRunà la section Référence des AWS CLI commandes.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
   /**
    * Starts a job run in AWS Glue.
    *
    * @param glueClient    the AWS Glue client to use for the job run
    * @param jobName       the name of the Glue job to run
    * @param inputDatabase the name of the input database
    * @param inputTable    the name of the input table
    * @param outBucket     the URL of the output S3 bucket
    * @throws GlueException if there is an error starting the job run
    */
   public static void startJob(GlueClient glueClient, String jobName, String
inputDatabase, String inputTable,
                               String outBucket) {
       try {
           Map<String, String> myMap = new HashMap<>();
           myMap.put("--input_database", inputDatabase);
           myMap.put("--input_table", inputTable);
           myMap.put("--output_bucket_url", outBucket);

           StartJobRunRequest runRequest = StartJobRunRequest.builder()
               .workerType(WorkerType.G_1_X)
               .numberOfWorkers(10)
               .arguments(myMap)
               .jobName(jobName)
               .build();

           StartJobRunResponse response = glueClient.startJobRun(runRequest);
           System.out.println("The request Id of the job is " +
response.responseMetadata().requestId());

       } catch (GlueException e) {
```

```
            throw e;
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous StartJobRunà la section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
const startJobRun = (jobName, dbName, tableName, bucketName) => {
  const client = new GlueClient({});

  const command = new StartJobRunCommand({
    JobName: jobName,
    Arguments: {
      "--input_database": dbName,
      "--input_table": tableName,
      "--output_bucket_url": `s3://${bucketName}/`,
    },
  });

  return client.send(command);
};
```

- Pour plus de détails sur l'API, reportez-vous StartJobRunà la section Référence des AWS SDK pour JavaScript API.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
        $jobName = 'test-job-' . $uniqid;

        $databaseName = "doc-example-database-$uniqid";

        $tables = $glueService->getTables($databaseName);

        $outputBucketUrl = "s3://$bucketName";
        $runId = $glueService->startJobRun($jobName, $databaseName, $tables,
$outputBucketUrl)['JobRunId'];

    public function startJobRun($jobName, $databaseName, $tables,
$outputBucketUrl): Result
    {
        return $this->glueClient->startJobRun([
            'JobName' => $jobName,
            'Arguments' => [
                'input_database' => $databaseName,
                'input_table' => $tables['TableList'][0]['Name'],
                'output_bucket_url' => $outputBucketUrl,
                '--input_database' => $databaseName,
                '--input_table' => $tables['TableList'][0]['Name'],
                '--output_bucket_url' => $outputBucketUrl,
            ],
        ]);
    }
```

- Pour plus de détails sur l'API, reportez-vous StartJobRunà la section Référence des AWS SDK pour PHP API.

Python

Kit SDK for Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client


    def start_job_run(self, name, input_database, input_table,
 output_bucket_name):
        """
        Starts a job run. A job run extracts data from the source, transforms it,
        and loads it to the output bucket.

        :param name: The name of the job definition.
        :param input_database: The name of the metadata database that contains
 tables
                               that describe the source data. This is typically
 created
                               by a crawler.
        :param input_table: The name of the table in the metadata database that
                            describes the source data.
        :param output_bucket_name: The S3 bucket where the output is written.
        :return: The ID of the job run.
        """
        try:
            # The custom Arguments that are passed to this function are used by
 the
            # Python ETL script to determine the location of input and output
 data.
```

```
            response = self.glue_client.start_job_run(
                JobName=name,
                Arguments={
                    "--input_database": input_database,
                    "--input_table": input_table,
                    "--output_bucket_url": f"s3://{output_bucket_name}/",
                },
            )
        except ClientError as err:
            logger.error(
                "Couldn't start job run %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response["JobRunId"]
```

- Pour plus de détails sur l'API, consultez StartJobRunle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
```

```
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Starts a job run for the specified job.
  #
  # @param name [String] The name of the job to start the run for.
  # @param input_database [String] The name of the input database for the job.
  # @param input_table [String] The name of the input table for the job.
  # @param output_bucket_name [String] The name of the output S3 bucket for the
 job.
  # @return [String] The ID of the started job run.
  def start_job_run(name, input_database, input_table, output_bucket_name)
    response = @glue_client.start_job_run(
      job_name: name,
      arguments: {
        '--input_database': input_database,
        '--input_table': input_table,
        '--output_bucket_url': "s3://#{output_bucket_name}/"
      }
    )
    response.job_run_id
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not start job run #{name}: \n#{e.message}")
    raise
  end
```

- Pour plus de détails sur l'API, reportez-vous StartJobRunà la section Référence des AWS SDK pour Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
        let job_run_output = glue
            .start_job_run()
            .job_name(self.job())
            .arguments("--input_database", self.database())
            .arguments(
                "--input_table",
                self.tables
                    .first()
                    .ok_or_else(|| GlueMvpError::Unknown("Missing crawler
table".into()))?
                    .name(),
            )
            .arguments("--output_bucket_url", self.bucket())
            .send()
            .await
            .map_err(GlueMvpError::from_glue_sdk)?;

        let job = job_run_output
            .job_run_id()
            .ok_or_else(|| GlueMvpError::Unknown("Missing run id from just
started job".into()))?
            .to_string();
```

- Pour plus de détails sur l'API, voir StartJobRunla section de référence de l'API AWS SDK for Rust.

SAP ABAP

Kit SDK pour SAP ABAP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    TRY.
        " iv_job_name = 'my-etl-job'
        " iv_input_database = 'my-database'
        " iv_input_table = 'my-table'
```

```
        " iv_output_bucket_url = 's3://example-output-bucket/'

        DATA lt_arguments TYPE /aws1/cl_glugenericmap_w=>tt_genericmap.
        lt_arguments = VALUE #(
          ( VALUE /aws1/cl_glugenericmap_w=>ts_genericmap_maprow(
            key = '--input_database'
            value = NEW /aws1/cl_glugenericmap_w( iv_value =
iv_input_database ) ) )
          ( VALUE /aws1/cl_glugenericmap_w=>ts_genericmap_maprow(
            key = '--input_table'
            value = NEW /aws1/cl_glugenericmap_w( iv_value = iv_input_table ) ) )
          ( VALUE /aws1/cl_glugenericmap_w=>ts_genericmap_maprow(
            key = '--output_bucket_url'
            value = NEW /aws1/cl_glugenericmap_w( iv_value =
iv_output_bucket_url ) ) ) ).

        DATA(oo_result) = lo_glu->startjobrun(
          iv_jobname = iv_job_name
          it_arguments = lt_arguments ).
        ov_job_run_id = oo_result->get_jobrunid( ).
        MESSAGE 'Job run started successfully.' TYPE 'I'.
      CATCH /aws1/cx_gluconcurrentrunsex00.
        MESSAGE 'Maximum concurrent runs exceeded.' TYPE 'E'.
      CATCH /aws1/cx_gluentitynotfoundex.
        MESSAGE 'Job does not exist.' TYPE 'E'.
      CATCH /aws1/cx_gluinvalidinputex INTO DATA(lo_invalid_ex).
        DATA(lv_invalid_error) = lo_invalid_ex->if_message~get_longtext( ).
        MESSAGE lv_invalid_error TYPE 'E'.
      CATCH /aws1/cx_gluinternalserviceex INTO DATA(lo_internal_ex).
        DATA(lv_internal_error) = lo_internal_ex->if_message~get_longtext( ).
        MESSAGE lv_internal_error TYPE 'E'.
      CATCH /aws1/cx_gluoperationtimeoutex INTO DATA(lo_timeout_ex).
        DATA(lv_timeout_error) = lo_timeout_ex->if_message~get_longtext( ).
        MESSAGE lv_timeout_error TYPE 'E'.
      CATCH /aws1/cx_gluresrcnumlmtexcdex INTO DATA(lo_limit_ex).
        DATA(lv_limit_error) = lo_limit_ex->if_message~get_longtext( ).
        MESSAGE lv_limit_error TYPE 'E'.
    ENDTRY.
```

- Pour plus de détails sur l'API, reportez-vous StartJobRunà la section de référence du AWS SDK pour l'API SAP ABAP.

Swift

Kit SDK pour Swift

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import AWSClientRuntime
import AWSGlue

    /// Start an AWS Glue job run.
    ///
    /// - Parameters:
    ///   - glueClient: The AWS Glue client to use.
    ///   - jobName: The name of the job to run.
    ///   - databaseName: The name of the AWS Glue database to run the job
 against.
    ///   - tableName: The name of the table in the database to run the job
 against.
    ///   - outputURL: The AWS S3 URI of the bucket location into which to
    ///     write the resulting output.
    ///
    /// - Returns: `true` if the job run is started successfully, otherwise
 `false`.
    func startJobRun(glueClient: GlueClient, name jobName: String, databaseName:
 String,
                     tableName: String, outputURL: String) async -> String? {
        do {
            let output = try await glueClient.startJobRun(
                input: StartJobRunInput(
                    arguments: [
                        "--input_database": databaseName,
                        "--input_table": tableName,
                        "--output_bucket_url": outputURL
                    ],
                    jobName: jobName,
                    numberOfWorkers: 10,
                    workerType: .g1x
                )
```

```
            )

            guard let id = output.jobRunId else {
                return nil
            }

            return id
        } catch {
            return nil
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous StartJobRunà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation de ce service avec un AWS SDK. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit SDK.

Sécurité dans AWS Glue

La sécurité du cloud AWS est la priorité absolue. En tant que AWS client, vous bénéficiez d'un centre de données et d'une architecture réseau conçus pour répondre aux exigences des entreprises les plus sensibles en matière de sécurité.

La sécurité est une responsabilité partagée entre vous AWS et vous. Le modèle de responsabilité partagée décrit cette notion par les termes sécurité du cloud et sécurité dans le cloud :

- Sécurité du cloud : AWS est chargée de protéger l'infrastructure qui exécute les AWS services dans le AWS cloud. AWS vous fournit également des services que vous pouvez utiliser en toute sécurité. Des auditeurs tiers testent et vérifient régulièrement l'efficacité de notre sécurité dans le cadre des programmes de conformitéAWS. Pour en savoir plus sur les programmes de conformité qui s'appliquent à AWS Glue, consultez la section AWS Services concernés par programme de conformité.
- Sécurité dans le cloud — Votre responsabilité est déterminée par le AWS service que vous utilisez. Vous êtes également responsable d'autres facteurs, y compris de la sensibilité de vos données, des exigences de votre entreprise, ainsi que de la législation et de la réglementation applicables.

Cette documentation vous aide à comprendre comment appliquer le modèle de responsabilité partagée lors de l'utilisation de AWS Glue. Les rubriques suivantes expliquent comment configurer AWS Glue pour répondre à vos objectifs de sécurité et de conformité. Vous apprendrez également à utiliser d'autres AWS services qui vous aident à surveiller et à sécuriser vos AWS Glue ressources.

Rubriques

- Protection des données dans AWS Glue
- Gestion des identités et des accès pour AWS Glue
- AWS Lake Formation modèles de contrôle d'accès
- Utilisation d'Amazon S3 Access Grants avec AWS Glue
- Propagation d'identité fiable avec AWS Glue ETL
- Connexion et surveillance AWS Glue
- Validation de conformité pour AWS Glue
- Résilience dans AWS Glue
- Sécurité de l'infrastructure dans AWS Glue

Protection des données dans AWS Glue

AWS Glue propose plusieurs fonctionnalités qui sont conçues pour vous aider à protéger vos données.

Rubriques

- Chiffrement de données au repos
- chiffrement des données en transit
- Conformité FIPS
- Gestion des clés
- AWS Gluedépendance à l'égard d'autres AWS services
- Points de terminaison de développement

Chiffrement de données au repos

AWS Glue prend en charge le chiffrement des données au repos pour Création de tâches ETL visuelles etDéveloppement de scripts à l'aide de points de terminaison de développement. Vous pouvez configurer des tâches d'extraction, de transformation et de chargement (ETL) et des points de terminaison de développement pour utiliser des clés AWS Key Management Service (AWS KMS) pour l'écriture de données chiffrées au repos. Vous pouvez également chiffrer les métadonnées stockées dans les clés AWS Glue Data Catalog d'utilisation que vous utilisez pour gérer AWS KMS. En outre, vous pouvez utiliser des AWS KMS clés pour chiffrer les signets de tâches et les journaux générés par les robots d'exploration et les tâches ETL.

Vous pouvez chiffrer des objets de métadonnées AWS Glue Data Catalog en plus des données écrites sur Amazon Simple Storage Service (Amazon S3) et CloudWatch Amazon Logs par des jobs, des robots d'exploration et des points de terminaison de développement. Lorsque vous créez des tâches, des crawlers et des points de terminaison de développement dans AWS Glue, vous pouvez fournir des paramètres de chiffrement en attachant une configuration de sécurité. Les configurations de sécurité contiennent des clés de chiffrement côté serveur (SSE-S3) gérées par Amazon S3 ou des clés principales du client () stockées dans (SSE-KMSCMKs). AWS KMS Vous pouvez créer des configurations de sécurité à l'aide de la console AWS Glue.

Vous pouvez activer le chiffrement de la totalité du catalogue de données dans votre compte. Pour ce faire, spécifiez « CMKs stocké dans AWS KMS ».

> **Important**
>
> AWS Glue ne prend en charge que les clés symétriques gérées par le client. Pour plus d'informations, voir Customer Managed Keys (CMKs) dans le Guide du AWS Key Management Service développeur.

Une fois le chiffrement activé, lorsque vous ajoutez des objets du catalogue de données, exécutez des crawlers ou des tâches, ou bien démarrez des points de terminaison de développement, les clés SSE-S3 ou SSE-KMS sont utilisées pour écrire les données au repos. En outre, vous pouvez configurer AWS Glue pour accéder uniquement aux stockages de données JDBC (Java Database Connectivity) par le biais d'un protocole TLS (Transport Layer Security) approuvé.

Dans AWS Glue, vous contrôlez les paramètres de chiffrement dans les emplacements suivants :

- Paramètres de votre catalogue de données.
- Les configurations de sécurité que vous créez.
- Le paramètre de chiffrement côté serveur (SSE-S3 ou SSE-KMS) qui est transmis à votre tâche d'extraction, de transformation et de chargement (ETL) AWS Glue.

Pour plus d'informations sur la mise en place du chiffrement, consultez Configuration du chiffrement dans AWS Glue.

Rubriques

- Chiffrement de votre catalogue de données
- Chiffrement des mots de passe de connexion
- Chiffrement des données écrites par AWS Glue

Chiffrement de votre catalogue de données

AWS Glue Data Catalog le chiffrement renforce la sécurité de vos données sensibles. AWS Glue s'intègre à AWS Key Management Service (AWS KMS) pour chiffrer les métadonnées stockées dans le catalogue de données. Vous pouvez activer ou désactiver les paramètres de chiffrement pour les ressources du catalogue de données à l'aide de la AWS Glue console ou du AWS CLI.

Lorsque vous activez le chiffrement pour votre catalogue de données, tous les nouveaux objets que vous créez sont chiffrés. Lorsque vous désactivez le chiffrement, les nouveaux objets que vous créez ne sont pas chiffrés, mais les objets chiffrés existants restent chiffrés.

Vous pouvez chiffrer l'intégralité de votre catalogue de données à l'aide de clés de chiffrement AWS gérées ou de clés de chiffrement gérées par le client. Pour plus d'informations sur les types de clés et les états, consultez AWS Key Management Service les concepts du Guide du AWS Key Management Service développeur.

> **Note**
>
> Lorsque vous utilisez le catalogue de données chiffré avec un robot, vous devez conserver les paramètres de chiffrement. La suppression des paramètres de chiffrement après qu'un robot a traité un catalogue chiffré entraîne des erreurs. Si vous devez supprimer des paramètres de chiffrement, créez un autre robot au lieu de modifier celui qui existe déjà.

AWS clés gérées

AWS les clés gérées sont des clés KMS de votre compte qui sont créées, gérées et utilisées en votre nom par un AWS service intégré à AWS KMS. Vous pouvez consulter les clés AWS gérées de votre compte, consulter leurs politiques clés et vérifier leur utilisation dans AWS CloudTrail les journaux. Toutefois, vous ne pouvez pas gérer ces clés ni modifier leurs autorisations.

Encryption at rest s'intègre AWS KMS automatiquement AWS à la gestion AWS Glue des clés gérées utilisées pour chiffrer vos métadonnées. Si aucune clé AWS gérée n'existe lorsque vous activez le chiffrement des métadonnées, une nouvelle clé est AWS KMS automatiquement créée pour vous.

Pour en savoir plus, consultez Clés gérées par AWS.

Clés gérées par le client

Les clés gérées par le client sont des clés KMS Compte AWS que vous créez, détenez et gérez. Vous avez un contrôle total sur ces clés KMS. Vous pouvez effectuer les actions suivantes :

- Établir et maintenir leurs stratégies de clé, leurs politiques IAM et leurs autorisations
- Les activer et les désactiver
- Effectuer une rotation de leur matériel de chiffrement
- Ajouter des balises

- Créer des alias qui y font référence
- Les planifier en vue de leur suppression

Pour plus d'informations sur la gestion des autorisations d'une clé gérée par le client, consultez Clés gérées par le client.

> **Important**
>
> AWS Glue prend uniquement en charge les clés symétriques gérées par le client. La liste de clés KMS n'affiche que des clés symétriques. Toutefois, si vous sélectionnez Choisir un ARN de clé KMS, la console vous permet d'entrer un ARN pour n'importe quel type de clé. Assurez-vous de n'entrer que ARNs pour les clés symétriques.
> Pour créer une CMK symétrique, suivez les étapes de création de clés symétriques gérées par le client dans le Guide du développeur AWS Key Management Service .

Lorsque vous activez le chiffrement du catalogue de données au repos, les types de ressources suivants sont chiffrés à l'aide de clés KMS :

- Bases de données
- Tables
- Partitions
- Versions de table
- Statistiques de colonne
- Fonctions définies par l'utilisateur
- Affichages du catalogue de données

AWS Glue contexte de chiffrement

Un contexte de chiffrement est un ensemble facultatif de paires clé-valeur qui contiennent des informations contextuelles supplémentaires sur les données. AWS KMS utilise le contexte de chiffrement en tant que données authentifiées supplémentaires pour prendre en charge le chiffrement authentifié. Lorsque vous incluez un contexte de chiffrement dans une demande de chiffrement de données, AWS KMS lie le contexte de chiffrement aux données chiffrées. Pour déchiffrer les données, vous devez inclure le même contexte de chiffrement dans la demande. AWS Glue utilise le

même contexte de chiffrement dans toutes les opérations AWS KMS cryptographiques, où la clé est `glue_catalog_id` et la valeur est le`catalogId`.

```
"encryptionContext": {
    "glue_catalog_id": "111122223333"
}
```

Lorsque vous utilisez une clé AWS gérée ou une clé symétrique gérée par le client pour chiffrer votre catalogue de données, vous pouvez également utiliser le contexte de chiffrement dans les enregistrements d'audit et les journaux pour identifier la manière dont la clé est utilisée. Le contexte de chiffrement apparaît également dans les journaux générés par AWS CloudTrail ou dans Amazon CloudWatch les journaux.

Activation du chiffrement

Vous pouvez activer le chiffrement de vos AWS Glue Data Catalog objets dans les paramètres du catalogue de données de la AWS Glue console ou en utilisant le AWS CLI.

Console

Pour activer le chiffrement à l'aide de la console

1. Connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/.
2. Dans le panneau de navigation, choisissez catalogue de données.
3. Sur la page des paramètres du catalogue de données, cochez la case Chiffrement des métadonnées et choisissez une AWS KMS clé.

   Lorsque vous activez le chiffrement, si vous ne spécifiez pas de clé gérée par le client, les paramètres de chiffrement utilisent une clé KMS AWS gérée.
4. (Facultatif) Lorsque vous utilisez une clé gérée par le client pour chiffrer votre catalogue de données, celui-ci propose une option permettant d'enregistrer un rôle IAM pour chiffrer et déchiffrer les ressources. Vous devez accorder à votre rôle IAM des autorisations qui AWS Glue peuvent être assumées en votre nom. Cela inclut AWS KMS les autorisations de chiffrement et de déchiffrement des données.

   Lorsque vous créez une nouvelle ressource dans le catalogue de données, elle AWS Glue assume le rôle IAM fourni pour chiffrer les données. De même, lorsqu'un consommateur accède à la ressource, il AWS Glue assume le rôle IAM pour déchiffrer les données. Si vous

enregistrez un rôle IAM avec les autorisations requises, le principal appelant n'a plus besoin d'autorisations pour accéder à la clé et déchiffrer les données.

> **Important**
>
> Vous ne pouvez déléguer des opérations KMS à un rôle IAM que si vous utilisez une clé gérée par le client pour chiffrer les ressources du catalogue de données. La fonctionnalité de délégation de rôles KMS ne prend pas en charge l'utilisation de clés gérées par AWS pour chiffrer les ressources du catalogue de données pour le moment.

> **Warning**
>
> Lorsque vous activez un rôle IAM pour déléguer les opérations KMS, vous ne pouvez plus accéder aux ressources du catalogue de données qui étaient précédemment chiffrées à l'aide d'une clé AWS gérée.

a. Pour activer un rôle IAM AWS Glue capable de chiffrer et de déchiffrer les données en votre nom, sélectionnez l'option Déléguer les opérations KMS à un rôle IAM.

b. Ensuite, choisissez un rôle IAM.

   Pour créer un rôle IAM, consultez Création d'un rôle IAM pour AWS Glue.

   Le rôle IAM censé accéder AWS Glue au catalogue de données doit disposer des autorisations nécessaires pour chiffrer et déchiffrer les métadonnées du catalogue de données. Vous pouvez créer un rôle IAM et y attacher les politiques en ligne suivantes :

   - Ajoutez la politique suivante pour inclure AWS KMS les autorisations permettant de chiffrer et de déchiffrer le catalogue de données.

     JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
```

```
      "Action": [
        "kms:Decrypt",
        "kms:Encrypt",
        "kms:GenerateDataKey"
      ],
      "Resource": "arn:aws:kms:us-east-1:111122223333:key/<key-id>"
    }
  ]
}
```

- Ajoutez ensuite la politique de confiance suivante au rôle pour que le AWS Glue service assume le rôle IAM.

  JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": "glue.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

- Ajoutez ensuite l'autorisation `iam:PassRole` au rôle IAM.

  JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iam:PassRole"
      ],
      "Resource": [
        "arn:aws:iam::111122223333:role/<encryption-role-name>"
      ]
```

```
    }
  ]
}
```

Lorsque vous activez le chiffrement, si vous n'avez pas spécifié de rôle IAM AWS Glue à assumer, le principal accédant au catalogue de données doit être autorisé à effectuer les opérations d'API suivantes :

- `kms:Decrypt`
- `kms:Encrypt`
- `kms:GenerateDataKey`

AWS CLI

Pour activer le chiffrement à l'aide du SDK ou AWS CLI

- Utilisez l'opération d'API `PutDataCatalogEncryptionSettings`. Si aucune clé n'est spécifiée, AWS Glue utilise une clé de chiffrement AWS gérée pour le compte client afin de chiffrer le catalogue de données.

```
aws glue put-data-catalog-encryption-settings \
  --data-catalog-encryption-settings '{
      "EncryptionAtRest": {
      "CatalogEncryptionMode": "SSE-KMS-WITH-SERVICE-ROLE",
      "SseAwsKmsKeyId": "arn:aws:kms:<region>:<account-id>:key/<key-id>",
      "CatalogEncryptionServiceRole":"arn:aws:iam::<account-
id>:role/<encryption-role-name>"
    }

  }'
```

Une fois le chiffrement activé, tous les objets que vous créez dans les objets du catalogue de données sont chiffrés. Si vous désactivez ce paramètre, les objets que vous créez dans le catalogue de données ne sont plus chiffrés. Vous pouvez continuer à accéder aux objets chiffrés existants dans le catalogue de données avec les autorisations KMS requises.

> **Important**
>
> La AWS KMS clé doit rester disponible dans le magasin de AWS KMS clés pour tous les objets chiffrés avec elle dans le catalogue de données. Si vous supprimez la clé, les objets ne peuvent plus être déchiffrés. Dans certains scénarios, vous pouvez opter pour cette solution pour empêcher l'accès aux métadonnées du catalogue de données.

Surveillance de vos clés KMS pour AWS Glue

Lorsque vous utilisez des clés KMS avec les ressources de votre catalogue de données, vous pouvez utiliser AWS CloudTrail Amazon CloudWatch nos journaux pour suivre les demandes AWS Glue envoyées à AWS KMS. AWS CloudTrail surveille et enregistre les opérations KMS qui AWS Glue appellent à accéder à des données chiffrées par vos clés KMS.

Les exemples suivants sont AWS CloudTrail des événements relatifs aux `GenerateDataKey` opérations `Decrypt` et.

Decrypt

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAXPHTESTANDEXAMPLE:Sampleuser01",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAXPHTESTANDEXAMPLE",
                "arn": "arn:aws:iam::111122223333:role/Admin",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2024-01-10T14:33:56Z",
                "mfaAuthenticated": "false"
```

```
            }
        },
        "invokedBy": "glue.amazonaws.com"
    },
    "eventTime": "2024-01-10T15:18:11Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Decrypt",
    "awsRegion": "eu-west-2",
    "sourceIPAddress": "glue.amazonaws.com",
    "userAgent": "glue.amazonaws.com",
    "requestParameters": {
        "encryptionContext": {
            "glue_catalog_id": "111122223333"
        },
        "encryptionAlgorithm": "SYMMETRIC_DEFAULT"
    },
    "responseElements": null,
    "requestID": "43b019aa-34b8-4798-9b98-ee968b2d63df",
    "eventID": "d7614763-d3fe-4f84-a1e1-3ca4d2a5bbd5",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:<region>:111122223333:key/<key-id>"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management",
    "sessionCredentialFromConsole": "true"
}
```

GenerateDataKey

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId":
 "AROAXPHTESTANDEXAMPLE:V_00_GLUE_KMS_GENERATE_DATA_KEY_111122223333",
```

```
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/
V_00_GLUE_KMS_GENERATE_DATA_KEY_111122223333",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAXPHTESTANDEXAMPLE",
                "arn": "arn:aws:iam::111122223333:role/Admin",
                "accountId": "AKIAIOSFODNN7EXAMPLE",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2024-01-05T21:15:47Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "glue.amazonaws.com"
    },
    "eventTime": "2024-01-05T21:15:47Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKey",
    "awsRegion": "eu-west-2",
    "sourceIPAddress": "glue.amazonaws.com",
    "userAgent": "glue.amazonaws.com",
    "requestParameters": {
        "keyId": "arn:aws:kms:eu-west-2:AKIAIOSFODNN7EXAMPLE:key/
AKIAIOSFODNN7EXAMPLE",
        "encryptionContext": {
            "glue_catalog_id": "111122223333"
        },
        "keySpec": "AES_256"
    },
    "responseElements": null,
    "requestID": "64d1783a-4b62-44ba-b0ab-388b50188070",
    "eventID": "1c73689b-2ef2-443b-aed7-8c126585ca5e",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:eu-west-2:111122223333:key/AKIAIOSFODNN7EXAMPLE"
        }
```

```
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

Chiffrement des mots de passe de connexion

Vous pouvez récupérer les mots de passe de connexion dans AWS Glue Data Catalog le à l'aide des opérations `GetConnections` d'API `GetConnection` et. Ces mots de passe sont stockés dans la connexion du catalogue de données et sont utilisés lorsque AWS Glue se connecte à un magasin de données Java Database Connectivity (JDBC). Lorsque la connexion a été créée ou mise à jour, une option dans les paramètres du catalogue de données déterminait si le mot de passe était crypté et, dans l'affirmative, quelle clé AWS Key Management Service (AWS KMS) était spécifiée.

Dans la console AWS Glue, vous pouvez activer cette option sur la page des paramètres de catalogue de données.

Pour chiffrer des mots de passe de connexion

1. Connectez-vous à la AWS Glue console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/glue/.
2. Choisissez Settings (Paramètres) dans le volet de navigation.
3. Dans la page Data catalog settings (Paramètres de catalogue de données), sélectionnez Encrypt connection passwords (Chiffrer les mots de passe de connexion) et choisissez une clé AWS KMS .

> **Important**
>
> AWS Glueprend uniquement en charge les clés principales symétriques du client (CMKs). La liste de AWS KMS key (clé KMS) n'affiche que des clés symétriques.

Toutefois, si vous sélectionnez Choisir un ARN de AWS KMS clé, la console vous permet de saisir un ARN pour n'importe quel type de clé. Assurez-vous de n'entrer que ARNs pour les clés symétriques.

Pour de plus amples informations, veuillez consulter Paramètres du catalogue de données.

Chiffrement des données écrites par AWS Glue

Une configuration de sécurité est un ensemble de propriétés de sécurité pouvant être utilisées par AWS Glue. Vous pouvez utiliser une configuration de sécurité pour chiffrer les données au repos. Les scénarios suivants illustrent les différentes façons dont vous pouvez utiliser une configuration de sécurité.

- Associez une configuration de sécurité à un AWS Glue robot d'exploration pour écrire des Amazon CloudWatch Logs chiffrés. Pour plus d'informations sur l'attachement de configurations de sécurité aux robots, consultez the section called "Configurer les paramètres de sécurité".
- Associez une configuration de sécurité à une tâche d'extraction, de transformation et de chargement (ETL) pour écrire des cibles Amazon Simple Storage Service (Amazon S3) chiffrées et CloudWatch des journaux chiffrés.
- Attacher une configuration de sécurité à une tâche ETL pour écrire ses signets de tâche sous forme de données Amazon S3 chiffrées.
- Attacher une configuration de sécurité à un point de terminaison de développement pour écrire des cibles Amazon S3 chiffrées.

Important

Actuellement, une configuration de sécurité remplace tout paramètre de chiffrement côté serveur (SSE-S3) qui est transmis en tant que paramètre de tâche ETL. Par conséquent, si une configuration de sécurité et un paramètre SSE-S3 sont associés à une tâche, le paramètre SSE-S3 est ignoré.

Pour plus d'informations sur les configurations de sécurité, consultez Gestion des configurations de sécurité sur la console AWS Glue.

Rubriques

- Configuration d'AWS Glue pour utiliser des configurations de sécurité
- Création d'un itinéraire AWS KMS pour les tâches VPC et les robots d'exploration
- Gestion des configurations de sécurité sur la console AWS Glue

Configuration d'AWS Glue pour utiliser des configurations de sécurité

Exécutez les étapes suivantes pour configurer votre environnement AWS Glue pour utiliser des configurations de sécurité.

1. Créez ou mettez à jour vos clés AWS Key Management Service (AWS KMS) pour accorder AWS KMS des autorisations aux rôles IAM qui sont transmis aux AWS Glue robots d'exploration et aux tâches pour chiffrer CloudWatch les journaux. Pour plus d'informations, consultez la section Chiffrer les données de journal dans CloudWatch les journaux à l'aide AWS KMS du guide de l'utilisateur Amazon CloudWatch Logs.

   Dans l'exemple suivant, *"role1""role2"*, et *"role3"* sont des rôles IAM transmis aux robots d'exploration et aux tâches.

```
{
       "Effect": "Allow",
       "Principal": { "Service": "logs.region.amazonaws.com",
       "AWS": [
                "role1",
                "role2",
                "role3"
             ] },
                    "Action": [
                           "kms:Encrypt*",
                           "kms:Decrypt*",
                           "kms:ReEncrypt*",
                           "kms:GenerateDataKey*",
                           "kms:Describe*"
                    ],
                    "Resource": "*"
}
```

L'`Service`instruction, représentée sous la forme`"Service": "logs.region.amazonaws.com"`, est obligatoire si vous utilisez la clé pour chiffrer CloudWatch les journaux.

2. Assurez-vous que la AWS KMS clé est bien `ENABLED` là avant de l'utiliser.

> **Note**
>
> Si vous utilisez Iceberg comme cadre de lac de données, les tables Iceberg disposent de leurs propres mécanismes pour activer le chiffrement côté serveur. Vous devez activer ces configurations en plus des configurations AWS Glue de sécurité. Pour activer le chiffrement côté serveur sur les tables Iceberg, consultez les conseils de la documentation d'Iceberg.

Création d'un itinéraire AWS KMS pour les tâches VPC et les robots d'exploration

Vous pouvez vous connecter directement à AWS KMS via un point de terminaison privé dans votre VPC au lieu de vous connecter via Internet. Lorsque vous utilisez un point de terminaison VPC, la communication entre votre VPC et celui-ci AWS KMS s'effectue entièrement au sein du réseau. AWS

Vous pouvez créer un point de terminaison AWS KMS VPC au sein d'un VPC. Sans cette étape, vos tâches ou crawlers peuvent échouer avec un `kms timeout` sur les tâches ou une `internal service exception` sur les crawlers. Pour obtenir des instructions détaillées, consultez la section Connexion AWS KMS via un point de terminaison VPC dans le guide du AWS Key Management Service développeur.

Au fur et à mesure que vous suivez ces instructions, sur la console VPC,vous devez effectuer les opérations suivantes :

- Sélectionnez Enable Private DNS name (Activer le nom DNS privé).
- Choisissez le groupe de sécurité (avec une règle à référence circulaire) que vous utilisez pour votre tâche ou votre crawler qui accède à Java Database Connectivity (JDBC). Pour plus d'informations sur les connexions AWS Glue, consultez Connexion aux données.

Lorsque vous ajoutez une configuration de sécurité à un robot ou à une tâche qui accède aux magasins de données JDBC, vous AWS Glue devez disposer d'un itinéraire vers le point de terminaison. AWS KMS Vous pouvez fournir l'itinéraire à l'aide d'une passerelle de traduction

d'adresses réseau (NAT) ou d'un point de AWS KMS terminaison VPC. Pour créer une passerelle NAT, veuillez consulter Passerelles NAT dans le Guide de l'utilisateur Amazon VPC.

Gestion des configurations de sécurité sur la console AWS Glue

> **Warning**
>
> AWS Glue les configurations de sécurité ne sont actuellement pas prises en charge dans les tâches Ray.

Une configuration de sécurité dans AWS Glue contient les propriétés nécessaires lorsque vous écrivez des données chiffrées. Vous créez des configurations de sécurité sur la console AWS Glue pour fournir les propriétés de chiffrement qui sont utilisés par les crawlers, les tâches et des points de terminaison de développement.

Pour afficher la liste de toutes les configurations de sécurité que vous avez créées, ouvrez la console AWS Glue à l'adresse https://console.aws.amazon.com/glue/ et choisissez Configurations de sécurité dans le volet de navigation.

La liste Configurations de sécurité affiche les propriétés suivantes pour chaque configuration :

Nom

Nom unique fourni lors de la création de la configuration. Le nom peut contenir des lettres (A à Z), des chiffres (0 à 9), des tirets (-) ou des traits de soulignement (_) et jusqu'à 255 caractères.

Activer le chiffrement Amazon S3

Si cette option est activée, le mode de chiffrement Amazon Simple Storage Service (Amazon S3) tel que `SSE-KMS` ou `SSE-S3` est activé pour le magasin de métadonnées du catalogue de données.

Activer le chiffrement CloudWatch des journaux Amazon

S'il est activé, le mode de chiffrement Amazon S3, tel qu'`SSE-KMS`il est activé lors de l'écriture de journaux sur Amazon CloudWatch.

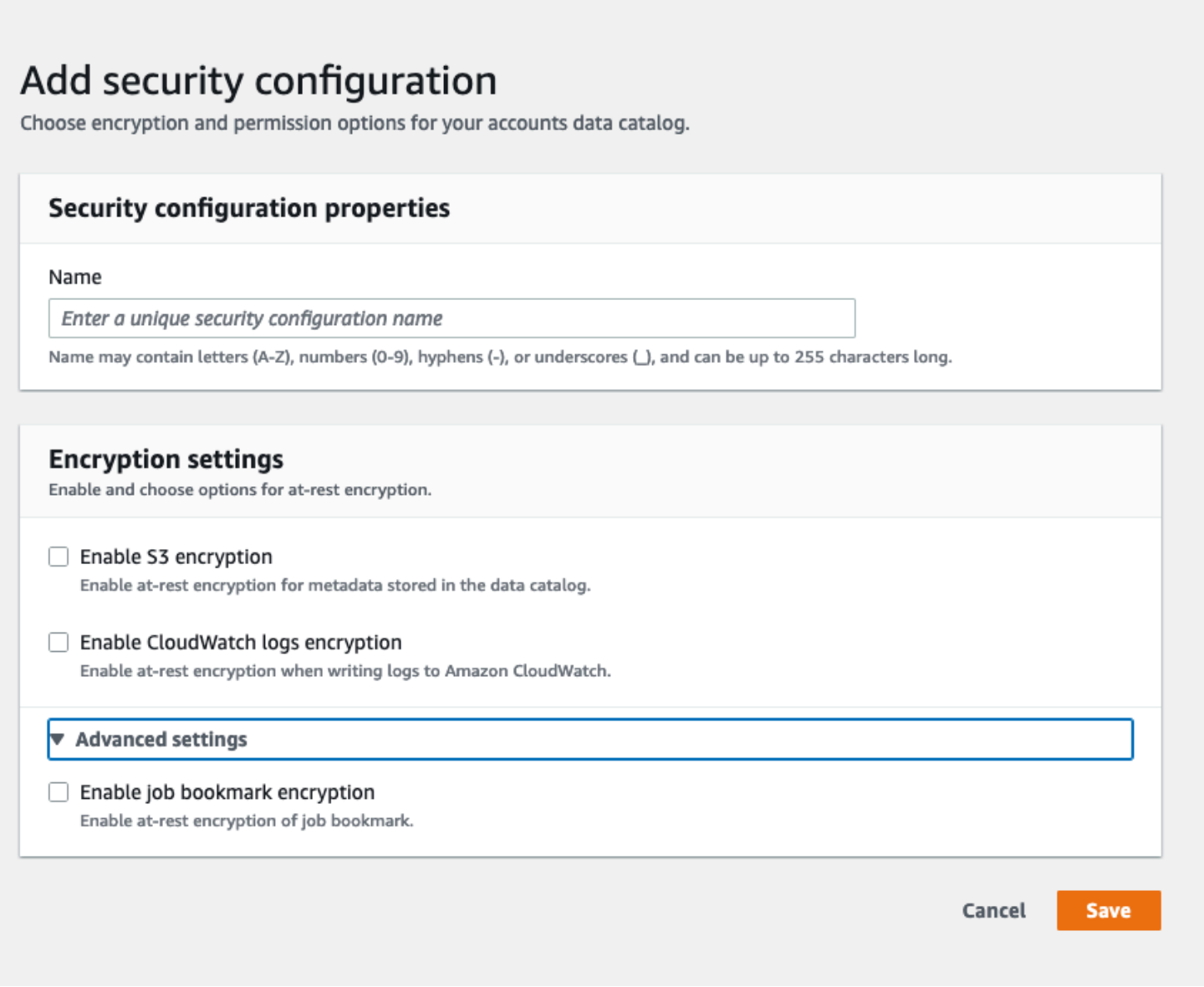
Paramètres avancés : activer le chiffrement des signets de tâches

Si cette option est activée, le mode de chiffrement Amazon S3 tel que `CSE-KMS` est activé lorsque les tâches sont marquées d'un signet.

Vous pouvez ajouter ou supprimer des configurations dans la section Configurations de sécurité sur la console. Pour afficher plus de détails sur une configuration, choisissez son nom dans la liste. Les détails incluent les informations que vous avez définies lors de la création de la configuration.

Ajout d'une configuration de sécurité

Pour ajouter une configuration de sécurité à l'aide de la console AWS Glue, dans la page Security Configurations (Configurations de sécurité), choisissez Add security configuration (Ajouter une configuration de sécurité).



Propriétés de configuration de sécurité

Saisissez un nom de configuration de sécurité unique. Le nom peut contenir des lettres (A à Z), des chiffres (0 à 9), des tirets (-) ou des traits de soulignement (_) et jusqu'à 255 caractères.

Paramètres de chiffrement

Vous pouvez activer le chiffrement au repos pour les métadonnées stockées dans le catalogue de données d'Amazon S3 et les journaux d'Amazon CloudWatch. Pour configurer le chiffrement des données et des métadonnées à l'aide des clés AWS Key Management Service (AWS KMS) sur la AWS Glue console, ajoutez une politique à l'utilisateur de la console. Cette politique doit spécifier les ressources autorisées sous forme de noms de ressources Amazon clés (ARNs) utilisés pour chiffrer les magasins de données Amazon S3, comme dans l'exemple suivant.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "kms:GenerateDataKey",
      "kms:Decrypt",
      "kms:Encrypt"
    ],
    "Resource": "arn:aws:kms:us-east-1:111122223333:key/key-id"
  }
}
```

> **Important**
>
> Lorsqu'une configuration de sécurité est attachée à un robot ou à une tâche, le rôle IAM transmis doit disposer AWS KMS d'autorisations. Pour de plus amples informations, veuillez consulter Chiffrement des données écrites par AWS Glue.

Lorsque vous définissez une configuration, vous pouvez fournir les valeurs des propriétés suivantes :

Activer le chiffrement S3

Lorsque vous écrivez des données Amazon S3, vous utilisez soit le chiffrement côté serveur avec des clés gérées par Amazon S3 (SSE-S3), soit le chiffrement côté serveur avec AWS KMS des clés gérées (SSE-KMS). Ce champ est facultatif. Pour autoriser l'accès à Amazon S3, choisissez une AWS KMS clé ou choisissez Enter a key ARN et fournissez l'ARN de la clé. Entrez l'ARN sous la forme `arn:aws:kms:`*`region`*`:`*`account-id`*`:key/`*`key-id`*. Vous pouvez également fournir l'ARN sous la forme d'un alias de clé, (par exemple,) `arn:aws:kms:`*`region`*`:`*`account-id`*`:alias/`*`alias-name`*.

Si vous activez l'interface utilisateur Spark pour votre tâche, le fichier journal de l'interface utilisateur Spark chargé sur Amazon S3 sera appliqué avec le même chiffrement.

> **Important**
>
> AWS Gluene prend en charge que les clés principales symétriques du client (CMKs). La liste de AWS KMS key (clé KMS) n'affiche que des clés symétriques. Toutefois, si vous sélectionnez Choisir un ARN de AWS KMS clé, la console vous permet de saisir un ARN pour n'importe quel type de clé. Assurez-vous de n'entrer que ARNs pour les clés symétriques.

Activer le chiffrement CloudWatch des journaux

Le chiffrement côté serveur (SSE-KMS) est utilisé pour chiffrer les journaux. CloudWatch Ce champ est facultatif. Pour l'activer, choisissez une AWS KMS clé ou choisissez Enter a key ARN et indiquez l'ARN de la clé. Entrez l'ARN sous la forme `arn:aws:kms:`*`region`*`:`*`account-id`*`:key/`*`key-id`*. Vous pouvez également fournir l'ARN sous la forme d'un alias de clé, (par exemple,) `arn:aws:kms:`*`region`*`:`*`account-id`*`:alias/`*`alias-name`*.

Paramètres avancés : chiffrement des signets de tâches

Le chiffrement côté client (CSE-KMS) est utilisé pour chiffrer les signets de tâche. Ce champ est facultatif. Les données de signet sont chiffrées avant d'être envoyées à Amazon S3 en vue de leur stockage. Pour l'activer, choisissez une AWS KMS clé ou choisissez Enter a key ARN et indiquez l'ARN de la clé. Entrez l'ARN sous la forme `arn:aws:kms:`*`region`*`:`*`account-id`*`:key/`*`key-id`*. Vous pouvez également fournir l'ARN sous la forme d'un alias de clé, (par exemple,) `arn:aws:kms:`*`region`*`:`*`account-id`*`:alias/`*`alias-name`*.

Pour en savoir plus, consulter les rubriques suivantes dans le Guide de l'utilisateur d'Amazon Simple Storage Service :

- Pour en savoir plus sur `SSE-S3`, consultez Protecting Data Using Server-Side Encryption with Amazon S3-Managed Encryption Keys (SSE-S3) (Protection des données à l'aide du chiffrement côté serveur avec les clés de chiffrement gérées par Amazon S3 (SSE-S3)).
- Pour plus d'informations sur`SSE-KMS`, consultez la section Protection des données à l'aide du chiffrement côté serveur avec. AWS KMS keys
- Pour plus d'informations sur `CSE-KMS`, veuillez consulter Using a KMS key stored in AWS KMS.

chiffrement des données en transit

AWS fournit un chiffrement TLS (Transport Layer Security) pour les données en mouvement. Vous pouvez configurer les paramètres de chiffrement pour les crawlers, les tâches ETL et les points de terminaison de développement à l'aide de configurations de sécurité dans AWS Glue. Vous pouvez activer le AWS Glue Data Catalog chiffrement via les paramètres du catalogue de données.

Depuis le 4 septembre 2018 AWS KMS (apportez votre propre clé et chiffrez côté serveur), l'AWS GlueETL AWS Glue Data Catalog est pris en charge.

Conformité FIPS

Si vous avez besoin de modules cryptographiques validés par la norme FIPS 140-2 pour accéder AWS via une interface de ligne de commande ou une API, utilisez un point de terminaison FIPS. Pour plus d'informations sur les points de terminaison FIPS (Federal Information Processing Standard) disponibles, consultez Federal Information Processing Standard (FIPS) 140-2 (Normes de traitement de l'information fédérale).

Gestion des clés

Vous pouvez utiliser Gestion des identités et des accès AWS (IAM) AWS Glue pour définir des utilisateurs, AWS des ressources, des groupes, des rôles et des politiques précises concernant l'accès, le refus, etc.

Vous pouvez définir l'accès aux métadonnées en utilisant à la fois les stratégies basées sur les ressources et les stratégies basées sur les identités, en fonction des besoins de votre organisation.

Les stratégies basées sur les ressources répertorient les principaux dont l'accès à vos ressources est autorisé ou refusé, ce qui vous permet de configurer des stratégies comme un accès entre comptes. Les stratégies d'identité sont spécifiquement associées à des utilisateurs, des groupes et des rôles au sein d'IAM.

step-by-stepPar exemple, consultez Restreindre l'accès à vos AWS Glue Data Catalog autorisations IAM au niveau des ressources et à des politiques basées sur les ressources sur le blog Big Data. AWS

La partie relative aux accès affinés de la stratégie est définie dans la clause `Resource`. Cette partie définit à la fois l' AWS Glue Data Catalog objet sur lequel l'action peut être effectuée et les objets résultants renvoyés par cette opération.

Un point de terminaison de développement est un environnement que vous pouvez utiliser pour développer et tester vos scripts AWS Glue. Vous pouvez ajouter ou supprimer la clé SSH d'un point de terminaison de développement, ou en effectuer une rotation.

Depuis le 4 septembre 2018 AWS KMS (apportez votre propre clé et chiffrez côté serveur), l'AWS GlueETL AWS Glue Data Catalog est pris en charge.

AWS Gluedépendance à l'égard d'autres AWS services

Pour qu'un utilisateur puisse utiliser la AWS Glue console, il doit disposer d'un ensemble minimal d'autorisations lui permettant d'utiliser les AWS Glue ressources de son AWS compte. Outre ces autorisations AWS Glue, la console nécessite les autorisations des services suivants :

- Autorisations Amazon CloudWatch Logs pour afficher les journaux.
- Gestion des identités et des accès AWS Autorisations (IAM) permettant de répertorier et de transmettre des rôles.
- CloudFormation autorisations pour travailler avec des piles.
- Autorisations Amazon Elastic Compute Cloud (Amazon EC2) permettant de répertorier des clouds privés virtuels (VPCs), des sous-réseaux, des groupes de sécurité, des instances et d'autres objets (pour configurer des EC2 éléments Amazon, par exemple VPCs lors de l'exécution de tâches, de robots d'exploration et de la création de points de terminaison de développement).
- Autorisations Amazon Simple Storage Service (Amazon S3) pour répertorier les compartiments et les objets, et pour récupérer et enregistrer les scripts.
- Autorisations Amazon Redshift requises pour travailler avec des clusters.

- Autorisations Amazon Relational Database Service (Amazon RDS) pour répertorier les instances.

Points de terminaison de développement

Un point de terminaison de développement est un environnement que vous pouvez utiliser pour développer et tester vos scripts AWS Glue. Vous pouvez utiliser AWS Glue pour créer, modifier et supprimer des points de terminaison de développement. Vous pouvez répertorier tous les points de terminaison de développement créés. Vous pouvez ajouter ou supprimer la clé SSH d'un point de terminaison de développement, ou en effectuer une rotation. Vous pouvez également créer des blocs-notes qui utilisent le point de terminaison de développement.

Vous fournissez les valeurs de configuration pour mettre en service les environnements de développement. Ces valeurs indiquent à AWS Glue comment configurer le réseau afin que vous puissiez accéder au point de terminaison de développement en toute sécurité et votre point de terminaison peut accéder à vos magasins de données. Ensuite, vous pouvez créer un bloc-notes qui se connecte au point de terminaison de développement. Vous utilisez votre bloc-notes pour créer et tester votre script ETL.

Utilisez un rôle Gestion des identités et des accès AWS (IAM) doté d'autorisations similaires au rôle IAM que vous utilisez pour exécuter des tâches AWS Glue ETL. Utilisez un cloud privé virtuel (VPC), un sous-réseau et un groupe de sécurité pour créer un point de terminaison de développement qui peut se connecter à vos ressources de données en toute sécurité. Vous générez une paire de clés SSH pour vous connecter à l'environnement de développement à l'aide de SSH.

Vous pouvez créer des points de terminaison de développement pour les données Amazon S3 et au sein d'un VPC que vous pouvez utiliser pour accéder à des ensembles de données à l'aide de JDBC.

Vous pouvez installer un client de bloc-notes Jupyter sur votre ordinateur local et l'utiliser pour déboguer et tester les scripts ETL sur un point de terminaison de développement. Vous pouvez également utiliser un bloc-notes Sagemaker pour créer des scripts ETL JupyterLab en AWS un. Consultez la section Utiliser un SageMaker bloc-notes avec votre terminal de développement.

AWS Glueétiquette EC2 les instances Amazon avec un nom préfixé para`aws-glue-dev-endpoint`.

Vous pouvez configurer un serveur bloc-notes sur un terminal de développement pour qu'il fonctionne PySpark avec des AWS Glue extensions.

Gestion des identités et des accès pour AWS Glue

Gestion des identités et des accès AWS (IAM) est un outil Service AWS qui permet à un administrateur de contrôler en toute sécurité l'accès aux AWS ressources. Les administrateurs IAM contrôlent qui peut être authentifié (connecté) et autorisé (autorisé) à utiliser les ressources AWS Glue. IAM est un Service AWS outil que vous pouvez utiliser sans frais supplémentaires.

> **Note**
>
> Vous pouvez autoriser l'accès à vos données dans le catalogue de données AWS Glue en utilisant AWS Glue des méthodes ou AWS Lake Formation des autorisations. Vous pouvez utiliser des politiques Gestion des identités et des accès AWS (IAM) pour définir un contrôle d'accès précis à l'aide de méthodes. AWS Glue Lake Formation utilise un modèle d'autorisations `GRANT/REVOKE` plus simple similaire aux commandes `GRANT/REVOKE` dans un système de base de données relationnelle.
>
> Cette section inclut des informations sur l'utilisation des méthodes AWS Glue. Pour plus d'informations sur l'utilisation des octrois Lake Formation, veuillez consulter la rubrique Octroi des autorisations Lake Formation dans le Guide du développeur AWS Lake Formation .

Rubriques

- Public ciblé
- Authentification par des identités
- Gestion de l'accès à l'aide de politiques
- Comment AWS Glue fonctionne avec IAM
- Configuration des autorisations IAM pour AWS Glue
- AWS Exemples de politiques de contrôle d'accès Glue
- Octroi de politiques AWS gérées pour AWS Glue
- Octroi de politiques à portée dynamique pour l'exécution des tâches
- Spécification de AWS Glue la ressource ARNs
- Octroi d'un accès intercompte
- Résolution des problèmes d'identité et d'accès à AWS Glue

Public ciblé

La façon dont vous utilisez Gestion des identités et des accès AWS (IAM) varie en fonction de votre rôle :

- Utilisateur du service : demandez des autorisations à votre administrateur si vous ne pouvez pas accéder aux fonctionnalités (voir Résolution des problèmes d'identité et d'accès à AWS Glue)
- Administrateur du service : déterminez l'accès des utilisateurs et soumettez les demandes d'autorisation (voir Comment AWS Glue fonctionne avec IAM)
- Administrateur IAM : rédigez des politiques pour gérer l'accès (voir Exemples de politiques basées sur l'identité pour Glue AWS)

Authentification par des identités

L'authentification est la façon dont vous vous connectez à AWS l'aide de vos informations d'identification. Vous devez être authentifié en tant qu'utilisateur IAM ou en assumant un rôle IAM. Utilisateur racine d'un compte AWS

Vous pouvez vous connecter en tant qu'identité fédérée à l'aide d'informations d'identification provenant d'une source d'identité telle que AWS IAM Identity Center (IAM Identity Center), d'une authentification unique ou d'informations d'identification. Google/Facebook Pour plus d'informations sur la connexion, consultez Connexion à votre Compte AWS dans le Guide de l'utilisateur Connexion à AWS .

Pour l'accès par programmation, AWS fournit un SDK et une CLI pour signer les demandes de manière cryptographique. Pour plus d'informations, consultez Signature AWS Version 4 pour les demandes d'API dans le Guide de l'utilisateur IAM.

Compte AWS utilisateur root

Lorsque vous créez un Compte AWS, vous commencez par une seule identité de connexion appelée utilisateur Compte AWS root qui dispose d'un accès complet à toutes Services AWS les ressources. Il est vivement déconseillé d'utiliser l'utilisateur racine pour vos tâches quotidiennes. Pour les tâches qui requièrent des informations d'identification de l'utilisateur racine, consultez Tâches qui requièrent les informations d'identification de l'utilisateur racine dans le Guide de l'utilisateur IAM.

Identité fédérée

Il est recommandé d'obliger les utilisateurs humains à utiliser la fédération avec un fournisseur d'identité pour accéder à Services AWS l'aide d'informations d'identification temporaires.

Une identité fédérée est un utilisateur provenant de votre annuaire d'entreprise, de votre fournisseur d'identité Web ou Directory Service qui y accède à Services AWS l'aide d'informations d'identification

provenant d'une source d'identité. Les identités fédérées assument des rôles qui fournissent des informations d'identification temporaires.

Pour une gestion des accès centralisée, nous vous recommandons d'utiliser AWS IAM Identity Center. Pour plus d'informations, consultez Qu'est-ce que IAM Identity Center ? dans le Guide de l'utilisateur AWS IAM Identity Center .

Utilisateurs et groupes IAM

Un utilisateur IAM est une identité qui dispose d'autorisations spécifiques pour une seule personne ou application. Nous vous recommandons d'utiliser ces informations d'identification temporaires au lieu des utilisateurs IAM avec des informations d'identification à long terme. Pour plus d'informations, voir Exiger des utilisateurs humains qu'ils utilisent la fédération avec un fournisseur d'identité pour accéder à AWS l'aide d'informations d'identification temporaires dans le guide de l'utilisateur IAM.

Les groupes IAM spécifient une collection d'utilisateurs IAM et permettent de gérer plus facilement les autorisations pour de grands ensembles d'utilisateurs. Pour plus d'informations, consultez Cas d'utilisation pour les utilisateurs IAM dans le Guide de l'utilisateur IAM.

Rôles IAM

Un rôle IAM est une identité dotée d'autorisations spécifiques qui fournit des informations d'identification temporaires. Vous pouvez assumer un rôle en passant d'un rôle d'utilisateur à un rôle IAM (console) ou en appelant une opération d' AWS API AWS CLI ou d'API. Pour plus d'informations, consultez Méthodes pour endosser un rôle dans le Guide de l'utilisateur IAM.

Les rôles IAM sont utiles pour l'accès des utilisateurs fédérés, les autorisations temporaires des utilisateurs IAM, les accès intercompte, les accès entre services et les applications exécutées sur Amazon EC2. Pour plus d'informations, consultez Accès intercompte aux ressources dans IAM dans le Guide de l'utilisateur IAM.

Gestion de l'accès à l'aide de politiques

Vous contrôlez l'accès en AWS créant des politiques et en les associant à AWS des identités ou à des ressources. Une politique définit les autorisations lorsqu'elles sont associées à une identité ou à une ressource. AWS évalue ces politiques lorsqu'un directeur fait une demande. La plupart des politiques sont stockées AWS sous forme de documents JSON. Pour plus d'informations les documents de politique JSON, consultez Vue d'ensemble des politiques JSON dans le Guide de l'utilisateur IAM.

À l'aide de politiques, les administrateurs précisent qui a accès à quoi en définissant quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

Par défaut, les utilisateurs et les rôles ne disposent d'aucune autorisation. Un administrateur IAM crée des politiques IAM et les ajoute aux rôles, que les utilisateurs peuvent ensuite assumer. Les politiques IAM définissent les autorisations quelle que soit la méthode que vous utilisez pour exécuter l'opération.

Politiques basées sur l'identité

Les stratégies basées sur l'identité sont des documents de stratégie d'autorisations JSON que vous attachez à une identité (utilisateur, groupe ou rôle). Ces politiques contrôlent les actions que peuvent exécuter ces identités, sur quelles ressources et dans quelles conditions. Pour découvrir comment créer une politique basée sur l'identité, consultez Définition d'autorisations IAM personnalisées avec des politiques gérées par le client dans le Guide de l'utilisateur IAM.

Les politiques basées sur l'identité peuvent être des politiques intégrées (intégrées directement dans une seule identité) ou des politiques gérées (politiques autonomes associées à plusieurs identités). Pour découvrir comment choisir entre des politiques gérées et en ligne, consultez Choix entre les politiques gérées et les politiques en ligne dans le Guide de l'utilisateur IAM.

Politiques basées sur les ressources

Les politiques basées sur les ressources sont des documents de politique JSON que vous attachez à une ressource. Les exemples incluent les politiques de confiance de rôle IAM et les stratégies de compartiment Amazon S3. Dans les services qui sont compatibles avec les politiques basées sur les ressources, les administrateurs de service peuvent les utiliser pour contrôler l'accès à une ressource spécifique. Vous devez spécifier un principal dans une politique basée sur les ressources.

Les politiques basées sur les ressources sont des politiques en ligne situées dans ce service. Vous ne pouvez pas utiliser les politiques AWS gérées par IAM dans une stratégie basée sur les ressources.

Autres types de politique

AWS prend en charge des types de politiques supplémentaires qui peuvent définir les autorisations maximales accordées par les types de politiques les plus courants :

- Limites d'autorisations : une limite des autorisations définit le nombre maximum d'autorisations qu'une politique basée sur l'identité peut accorder à une entité IAM. Pour plus d'informations, consultez Limites d'autorisations pour des entités IAM dans le Guide de l'utilisateur IAM.

- Politiques de contrôle des services (SCPs) — Spécifiez les autorisations maximales pour une organisation ou une unité organisationnelle dans AWS Organizations. Pour plus d'informations, consultez Politiques de contrôle de service dans le Guide de l'utilisateur AWS Organizations .
- Politiques de contrôle des ressources (RCPs) : définissez le maximum d'autorisations disponibles pour les ressources de vos comptes. Pour plus d'informations, voir Politiques de contrôle des ressources (RCPs) dans le guide de AWS Organizations l'utilisateur.
- Politiques de session : politiques avancées que vous passez en tant que paramètre lorsque vous créez par programmation une session temporaire pour un rôle ou un utilisateur fédéré. Pour plus d'informations, consultez Politiques de session dans le Guide de l'utilisateur IAM.

Plusieurs types de politique

Lorsque plusieurs types de politiques s'appliquent à la requête, les autorisations en résultant sont plus compliquées à comprendre. Pour savoir comment AWS déterminer s'il faut autoriser une demande lorsque plusieurs types de politiques sont impliqués, consultez la section Logique d'évaluation des politiques dans le guide de l'utilisateur IAM.

Comment AWS Glue fonctionne avec IAM

Avant d'utiliser IAM pour gérer l'accès à AWS Glue, découvrez quelles fonctionnalités IAM peuvent être utilisées avec Glue AWS .

Fonctionnalités IAM que vous pouvez utiliser avec Glue AWS

Fonctionnalité IAM	AWS Support Glue
Politiques basées sur l'identité	Oui
Politiques basées sur les ressources	Partielle
Actions de politique	Oui
Ressources de politique	Oui
Clés de condition de politique (spécifiques au service)	Oui
ACLs	Non

Fonctionnalité IAM	AWS Support Glue
ABAC (identifications dans les politiques)	Partielle
Informations d'identification temporaires	Oui
Autorisations de principals	Non
Rôles de service	Oui
Rôles liés à un service	Non

Pour obtenir une vue d'ensemble de la façon dont AWS Glue et les autres AWS services fonctionnent avec la plupart des fonctionnalités IAM, consultez la section AWS Services compatibles avec IAM dans le guide de l'utilisateur IAM.

Politiques basées sur l'identité pour Glue AWS

Prend en charge les politiques basées sur l'identité : oui

Les politiques basées sur l'identité sont des documents de politique d'autorisations JSON que vous pouvez attacher à une identité telle qu'un utilisateur, un groupe d'utilisateurs ou un rôle IAM. Ces politiques contrôlent quel type d'actions des utilisateurs et des rôles peuvent exécuter, sur quelles ressources et dans quelles conditions. Pour découvrir comment créer une politique basée sur l'identité, consultez Définition d'autorisations IAM personnalisées avec des politiques gérées par le client dans le Guide de l'utilisateur IAM.

Avec les politiques IAM basées sur l'identité, vous pouvez spécifier des actions et ressources autorisées ou refusées, ainsi que les conditions dans lesquelles les actions sont autorisées ou refusées. Pour découvrir tous les éléments que vous utilisez dans une politique JSON, consultez Références des éléments de politique JSON IAM dans le Guide de l'utilisateur IAM.

AWS Glue prend en charge les politiques basées sur l'identité (politiques IAM) pour toutes les opérations AWS Glue. En attachant une politique, vous pouvez accorder des autorisations pour créer, consulter ou modifier une ressource AWS Glue telle qu'une table dans le AWS Glue Data Catalog.

Exemples de politiques basées sur l'identité pour Glue AWS

Pour consulter des exemples de politiques basées sur l'identité de AWS Glue, consultez. Exemples de politiques basées sur l'identité pour Glue AWS

Politiques basées sur les ressources au sein de Glue AWS

Prise en charge des politiques basées sur les ressources : partielle

Les politiques basées sur les ressources sont des documents de politique JSON que vous attachez à une ressource. Par exemple, les politiques de confiance de rôle IAM et les politiques de compartiment Amazon S3 sont des politiques basées sur les ressources. Dans les services qui sont compatibles avec les politiques basées sur les ressources, les administrateurs de service peuvent les utiliser pour contrôler l'accès à une ressource spécifique. Pour la ressource dans laquelle se trouve la politique, cette dernière définit quel type d'actions un principal spécifié peut effectuer sur cette ressource et dans quelles conditions. Vous devez spécifier un principal dans une politique basée sur les ressources. Les principaux peuvent inclure des comptes, des utilisateurs, des rôles, des utilisateurs fédérés ou. Services AWS

Pour permettre un accès intercompte, vous pouvez spécifier un compte entier ou des entités IAM dans un autre compte en tant que principal dans une politique basée sur les ressources. Pour plus d'informations, consultez Accès intercompte aux ressources dans IAM dans le Guide de l'utilisateur IAM.

> **Note**
>
> Vous ne pouvez utiliser qu'une politique de ressource AWS Glue pour gérer les autorisations des ressources du catalogue de données. Vous ne pouvez pas l'attacher à d'autres ressources AWS Glue telles que les tâches, les déclencheurs, les points de terminaison de développement, les crawlers ou les classificateurs.
> Une seule politique basée sur les ressources est autorisée par catalogue et sa taille est limitée à 10 Ko.

Dans AWS Glue, une politique de ressources est attachée à un catalogue, qui est un conteneur virtuel pour tous les types de ressources de catalogue de données mentionnés précédemment. Chaque AWS compte possède un seul catalogue dans une AWS région dont l'identifiant du catalogue est le même que celui du AWS compte. Vous ne pouvez pas supprimer ou modifier un catalogue.

Une politique de ressources est évaluée pour tous les appels d'API au catalogue où le principal de l'appelant est inclus dans le bloc `"Principal"` du document de politique.

Pour consulter des exemples de politiques basées sur les ressources de AWS Glue, consultez. Exemples de politiques basées sur les ressources pour Glue AWS

Actions politiques pour AWS Glue

Prend en charge les actions de politique : oui

Les administrateurs peuvent utiliser les politiques AWS JSON pour spécifier qui a accès à quoi. C'est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

L'élément `Action` d'une politique JSON décrit les actions que vous pouvez utiliser pour autoriser ou refuser l'accès à une politique. Intégration d'actions dans une politique afin d'accorder l'autorisation d'exécuter les opérations associées.

Pour consulter la liste des actions AWS Glue, consultez la section Actions définies par AWS Glue dans le Service Authorization Reference.

Les actions politiques dans AWS Glue utilisent le préfixe suivant avant l'action :

```
glue
```

Pour indiquer plusieurs actions dans une seule déclaration, séparez-les par des virgules.

```
"Action": [
      "glue:action1",
      "glue:action2"
         ]
```

Vous pouvez aussi spécifier plusieurs actions à l'aide de caractères génériques (*). Par exemple, pour spécifier toutes les actions qui commencent par le mot `Get`, incluez l'action suivante :

```
"Action": "glue:Get*"
```

Pour afficher des exemples de stratégies, veuillez consulter AWS Exemples de politiques de contrôle d'accès Glue.

Ressources politiques pour AWS Glue

Prend en charge les ressources de politique : oui

Les administrateurs peuvent utiliser les politiques AWS JSON pour spécifier qui a accès à quoi. C'est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

L'élément de politique JSON `Resource` indique le ou les objets auxquels l'action s'applique. Il est recommandé de définir une ressource à l'aide de son Amazon Resource Name (ARN). Pour les actions qui ne sont pas compatibles avec les autorisations de niveau ressource, utilisez un caractère générique (*) afin d'indiquer que l'instruction s'applique à toutes les ressources.

```
"Resource": "*"
```

Pour plus d'informations sur la façon de contrôler l'accès aux ressources AWS Glue à ARNs l'aide deSpécification de AWS Glue la ressource ARNs.

Pour consulter la liste des types de ressources AWS Glue et leurs caractéristiques ARNs, consultez la section Ressources définies par AWS Glue dans le Service Authorization Reference. Pour savoir quelles actions vous pouvez utiliser pour spécifier l'ARN de chaque ressource, consultez la section Actions définies par AWS Glue.

Clés de conditions de politique pour AWS Glue

Prend en charge les clés de condition de politique spécifiques au service : oui

Les administrateurs peuvent utiliser les politiques AWS JSON pour spécifier qui a accès à quoi. C'est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

L'élément `Condition` indique à quel moment les instructions s'exécutent en fonction de critères définis. Vous pouvez créer des expressions conditionnelles qui utilisent des opérateurs de condition, tels que les signes égal ou inférieur à, pour faire correspondre la condition de la politique aux valeurs de la demande. Pour voir toutes les clés de condition AWS globales, voir les clés de contexte de condition AWS globales dans le guide de l'utilisateur IAM.

Pour consulter la liste des clés de condition AWS Glue, voir Clés de condition pour AWS Glue dans le Service Authorization Reference. Pour savoir avec quelles actions et ressources vous pouvez utiliser une clé de condition, consultez Actions définies par AWS Glue.

Pour afficher des exemples de stratégies, veuillez consulter Contrôler les paramètres à l'aide de clés de condition ou de contexte.

ACLs dans AWS Glue

Supports ACLs : Non

Les listes de contrôle d'accès (ACLs) contrôlent les principaux (membres du compte, utilisateurs ou rôles) autorisés à accéder à une ressource. ACLs sont similaires aux politiques basées sur les ressources, bien qu'elles n'utilisent pas le format de document de politique JSON.

ABAC avec Glue AWS

Prend en charge ABAC (identifications dans les politiques) : partiellement

Le contrôle d'accès par attributs (ABAC) est une stratégie d'autorisation qui définit les autorisations en fonction des attributs nommés balise. Vous pouvez associer des balises aux entités et aux AWS ressources IAM, puis concevoir des politiques ABAC pour autoriser les opérations lorsque la balise du principal correspond à la balise de la ressource.

Pour contrôler l'accès basé sur des étiquettes, vous devez fournir les informations d'étiquette dans l'élément de condition d'une politique utilisant les clés de condition `aws:ResourceTag/`*`key-name`*, `aws:RequestTag/`*`key-name`* ou `aws:TagKeys`.

Si un service prend en charge les trois clés de condition pour tous les types de ressources, alors la valeur pour ce service est Oui. Si un service prend en charge les trois clés de condition pour certains types de ressources uniquement, la valeur est Partielle.

Pour plus d'informations sur ABAC, consultez Définition d'autorisations avec l'autorisation ABAC dans le Guide de l'utilisateur IAM. Pour accéder à un didacticiel décrivant les étapes de configuration de l'ABAC, consultez Utilisation du contrôle d'accès par attributs (ABAC) dans le Guide de l'utilisateur IAM.

> **Important**
>
> Les clés de contexte de condition s'appliquent uniquement aux actions d'API AWS Glue sur des crawlers, des tâches, des déclencheurs et des points de terminaison de développement. Pour plus d'informations sur les opérations d'API concernées, consultez la section Clés de condition pour AWS Glue.
>
> Les opérations d'API du catalogue de données AWS Glue ne prennent actuellement pas en charge les clés de contexte de condition globales `aws:referer` et `aws:UserAgent`.

Pour visualiser un exemple de politique basée sur l'identité permettant de limiter l'accès à une ressource en fonction des balises de cette ressource, consultez Octoyer l'accès à l'aide de balises.

Utiliser des informations d'identification temporaires avec AWS Glue

Prend en charge les informations d'identification temporaires : oui

Les informations d'identification temporaires fournissent un accès à court terme aux AWS ressources et sont automatiquement créées lorsque vous utilisez la fédération ou que vous changez de rôle. AWS recommande de générer dynamiquement des informations d'identification temporaires au lieu d'utiliser des clés d'accès à long terme. Pour plus d'informations, consultez Informations d'identification de sécurité temporaires dans IAM et Services AWS compatibles avec IAM dans le Guide de l'utilisateur IAM.

Autorisations principales interservices pour AWS Glue

Prend en charge les sessions d'accès direct (FAS) : oui

Les sessions d'accès direct (FAS) utilisent les autorisations du principal appelant et Service AWS, combinées Service AWS à la demande d'envoi de demandes aux services en aval. Pour plus de détails sur la politique relative à la transmission de demandes FAS, consultez la section Sessions de transmission d'accès.

Rôles de service pour AWS Glue

Prend en charge les rôles de service : oui

Un rôle de service est un rôle IAM qu'un service endosse pour accomplir des actions en votre nom. Un administrateur IAM peut créer, modifier et supprimer un rôle de service à partir d'IAM. Pour plus d'informations, consultez Création d'un rôle pour la délégation d'autorisations à un Service AWS dans le Guide de l'utilisateur IAM.

> **Warning**
>
> La modification des autorisations associées à un rôle de service peut perturber les fonctionnalités de AWS Glue. Modifiez les rôles de service uniquement lorsque AWS Glue fournit des instructions à cet effet.

Pour obtenir des instructions détaillées sur la création d'un rôle de service pour AWS Glue, reportez-vous Étape 1 : créer une politique IAM pour le service AWS Glue aux sections etÉtape 2 : créer un rôle IAM pour AWS Glue.

Rôles liés à un service pour Glue AWS

Prend en charge les rôles liés à un service : non

Un rôle lié à un service est un type de rôle de service lié à un. Service AWS Le service peut endosser le rôle afin d'effectuer une action en votre nom. Les rôles liés à un service apparaissent dans votre Compte AWS répertoire et appartiennent au service. Un administrateur IAM peut consulter, mais ne peut pas modifier, les autorisations concernant les rôles liés à un service.

Pour plus d'informations sur la création ou la gestion des rôles liés à un service, consultez Services AWS qui fonctionnent avec IAM. Recherchez un service dans le tableau qui inclut un `Yes` dans la colonne Rôle lié à un service. Choisissez le lien Oui pour consulter la documentation du rôle lié à ce service.

Configuration des autorisations IAM pour AWS Glue

Vous utilisez Gestion des identités et des accès AWS (IAM) pour définir les politiques et les rôles AWS Glue utilisés pour accéder aux ressources. Les étapes suivantes vous guident à travers les différentes options de configuration des autorisations pour AWS Glue. En fonction de vos besoins professionnels, vous pouvez avoir besoin d'accroître ou de limiter l'accès à vos ressources.

> **Note**
>
> Pour commencer à utiliser les autorisations IAM de base pour la AWS Glue place, consultezConfiguration des autorisations IAM pour AWS Glue.

1. Créer une politique IAM pour le service AWS Glue : créez une politique de service qui autorise l'accès aux ressources AWS Glue.
2. Créer un rôle IAM pour AWS Glue : créez un rôle IAM et attachez la politique du service AWS Glue et une politique pour vos ressources Amazon Simple Storage Service (Amazon S3) utilisées par AWS Glue.
3. Attacher une politique aux utilisateurs ou groupes ayant accès à AWS Glue : attachez des politiques à tous les utilisateurs ou groupes qui se connectent à la console AWS Glue.

4. Créer une politique IAM pour les blocs-notes : créez une politique de serveur de bloc-notes à utiliser lors de la création de serveurs de bloc-notes sur les points de terminaison de développement.
5. Créer un rôle IAM pour les blocs-notes : créez un rôle IAM et attachez la politique de serveur de bloc-notes.
6. Création d'une politique IAM pour les blocs-notes Amazon SageMaker AI : créez une politique IAM à utiliser lors de la création de blocs-notes Amazon SageMaker AI sur les terminaux de développement.
7. Création d'un rôle IAM pour les blocs-notes Amazon SageMaker AI : créez un rôle IAM et joignez la politique d'octroi d'autorisations lors de la création de blocs-notes Amazon SageMaker AI sur les terminaux de développement.

Étape 1 : créer une politique IAM pour le service AWS Glue

Toute opération qui accède aux données d'une autre AWS ressource, telle que l'accès à vos objets dans Amazon S3, AWS Glue nécessite une autorisation pour accéder à la ressource en votre nom. Vous fournissez ces autorisations en utilisant Gestion des identités et des accès AWS (IAM).

> **Note**
>
> Vous pouvez ignorer cette étape si vous utilisez la politique AWS gérée**`AWSGlueServiceRole`**.

Au cours de cette étape, vous créez une politique similaire à `AWSGlueServiceRole`. Vous pouvez consulter la version la plus récente de `AWSGlueServiceRole` sur la console IAM.

Pour créer une politique IAM pour AWS Glue

Cette politique autorise certaines actions Amazon S3 à gérer des ressources de votre compte qui sont nécessaires pour AWS Glue lorsque celui-ci endosse le rôle à l'aide de cette politique. Certaines des ressources spécifiées dans cette politique font référence aux noms par défaut utilisés AWS Glue pour les compartiments Amazon S3, les scripts ETL Amazon S3, les CloudWatch journaux et les ressources Amazon EC2. Pour plus de simplicité, AWS Glue écrit certains objets Amazon S3 dans des compartiments de votre compte portant le préfixe `aws-glue-*` par défaut.

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.

2. Dans le volet de navigation de gauche, choisissez Politiques.
3. Sélectionnez Créer une politique.
4. Sur l'écran Créer une politique, accédez à un onglet pour modifier le fichier JSON. Créez un document de politique avec les instructions JSON suivantes, puis sélectionnez Examiner une politique.

> **Note**
>
> Ajoutez les permissions requises pour les ressources Amazon S3. Vous pouvez vouloir définir la portée de la section des ressources de votre politique d'accès uniquement aux ressources qui sont requises.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:*",
        "s3:GetBucketLocation",
        "s3:ListBucket",
        "s3:ListAllMyBuckets",
        "s3:GetBucketAcl",
        "ec2:DescribeVpcEndpoints",
        "ec2:DescribeRouteTables",
        "ec2:CreateNetworkInterface",
        "ec2:DeleteNetworkInterface",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeSubnets",
        "ec2:DescribeVpcAttribute",
        "iam:ListRolePolicies",
        "iam:GetRole",
        "iam:GetRolePolicy",
        "cloudwatch:PutMetricData"
      ],
      "Resource": [
```

```
        "*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:CreateBucket",
        "s3:PutBucketPublicAccessBlock"
      ],
      "Resource": [
        "arn:aws:s3:::aws-glue-*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:DeleteObject"
      ],
      "Resource": [
        "arn:aws:s3:::aws-glue-*/*",
        "arn:aws:s3:::*/*aws-glue-*/*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject"
      ],
      "Resource": [
        "arn:aws:s3:::crawler-public*",
        "arn:aws:s3:::aws-glue-*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents",
        "logs:AssociateKmsKey"
      ],
      "Resource": [
```

```
        "arn:aws:logs:*:*:log-group:/aws-glue/*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "ec2:CreateTags",
        "ec2:DeleteTags"
      ],
      "Condition": {
        "ForAllValues:StringEquals": {
          "aws:TagKeys": [
            "aws-glue-service-resource"
          ]
        }
      },
      "Resource": [
        "arn:aws:ec2:*:*:network-interface/*",
        "arn:aws:ec2:*:*:security-group/*",
        "arn:aws:ec2:*:*:instance/*"
      ]
    }
  ]
}
```

Le tableau suivant décrit les autorisations accordées par cette politique.

Action	Ressource	Description
`"glue:*"`	`"*"`	Accorde les autorisations pour exécuter toutes les opérations d'API AWS Glue.
`"s3:GetBucketLocation", "s3:ListBucket", "s3:ListAllMyBuckets", "s3:GetBucketAcl",`	`"*"`	Permet de répertorier les compartiments Amazon S3 des crawlers, des tâches, des points de terminaison de développement et des serveurs de bloc-notes.

Action	Ressource	Description
`"ec2:DescribeVpcEndpoints", "ec2:DescribeRouteTables", "ec2:CreateNetworkInterface", "ec2:DeleteNetworkInterface", "ec2:DescribeNetworkInterfaces", "ec2:DescribeSecurityGroups", "ec2:DescribeSubnets", "ec2:DescribeVpcAttribute",`	`"*"`	Permet de configurer des éléments du réseau Amazon EC2, tels que des clouds privés virtuels (VPCs) lors de l'exécution de tâches, de robots d'exploration et de points de terminaison de développement.
`"iam:ListRolePolicies", "iam:GetRole", "iam:GetRolePolicy"`	`"*"`	Permet de répertorier les rôles IAM des crawlers, des tâches, des points de terminaison de développement et des serveurs de bloc-notes.
`"cloudwatch:PutMetricData"`	`"*"`	Permet d'écrire CloudWatch des métriques pour les tâches.

Action	Ressource	Description
`"s3:CreateBucket", "s3:PutBucketPublicAccessBlock"`	`"arn:aws:s3:::aws-glue-*"`	Permet de créer de compartiments Amazon S3 dans votre compte à partir de tâches et de serveurs de bloc-notes. Convention de dénomination : utilise les dossiers Amazon S3 nommés aws-glue-. Active AWS Glue pour créer des compartiments qui bloquent l'accès public.
`"s3:GetObject", "s3:PutObject", "s3:DeleteObject"`	`"arn:aws:s3:::aws-glue-*/*", "arn:aws:s3:::*/*aws-glue-*/*"`	Permet d'obtenir, d'ajouter et de supprimer des objets Amazon S3 dans votre compte lors du stockage d'objets, tel que des scripts ETL et des emplacements de serveurs de bloc-notes. Convention de dénomination : accorde l'autorisation aux compartiments ou dossiers Amazon S3 dont les noms portent le préfixe aws-glue-.

Action	Ressource	Description
`"s3:GetObject"`	`"arn:aws:s3:::crawler-public*", "arn:aws:s3:::aws-glue-*"`	Permet d'obtenir des objets Amazon S3 utilisés par des exemples et des didacticiels d'crawlers et de tâches. Convention de dénominat ion : les noms de compartim ents Amazon S3 dont le nom commence par crawler-public et aws-glue-.
`"logs:CreateLogGroup", "logs:CreateLogStream", "logs:PutLogEvents"`	`"arn:aws:logs:*:*:log-group:/aws-glue/*"`	Permet d'écrire des journaux dans CloudWatch Logs. Convention de dénominat ion : AWS Glue écrit des journaux dans des groupes de journaux dont le nom commence par aws-glue.
`"ec2:CreateTags", "ec2:DeleteTags"`	`"arn:aws:ec2:*:*:network-interface/*", "arn:aws:ec2:*:*:security-group/*", "arn:aws:ec2:*:*:instance/*"`	Permet de baliser des ressources Amazon EC2 créées pour des points de terminaison de développe ment. Convention de dénominat ion : AWS Glue étiquette les interfaces réseau, les groupes de sécurité et les instances Amazon EC2 avec. aws-glue-service-r esource

5. Sur l'écran Review policy (Examiner la politique), saisissez votre Policy Name (Nom de politique), par exemple, GlueServiceRolePolicy. Saisissez éventuellement une description, et lorsque vous êtes satisfait de la politique, sélectionnez Créer une politique.

Étape 2 : créer un rôle IAM pour AWS Glue

Vous devez accorder à votre rôle IAM des autorisations dont AWS Glue peut disposer lors de l'appel d'autres services en votre nom. Cela inclut l'accès à Amazon S3 pour toutes les sources, cibles, scripts et répertoires temporaires que vous utilisez avec AWS Glue. Une autorisation est requise pour les crawlers, les tâches et les points de terminaison de développement.

Vous fournissez ces autorisations en utilisant Gestion des identités et des accès AWS (IAM). Ajoutez une politique au rôle IAM que vous transmettez à AWS Glue.

Pour créer un rôle IAM dans l'éditeur de tâches

1. Lorsque vous créez une tâche dans la AWS Glue console, recherchez la section des rôles.
2. Choisissez Create New Role (Créer un nouveau rôle).
3. Un formulaire de création de rôle intégré s'ouvre, vous permettant de :
   - Spécifiez le nom du rôle ; par exemple,`AWSGlueServiceRoleDefault`.
   - La politique gérée `AWSGlueServiceRole` est automatiquement sélectionnée.
   - Passez en revue la politique de confiance pour assumer le rôle.
   - Ajoutez des balises facultatives pour les métadonnées.
4. Choisissez Créer un rôle.
5. Le rôle nouvellement créé est automatiquement sélectionné pour votre tâche.

Vous pouvez également utiliser la console IAM pour créer le rôle :

Pour créer un rôle IAM pour AWS Glue utiliser la console IAM

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le volet de navigation de gauche, choisissez Rôles.
3. Choisissez Créer un rôle.

4. Choisissez Service AWS comme type d'entité de confiance. Ensuite, pour le service ou le cas d'utilisation, recherchez et choisissez AWS Glue. Choisissez Suivant.
5. Sur la page Ajouter des autorisations, choisissez les politiques qui contiennent les autorisations requises ; par exemple, la politique AWS gérée `AWSGlueServiceRole` pour les AWS Glue autorisations générales et la politique AWS gérée AmazonS3 FullAccess pour l'accès aux ressources Amazon S3. Ensuite, sélectionnez Suivant.

> **Note**
>
> Assurez-vous que l'une des politiques de ce rôle accorde des autorisations à vos sources et cibles Amazon S3. Vous pouvez vouloir fournir votre propre politique pour l'accès à certaines ressources Amazon S3. Les sources de données nécessitent les autorisations `s3:ListBucket` et `s3:GetObject`. Les sources de données nécessitent les autorisations `s3:ListBucket`, `s3:PutObject` et `s3:DeleteObject`. Pour plus d'informations sur la création d'une politique Amazon S3 pour vos ressources, consultez Spécification des ressources d'une politique. Pour obtenir un exemple de politique Amazon S3, consultez notre billet de blog Writing IAM Policies: How to Grant Access to an Amazon S3 Bucket.
> Si vous prévoyez d'accéder à des sources et cibles Amazon S3 chiffrées avec SSE-KMS, attachez une politique permettant aux crawlers, tâches et points de terminaison de développement AWS Glue de déchiffrer les données. Pour plus d'informations, consultez la section Protection des données à l'aide du chiffrement côté serveur avec des clés AWS KMS gérées (SSE-KMS).
> Voici un exemple.
>
> JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "kms:Decrypt"
      ],
      "Resource": [
        "arn:aws:kms:*:111122223333:key/key-id"
      ]
```

```
    }
  ]
}
```

6. Donnez un nom à votre rôle et ajoutez une description (facultatif), puis passez en revue la politique de confiance et les autorisations. Pour Role name (Nom du rôle), indiquez le nom de votre rôle, par exemple, `AWSGlueServiceRoleDefault`. Créez le rôle avec un nom portant comme préfixe la chaîne `AWSGlueServiceRole` afin de permettre au rôle d'être transmis entre les utilisateurs de la console et le service. Les politiques fournies par AWS Glue prévoient que les rôles de service IAM commencent par `AWSGlueServiceRole`. Sinon, vous devez ajouter une politique accordant à vos utilisateurs l'autorisation `iam:PassRole` afin que les rôles IAM correspondent à votre convention de dénomination. Choisissez Create Role (Créer un rôle).

   > **Note**
   >
   > Lorsque vous créez un bloc-notes avec un rôle, ce rôle est ensuite transmis à des séances interactives afin que le même rôle puisse être utilisé aux deux endroits. En tant que tel, l'autorisation `iam:PassRole` doit faire partie de la politique du rôle.
   > Créez une politique pour votre rôle à l'aide de l'exemple suivant. Remplacez le numéro de compte par le vôtre et le nom du rôle.
   >
   > JSON
   >
   > ```
   > {
   >   "Version":"2012-10-17",
   >   "Statement": [
   >     {
   >       "Effect": "Allow",
   >       "Action": "iam:PassRole",
   >       "Resource": "arn:aws:iam::090000000210:role/<role_name>"
   >     }
   >   ]
   > }
   > ```

7. Ajoutez des balises à votre rôle (facultatif). Les balises sont des paires clé-valeur que vous pouvez ajouter aux AWS ressources pour identifier, organiser ou rechercher des ressources. Puis, choisissez Créer un rôle.

Étape 3 : attacher une politique aux utilisateurs ou aux groupes accédant à AWS Glue

L'administrateur doit attribuer des autorisations à tous les utilisateurs, groupes ou rôles à l'aide de la AWS Glue console ou AWS Command Line Interface (AWS CLI). Vous fournissez ces autorisations grâce à Gestion des identités et des accès AWS (IAM) par le biais des politiques. Cette étape décrit l'attribution d'autorisations à des utilisateurs ou groupes.

Une fois cette étape terminée, les politiques suivantes seront associées à votre utilisateur ou groupe :

- La politique AWS gérée `AWSGlueConsoleFullAccess` ou la politique personnalisée GlueConsoleAccessPolicy
- **`AWSGlueConsoleSageMakerNotebookFullAccess`**
- **`CloudWatchLogsReadOnlyAccess`**
- **`AWSCloudFormationReadOnlyAccess`**
- **`AmazonAthenaFullAccess`**

Associer une politique en ligne et l'intégrer à un utilisateur ou à un groupe

Vous pouvez associer une politique AWS gérée ou une politique intégrée à un utilisateur ou à un groupe pour accéder à la AWS Glue console. Certaines des ressources spécifiées dans cette politique font référence aux noms par défaut utilisés AWS Glue pour les compartiments Amazon S3, les scripts ETL Amazon S3 CloudFormation, CloudWatch les journaux et les ressources Amazon EC2. Pour plus de simplicité, AWS Glue écrit certains objets Amazon S3 dans des compartiments de votre compte portant le préfixe `aws-glue-*` par défaut.

> **Note**
>
> Vous pouvez ignorer cette étape si vous utilisez la politique AWS gérée **`AWSGlueConsoleFullAccess`**.

> **Important**
>
> AWS Glue a besoin d'une autorisation pour endosser un rôle utilisé pour effectuer des tâches en votre nom. Pour cela, vous ajoutez les autorisations **`iam:PassRole`** à vos utilisateurs ou groupes AWS Glue. Cette politique accorde l'autorisation aux rôles commençant par `AWSGlueServiceRole` pour les rôles de service AWS Glue et

`AWSGlueServiceNotebookRole` pour les rôles qui sont requis lorsque vous créez un serveur de bloc-notes. Vous pouvez également créer votre propre politique pour les autorisations `iam:PassRole`, conforme à votre convention de dénomination. Conformément aux meilleures pratiques de sécurité, il est recommandé de restreindre l'accès en renforçant les politiques afin de restreindre davantage l'accès aux compartiments et aux groupes de Amazon CloudWatch journaux Amazon S3. Pour obtenir un exemple de politique Amazon S3, consultez notre billet de blog Writing IAM Policies: How to Grant Access to an Amazon S3 Bucket.

Au cours de cette étape, vous créez une politique similaire à `AWSGlueConsoleFullAccess`. Vous pouvez consulter la version la plus récente de `AWSGlueConsoleFullAccess` sur la console IAM.

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le volet de navigation, choisissez Utilisateurs ou Groupes d'utilisateurs.
3. Dans la liste, sélectionnez le nom de l'utilisateur ou du groupe auquel intégrer une politique.
4. Sélectionnez l'onglet Autorisations et, si nécessaire, développez la section Politiques d'autorisations.
5. Cliquez sur le lien Add Inline policy (Ajouter une politique en ligne).
6. Sur l'écran Créer une politique, accédez à un onglet pour modifier le fichier JSON. Créez un document de politique avec les instructions JSON suivantes, puis sélectionnez Examiner une politique.

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:*",
        "redshift:DescribeClusters",
        "redshift:DescribeClusterSubnetGroups",
        "iam:ListRoles",
        "iam:ListUsers",
```

```
        "iam:ListGroups",
        "iam:ListRolePolicies",
        "iam:GetRole",
        "iam:GetRolePolicy",
        "iam:ListAttachedRolePolicies",
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeSubnets",
        "ec2:DescribeVpcs",
        "ec2:DescribeVpcEndpoints",
        "ec2:DescribeRouteTables",
        "ec2:DescribeVpcAttribute",
        "ec2:DescribeKeyPairs",
        "ec2:DescribeInstances",
        "rds:DescribeDBInstances",
        "rds:DescribeDBClusters",
        "rds:DescribeDBSubnetGroups",
        "s3:ListAllMyBuckets",
        "s3:ListBucket",
        "s3:GetBucketAcl",
        "s3:GetBucketLocation",
        "cloudformation:DescribeStacks",
        "cloudformation:GetTemplateSummary",
        "dynamodb:ListTables",
        "kms:ListAliases",
        "kms:DescribeKey",
        "cloudwatch:GetMetricData",
        "cloudwatch:ListDashboards"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": [
        "arn:aws:s3:::*/*aws-glue-*/*",
        "arn:aws:s3:::aws-glue-*"
      ]
    },
    {
```

```
      "Effect": "Allow",
      "Action": [
        "tag:GetResources"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:CreateBucket",
        "s3:PutBucketPublicAccessBlock"
      ],
      "Resource": [
        "arn:aws:s3:::aws-glue-*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "logs:GetLogEvents"
      ],
      "Resource": [
        "arn:aws:logs:*:*:/aws-glue/*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "cloudformation:CreateStack",
        "cloudformation:DeleteStack"
      ],
      "Resource": "arn:aws:cloudformation:*:*:stack/aws-glue*/*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "ec2:RunInstances"
      ],
      "Resource": [
        "arn:aws:ec2:*:*:instance/*",
        "arn:aws:ec2:*:*:key-pair/*",
        "arn:aws:ec2:*:*:image/*",
```

```
        "arn:aws:ec2:*:*:security-group/*",
        "arn:aws:ec2:*:*:network-interface/*",
        "arn:aws:ec2:*:*:subnet/*",
        "arn:aws:ec2:*:*:volume/*"
      ]
    },
    {
      "Action": [
        "iam:PassRole"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:iam::*:role/AWSGlueServiceRole*",
      "Condition": {
        "StringLike": {
          "iam:PassedToService": [
            "glue.amazonaws.com"
          ]
        }
      }
    },
    {
      "Action": [
        "iam:PassRole"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:iam::*:role/AWSGlueServiceNotebookRole*",
      "Condition": {
        "StringLike": {
          "iam:PassedToService": [
            "ec2.amazonaws.com"
          ]
        }
      }
    },
    {
      "Action": [
        "iam:PassRole"
      ],
      "Effect": "Allow",
      "Resource": [
        "arn:aws:iam::*:role/service-role/AWSGlueServiceRole*"
      ],
      "Condition": {
        "StringLike": {
```

```
          "iam:PassedToService": [
            "glue.amazonaws.com"
          ]
        }
      }
    }
  ]
}
```

Le tableau suivant décrit les autorisations accordées par cette politique.

Action	Ressource	Description
`"glue:*"`	`"*"`	Accorde les autorisations pour exécuter toutes les opérations d'API AWS Glue. Si vous avez déjà créé votre politique sans l'action `"glue:*"`, vous devez ajouter les autorisations individuelles suivantes à votre politique : • « colle : ListCrawlers » • « colle : BatchGetCrawlers » • « colle : ListTriggers » • « colle : BatchGetTriggers » • « colle : ListDevEndpoints » • « colle : BatchGetDevEndpoints » • « colle : ListJobs » • « colle : BatchGetJobs »
`"redshift:DescribeClusters", "redshift:DescribeClusterSubnetGroups"`	`"*"`	Permet de créer des connexions à Amazon Redshift.

Action	Ressource	Description
`"iam:ListRoles", "iam:ListRolePolicies", "iam:GetRole", "iam:GetRolePolicy", "iam:ListAttachedRolePolicies"`	`"*"`	Permet de répertorier les rôles IAM lors de l'utilisation d'crawlers, de tâches, de points de terminaison de développement et de serveurs de bloc-notes.
`"ec2:DescribeSecurityGroups", "ec2:DescribeSubnets", "ec2:DescribeVpcs", "ec2:DescribeVpcEndpoints", "ec2:DescribeRouteTables", "ec2:DescribeVpcAttribute", "ec2:DescribeKeyPairs", "ec2:DescribeInstances"`	`"*"`	Permet de configurer des éléments du réseau Amazon EC2, par exemple lors de l'exécution de tâches VPCs, de robots d'exploration et de points de terminaison de développement.
`"rds:DescribeDBInstances"`	`"*"`	Permet de créer des connexions à Amazon RDS.
`"s3:ListAllMyBuckets", "s3:ListBucket", "s3:GetBucketAcl", "s3:GetBucketLocation"`	`"*"`	Permet de répertorier les compartiments Amazon S3 lors de l'utilisation d'crawlers, de tâches, de points de terminaison de développement et de serveurs de bloc-notes.
`"dynamodb:ListTables"`	`"*"`	Autorise la création d'une liste des tables DynamoDB.

Action	Ressource	Description
`"kms:ListAliases", "kms:DescribeKey"`	`"*"`	Permet l'utilisation de clés KMS.
`"cloudwatch:GetMetricData", "cloudwatch:ListDashboards"`	`"*"`	Permet de travailler avec CloudWatch des métriques.
`"s3:GetObject", "s3:PutObject"`	`"arn:aws:s3::: aws-glue-*/*", "arn:aws:s3::: */*aws-glue-*/*", "arn:aws:s3::: aws-glue-*"`	Permet d'obtenir et d'ajouter des objets Amazon S3 dans votre compte lors du stockage d'objets, tel que des scripts ETL et des emplacements de serveurs de bloc-notes. Convention de dénomination : accorde l'autorisation aux compartiments ou dossiers Amazon S3 dont les noms portent le préfixe aws-glue-.
`"tag:GetResources"`	`"*"`	Permet de récupérer les AWS tags.

Action	Ressource	Description
`"s3:CreateBucket", "s3:PutBucketPublicAccessBlock"`	`"arn:aws:s3::: aws-glue-*"`	Permet de créer un compartiment Amazon S3 dans votre compte lors du stockage d'objets, tel que des scripts ETL et des emplacements de serveurs de bloc-notes Convention de dénomination : accorde l'autorisation aux compartiments Amazon S3 dont les noms portent le préfixe aws-glue-. Active AWS Glue pour créer des compartiments qui bloquent l'accès public.
`"logs:GetLogEvents"`	`"arn:aws:logs:*:*: /aws-glue/*"`	Permet de récupérer les CloudWatch journaux. Convention de dénomination : AWS Glue écrit des journaux dans des groupes de journaux dont le nom commence par aws-glue-.

Action	Ressource	Description
`"cloudformation:CreateStack", "cloudformation:DeleteStack"`	`"arn:aws:cloudformation:*:*:stack/aws-glue*/*"`	Permet de gérer les CloudFormation piles lorsque vous travaillez avec des serveurs portables. Convention de dénomination : AWS Glue crée des piles dont le nom commence par aws-glue.
`"ec2:RunInstances"`	`"arn:aws:ec2:*:*:instance/*", "arn:aws:ec2:*:*:key-pair/*", "arn:aws:ec2:*:*:image/*", "arn:aws:ec2:*:*:security-group/*", "arn:aws:ec2:*:*:network-interface/*", "arn:aws:ec2:*:*:subnet/*", "arn:aws:ec2:*:*:volume/*"`	Autorise l'exécution de points de terminaison de développement et de serveurs de blocs-notes.
`"iam:PassRole"`	`"arn:aws:iam::*:role/ AWSGlueServiceRole*"`	Permet à AWS Glue de disposer de l'autorisation `PassRole` pour les rôles commençant par `AWSGlueServiceRole` .

Action	Ressource	Description
`"iam:PassRole"`	`"arn:aws:iam::*:role/ AWSGlueServiceNotebookRole*"`	Permet à Amazon EC2 de disposer de l'autorisation `PassRole` pour les rôles commençant par `AWSGlueServiceNotebookRole` .
`"iam:PassRole"`	`"arn:aws:iam::*:role/service-role/AWSGlueServiceRole*"`	Permet à AWS Glue de disposer de l'autorisation `PassRole` pour les rôles commençant par `service-role/AWSGlueServiceRole` .

7. Sur l'écran Réviser la politique, entrez le nom de la politique, par exemple GlueConsoleAccessPolicy. Lorsque vous êtes satisfait de la politique, sélectionnez Create policy (Créer une politique). Assurez-vous qu'aucune erreur ne s'affiche dans un cadre rouge en haut de l'écran. Corrigez les erreurs signalées.

> **Note**
>
> Si l'option Use autoformatting est sélectionnée, la politique est reformatée chaque fois que vous ouvrez une politique ou sélectionnez Validate Policy.

Pour joindre la politique AWSGlue ConsoleFullAccess gérée

Vous pouvez attacher la politique `AWSGlueConsoleFullAccess` afin d'accorder les autorisations requises par l'utilisateur de la console AWS Glue.

> **Note**
>
> Vous pouvez ignorer cette étape si vous avez créé votre propre politique pour l'accès à la console AWS Glue.

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, choisissez Politiques.
3. Dans la liste des politiques, cochez la case à côté de AWSGlueConsoleFullAccess. Vous pouvez utiliser le menu Filtre et la zone de recherche pour filtrer la liste de politiques.
4. Sélectionnez Policy Actions (Actions de politique), puis sélectionnez Attach (Attacher).
5. Sélectionnez l'utilisateur auquel attacher la politique. Vous pouvez utiliser le menu Filtre et la zone de recherche pour filtrer la liste des entités du principal. Après avoir choisi l'utilisateur auquel attacher la politique, sélectionnez Attacher la politique.

Pour attacher la politique gérée par **`AWSGlueConsoleSageMakerNotebookFullAccess`**

Vous pouvez associer la `AWSGlueConsoleSageMakerNotebookFullAccess` politique à un utilisateur pour gérer les blocs-notes SageMaker IA créés sur la AWS Glue console. Outre les autres autorisations de AWS Glue console requises, cette politique accorde l'accès aux ressources nécessaires à la gestion des blocs-notes SageMaker AI.

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, choisissez Politiques.
3. Dans la liste des politiques, cochez la case à côté de AWSGlueConsoleSageMakerNotebookFullAccess. Vous pouvez utiliser le menu Filtre et la zone de recherche pour filtrer la liste de politiques.
4. Sélectionnez Policy Actions (Actions de politique), puis sélectionnez Attach (Attacher).
5. Sélectionnez l'utilisateur auquel attacher la politique. Vous pouvez utiliser le menu Filtre et la zone de recherche pour filtrer la liste des entités du principal. Après avoir choisi l'utilisateur auquel attacher la politique, sélectionnez Attach policy (Attacher la politique).

Pour joindre la politique CloudWatchLogsReadOnlyAccess gérée

Vous pouvez associer la CloudWatchLogsReadOnlyAccesspolitique à un utilisateur pour consulter les journaux créés par AWS Glue la console CloudWatch Logs.

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.

2. Dans le panneau de navigation, choisissez Politiques.
3. Dans la liste des politiques, cochez la case à côté de CloudWatchLogsReadOnlyAccess. Vous pouvez utiliser le menu Filtre et la zone de recherche pour filtrer la liste de politiques.
4. Sélectionnez Policy Actions (Actions de politique), puis sélectionnez Attach (Attacher).
5. Sélectionnez l'utilisateur auquel attacher la politique. Vous pouvez utiliser le menu Filtre et la zone de recherche pour filtrer la liste des entités du principal. Après avoir choisi l'utilisateur auquel attacher la politique, sélectionnez Attach policy (Attacher la politique).

Pour joindre la politique AWSCloud FormationReadOnlyAccess gérée

Vous pouvez associer la AWSCloudFormationReadOnlyAccesspolitique à un utilisateur pour afficher les CloudFormation piles utilisées par celui-ci AWS Glue sur la CloudFormation console.

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, choisissez Politiques.
3. Dans la liste des politiques, cochez la case à côté de AWSCloudFormationReadOnlyAccess. Vous pouvez utiliser le menu Filtre et la zone de recherche pour filtrer la liste de politiques.
4. Sélectionnez Policy Actions (Actions de politique), puis sélectionnez Attach (Attacher).
5. Sélectionnez l'utilisateur auquel attacher la politique. Vous pouvez utiliser le menu Filtre et la zone de recherche pour filtrer la liste des entités du principal. Après avoir choisi l'utilisateur auquel attacher la politique, sélectionnez Attach policy (Attacher la politique).

Pour joindre la politique AmazonAthenaFullAccess gérée

Vous pouvez associer la AmazonAthenaFullAccesspolitique à un utilisateur pour qu'il puisse consulter les données Amazon S3 dans la console Athena.

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation, choisissez Politiques.
3. Dans la liste des politiques, cochez la case à côté de AmazonAthenaFullAccess. Vous pouvez utiliser le menu Filtre et la zone de recherche pour filtrer la liste de politiques.
4. Sélectionnez Policy Actions (Actions de politique), puis sélectionnez Attach (Attacher).

5. Sélectionnez l'utilisateur auquel attacher la politique. Vous pouvez utiliser le menu Filtre et la zone de recherche pour filtrer la liste des entités du principal. Après avoir choisi l'utilisateur auquel attacher la politique, sélectionnez Attach policy (Attacher la politique).

Étape 4 : créer une politique IAM pour les serveurs de blocs-notes

Si vous prévoyez d'utiliser des bloc-notes avec des points de terminaison de développement, vous devez spécifier des autorisations lorsque vous créez le serveur de bloc-notes. Vous fournissez ces autorisations à l'aide d' Gestion des identités et des accès AWS (IAM).

Cette politique autorise certaines actions Amazon S3 à gérer des ressources de votre compte qui sont nécessaires pour AWS Glue lorsque celui-ci endosse le rôle à l'aide de cette politique. Certaines des ressources spécifiées dans cette politique font référence aux noms par défaut utilisés par AWS Glue pour les compartiments Amazon S3, les scripts ETL Amazon S3 et les ressources Amazon EC2. Pour plus de simplicité, AWS Glue écrit par défaut certains objets Amazon S3 dans des compartiments de votre compte portant le préfixe `aws-glue-*`.

> **Note**
>
> Vous pouvez ignorer cette étape si vous utilisez la politique AWS gérée **`AWSGlueServiceNotebookRole`**.

Au cours de cette étape, vous créez une politique similaire à `AWSGlueServiceNotebookRole`. Vous pouvez consulter la version la plus récente de `AWSGlueServiceNotebookRole` sur la console IAM.

Pour créer une politique IAM pour les bloc-notes

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le volet de navigation de gauche, choisissez Politiques.
3. Sélectionnez Créer une politique.
4. Sur l'écran Créer une politique, accédez à un onglet pour modifier le fichier JSON. Créez un document de politique avec les instructions JSON suivantes, puis sélectionnez Examiner une politique.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:CreateDatabase",
        "glue:CreatePartition",
        "glue:CreateTable",
        "glue:DeleteDatabase",
        "glue:DeletePartition",
        "glue:DeleteTable",
        "glue:GetDatabase",
        "glue:GetDatabases",
        "glue:GetPartition",
        "glue:GetPartitions",
        "glue:GetTable",
        "glue:GetTableVersions",
        "glue:GetTables",
        "glue:UpdateDatabase",
        "glue:UpdatePartition",
        "glue:UpdateTable",
        "glue:GetJobBookmark",
        "glue:ResetJobBookmark",
        "glue:CreateConnection",
        "glue:CreateJob",
        "glue:DeleteConnection",
        "glue:DeleteJob",
        "glue:GetConnection",
        "glue:GetConnections",
        "glue:GetDevEndpoint",
        "glue:GetDevEndpoints",
        "glue:GetJob",
        "glue:GetJobs",
        "glue:UpdateJob",
        "glue:BatchDeleteConnection",
        "glue:UpdateConnection",
        "glue:GetUserDefinedFunction",
        "glue:UpdateUserDefinedFunction",
        "glue:GetUserDefinedFunctions",
```

```
        "glue:DeleteUserDefinedFunction",
        "glue:CreateUserDefinedFunction",
        "glue:BatchGetPartition",
        "glue:BatchDeletePartition",
        "glue:BatchCreatePartition",
        "glue:BatchDeleteTable",
        "glue:UpdateDevEndpoint",
        "s3:GetBucketLocation",
        "s3:ListBucket",
        "s3:ListAllMyBuckets",
        "s3:GetBucketAcl"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject"
      ],
      "Resource": [
        "arn:aws:s3:::crawler-public*",
        "arn:aws:s3:::aws-glue*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:PutObject",
        "s3:DeleteObject"
      ],
      "Resource": [
        "arn:aws:s3:::aws-glue*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "ec2:CreateTags",
        "ec2:DeleteTags"
      ],
      "Condition": {
        "ForAllValues:StringEquals": {
```

```
          "aws:TagKeys": [
            "aws-glue-service-resource"
          ]
        }
      },
      "Resource": [
        "arn:aws:ec2:*:*:network-interface/*",
        "arn:aws:ec2:*:*:security-group/*",
        "arn:aws:ec2:*:*:instance/*"
      ]
    }
  ]
}
```

Le tableau suivant décrit les autorisations accordées par cette politique.

Action	Ressource	Description
`"glue:*"`	`"*"`	Accorde les autorisations pour exécuter toutes les opérations d'API AWS Glue.
`"s3:GetBucketLocation", "s3:ListBucket", "s3:ListAllMyBuckets", "s3:GetBucketAcl"`	`"*"`	Permet de répertorier les compartiments Amazon S3 à partir des serveurs de bloc-notes.

Action	Ressource	Description
`"s3:GetObject"`	`"arn:aws:s3:::crawler-public*", "arn:aws:s3:::aws-glue-*"`	Permet d'obtenir des objets Amazon S3 utilisés par des exemples et des didactiels de bloc-notes. Convention de dénomination : les noms de compartiments Amazon S3 dont le nom commence par crawler-public et aws-glue-.
`"s3:PutObject", "s3:DeleteObject"`	`"arn:aws:s3:::aws-glue*"`	Permet d'ajouter et de supprimer des objets Amazon S3 dans votre compte à partir des bloc-notes. Convention de dénomination : utilise les dossiers Amazon S3 nommés aws-glue-.
`"ec2:CreateTags", "ec2:DeleteTags"`	`"arn:aws:ec2:*:*:network-interface/*", "arn:aws:ec2:*:*:security-group/*", "arn:aws:ec2:*:*:instance/*"`	Permet de baliser des ressources Amazon EC2 créées pour des serveurs de bloc-notes. Convention de dénomination : AWS Glue étiquette les instances Amazon EC2 avec. aws-glue-service-resource

5. Sur l'écran Review policy (Examiner la politique), saisissez votre Policy Name (Nom de politique), par exemple, GlueServiceNotebookPolicyDefault. Saisissez éventuellement une description, et lorsque vous êtes satisfait de la politique, sélectionnez Créer une politique.

Étape 5 : créer un rôle IAM pour les serveurs de blocs-notes

Si vous prévoyez d'utiliser des bloc-notes avec des points de terminaison de développement, vous devez accorder les autorisations de rôles IAM. Vous fournissez ces autorisations à l'aide d' Gestion des identités et des accès AWS IAM, via un rôle IAM.

> **Note**
>
> Lorsque vous créez un rôle IAM à l'aide de la console IAM, celle-ci crée automatiquement un profil d'instance et lui attribue le même nom qu'au rôle auquel il correspond.

Pour créer un rôle IAM pour des bloc-notes

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le volet de navigation de gauche, choisissez Rôles.
3. Choisissez Créer un rôle.
4. Pour le type de rôle, sélectionnez Service AWS , recherchez et sélectionnez EC2, sélectionnez le cas d'utilisation EC2, puis Suivant : autorisations.
5. Sur la page Joindre la politique d'autorisations, choisissez les politiques qui contiennent les autorisations requises ; par exemple, AWSGlueServiceNotebookRolepour les AWS Glue autorisations générales et la politique AWS gérée AmazonS3 FullAccess pour l'accès aux ressources Amazon S3. Choisissez ensuite Next: Review.

   > **Note**
   >
   > Assurez-vous que l'une des politiques de ce rôle accorde des autorisations à vos sources et cibles Amazon S3. Confirmez également que votre politique autorise un accès complet à l'emplacement dans lequel vous stockez votre bloc-notes lors de la création d'un serveur de bloc-notes. Vous pouvez vouloir fournir votre propre politique pour l'accès à certaines ressources Amazon S3. Pour plus d'informations sur la création

d'une politique Amazon S3 pour vos ressources, consultez Spécification des ressources d'une politique.
Si vous prévoyez d'accéder à des sources et cibles Amazon S3 chiffrées avec SSE-KMS, attachez une politique permettant aux bloc-notes de déchiffrer les données. Pour plus d'informations, consultez la section Protection des données à l'aide du chiffrement côté serveur avec des clés AWS KMS gérées (SSE-KMS).
Voici un exemple.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "kms:Decrypt"
      ],
      "Resource": [
        "arn:aws:kms:*:111122223333:key/key-id"
      ]
    }
  ]
}
```

6. Dans le champ Role name (Nom de rôle), saisissez un nom pour votre rôle. Créez le rôle avec un nom portant comme préfixe la chaîne `AWSGlueServiceNotebookRole` afin de permettre au rôle d'être transmis entre les utilisateurs de la console et le serveur de bloc-notes. Les politiques fournies par AWS Glue prévoient que les rôles de service IAM commencent par `AWSGlueServiceNotebookRole`. Sinon, vous devez ajouter une politique à vos utilisateurs accordant l'autorisation `iam:PassRole` aux rôles IAM de correspondre à votre convention de dénomination. Par exemple, saisissez `AWSGlueServiceNotebookRoleDefault`. Puis choisissez Create role (Créer un rôle).

Étape 6 : créer une politique IAM pour les ordinateurs portables dotés d' SageMaker intelligence artificielle

Si vous envisagez d'utiliser des blocs-notes SageMaker AI avec des terminaux de développement, vous devez spécifier des autorisations lors de la création du bloc-notes. Vous fournissez ces autorisations à l'aide d' Gestion des identités et des accès AWS (IAM).

Pour créer une politique IAM pour les ordinateurs portables dotés d' SageMaker intelligence artificielle

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le volet de navigation de gauche, choisissez Politiques.
3. Sélectionnez Créer une politique.
4. Sur la page Créer une politique, accédez à un onglet pour modifier le fichier JSON. Créez votre document de politique avec les instructions JSON suivantes. Modifiez *bucket-nameregion-code*, et *account-id* pour votre environnement.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": [
                "s3:ListBucket"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket"
            ]
        },
        {
            "Action": [
                "s3:GetObject"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket*"
            ]
```

```
        },
        {
            "Action": [
                "logs:CreateLogStream",
                "logs:DescribeLogStreams",
                "logs:PutLogEvents",
                "logs:CreateLogGroup"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:logs:us-east-1:111122223333:log-group:/aws/
sagemaker/*",
                "arn:aws:logs:us-east-1:111122223333:log-group:/aws/
sagemaker/*:log-stream:aws-glue-*"
            ]
        },
        {
            "Action": [
                "glue:UpdateDevEndpoint",
                "glue:GetDevEndpoint",
                "glue:GetDevEndpoints"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:glue:us-east-1:111122223333:devEndpoint/*"
            ]
        },
        {
            "Action": [
                "sagemaker:ListTags"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:sagemaker:us-east-1:111122223333:notebook-instance/
*"
            ]
        }
    ]
}
```

Sélectionnez ensuite Examiner une politique.

Le tableau suivant décrit les autorisations accordées par cette politique.

Action	Ressource	Description
"s3:ListBucket*"	"arn:aws:s3::: *bucket-name* "	Octroie l'autorisation de répertorier les compartiments Amazon S3.
"s3:GetObject"	"arn:aws:s3::: *bucket-name* *"	Accorde l'autorisation d'obtenir des objets Amazon S3 utilisés par les blocs-notes dotés d' SageMaker intelligence artificielle.
"logs:CreateLogStream", "logs:DescribeLogStreams", "logs:PutLogEvents", "logs:CreateLogGroup"	"arn:aws:logs: *region-code* :*account-id* :log-group:/aws/sagemaker/*", "arn:aws:logs: *region-code* :*account-id* :log-group:/aws/sagemaker/*:log-stream:aws-glue-*"	Accorde l'autorisation d'écrire des CloudWatch journaux sur Amazon Logs à partir de carnets de notes. Convention de dénomination : écrit dans les groupes de journaux dont le nom commence par aws-glue.
"glue:UpdateDevEndpoint", "glue:GetDevEndpoint", "glue:GetDevEndpoints"	"arn:aws:glue: *region-code* :*account-id* :devEndpoint/*"	Accorde l'autorisation d'utiliser un point de développement à partir de blocs-notes d' SageMaker IA.
"sagemaker:ListTags"	"arn:aws:sagemaker : *region-code* :*account-id* :notebook-instance /*"	Accorde l'autorisation de renvoyer des balises pour une ressource d' SageMaker IA. La `aws-glue-dev-endpoint` balise est requise sur le bloc-notes SageMaker AI pour connecter le bloc-notes à un terminal de développement.

5. Sur l'écran Review policy (Examiner la politique), saisissez votre Policy Name (Nom de politique), par exemple, `AWSGlueSageMakerNotebook`. Saisissez éventuellement une description, et lorsque vous êtes satisfait de la politique, sélectionnez Créer une politique.

Étape 7 : créer un rôle IAM pour les blocs-notes SageMaker AI

Si vous envisagez d'utiliser des blocs-notes SageMaker AI avec des points de terminaison de développement, vous devez accorder des autorisations au rôle IAM. Vous fournissez ces autorisations en utilisant Gestion des identités et des accès AWS (IAM), via un rôle IAM.

Pour créer un rôle IAM pour les blocs-notes SageMaker AI

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le volet de navigation de gauche, choisissez Rôles.
3. Choisissez Créer un rôle.
4. Pour le type de rôle, choisissez AWS Service, recherchez et choisissez SageMaker, puis choisissez le cas d'utilisation SageMaker - Exécution. Choisissez ensuite Suivant : Autorisations.
5. Sur la page Joindre une politique d'autorisations, choisissez les politiques contenant les autorisations requises ; par exemple, AmazonSageMakerFullAccess. Choisissez Suivant : Vérification.

   Si vous prévoyez d'accéder à des sources et cibles Amazon S3 qui sont chiffrées avec SSE-KMS, vous devrez attacher une politique permettant aux blocs-notes de déchiffrer les données, comme illustré dans l'exemple suivant. Pour plus d'informations, voir Protection des données à l'aide du chiffrement côté serveur avec des clés AWS KMS gérées (SSE-KMS).

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "kms:Decrypt"
      ],
      "Resource": [
```

```
            "arn:aws:kms:*:111122223333:key/key-id"
        ]
      }
    ]
}
```

6. Dans le champ Role name (Nom de rôle), saisissez un nom pour votre rôle. Pour autoriser le transfert du rôle des utilisateurs de la console à l' SageMaker IA, utilisez un nom préfixé par la chaîne`AWSGlueServiceSageMakerNotebookRole`. AWS Glueà condition que les politiques prévoient que les rôles IAM commencent `AWSGlueServiceSageMakerNotebookRole` par. Sinon, vous devez ajouter une politique à vos utilisateurs accordant l'autorisation `iam:PassRole` aux rôles IAM de correspondre à votre convention de dénomination.

   Par exemple, saisissez `AWSGlueServiceSageMakerNotebookRole-Default`, puis sélectionnez Create role (Créer le rôle).
7. Après avoir créé le rôle, joignez la politique qui autorise les autorisations supplémentaires requises pour créer des blocs-notes SageMaker AI à partir deAWS Glue.

   Ouvrez le rôle que vous venez de créer, `AWSGlueServiceSageMakerNotebookRole-Default`, puis sélectionnez Attach policies (Attacher des politiques). Attachez la politique nommée `AWSGlueSageMakerNotebook` que vous avez créée au rôle.

AWS Exemples de politiques de contrôle d'accès Glue

Cette section contient des exemples de politiques de contrôle d'accès basées sur l'identité (IAM) et de politiques basées sur les ressources AWS Glue.

Table des matières

- Exemples de politiques basées sur l'identité pour Glue AWS
  - Bonnes pratiques en matière de politiques
  - Les autorisations au niveau des ressources s'appliquent uniquement à des objets AWS Glue spécifiques
  - Utilisation de la console AWS Glue
  - Autorisation accordée aux utilisateurs pour afficher leurs propres autorisations
  - Octroyer une autorisation en lecture seule à une table
  - Filtrer les tables par GetTables autorisation
  - Octroyer un accès complet à une table et à toutes les partitions

- Contrôler l'accès par préfixe du nom et refus explicite
- Octoyer l'accès à l'aide de balises
- Refuser l'accès à l'aide de balises
- Utiliser des balises avec les opérations d'API de liste et de lot
- Contrôler les paramètres à l'aide de clés de condition ou de contexte
  - Contrôler des politiques qui contrôlent les paramètres à l'aide des clés de condition
  - Contrôler des politiques qui contrôlent les paramètres à l'aide des clés de contexte
- Refus de la possibilité de créer des sessions d'aperçu des données à une identité
- Exemples de politiques basées sur les ressources pour Glue AWS
  - Considérations relatives à l'utilisation de politiques basées sur les ressources avec Glue AWS
  - Utiliser une politique de ressource pour contrôler l'accès dans le même compte

Exemples de politiques basées sur l'identité pour Glue AWS

Par défaut, les utilisateurs et les rôles ne sont pas autorisés à créer ou à modifier les ressources AWS Glue. Pour octroyer aux utilisateurs des autorisations d'effectuer des actions sur les ressources dont ils ont besoin, un administrateur IAM peut créer des politiques IAM.

Pour apprendre à créer une politique basée sur l'identité IAM à l'aide de ces exemples de documents de politique JSON, consultez Création de politiques IAM (console) dans le Guide de l'utilisateur IAM.

Pour plus de détails sur les actions et les types de ressources définis par AWS Glue, y compris le ARNs format de chaque type de ressource, voir Actions, ressources et clés de condition pour AWS Glue dans le Service Authorization Reference.

> **Note**
>
> Les exemples fournis dans cette section utilisent tous la région `us-west-2`. Vous pouvez le remplacer par la AWS région que vous souhaitez utiliser.

Rubriques

- Bonnes pratiques en matière de politiques
- Les autorisations au niveau des ressources s'appliquent uniquement à des objets AWS Glue spécifiques

- Utilisation de la console AWS Glue
- Autorisation accordée aux utilisateurs pour afficher leurs propres autorisations
- Octroyer une autorisation en lecture seule à une table
- Filtrer les tables par GetTables autorisation
- Octroyer un accès complet à une table et à toutes les partitions
- Contrôler l'accès par préfixe du nom et refus explicite
- Octoyer l'accès à l'aide de balises
- Refuser l'accès à l'aide de balises
- Utiliser des balises avec les opérations d'API de liste et de lot
- Contrôler les paramètres à l'aide de clés de condition ou de contexte
- Refus de la possibilité de créer des sessions d'aperçu des données à une identité

Bonnes pratiques en matière de politiques

Les politiques basées sur l'identité déterminent si quelqu'un peut créer, accéder ou supprimer les ressources AWS Glue de votre compte. Ces actions peuvent entraîner des frais pour votre Compte AWS. Lorsque vous créez ou modifiez des politiques basées sur l'identité, suivez ces instructions et recommandations :

- Commencez AWS par les politiques gérées et passez aux autorisations du moindre privilège : pour commencer à accorder des autorisations à vos utilisateurs et à vos charges de travail, utilisez les politiques AWS gérées qui accordent des autorisations pour de nombreux cas d'utilisation courants. Ils sont disponibles dans votre Compte AWS. Nous vous recommandons de réduire davantage les autorisations en définissant des politiques gérées par les AWS clients spécifiques à vos cas d'utilisation. Pour plus d'informations, consultez politiques gérées par AWS ou politiques gérées par AWS pour les activités professionnelles dans le Guide de l'utilisateur IAM.
- Accordez les autorisations de moindre privilège : lorsque vous définissez des autorisations avec des politiques IAM, accordez uniquement les autorisations nécessaires à l'exécution d'une seule tâche. Pour ce faire, vous définissez les actions qui peuvent être entreprises sur des ressources spécifiques dans des conditions spécifiques, également appelées autorisations de moindre privilège. Pour plus d'informations sur l'utilisation d'IAM pour appliquer des autorisations, consultez politiques et autorisations dans IAM dans le Guide de l'utilisateur IAM.
- Utilisez des conditions dans les politiques IAM pour restreindre davantage l'accès : vous pouvez ajouter une condition à vos politiques afin de limiter l'accès aux actions et aux ressources. Par exemple, vous pouvez écrire une condition de politique pour spécifier que toutes les demandes

doivent être envoyées via SSL. Vous pouvez également utiliser des conditions pour accorder l'accès aux actions de service si elles sont utilisées par le biais d'un service spécifique Service AWS, tel que CloudFormation. Pour plus d'informations, consultez Conditions pour éléments de politique JSON IAM dans le Guide de l'utilisateur IAM.

- Utilisez l'Analyseur d'accès IAM pour valider vos politiques IAM afin de garantir des autorisations sécurisées et fonctionnelles : l'Analyseur d'accès IAM valide les politiques nouvelles et existantes de manière à ce que les politiques IAM respectent le langage de politique IAM (JSON) et les bonnes pratiques IAM. IAM Access Analyzer fournit plus de 100 vérifications de politiques et des recommandations exploitables pour vous aider à créer des politiques sécurisées et fonctionnelles. Pour plus d'informations, consultez Validation de politiques avec IAM Access Analyzer dans le Guide de l'utilisateur IAM.
- Exiger l'authentification multifactorielle (MFA) : si vous avez un scénario qui nécessite des utilisateurs IAM ou un utilisateur root, activez l'authentification MFA pour une sécurité accrue. Compte AWS Pour exiger la MFA lorsque des opérations d'API sont appelées, ajoutez des conditions MFA à vos politiques. Pour plus d'informations, consultez Sécurisation de l'accès aux API avec MFA dans le Guide de l'utilisateur IAM.

Pour plus d'informations sur les bonnes pratiques dans IAM, consultez Bonnes pratiques de sécurité dans IAM dans le Guide de l'utilisateur IAM.

Les autorisations au niveau des ressources s'appliquent uniquement à des objets AWS Glue spécifiques

Vous pouvez uniquement définir un contrôle précis pour certains objets dans AWS Glue. Vous devez donc écrire la politique IAM de votre client afin que les opérations d'API qui autorisent Amazon Resource Names (ARNs) pour l'`Resource`instruction ne soient pas mélangées avec les opérations d'API qui ne l'autorisent ARNs pas.

Par exemple, la politique IAM suivante autorise les opérations d'API pour `GetClassifier` et `GetJobRun`. Il le définit `Resource` comme `*` étant donné qu'il AWS Glue n'autorise pas ARNs les classificateurs et les exécutions de tâches. Parce que ARNs sont autorisées pour des opérations d'API spécifiques telles que `GetDatabase` et`GetTable`, ARNs peuvent être spécifiées dans la seconde moitié de la politique.

JSON

```
{
```

```
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:GetClassifier*",
        "glue:GetJobRun*"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "glue:Get*"
      ],
      "Resource": [
        "arn:aws:glue:us-east-1:111122223333:catalog",
        "arn:aws:glue:us-east-1:111122223333:database/default",
        "arn:aws:glue:us-east-1:111122223333:table/default/e*1*",
        "arn:aws:glue:us-east-1:111122223333:connection/connection2"
      ]
    }
  ]
}
```

Pour obtenir la liste des AWS Glue objets autorisés ARNs, voirSpécification de AWS Glue la ressource ARNs.

Utilisation de la console AWS Glue

Pour accéder à la console AWS Glue, vous devez disposer d'un minimum d'autorisations. Ces autorisations doivent vous permettre de répertorier et de consulter les détails des ressources AWS Glue de votre Compte AWS. Si vous créez une politique basée sur l'identité qui est plus restrictive que l'ensemble minimum d'autorisations requis, la console ne fonctionnera pas comme prévu pour les entités (utilisateurs ou rôles) tributaires de cette politique.

Il n'est pas nécessaire d'accorder des autorisations de console minimales aux utilisateurs qui appellent uniquement l'API AWS CLI ou l' AWS API. Autorisez plutôt l'accès à uniquement aux actions qui correspondent à l'opération d'API qu'ils tentent d'effectuer.

Pour garantir que les utilisateurs et les rôles peuvent toujours utiliser la console AWS Glue, associez également la AWS Glue *ConsoleAccess* ou la politique *ReadOnly* AWS gérée aux entités. Pour plus d'informations, consultez Ajout d'autorisations à un utilisateur dans le Guide de l'utilisateur IAM.

Pour qu'un utilisateur puisse utiliser la AWS Glue console, il doit disposer d'un ensemble minimal d'autorisations lui permettant d'utiliser les AWS Glue ressources de son AWS compte. Outre ces autorisations AWS Glue, la console nécessite les autorisations des services suivants :

- Autorisations Amazon CloudWatch Logs pour afficher les journaux.
- Gestion des identités et des accès AWS (IAM) autorisations permettant de répertorier et de transmettre des rôles.
- AWS CloudFormation autorisations pour travailler avec des piles.
- Autorisations Amazon Elastic Compute Cloud (Amazon EC2) permettant de VPCs répertorier des sous-réseaux, des groupes de sécurité, des instances et d'autres objets.
- Autorisations Amazon Simple Storage Service (Amazon S3) pour répertorier les compartiments et les objets, et pour récupérer et enregistrer les scripts.
- Autorisations Amazon Redshift requises pour travailler avec des clusters.
- Autorisations Amazon Relational Database Service (Amazon RDS) pour répertorier les instances.

Pour plus d'informations sur les autorisations dont vos utilisateurs ont besoin pour afficher la console AWS Glue et l'utiliser, consultez Étape 3 : attacher une politique aux utilisateurs ou aux groupes accédant à AWS Glue.

Si vous créez une politique IAM plus restrictive que les autorisations minimales requises, la console ne fonctionnera pas comme prévu pour les utilisateurs dotés de cette politique IAM. Pour garantir que ces utilisateurs pourront continuer à utiliser la console AWS Glue, attachez également la politique gérée par `AWSGlueConsoleFullAccess`, comme décrit dans AWS politiques gérées (prédéfinies) pour AWS Glue.

Autorisation accordée aux utilisateurs pour afficher leurs propres autorisations

Cet exemple montre comment créer une politique qui permet aux utilisateurs IAM d'afficher les politiques en ligne et gérées attachées à leur identité d'utilisateur. Cette politique inclut les autorisations permettant d'effectuer cette action sur la console ou par programmation à l'aide de l'API AWS CLI or AWS .

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

Octroyer une autorisation en lecture seule à une table

La politique suivante accorde une autorisation en lecture seule à une table nommée `books` dans la base de données nommée `db1`. Pour plus d'informations sur la ressource Amazon Resource Names (ARNs), consultez[Catalogue de données ARNs](#).

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
```

```
    {
      "Sid": "GetTablesActionOnBooks",
      "Effect": "Allow",
      "Action": [
        "glue:GetTables",
        "glue:GetTable"
      ],
      "Resource": [
        "arn:aws:glue:us-east-1:111122223333:catalog",
        "arn:aws:glue:us-east-1:111122223333:database/db1",
        "arn:aws:glue:us-east-1:111122223333:table/db1/books"
      ]
    }
  ]
}
```

Cette politique accorde l'autorisation en lecture seule à une table nommée `books` dans la base de données nommée `db1`. Pour accorder une autorisation `Get` à une table, cette autorisation est également requise pour le catalogue et les ressources de base de données.

La politique suivante accorde les autorisations nécessaires minimales pour créer la table `tb1` dans la base de données `db1` :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:CreateTable"
      ],
      "Resource": [
        "arn:aws:glue:us-east-1:111122223333:table/db1/tbl1",
        "arn:aws:glue:us-east-1:111122223333:database/db1",
        "arn:aws:glue:us-east-1:111122223333:catalog"
      ]
    }
  ]
}
```

Filtrer les tables par GetTables autorisation

Supposons qu'il y ait trois tables, `customers`, `stores` et `store_sales` dans la base de données db1. La politique suivante octroie l'autorisation `GetTables` à `stores` et `store_sales`, mais pas à `customers`. Lorsque vous appelez `GetTables` avec cette politique, le résultat contient uniquement les deux tables autorisées (la table `customers` n'est pas renvoyée).

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "GetTablesExample",
      "Effect": "Allow",
      "Action": [
        "glue:GetTables"
      ],
      "Resource": [
        "arn:aws:glue:us-east-1:111122223333:catalog",
        "arn:aws:glue:us-east-1:111122223333:database/db1",
        "arn:aws:glue:us-east-1:111122223333:table/db1/store_sales",
        "arn:aws:glue:us-east-1:111122223333:table/db1/stores"
      ]
    }
  ]
}
```

Vous pouvez simplifier la politique précédente en utilisant `store*` pour correspondre à tous les noms de table qui commencent par `store`.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "GetTablesExample2",
      "Effect": "Allow",
      "Action": [
```

```
        "glue:GetTables"
      ],
      "Resource": [
        "arn:aws:glue:us-east-1:111122223333:catalog",
        "arn:aws:glue:us-east-1:111122223333:database/db1",
        "arn:aws:glue:us-east-1:111122223333:table/db1/store*"
      ]
    }
  ]
}
```

De même, si vous utilisez `/db1/*` pour une correspondance avec toutes les tables `db1`, la politique suivante accorde l'accès `GetTables` à toutes les tables de `db1`.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "GetTablesReturnAll",
      "Effect": "Allow",
      "Action": [
        "glue:GetTables"
      ],
      "Resource": [
        "arn:aws:glue:us-east-1:111122223333:catalog",
        "arn:aws:glue:us-east-1:111122223333:database/db1",
        "arn:aws:glue:us-east-1:111122223333:table/db1/*"
      ]
    }
  ]
}
```

Si aucun ARN de table n'est fourni, un appel de `GetTables` réussit, mais renvoie une liste vide.

JSON

```
{
```

```
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "GetTablesEmptyResults",
      "Effect": "Allow",
      "Action": [
        "glue:GetTables"
      ],
      "Resource": [
        "arn:aws:glue:us-east-1:111122223333:catalog",
        "arn:aws:glue:us-east-1:111122223333:database/db1"
      ]
    }
  ]
}
```

Si l'ARN de la base de données est manquant dans la politique, un appel à `GetTables` échoue avec une erreur`AccessDeniedException`.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "GetTablesAccessDeny",
      "Effect": "Allow",
      "Action": [
        "glue:GetTables"
      ],
      "Resource": [
        "arn:aws:glue:us-east-1:111122223333:catalog",
        "arn:aws:glue:us-east-1:111122223333:table/db1/*"
      ]
    }
  ]
}
```

Octroyer un accès complet à une table et à toutes les partitions

La politique suivante accorde toutes les autorisations sur une table nommée `books` dans la base de données `db1`. Cela inclut les autorisations de lecture et d'écriture sur la table elle-même, sur les versions archivées et sur toutes ses partitions.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "FullAccessOnTable",
      "Effect": "Allow",
      "Action": [
        "glue:CreateTable",
        "glue:GetTable",
        "glue:GetTables",
        "glue:UpdateTable",
        "glue:DeleteTable",
        "glue:BatchDeleteTable",
        "glue:GetTableVersion",
        "glue:GetTableVersions",
        "glue:DeleteTableVersion",
        "glue:BatchDeleteTableVersion",
        "glue:CreatePartition",
        "glue:BatchCreatePartition",
        "glue:GetPartition",
        "glue:GetPartitions",
        "glue:BatchGetPartition",
        "glue:UpdatePartition",
        "glue:DeletePartition",
        "glue:BatchDeletePartition"
      ],
      "Resource": [
        "arn:aws:glue:us-east-1:111122223333:catalog",
        "arn:aws:glue:us-east-1:111122223333:database/db1",
        "arn:aws:glue:us-east-1:111122223333:table/db1/books"
      ]
    }
  ]
}
```

La politique précédente peut être simplifiée dans la pratique.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "FullAccessOnTable",
      "Effect": "Allow",
      "Action": [
        "glue:*Table*",
        "glue:*Partition*"
      ],
      "Resource": [
        "arn:aws:glue:us-east-1:111122223333:catalog",
        "arn:aws:glue:us-east-1:111122223333:database/db1",
        "arn:aws:glue:us-east-1:111122223333:table/db1/books"
      ]
    }
  ]
}
```

Notez que la granularité minimum d'un contrôle d'accès précis se trouve au niveau de la table. Cela signifie que vous ne pouvez pas accorder à un utilisateur l'accès à certaines partitions dans une table, mais pas à d'autres, ou à certaines colonnes de la table, mais pas à d'autres. Un utilisateur a soit accès à toute la table, soit à aucune partie de la table.

Contrôler l'accès par préfixe du nom et refus explicite

Dans cet exemple, supposons que les bases de données et les tables de votre AWS Glue Data Catalog soient organisées à l'aide de préfixes de nom. Les bases de données à l'étape de développement ont le préfixe de nom `dev-` et celles en production ont le préfixe de nom `prod-`. Vous pouvez utiliser la politique suivante pour accorder aux développeurs un accès complet à toutes les bases de données UDFs, tables, etc., qui portent le `dev-` préfixe. Mais vous accordez un accès en lecture seule à tout avec le préfixe `prod-`.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "DevAndProdFullAccess",
      "Effect": "Allow",
      "Action": [
        "glue:*Database*",
        "glue:*Table*",
        "glue:*Partition*",
        "glue:*UserDefinedFunction*",
        "glue:*Connection*"
      ],
      "Resource": [
        "arn:aws:glue:us-east-1:111122223333:catalog",
        "arn:aws:glue:us-east-1:111122223333:database/dev-*",
        "arn:aws:glue:us-east-1:111122223333:database/prod-*",
        "arn:aws:glue:us-east-1:111122223333:table/dev-*/*",
        "arn:aws:glue:us-east-1:111122223333:table/*/dev-*",
        "arn:aws:glue:us-east-1:111122223333:table/prod-*/*",
        "arn:aws:glue:us-east-1:111122223333:table/*/prod-*",
        "arn:aws:glue:us-east-1:111122223333:userDefinedFunction/dev-*/*",
        "arn:aws:glue:us-east-1:111122223333:userDefinedFunction/*/dev-*",
        "arn:aws:glue:us-east-1:111122223333:userDefinedFunction/prod-*/*",
        "arn:aws:glue:us-east-1:111122223333:userDefinedFunction/*/prod-*",
        "arn:aws:glue:us-east-1:111122223333:connection/dev-*",
        "arn:aws:glue:us-east-1:111122223333:connection/prod-*"
      ]
    },
    {
      "Sid": "ProdWriteDeny",
      "Effect": "Deny",
      "Action": [
        "glue:*Create*",
        "glue:*Update*",
        "glue:*Delete*"
      ],
      "Resource": [
        "arn:aws:glue:us-east-1:111122223333:database/prod-*",
        "arn:aws:glue:us-east-1:111122223333:table/prod-*/*",
        "arn:aws:glue:us-east-1:111122223333:table/*/prod-*",
```

```
        "arn:aws:glue:us-east-1:111122223333:userDefinedFunction/prod-*/*",
        "arn:aws:glue:us-east-1:111122223333:userDefinedFunction/*/prod-*",
        "arn:aws:glue:us-east-1:111122223333:connection/prod-*"
      ]
    }
  ]
}
```

La deuxième instruction de la politique précédente utilise le explicite `deny`. Vous pouvez utiliser le `deny` explicite pour remplacer toutes les autorisations `allow` qui sont accordées au principal. Cela vous permet de verrouiller l'accès aux ressources critiques et d'éviter qu'une autre politique accorde un accès à celles-ci par accident.

Dans l'exemple précédent, même si la première instruction accorde un accès complet aux ressources `prod-`, la deuxième instruction révoque explicitement l'accès en écriture à celles-ci, laissant uniquement l'accès en lecture aux ressources `prod-`.

Octoyer l'accès à l'aide de balises

Par exemple, supposons que vous souhaitiez limiter l'accès à un déclencheur `t2` à un utilisateur spécifique nommé `Tom` dans votre compte. Tous les autres utilisateurs, y compris `Sam`, ont accès au déclencheur `t1`. Les déclencheurs `t1` et `t2` ont les propriétés suivantes.

```
aws glue get-triggers
{
    "Triggers": [
        {
            "State": "CREATED",
            "Type": "SCHEDULED",
            "Name": "t1",
            "Actions": [
                {
                    "JobName": "j1"
                }
            ],
            "Schedule": "cron(0 0/1 * * ? *)"
        },
        {
            "State": "CREATED",
            "Type": "SCHEDULED",
            "Name": "t2",
```

```
            "Actions": [
                {
                    "JobName": "j1"
                }
            ],
            "Schedule": "cron(0 0/1 * * ? *)"
        }
    ]
}
```

L'administrateur AWS Glue a attaché une valeur de balise `Tom` (`aws:ResourceTag/Name": "Tom"`) au déclencheur `t2`. L'administrateur AWS Glue a également donné à Tom une politique IAM avec une instruction de condition basée sur la balise. Par conséquent, Tom peut uniquement utiliser une opération AWS Glue qui agit sur les ressources avec la valeur de balise `Tom`.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "glue:*",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Name": "Tom"
        }
      }
    }
  ]
}
```

Lorsque Tom essaie d'accéder au déclencheur `t1`, il reçoit un message d'accès refusé. Dans le même temps, il parvient à récupérer le déclencheur `t2`.

```
aws glue get-trigger --name t1
```

```
An error occurred (AccessDeniedException) when calling the GetTrigger operation:
 User: Tom is not authorized to perform: glue:GetTrigger on resource: arn:aws:glue:us-
east-1:123456789012:trigger/t1

aws glue get-trigger --name t2
{
    "Trigger": {
        "State": "CREATED",
        "Type": "SCHEDULED",
        "Name": "t2",
        "Actions": [
            {
                "JobName": "j1"
            }
        ],
        "Schedule": "cron(0 0/1 * * ? *)"
    }
}
```

Tom ne peut pas utiliser l'opération d'API plurielle `GetTriggers` pour répertorier les déclencheurs, car celle-ci ne prend pas en charge le filtrage des balises.

Pour accorder à Tom l'accès à `GetTriggers`, l'administrateur AWS Glue crée une politique qui fractionne les autorisations en deux sections. Une section autorise Tom à accéder à tous les déclencheurs avec l'opération d'API `GetTriggers`. La deuxième section autorise Tom à accéder aux opérations d'API qui sont balisées avec la valeur `Tom`. Avec cette politique, Tom est autorisé à accéder à la fois à `GetTriggers` et à `GetTrigger` pour déclencher `t2`.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "glue:GetTriggers",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "glue:*",
      "Resource": "*",
```

```
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Name": "Tom"
        }
      }
    }
  ]
}
```

Refuser l'accès à l'aide de balises

Une autre approche de politique de ressource consiste à refuser explicitement l'accès aux ressources.

> **Important**
>
> Une politique de refus explicite ne fonctionne pas pour les opérations d'API plurielles telles que `GetTriggers`.

Dans l'exemple de politique suivant, toutes les opérations de tâche AWS Glue sont autorisées. Cependant, la deuxième instruction `Effect` refuse explicitement l'accès aux tâches balisées avec la clé `Team` et la valeur `Special`.

Lorsqu'un administrateur attache la politique suivante à une identité, cette dernière peut accéder à toutes les tâches, sauf celles balisées avec la clé `Team` et la valeur `Special`.

Utiliser des balises avec les opérations d'API de liste et de lot

Une troisième approche possible pour écrire une politique de ressource consiste à autoriser l'accès aux ressources à l'aide d'une opération d'API `List` pour répertorier les ressources pour une valeur de balise. Ensuite, utilisez l'opération d'API `Batch` correspondante pour autoriser l'accès aux détails des ressources spécifiques. Avec cette approche, l'administrateur n'a pas besoin d'autoriser l'accès aux opérations d'API `GetCrawlers`, `GetDevEndpoints`, `GetJobs` ou `GetTriggers` plurielles. Au lieu de cela, vous pouvez permettre de répertorier les ressources avec les opérations d'API suivantes :

- `ListCrawlers`
- `ListDevEndpoints`

- `ListJobs`
- `ListTriggers`

De plus, vous pouvez permettre d'obtenir des détails sur les ressources individuelles avec les opérations d'API suivantes :

- `BatchGetCrawlers`
- `BatchGetDevEndpoints`
- `BatchGetJobs`
- `BatchGetTriggers`

En tant qu'administrateur, pour utiliser cette approche, vous pouvez effectuer les opérations suivantes :

1. Ajouter des balises à vos crawlers, points de terminaison de développement, tâches et déclencheurs.
2. Refuser l'accès utilisateur aux opérations d'API `Get` telles que `GetCrawlers`, `GetDevEndponts`, `GetJobs` et `GetTriggers`.
3. Pour permettre aux utilisateurs de déterminer à quelles ressources balisées ils ont accès, autorisez l'accès utilisateur aux opérations d'API `List` telles que `ListCrawlers`, `ListDevEndponts`, `ListJobs` et `ListTriggers`.
4. Refusez l'accès des utilisateurs au AWS Glue balisage APIs, tel que `TagResource` et`UntagResource`.
5. Autorisez l'utilisateur à accéder aux détails de ressources à l'aide des opérations d'API `BatchGet` telles que `BatchGetCrawlers`, `BatchGetDevEndponts`, `BatchGetJobs` et `BatchGetTriggers`.

Par exemple, lorsque vous appelez l'opération `ListCrawlers`, fournissez une valeur de balise pour correspondre au nom d'utilisateur. Ensuite, le résultat est une liste d'crawlers qui correspondent aux valeurs de balise fournies. Fournissez la liste de noms à `BatchGetCrawlers` pour obtenir des détails sur chaque crawler avec la balise donnée.

Par exemple, si Tom ne doit être en mesure de récupérer que les détails des déclencheurs balisés avec `Tom`, l'administrateur peut ajouter des balises aux déclencheurs pour `Tom`, refuser à tous les

utilisateurs l'accès à l'opération d'API `GetTriggers` et autoriser tous les utilisateurs à accéder à `ListTriggers` et à `BatchGetTriggers`.

Voici la politique de ressources que l'administrateur AWS Glue accorde à Tom. Dans la première section de la politique, les opérations d'API AWS Glue sont refusées pour `GetTriggers`. Dans la deuxième section de la politique, les opérations `ListTriggers` sont autorisées pour toutes les ressources. Toutefois, dans la troisième section, les ressources balisées avec `Tom` sont autorisées à accéder avec l'accès `BatchGetTriggers`.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "glue:GetTriggers",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "glue:ListTriggers"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "glue:BatchGetTriggers"
      ],
      "Resource": [
        "*"
      ],
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Name": "Tom"
        }
      }
    }
```

```
  ]
}
```

En utilisant les mêmes déclencheurs que dans l'exemple précédent, Tom peut accéder au déclencheur `t2`, mais pas au déclencheur `t1`. L'exemple suivant montre les résultats lorsque Tom essaie d'accéder à `t1` et `t2` avec `BatchGetTriggers`.

```
aws glue batch-get-triggers --trigger-names t2
{
    "Triggers": {
        "State": "CREATED",
        "Type": "SCHEDULED",
        "Name": "t2",
        "Actions": [
            {
                "JobName": "j2"
            }
        ],
        "Schedule": "cron(0 0/1 * * ? *)"
    }
}

aws glue batch-get-triggers  --trigger-names t1

An error occurred (AccessDeniedException) when calling the BatchGetTriggers operation:
 No access to any requested resource.
```

L'exemple suivant montre les résultats lorsque Tom essaie d'accéder au déclencheur `t2` et au déclencheur `t3` (qui n'existe pas) dans le même appel `BatchGetTriggers`. Notez que, comme Tom a accès au déclencheur `t2` et qu'il existe, seul `t2` est renvoyé. Tom est autorisé à accéder au déclencheur `t3`, mais le déclencheur `t3` n'existe pas, si bien que `t3` est renvoyé comme réponse dans une liste de `"TriggersNotFound": []`.

```
aws glue batch-get-triggers --trigger-names t2 t3
{
    "Triggers": {
        "State": "CREATED",
        "Type": "SCHEDULED",
        "Name": "t2",
        "Actions": [
            {
```

```
                "JobName": "j2"
            }
        ],
        "TriggersNotFound": ["t3"],
        "Schedule": "cron(0 0/1 * * ? *)"
    }
}
```

Contrôler les paramètres à l'aide de clés de condition ou de contexte

Vous pouvez utiliser des clés de condition ou des clés contextuelles lorsque vous accordez des autorisations de création et de mise à jour de tâches. Ces sections traitent des clés :

- Contrôler des politiques qui contrôlent les paramètres à l'aide des clés de condition
- Contrôler des politiques qui contrôlent les paramètres à l'aide des clés de contexte

Contrôler des politiques qui contrôlent les paramètres à l'aide des clés de condition

AWS Glue fournit trois clés de condition IAM `glue:VpcIdsglue:SubnetIds`, et`glue:SecurityGroupIds`. Vous pouvez utiliser les clés de condition dans les politiques IAM lorsque vous accordez des autorisations de création et de mise à jour de tâches. Vous pouvez utiliser ce paramètre pour vous assurer que les tâches ou les sessions ne sont pas créées (ou mises à jour vers) pour s'exécuter en dehors de l'environnement VPC souhaité. Les informations sur les paramètres du VPC ne sont pas une entrée directe provenant de la demande `CreateJob`, mais déduites du champ « connexions » de la tâche qui pointe vers une connexion AWS Glue.

Exemple d'utilisation

Créez une connexion de type AWS Glue réseau nommée « traffic-monitored-connection » avec le VpcId « vpc-id1234 » souhaité, et. SubnetIds SecurityGroupIds

Spécifiez la condition des clés de condition pour le `CreateJob` et `UpdateJob` dans la politique IAM.

```
{
  "Effect": "Allow",
  "Action": [
    "glue:CreateJob",
    "glue:UpdateJob"
  ],
  "Resource": [
```

```
    "*"
  ],
  "Condition": {
    "ForAnyValue:StringLike": {
      "glue:VpcIds": [
        "vpc-id1234"
      ]
    }
  }
}
```

Vous pouvez créer une politique IAM similaire pour interdire la création d'une tâche AWS Glue sans spécifier d'informations de connexion.

Restreindre les sessions sur VPCs

Pour obliger les sessions créées à s'exécuter au sein d'un VPC spécifique, vous limitez l'autorisation des rôles en ajoutant un effet `Deny` sur l'action `glue:CreateSession` à la condition que le glue:vpc-id ne soit pas égal à vpc-<123>. Par exemple :

```
"Effect": "Deny",
"Action": [
    "glue:CreateSession"
 ],
"Condition": {
    "StringNotEquals" : {"glue:VpcIds" : ["vpc-123"]}
}
```

Vous pouvez également obliger les sessions créées à s'exécuter au sein d'un VPC en ajoutant un effet `Deny` sur l'action `glue:CreateSession` à la condition que le `glue:vpc-id` soit nul. Par exemple :

```
{
    "Effect": "Deny",
    "Action": [
        "glue:CreateSession"
    ],
      "Condition": {
        "Null": {"glue:VpcIds": true}
    }
},
```

```
{
    "Effect": "Allow",
    "Action": [
        "glue:CreateSession"
    ],
    "Resource": ["*"]
}
```

Contrôler des politiques qui contrôlent les paramètres à l'aide des clés de contexte

AWS Glue fournit une clé de contexte (`glue:CredentialIssuingService=glue.amazonaws.com`) à chaque session de rôle, qui est mise à la AWS Glue disposition du poste de travail et du point de terminaison du développeur. Cela vous permet de mettre en œuvre des contrôles de sécurité pour les actions entreprises par les AWS Glue scripts. AWS Gluefournit une autre clé de contexte (`glue:RoleAssumedBy=glue.amazonaws.com`) à chaque session de rôle AWS Glue qui appelle un autre AWS service au nom du client (pas par un job/dev point de terminaison, mais directement par le AWS Glue service).

Exemple d'utilisation

Indiquez l'autorisation conditionnelle dans la politique IAM et attachez-la au rôle à utiliser par une tâche AWS Glue. Cela garantit que certaines actions sont allowed/denied basées sur le fait que la session de rôle est utilisée ou non pour un environnement d'exécution de AWS Glue tâches.

```
{
    "Effect": "Allow",
    "Action": "s3:GetObject",
    "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
    "Condition": {
        "StringEquals": {
            "glue:CredentialIssuingService": "glue.amazonaws.com"
        }
    }
}
```

Refus de la possibilité de créer des sessions d'aperçu des données à une identité

Cette section contient un exemple de politique IAM utilisé pour refuser à une identité la possibilité de créer des sessions d'aperçu des données. Associez cette politique à l'identité, qui est distincte du rôle utilisé par la session d'aperçu des données lors de son exécution.

```
{
    "Sid": "DatapreviewDeny",
    "Effect": "Deny",
     "Action": [
           "glue:CreateSession"
     ],
     "Resource": [
          "arn:aws:glue:*:*:session/glue-studio-datapreview*"
      ]
 }
```

Exemples de politiques basées sur les ressources pour Glue AWS

Cette section contient des exemples de politiques basées sur les ressources, y compris les politiques qui accordent l'accès intercompte.

Les exemples utilisent le AWS Command Line Interface (AWS CLI) pour interagir avec les opérations AWS Glue de l'API de service. Vous pouvez effectuer les mêmes opérations sur la AWS Glue console ou à l'aide de l'un des AWS SDKs.

> **Important**
>
> En modifiant une politique de ressource AWS Glue, vous pouvez accidentellement annuler les autorisations des utilisateurs AWS Glue existants dans votre compte et provoquer des interruptions inattendues. Essayez ces exemples uniquement dans les comptes de développement ou de test et assurez-vous qu'ils n'interrompent aucun workflow existant avant d'apporter des modifications.

Rubriques

- Considérations relatives à l'utilisation de politiques basées sur les ressources avec Glue AWS
- Utiliser une politique de ressource pour contrôler l'accès dans le même compte

Considérations relatives à l'utilisation de politiques basées sur les ressources avec Glue AWS

> **Note**
>
> Les politiques IAM et la politique de ressource AWS Glue prennent quelques secondes pour être appliquées. Une fois que vous avez attaché une nouvelle politique, vous pouvez

remarquer que l'ancienne politique est toujours en vigueur jusqu'à ce que la nouvelle politique ait été appliquée à tout le système.

Vous utilisez un document de politique écrit au format JSON pour créer ou modifier une politique de ressource. La syntaxe de la politique est la même que pour une politique IAM basée sur l'identité (veuillez consulter la Référence de politique JSON IAM), avec les exceptions suivantes :

- Un bloc `"Principal"` ou `"NotPrincipal"` est requis pour chaque déclaration de politique.
- Le `"Principal"` ou `"NotPrincipal"` doit identifier les principaux existants valides. Les modèles de caractères génériques (tels que `arn:aws:iam::`*`account-id`*`:user/*`) ne sont pas autorisés.
- Le `"Resource"` bloc de la politique nécessite que toutes les ressources correspondent ARNs à la syntaxe d'expression régulière suivante (où la première `%s` est le*`region`*, et la seconde `%s` est le*`account-id`*) :

  ```
  *arn:aws:glue:%s:%s:(\*|[a-zA-Z\*]+\/?.*)
  ```

  Par exemple, `arn:aws:glue:us-west-2:`*`account-id`*`:*` et `arn:aws:glue:us-west-2:`*`account-id`*`:database/default` sont autorisés, mais `*` est pas autorisé.
- Contrairement aux politiques basées sur l'identité, une politique de AWS Glue ressources ne doit contenir que les Amazon Resource Names (ARNs) des ressources appartenant au catalogue auquel la politique est attachée. Cela commence ARNs toujours par`arn:aws:glue:`.
- Une politique ne peut pas faire en sorte que l'identité qui la crée ne puisse pas créer ou modifier de politiques.
- La taille d'un document JSON de politique de ressource ne peut pas dépasser 10 Ko.

Utiliser une politique de ressource pour contrôler l'accès dans le même compte

Dans cet exemple, un utilisateur administrateur du Compte A crée une politique de ressource qui accorde à l'utilisateur IAM `Alice` du Compte A un accès complet au catalogue. Alice n'a aucune politique IAM attachée.

Pour ce faire, l'utilisateur admin exécute la AWS CLI commande suivante.

```
# Run as admin of Account A
```

```
$ aws glue put-resource-policy --profile administrator-name --region us-west-2 --
policy-in-json '{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Principal": {
        "AWS": [
          "arn:aws:iam::account-A-id:user/Alice"
        ]
      },
      "Effect": "Allow",
      "Action": [
        "glue:*"
      ],
      "Resource": [
        "arn:aws:glue:us-west-2:account-A-id:*"
      ]
    }
  ]
}'
```

Au lieu de saisir le document de politique JSON dans le cadre de votre AWS CLI commande, vous pouvez enregistrer un document de politique dans un fichier et référencer le chemin du fichier dans la AWS CLI commande, préfixé par`file://`. Voici un exemple de la façon dont vous pourriez procéder :

```
$ echo '{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Principal": {
        "AWS": [
          "arn:aws:iam::account-A-id:user/Alice"
        ]
      },
      "Effect": "Allow",
      "Action": [
        "glue:*"
      ],
      "Resource": [
        "arn:aws:glue:us-west-2:account-A-id:*"
      ]
    }
```

```
  ]
}' > /temp/policy.json

$ aws glue put-resource-policy --profile admin1 \
    --region us-west-2 --policy-in-json file:///temp/policy.json
```

Une fois que cette politique de ressource a été propagée, Alice peut accéder à toutes les ressources AWS Glue du Compte A, comme suit.

```
# Run as user Alice
$ aws glue create-database --profile alice --region us-west-2 --database-input '{
    "Name": "new_database",
    "Description": "A new database created by Alice",
    "LocationUri": "s3://amzn-s3-demo-bucket"
}'

$ aws glue get-table --profile alice --region us-west-2 --database-name "default" --
table-name "tbl1"}
```

En réponse à l'appel `get-table` d'Alice, le service AWS Glue renvoie ce qui suit.

```
{
  "Table": {
    "Name": "tbl1",
    "PartitionKeys": [],
    "StorageDescriptor": {
        ......
    },
    ......
  }
}
```

Octroi de politiques AWS gérées pour AWS Glue

Une politique AWS gérée est une politique autonome créée et administrée par AWS. AWS les politiques gérées sont conçues pour fournir des autorisations pour de nombreux cas d'utilisation courants afin que vous puissiez commencer à attribuer des autorisations aux utilisateurs, aux groupes et aux rôles.

N'oubliez pas que les politiques AWS gérées peuvent ne pas accorder d'autorisations de moindre privilège pour vos cas d'utilisation spécifiques, car elles sont accessibles à tous les AWS clients.

Nous vous recommandons de réduire encore les autorisations en définissant des politiques gérées par le client qui sont propres à vos cas d'utilisation.

Vous ne pouvez pas modifier les autorisations définies dans les politiques AWS gérées. Si les autorisations définies dans une politique AWS gérée sont AWS mises à jour, la mise à jour affecte toutes les identités principales (utilisateurs, groupes et rôles) auxquelles la politique est attachée. AWS est le plus susceptible de mettre à jour une politique AWS gérée lorsqu'une nouvelle politique Service AWS est lancée ou lorsque de nouvelles opérations d'API sont disponibles pour les services existants.

Pour plus d'informations, consultez Politiques gérées par AWS dans le Guide de l'utilisateur IAM.

AWS politiques gérées (prédéfinies) pour AWS Glue

AWS répond à de nombreux cas d'utilisation courants en fournissant des politiques IAM autonomes créées et administrées par. AWS Ces politiques AWS gérées accordent les autorisations nécessaires pour les cas d'utilisation courants afin que vous puissiez éviter d'avoir à rechercher les autorisations nécessaires. Pour plus d'informations, consultez Politiques gérées par AWS dans le Guide de l'utilisateur IAM.

Les politiques AWS gérées suivantes, que vous pouvez associer aux identités de votre compte, sont spécifiques à chaque scénario d'utilisation AWS Glue et sont regroupées par scénario d'utilisation :

- AWSGlueConsoleFullAccess— Accorde un accès complet aux AWS Glue ressources lorsqu'une identité à laquelle la politique est attachée utilise le AWS Management Console. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique est généralement attachée aux utilisateurs de la console AWS Glue.
- AWSGlueServiceRole— Accorde l'accès aux ressources dont AWS Glue les différents processus ont besoin pour s'exécuter en votre nom. Ces ressources incluent AWS Glue Amazon S3, IAM, CloudWatch Logs et Amazon EC2. Si vous suivez la convention de dénomination des ressources spécifiées dans la politique, les processus AWS Glue ont les autorisations requises. Cette politique est généralement attachée aux rôles spécifiés lorsque vous définissez les crawlers, les tâches et les points de terminaison de développement.
- AwsGlueSessionUserRestrictedServiceRole— Fournit un accès complet à toutes les AWS Glue ressources, à l'exception des sessions. Elle permet aux utilisateurs de créer et d'utiliser uniquement les séances interactives associées à l'utilisateur. Cette politique inclut les autres autorisations nécessaires AWS Glue pour gérer les AWS Glue ressources d'autres AWS services.

La politique permet également d'ajouter des balises aux AWS Glue ressources d'autres AWS services.

> **Note**
>
> Pour bénéficier de tous les avantages de la sécurité, n'accordez pas cette politique à un utilisateur qui a reçu la politique `AWSGlueServiceRole`, `AWSGlueConsoleFullAccess`, ou `AWSGlueConsoleSageMakerNotebookFullAccess`.

- AwsGlueSessionUserRestrictedPolicy— Permet de créer des sessions AWS Glue interactives à l'aide de l'opération `CreateSession` API uniquement si une clé de balise « propriétaire » et une valeur correspondant à l'ID AWS utilisateur du destinataire sont fournies. Cette politique d'identité est associée à l'utilisateur IAM qui appelle l'opération d'API `CreateSession`. Cette politique permet également à la personne assignée d'interagir avec les ressources de session AWS Glue interactives créées avec une balise « propriétaire » et une valeur correspondant à son ID AWS utilisateur. Cette politique refuse l'autorisation de modifier ou de supprimer les balises « propriétaire » d'une ressource AWS Glue de séance après la création de la séance.

  > **Note**
  >
  > Pour bénéficier de tous les avantages de la sécurité, n'accordez pas cette politique à un utilisateur qui a reçu la politique `AWSGlueServiceRole`, `AWSGlueConsoleFullAccess`, ou `AWSGlueConsoleSageMakerNotebookFullAccess`.

- AwsGlueSessionUserRestrictedNotebookServiceRole— Fournit un accès suffisant à la session du AWS Glue Studio bloc-notes pour interagir avec des ressources de session AWS Glue interactives spécifiques. Il s'agit de ressources créées avec la valeur de balise « owner » qui correspond à l'ID AWS utilisateur du principal (utilisateur ou rôle IAM) qui crée le bloc-notes. Pour en savoir plus sur ces balises, veuillez consulter le diagramme Valeurs de la clé du principal dans le Guide de l'utilisateur IAM.

  Cette politique de fonction du service est attachée au rôle spécifié par une commande magique dans le bloc-notes ou transmise en tant que rôle à l'opération d'API `CreateSession`. Cette politique permet également au principal de créer une session AWS Glue interactive à partir de l'interface du AWS Glue Studio bloc-notes uniquement si la clé de balise « propriétaire » et la valeur correspondent à AWS l'ID utilisateur du principal. Cette politique refuse l'autorisation de modifier ou de supprimer les balises « propriétaire » d'une ressource AWS Glue de séance après la création de la séance. Cette politique inclut également les autorisations d'écriture et de lecture à

partir de compartiments Amazon S3, de rédaction de CloudWatch journaux, ainsi que de création et de suppression de balises pour les ressources Amazon EC2 utilisées par. AWS Glue

> **Note**
>
> Pour obtenir tous les avantages en matière de sécurité, n'accordez pas cette politique à un rôle qui a été attribué à la politique `AWSGlueServiceRole`, `AWSGlueConsoleFullAccess`, ou `AWSGlueConsoleSageMakerNotebookFullAccess`.

- AwsGlueSessionUserRestrictedNotebookPolicy— Permet de créer une session AWS Glue interactive à partir de l'interface du AWS Glue Studio bloc-notes uniquement s'il existe une clé de balise « propriétaire » et une valeur correspondant à IDof l' AWS utilisateur principal (utilisateur ou rôle IAM) qui crée le bloc-notes. Pour en savoir plus sur ces balises, veuillez consulter le diagramme Valeurs de la clé du principal dans le Guide de l'utilisateur IAM.

  Cette politique est attachée au principal (utilisateur ou rôle IAM) qui crée des séances à partir de l'interface de bloc-notes AWS Glue Studio. Cette politique permet également un accès suffisant au bloc-notes AWS Glue Studio pour interagir avec des ressources de séances interactives AWS Glue spécifiques. Il s'agit de ressources créées avec la valeur de balise « owner » qui correspond à AWS l'ID utilisateur du principal. Cette politique refuse l'autorisation de modifier ou de supprimer les balises « propriétaire » d'une ressource AWS Glue de séance après la création de la séance.
- AWSGlueServiceNotebookRole— Accorde l'accès aux AWS Glue sessions démarrées dans un AWS Glue Studio bloc-notes. Cette politique permet de répertorier et d'obtenir des informations de session pour toutes les sessions, mais permet uniquement aux utilisateurs de créer et d'utiliser les sessions étiquetées avec leur ID AWS utilisateur. Cette politique refuse l'autorisation de modifier ou de supprimer les balises « propriétaire » des ressources de AWS Glue session étiquetées avec leur AWS ID.

  Attribuez cette politique à l' AWS utilisateur qui crée des tâches à l'aide de l'interface du bloc-notes dansAWS Glue Studio.
- AWSGlueConsoleSageMakerNotebookFullAccess— Accorde un accès complet à AWS Glue et aux ressources d' SageMaker IA lorsque l'identité à laquelle la politique est attachée utilise le AWS Management Console. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console qui gèrent des blocs-notes dotés d' SageMaker intelligence artificielle.

- AWSGlueSchemaRegistryFullAccess— Accorde un accès complet aux ressources du registre des AWS Glue schémas lorsque l'identité à laquelle la politique est attachée utilise le AWS Management Console ou AWS CLI. Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique est généralement attachée aux utilisateurs de la AWS Glue console ou aux utilisateurs AWS CLI qui gèrent le registre des AWS Glue schémas.
- AWSGlueSchemaRegistryReadonlyAccess— Accorde un accès en lecture seule aux ressources du registre des AWS Glue schémas lorsqu'une identité à laquelle la politique est attachée utilise le AWS Management Console ou. AWS CLI Si vous suivez la convention de dénomination pour les ressources spécifiées dans la politique, les utilisateurs bénéficient des capacités totales de la console. Cette politique s'applique généralement aux utilisateurs de la AWS Glue console ou à AWS CLI ceux qui utilisent le registre des AWS Glue schémas.

> **Note**
>
> Vous pouvez consulter ces politiques d'autorisations en vous connectant à la console IAM et en y recherchant des politiques spécifiques.

Vous pouvez également créer vos propres stratégies IAM personnalisées afin d'accorder des autorisations pour les actions et les ressources AWS Glue. Vous pouvez attacher ces politiques personnalisées aux utilisateurs ou groupes IAM qui nécessitent ces autorisations.

Pour créer une connexion avec une configuration VPC tout en utilisant un rôle IAM personnalisé, celui-ci doit disposer des actions d'accès VPC suivantes :

- responsable des secrets : GetSecretValue
- responsable des secrets : PutSecretValue
- responsable des secrets : DescribeSecret
- EC2 : CreateNetworkInterface
- EC2 : DeleteNetworkInterface
- EC2 : DescribeNetworkInterfaces
- EC2 : DescribeSubnets

AWS Glue les mises à jour dans les politiques AWS gérées

Consultez les détails des mises à jour apportées aux politiques AWS gérées pour AWS Glue depuis que ce service a commencé à suivre ces modifications. Pour recevoir des alertes automatiques concernant les modifications apportées à cette page, abonnez-vous au flux RSS sur la page d'historique du document AWS Glue.

Modifier	Description	Date
AwsGlueSessionUserRestrictedNotebookPolicy — Mise à jour mineure d'une politique existante.	Ajoutez une action d'autorisation pour `glue:TagResource` sur la clé de l'identification propriétaire. Nécessaire tag-on-create pour la prise en charge des sessions avec clé d'identification du propriétaire.	30 août 2024
AwsGlueSessionUserRestrictedNotebookServiceRole — Mise à jour mineure d'une politique existante.	Ajoutez une action d'autorisation pour `glue:TagResource` sur la clé de l'identification propriétaire. Nécessaire tag-on-create pour la prise en charge des sessions avec clé d'identification du propriétaire.	30 août 2024
AwsGlueSessionUserRestrictedPolicy — Mise à jour mineure d'une politique existante.	Ajoutez une action d'autorisation pour `glue:TagResource` sur la clé de l'identification propriétaire. Nécessaire tag-on-create pour la prise en charge des sessions avec clé d'identification du propriétaire.	5 août 2024
AwsGlueSessionUserRestrictedServiceRole — Mise à	Ajoutez une action d'autorisation pour `glue:TagResource` sur la clé de	5 août 2024

Modifier	Description	Date
jour mineure d'une politique existante.	l'identification propriétaire. Nécessaire tag-on-create pour la prise en charge des sessions avec clé d'identif ication du propriétaire.	
AwsGlueSessionUserRestricte dPolicy — Mise à jour mineure d'une politique existante.	Ajout de `glue:Star tCompletion` et de `glue:GetCompletion` à la politique. Nécessaire pour l'intégration des données Amazon Q dans AWS Glue.	30 avril 2024
AwsGlueSessionUserRestricte dNotebookServiceRole — Mise à jour mineure d'une politique existante.	Ajout de `glue:Star tCompletion` et de `glue:GetCompletion` à la politique. Nécessaire pour l'intégration des données Amazon Q dans AWS Glue.	30 avril 2024
AwsGlueSessionUserRestricte dServiceRole — Mise à jour mineure d'une politique existante.	Ajout de `glue:Star tCompletion` et de `glue:GetCompletion` à la politique. Nécessaire pour l'intégration des données Amazon Q dans AWS Glue.	30 avril 2024
AWSGlueServiceNote bookRole — Mise à jour mineure d'une politique existante.	Ajout de `glue:Star tCompletion` et de `glue:GetCompletion` à la politique. Nécessaire pour l'intégration des données Amazon Q dans AWS Glue.	30 janvier 2024

Modifier	Description	Date
AwsGlueSessionUserRestrictedNotebookPolicy — Mise à jour mineure d'une politique existante.	Ajout de `glue:StartCompletion` et de `glue:GetCompletion` à la politique. Nécessaire pour l'intégration des données Amazon Q dans AWS Glue.	29 novembre 2023
AWSGlueServiceNotebookRole — Mise à jour mineure d'une politique existante.	Ajouter une stratégie `codewhisperer:GenerateRecommendations` Nécessaire pour une nouvelle fonctionnalité dans laquelle AWS Glue génère CodeWhisperer des recommandations.	9 octobre 2023
AWSGlueServiceRole — Mise à jour mineure d'une politique existante.	Resserrez la portée des CloudWatch autorisations pour mieux refléter AWS la journalisation de Glue.	4 août 2023
AWSGlueConsoleFullAccess — Mise à jour mineure d'une politique existante.	Ajoutez les autorisations Lister et Décrire de la recette `databrew` à la stratégie. Nécessaire pour fournir un accès administratif complet aux nouvelles fonctionnalités permettant à AWS Glue d'accéder aux recettes.	9 mai 2023

Modifier	Description	Date
AWSGlueConsoleFullAccess — Mise à jour mineure d'une politique existante.	Ajouter une stratégie `cloudformation:ListStacks` Préserve les fonctionnalités existantes après modification des exigences CloudFormation d'autorisation.	28 mars 2023
Nouvelles politiques gérées ajoutées pour la fonctionnalité de séances interactives : • AwsGlueSessionUserRestrictedServiceRole • AwsGlueSessionUserRestrictedPolicy • AwsGlueSessionUserRestrictedNotebookServiceRole • AwsGlueSessionUserRestrictedNotebookPolicy	Ces politiques ont été conçues pour fournir une sécurité supplémentaire pour les séances interactives et les blocs-notes dans AWS Glue Studio. Les politiques limitent l'accès à l'opération d'API `CreateSession` pour que seul le propriétaire ait accès.	30 novembre 2021

Modifier	Description	Date
AWSGlueConsoleSageMakerNotebookFullAccess — Mise à jour d'une politique existante.	Suppression d'un ARN de ressource redondant (`arn:aws:s3:::aws-glue-*/*`) pour l'action qui accorde des read/write autorisations sur les compartiments Amazon S3 AWS Glue utilisés pour stocker des scripts et des fichiers temporaires. Correction d'un problème de syntaxe en remplaçant `"StringEquals"` par `"ForAnyValue:StringLike"` , et changement de place des lignes `"Effect": "Allow"` pour qu'elles précèdent la ligne `"Action":` dans chaque endroit où elles n'étaient pas dans l'ordre.	15 juillet 2021
AWSGlueConsoleFullAccess — Mise à jour d'une politique existante.	Suppression d'un ARN de ressource redondant (`arn:aws:s3:::aws-glue-*/*`) pour l'action qui accorde des read/write autorisations sur les compartiments Amazon S3 AWS Glue utilisés pour stocker des scripts et des fichiers temporaires.	15 juillet 2021

Modifier	Description	Date
AWS Glue a démarré le suivi des modifications.	AWS Gluea commencé à suivre les modifications apportées AWS à ses politique s gérées.	10 juin 2021

Octroi de politiques à portée dynamique pour l'exécution des tâches

AWS Glue propose une nouvelle fonctionnalité puissante : des politiques de session dynamiques pour l'exécution des tâches. Cette fonctionnalité vous permet de définir des autorisations personnalisées et détaillées pour chaque tâche exécutée sans créer plusieurs rôles IAM.

Lorsque vous démarrez une tâche Glue à l'aide de l'API `StartJobRun`, vous pouvez inclure une politique de session intégrée. Cette politique modifie temporairement les autorisations du rôle d'exécution de la tâche pendant la durée de l'exécution de cette tâche spécifique. Cela revient à utiliser des informations d'identification temporaires avec l'`AssumeRole`API dans d'autres AWS services.

- Sécurité renforcée : vous pouvez limiter les autorisations de la tâche au minimum nécessaire pour chaque exécution.
- Gestion simplifiée : élimine le besoin de créer et de gérer de nombreux rôles IAM pour différents scénarios.
- Flexibilité : vous pouvez ajuster les autorisations de manière dynamique en fonction des paramètres d'exécution ou des besoins propres au locataire.
- Capacité de mise à l'échelle : cette méthode excelle dans les environnements à locataires multiples où vous devez isoler les ressources entre les locataires.

Exemples d'autorisation d'utilisation de politiques à portée dynamique :

Les exemples suivants montrent comment accorder aux tâches un accès en lecture et en écriture uniquement à un chemin d'accès au compartiment Amazon S3 spécifique, le chemin étant déterminé dynamiquement par l'ID d'exécution de la tâche. Ils montrent comment mettre en œuvre des autorisations granulaires propres à l'exécution pour chaque tâche exécutée.

À partir de la CLI

```
aws glue start-job-run \
    --job-name "your-job-name" \
    --execution-role-session-policy '{
        "Version": "2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Action": [
                    "s3:GetObject",
                    "s3:PutObject"
                ],
                "Resource": [
                    "arn:aws:s3:::specific-bucket/${JobRunId}/*"
                ]
            }
        ]
    }'
```

Spécification de AWS Glue la ressource ARNs

DansAWS Glue, vous pouvez contrôler l'accès aux ressources à l'aide d'une politique Gestion des identités et des accès AWS (IAM). Dans une politique, vous utilisez un Amazon Resource Name (ARN) pour identifier la ressource à laquelle la politique s'applique. Toutes les ressources ne sont pas prises en AWS Glue charge ARNs.

Rubriques

- Catalogue de données ARNs
- ARNs pour les objets hors catalogue dans AWS Glue
- Contrôle d'accès pour les opérations d'API uniques AWS Glue non liées à un catalogue
- Contrôle d'accès pour les opérations d'API AWS Glue non liées à un catalogue qui extraient plusieurs éléments
- Contrôle d'accès pour les AWS Glue opérations d' BatchGet API hors catalogue

Catalogue de données ARNs

Les ressources du catalogue de données ont une structure hiérarchique, avec `catalog` comme racine.

```
arn:aws:glue:region:account-id:catalog
```

Chaque AWS compte possède un seul catalogue de données dans une AWS région avec l'identifiant de compte à 12 chiffres comme identifiant de catalogue. Les ressources sont ARNs associées à des ressources uniques, comme indiqué dans le tableau suivant.

Type de ressource	Format ARN
Catalogue	`arn:aws:glue: region:account-id :catalog` Par exemple : `arn:aws:glue:us-east-1:123456789012:catalog`
Base de données	`arn:aws:glue: region:account-id :database/ database name` Par exemple : `arn:aws:glue:us-east-1:123456789012:database/db1`
Table	`arn:aws:glue: region:account-id :table/database name/table name` Par exemple : `arn:aws:glue:us-east-1:123456789012:table/db1/tbl1`
Catalogue de tables S3 fédérées (tous les compartiments de tables)	`arn:aws:glue: region:account-id :catalog/s3tablescatalog` Par exemple : `arn:aws:glue:us-east-1:123456789012:catalog/s3tablescatalog`
Catalogue de compartiments de tables S3 fédérées (catalogue enfant)	`arn:aws:glue: region:account-id :catalog/s3tablescatalog/ bucket name` Par exemple : `arn:aws:glue:us-east-1:123456789012:catalog/s3tablescatalog/amzn-s3-demo-bucket1`
Base de données de tables S3 fédérées	`arn:aws:glue: region:account-id :database/s3tablescatalog/ child catalog name/database name`

Type de ressource	Format ARN
	Par exemple : `arn:aws:glue:us-east-1:123456789012:database/s3tablescatalog/amzn-s3-demo-bucket1/nsdb1`
Table S3 fédérée	`arn:aws:glue:` *region*`:`*account-id* `:table/s3tablescatalog/` *child catalog name*`/`*database name*`/`*table name* Par exemple : `arn:aws:glue:us-east-1:123456789012:table/s3tablescatalog/amzn-s3-demo-bucket1/nsdb1/s3tbl1`
Catalogue de tables S3 fédérées (un compartiment de tables unique enregistré avec Lake Formation)	`arn:aws:glue:` *region*`:`*account-id* `:catalog/`*catalog name* Par exemple : `arn:aws:glue:us-east-1:123456789012:catalog/amzn-s3-demo-bucket1`
Base de données de tables S3 fédérées	`arn:aws:glue:` *region*`:`*account-id* `:catalog/`*catalog name*`/`*database name* Par exemple : `arn:aws:glue:us-east-1:123456789012:database/amzn-s3-demo-bucket1/nsdb1`
Table S3 fédérée	`arn:aws:glue:` *region*`:`*account-id* `:catalog/`*catalog name*`/`*database name*`/`*table name* Par exemple : `arn:aws:glue:us-east-1:123456789012:table/amzn-s3-demo-bucket1/` *nsdb1*`/s3tbl1`
Catalogue fédéré ou géré (catalogue de premier niveau dans un catalogue multiple)	`arn:aws:glue:` *region*`:`*account-id* `:catalog/` *top-level catalog name* Par exemple : `arn:aws:glue:us-east-1:123456789012:catalog/nscatalog` Le format ARN d'un catalogue géré suit la même structure.

Type de ressource	Format ARN
Catalogue fédéré à plusieurs niveaux (catalogue enfant dans un catalogue à plusieurs niveaux)	`arn:aws:glue:` *`region`*`:`*`account-id`* `:catalog/` *`top-level catalog name`*`/`*`child catalog name`* Par exemple : `arn:aws:glue:us-east-1:123456789012:catalog/nscatalog/dbcatalog`
Base de données fédérée	`arn:aws:glue:` *`region`*`:`*`account-id`* `:database/` *`name space catalog name`*`/`*`child catalog name`*`/`*`database name`* Par exemple : `arn:aws:glue:us-east-1:123456789012:database/nscatalog/dbcatalog/schemadb`
Table fédérée	`arn:aws:glue:` *`region`*`:`*`account-id`* `:table/`*`name space catalog name`*`/`*`child catalog name`*`/`*`database name`*`/`*`table name`* Par exemple : `arn:aws:glue:us-east-1:123456789012:table/nscatalog/dbcatalog/schemadb/rstbl1`
Conteneur de liens du catalogue	`arn:aws:glue:` *`region`*`:`*`account-id`* `:catalog/`*`link container name`* Par exemple : `arn:aws:glue:glue:us-east-1:123456789012:catalog` /linkcontainer-example
Base de données	`arn:aws:glue:` *`region`*`:`*`account-id`* `:catalog/`*`link container name`*`/`*`database name`* Par exemple : `arn:aws:glue:glue:us-east-1:123456789012:database` /linkcontainer-example/link-db
Fonction définie par l'utilisateur	`arn:aws:glue:` *`region`*`:`*`account-id`* `:userDefinedFunction/` *`database name`*`/`*`user-defined function name`* Par exemple : `arn:aws:glue:us-east-1:123456789012:userDefinedFunction/db1/func1`

Type de ressource	Format ARN
Connexion	`arn:aws:glue:` *`region`*`:`*`account-id`* `:connection/` *`connection name`* Par exemple : `arn:aws:glue:us-east-1:123456789012:connection/connection1`
Séance interactive	`arn:aws:glue:` *`region`*`:`*`account-id`* `:session/` *`interactive session id`* Par exemple : `arn:aws:glue:us-east-1:123456789012:session/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111`

Pour permettre un contrôle d'accès précis, vous pouvez les utiliser ARNs dans vos politiques IAM et vos politiques de ressources pour accorder ou refuser l'accès à des ressources spécifiques. Les caractères génériques sont autorisés dans les politiques. Par exemple, l'ARN suivant correspond à toutes les tables de la base de données `default`.

```
arn:aws:glue:us-east-1:123456789012:table/default/*
```

> **Important**
>
> Toutes les opérations effectuées sur une ressource du catalogue de données requièrent une autorisation sur la ressource et tous les ancêtres de cette ressource. Par exemple, la création d'une partition pour une table nécessite l'autorisation sur la table, la base de données et le catalogue où se trouve la table. L'exemple suivant montre l'autorisation requise pour créer des partitions sur la table `PrivateTable` dans la base de données `PrivateDatabase` du catalogue de données.
>
> ```
> {
>    "Sid": "GrantCreatePartitions",
>    "Effect": "Allow",
>    "Action": [
>        "glue:BatchCreatePartitions"
>    ],
>    "Resource": [
>        "arn:aws:glue:us-east-1:123456789012:table/PrivateDatabase/PrivateTable",
> ```

```
        "arn:aws:glue:us-east-1:123456789012:database/PrivateDatabase",
        "arn:aws:glue:us-east-1:123456789012:catalog"
    ]
}
```

Outre l'autorisation sur la ressource et tous ses ancêtres, toutes les opérations de suppression nécessitent une autorisation sur tous les enfants de cette ressource. Par exemple, pour supprimer une base de données, il faut l'autorisation sur toutes les tables et les fonctions définies par l'utilisateur dans la base de données, ainsi que sur la base de données et le catalogue dans lequel se trouve la base de données. L'exemple suivant montre l'autorisation requise pour supprimer une base de données `PrivateDatabase` dans le catalogue de données.

```
{
   "Sid": "GrantDeleteDatabase",
   "Effect": "Allow",
   "Action": [
       "glue:DeleteDatabase"
   ],
   "Resource": [
       "arn:aws:glue:us-east-1:123456789012:table/PrivateDatabase/*",
       "arn:aws:glue:us-east-1:123456789012:userDefinedFunction/PrivateDatabase/
*",
       "arn:aws:glue:us-east-1:123456789012:database/PrivateDatabase",
       "arn:aws:glue:us-east-1:123456789012:catalog"
   ]
}
```

En résumé, les actions sur les ressources du catalogue de données suivent ces règles d'autorisation :

- Les actions sur le catalogue requièrent l'autorisation sur le catalogue uniquement.
- Les actions sur une base de données requièrent l'autorisation sur la base de données et le catalogue.
- Les actions de suppression sur une base de données requièrent l'autorisation sur la base de données et le catalogue, ainsi que sur toutes les tables et les fonctions définies par l'utilisateur dans la base de données.
- Les actions sur une table, une partition ou une version de table requièrent l'autorisation sur la table, la base de données et le catalogue.

- Les actions sur une fonction définie par l'utilisateur requièrent l'autorisation sur la fonction définie par l'utilisateur, la base de données et le catalogue.
- Les actions sur une connexion requièrent l'autorisation sur la connexion et le catalogue.

ARNs pour les objets hors catalogue dans AWS Glue

Certaines ressources AWS Glue autorisent les permissions au niveau des ressources pour contrôler l'accès à l'aide d'un ARN. Vous pouvez les utiliser ARNs dans vos politiques IAM pour permettre un contrôle d'accès précis. Le tableau suivant répertorie les ressources qui peuvent contenir des ressources ARNs.

Type de ressource	Format ARN
crawler	`arn:aws:glue:` *`region`*`:`*`account-id`* `:crawler/` *`crawler-name`* Par exemple : `arn:aws:glue:us-east-1:123456789012:crawler/mycrawler`
Tâche	`arn:aws:glue:` *`region`*`:`*`account-id`* `:job/`*`job-name`* Par exemple : `arn:aws:glue:us-east-1:123456789012:job/testjob`
Déclencheur	`arn:aws:glue:` *`region`*`:`*`account-id`* `:trigger/` *`trigger-name`* Par exemple : `arn:aws:glue:us-east-1:123456789012:trigger/sampletrigger`
Point de terminaison de développement	`arn:aws:glue:` *`region`*`:`*`account-id`* `:devEndpoint/` *`development-endpoint-name`* Par exemple : `arn:aws:glue:us-east-1:123456789012:devEndpoint/temporarydevendpoint`
Transformation de machine learning	`arn:aws:glue:` *`region`*`:`*`account-id`* `:mlTransform/` *`transform-id`*

Type de ressource	Format ARN
	Par exemple : `arn:aws:glue:us-east-1:123456789012:mlTransform/tfm-1234567890`

Contrôle d'accès pour les opérations d'API uniques AWS Glue non liées à un catalogue

Les opérations d'API AWS Glue non liées à un catalogue uniques agissent sur un seul élément (point de terminaison de développement). Des exemples sont `GetDevEndpoint`, `CreateUpdateDevEndpoint` et `UpdateDevEndpoint`. Pour ces opérations, une politique doit placer le nom de l'API dans le bloc `"action"` et l'ARN de la ressource dans le bloc `"resource"`.

Supposons que vous souhaitiez autoriser un utilisateur à appeler l'opération `GetDevEndpoint`. La politique suivante accorde les autorisations nécessaires minimales à un point de terminaison nommé `myDevEndpoint-1`.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "MinimumPermissions",
      "Effect": "Allow",
      "Action": "glue:GetDevEndpoint",
      "Resource": "arn:aws:glue:us-east-1:111122223333:devEndpoint/
myDevEndpoint-1"
    }
  ]
}
```

La politique suivante autorise l'accès `UpdateDevEndpoint` aux ressources qui correspondent à `myDevEndpoint-` avec un caractère générique (*).

JSON

```
{
```

```
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "PermissionWithWildcard",
      "Effect": "Allow",
      "Action": "glue:UpdateDevEndpoint",
      "Resource": "arn:aws:glue:us-east-1:111122223333:devEndpoint/myDevEndpoint-
*"
    }
  ]
}
```

Vous pouvez combiner les deux politiques, comme dans l'exemple suivant. Il se peut que vous constatiez `EntityNotFoundException` pour n'importe quel point de terminaison de développement dont le nom commence par A. Toutefois, une erreur d'accès refusé est renvoyé lorsque vous essayez d'accéder à d'autres points de terminaison de développement.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "CombinedPermissions",
      "Effect": "Allow",
      "Action": [
        "glue:UpdateDevEndpoint",
        "glue:GetDevEndpoint"
      ],
      "Resource": "arn:aws:glue:us-east-1:111122223333:devEndpoint/A*"
    }
  ]
}
```

Contrôle d'accès pour les opérations d'API AWS Glue non liées à un catalogue qui extraient plusieurs éléments

Certaines opérations d'API AWS Glue extraient plusieurs éléments (par exemple, plusieurs points de terminaison de développement) ; par exemple, `GetDevEndpoints`. Pour cette opération, vous ne pouvez spécifier qu'une ressource générique (*), et non une ressource spécifique ARNs.

Par exemple, pour inclure `GetDevEndpoints` dans la politique, la ressource doit être définie avec le caractère générique (*). Les opérations uniques (`GetDevEndpoint`, `CreateDevEndpoint` et `DeleteDevendpoint`) sont également définies pour toutes les ressources (*) de l'exemple.

```
{
            "Sid": "PluralAPIIncluded",
            "Effect": "Allow",
            "Action": [
                "glue:GetDevEndpoints",
                "glue:GetDevEndpoint",
                "glue:CreateDevEndpoint",
                "glue:UpdateDevEndpoint"
            ],
            "Resource": [
                "*"
            ]
}
```

Contrôle d'accès pour les AWS Glue opérations d' BatchGet API hors catalogue

Certaines opérations d'API AWS Glue extraient plusieurs éléments (par exemple, plusieurs points de terminaison de développement) ; par exemple, `BatchGetDevEndpoints`. Pour cette opération, vous pouvez spécifier un ARN pour limiter la portée des ressources qui sont accessibles.

Par exemple, pour autoriser l'accès à un point de terminaison de développement spécifique, incluez `BatchGetDevEndpoints` dans la politique avec son ARN de ressource.

```
{
            "Sid": "BatchGetAPIIncluded",
            "Effect": "Allow",
            "Action": [
                "glue:BatchGetDevEndpoints"
            ],
            "Resource": [
```

```
                "arn:aws:glue:us-east-1:123456789012:devEndpoint/de1"
            ]
}
```

Avec cette politique, vous pouvez accéder correctement au point de terminaison de développement nommé `de1`. Toutefois, si vous tentez d'accéder au point de terminaison de développement nommé `de2`, une erreur est renvoyée.

```
An error occurred (AccessDeniedException) when calling the BatchGetDevEndpoints
 operation: No access to any requested resource.
```

> **Important**
>
> Pour d'autres approches de la configuration des politiques IAM, telles que l'utilisation des opérations d'API `List` et `BatchGet`, consultez Exemples de politiques basées sur l'identité pour Glue AWS.

Octroi d'un accès intercompte

L'octroi d'un accès intercompte aux ressources du catalogue de données permet à vos tâches Extract-transform-load (ETL) d'interroger et de joindre des données provenant de comptes différents.

Rubriques

- Méthodes d'octroi d'un accès intercompte dans AWS Glue
- Ajouter ou mettre à jour la politique de ressource du catalogue de données
- Créer un appel d'API intercompte
- Créez un appel ETL intercompte
- Journalisation entre comptes CloudTrail
- Propriété et facturation des ressources intercomptes
- Limitations des accès inter-comptes

Méthodes d'octroi d'un accès intercompte dans AWS Glue

Vous pouvez accorder l'accès à vos données à des AWS comptes externes en utilisant AWS Glue des méthodes ou en utilisant des AWS Lake Formation subventions entre comptes. Les AWS Glue

méthodes utilisent des politiques Gestion des identités et des accès AWS (IAM) pour obtenir un contrôle d'accès précis. Lake Formation utilise un modèle d'autorisations `GRANT/REVOKE` plus simple similaire aux commandes `GRANT/REVOKE` dans un système de base de données relationnelle.

Cette section décrit l'utilisation des méthodes AWS Glue. Pour de plus amples informations sur l'utilisation des autorisations intercomptes Lake Formation, consultez Octroi des autorisations Lake Formation dans le Guide du développeur AWS Lake Formation .

Il existe deux méthodes AWS Glue pour accorder un accès intercompte à une ressource :

- Utilisation d'une politique de ressources de catalogue de données
- Utilisation d'un rôle IAM

Octroi d'un accès intercompte à l'aide d'une politique de ressource

Voici les étapes générales d'octroi d'un accès intercompte à l'aide d'une politique de ressource de catalogue de données :

1. Un administrateur (ou toute autre identité autorisée) d'un compte A attache une politique de ressource au catalogue de données dans le compte A. Cette politique accorde au compte B des autorisations intercomptes spécifiques pour effectuer des opérations sur une ressource dans le catalogue du compte A.
2. Un administrateur du compte B attache une politique IAM à une identité IAM dans le compte B qui délègue les autorisations reçues du compte A.

   L'identité du compte B a désormais accès à la ressource spécifiée dans le compte A.

   L'identité a besoin d'une autorisation accordée à la fois par le propriétaire de la ressource (compte A) et son compte parent (compte B) afin d'être en mesure d'accéder à la ressource.

Octroi d'un accès intercompte avec un rôle IAM

Voici les étapes générales d'octroi d'un accès intercompte à l'aide d'un rôle IAM :

1. Un administrateur (ou toute autre identité autorisée) dans le compte qui possède la ressource (compte A) crée un rôle IAM.
2. L'administrateur du Compte A attache une politique au rôle qui accorde des autorisations intercomptes pour l'accès à la ressource en question.

3. L'administrateur du compte A attache une politique d'approbation au rôle qui identifie une identité IAM dans un compte différent (compte B) comme principal pouvant assumer le rôle.

   Le principal de la politique de confiance peut également être un directeur de AWS service si vous souhaitez accorder à un AWS service l'autorisation d'assumer ce rôle.
4. Un administrateur dans le compte B délègue désormais des autorisations à une ou plusieurs identités IAM du compte B afin qu'elle puisse assumer ce rôle. Cela donne à ces identités du compte B l'accès à la ressource du compte A.

Pour plus d'informations sur l'utilisation d'IAM pour déléguer des autorisations, veuillez consulter la section Access management (français non garanti) dans le Guide de l'utilisateur IAM. Pour plus d'informations sur les utilisateurs, les groupes, les rôles et les autorisations, consultez Identités (utilisateurs, groupes et rôles) dans le Guide de l'utilisateur IAM.

Pour une comparaison entre ces deux approches, veuillez consulter la rubrique Différence entre les rôles IAM et les politiques basées sur les ressources dans le Guide de l'utilisateur IAM. AWS Glue prend en charge les deux options, avec la restriction qu'une politique de ressources peut uniquement accorder l'accès aux ressources du catalogue de données.

Par exemple, pour accorder au rôle `Dev` du compte B l'accès à la base de données `db1` du compte A, attachez la politique de ressources suivante au catalogue du compte A.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:GetDatabase"
      ],
      "Principal": {
        "AWS": [
          "arn:aws:iam::111122223333:role/Dev"
        ]
      },
      "Resource": [
        "arn:aws:glue:us-east-1:111122223333:catalog",
        "arn:aws:glue:us-east-1:111122223333:database/db1"
```

```
      ]
    }
  ]
}
```

En outre, le compte B devrait attacher la politique IAM suivante au rôle `Dev` avant d'obtenir l'accès à db1 sur le compte A.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:GetDatabase"
      ],
      "Resource": [
        "arn:aws:glue:us-east-1:111122223333:catalog",
        "arn:aws:glue:us-east-1:111122223333:database/db1"
      ]
    }
  ]
}
```

Ajouter ou mettre à jour la politique de ressource du catalogue de données

Vous pouvez ajouter ou mettre à jour la politique de ressources du catalogue de AWS Glue données à l'aide de la console, de l'API ou AWS Command Line Interface (AWS CLI).

> **Important**
>
> Si vous avez déjà accordé des autorisations intercomptes à partir de votre compte avec AWS Lake Formation, l'ajout ou la mise à jour de la politique de ressource du catalogue de données nécessite une étape supplémentaire. Pour en savoir plus, veuillez consulter la rubrique Gestion des autorisations intercomptes avec AWS Glue et Lake Formation dans le Guide du développeur AWS Lake Formation .

Pour déterminer s'il existe des octrois intercomptes Lake Formation, utilisez l'opération d'API `glue:GetResourcePolicies` ou l' AWS CLI. Si `glue:GetResourcePolicies` renvoie des politiques autres qu'une politique de catalogue de données existante, les octrois Lake Formation existent. Pour plus d'informations, consultez la section Affichage de toutes les subventions entre comptes à l'aide du fonctionnement de l' GetResourcePolicies API dans le Guide du AWS Lake Formation développeur.

Pour ajouter ou mettre à jour la politique de ressource du catalogue de données (console)

1. Ouvrez la console AWS Glue à l'adresse https://console.aws.amazon.com/glue/.

   Connectez-vous en tant qu'utilisateur administratif Gestion des identités et des accès AWS (IAM) `glue:PutResourcePolicy` autorisé.
2. Dans le panneau de navigation, sélectionnez Settings (Paramètres).
3. Dans la page Paramètres du catalogue de données, sous Permissions (Autorisations), collez une politique de ressource dans la zone de texte. Ensuite, choisissez Save (Enregistrer).

   Si la console affiche une alerte indiquant que les autorisations de la politique s'ajouteront aux autorisations accordées à l'aide de Lake Formation, sélectionnez Proceed (Continuer).

Pour ajouter ou mettre à jour la politique de ressource du catalogue de données (AWS CLI)

- Soumettez une commande `aws glue put-resource-policy`. Si des autorisations Lake Formation existent déjà, assurez-vous d'inclure l'option `--enable-hybrid` avec la valeur `'TRUE'`.

  Pour obtenir des exemples d'utilisation de cette commande, veuillez consulter Exemples de politiques basées sur les ressources pour Glue AWS.

Créer un appel d'API intercompte

Toutes les opérations AWS Glue Data Catalog ont un champ `CatalogId`. Si les autorisations requises ont été accordées pour activer l'accès intercompte, un utilisateur peut créer des appels d'API du catalogue de données intercompte. L'appelant effectue cette opération en transmettant l'ID de compte AWS cible dans `CatalogId` afin d'accéder à la ressource de ce compte de destination.

Si aucune valeur `CatalogId` n'est fournie, AWS Glue utilise l'ID de compte de l'appelant par défaut, et l'appel n'est pas intercompte.

Créez un appel ETL intercompte

Certains AWS Glue PySpark et Scala APIs ont un champ d'ID de catalogue. Si toutes les autorisations requises ont été accordées pour permettre l'accès entre comptes, une tâche ETL peut effectuer PySpark et Scala appeler des opérations d'API entre comptes en transmettant l'ID du AWS compte cible dans le champ ID du catalogue pour accéder aux ressources du catalogue de données d'un compte cible.

Si aucun ID de catalogue n'est fourni, AWS Glue utilise l'ID de compte de l'appelant par défaut, et l'appel n'est pas intercompte.

Pour PySpark APIs ce support`catalog_id`, voirGlueContext classe. Pour APIs ce support Scala`catalogId`, voirAWS GlueScala GlueContext APIs.

L'exemple suivant montre les autorisations requises par le bénéficiaire pour exécuter une tâche ETL. Dans cet exemple, *`grantee-account-id`* est `catalog-id` le client qui exécute le travail et *`grantor-account-id`* le propriétaire de la ressource. Cet exemple accorde l'autorisation à toutes les ressources de catalogue dans le compte du concédant. Pour limiter l'étendue des ressources accordées, vous pouvez fournir des informations spécifiques au catalogue, ARNs à la base de données, à la table et à la connexion.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:GetConnection",
        "glue:GetDatabase",
        "glue:GetTable",
        "glue:GetPartition"
      ],
      "Principal": {
        "AWS": [
          "arn:aws:iam::111122223333:root"
```

```
        ]
      },
      "Resource": [
        "arn:aws:glue:us-east-1:111122223333:*"
      ]
    }
  ]
}
```

> **Note**
>
> Si la table dans le compte du concédant pointe vers un emplacement Amazon S3, qui est également dans le compte du concédant, le rôle IAM utilisé pour exécuter une tâche ETL dans le compte du concédant doit avoir l'autorisation de répertorier et d'obtenir les objets à partir du compte du concédant.

Étant donné que le client du Compte A est déjà autorisé à créer et exécuter des tâches ETL, les étapes élémentaires de configuration d'une tâche ETL pour l'accès intercompte sont les suivantes :

1. Autorisez l'accès intercompte aux données (ignorez cette étape si l'accès intercompte Amazon S3 est déjà configuré).
   a. Mettez à jour la politique de compartiment Amazon S3 du compte B pour autoriser un accès intercompte à partir du compte A.
   b. Mettez à jour la politique IAM du compte A pour autoriser l'accès au compartiment du Compte B.
2. Autoriser l'accès intercompte aux catalogues de données.
   a. Créez ou mettez à jour la politique de ressource attachée au catalogue de données dans le compte B pour autoriser l'accès depuis le compte A.
   b. Mettez à jour la politique IAM du compte A pour autoriser l'accès au catalogue de données dans le compte B.

Journalisation entre comptes CloudTrail

Lorsqu'une tâche d'AWS Glueextraction, de transformation et de chargement (ETL) accède aux données sous-jacentes d'une table de catalogue de données partagée via des autorisations

AWS Lake Formation entre comptes, il existe un comportement de AWS CloudTrail journalisation supplémentaire.

Aux fins de cette discussion, le AWS compte qui a partagé la table est le compte propriétaire, et le compte avec lequel la table a été partagée est le compte destinataire. Lorsqu'une tâche ETL du compte destinataire accède aux données de la table du compte propriétaire, l' CloudTrail événement d'accès aux données ajouté aux journaux du compte destinataire est copié dans les journaux du CloudTrail compte propriétaire. Cela permet aux comptes propriétaires de suivre les accès aux données par les différents comptes destinataires. Par défaut, les CloudTrail événements n'incluent pas d'identifiant principal lisible par l'homme (ARN principal). Un administrateur du compte destinataire peut choisir d'inclure l'ARN du principal dans les journaux.

Pour plus d'informations, consultez la section CloudTrailConnexion entre comptes dans le Guide du AWS Lake Formation développeur.

Voir aussi

- the section called "Journalisation et surveillance"

Propriété et facturation des ressources intercomptes

Lorsqu'un utilisateur d'un AWS compte (compte A) crée une nouvelle ressource telle qu'une base de données dans un autre compte (compte B), cette ressource appartient alors au compte B, le compte sur lequel elle a été créée. L'administrateur du Compte B obtient automatiquement des autorisations complètes pour accéder à la nouvelle ressource, y compris la lecture, l'écriture et l'octroi d'autorisations d'accès à un compte tiers. L'utilisateur du Compte A peut accéder à la ressource qu'il vient de créer uniquement s'il dispose des autorisations appropriées accordées par le Compte B.

Les coûts de stockage et les autres coûts directement liés à la nouvelle ressource sont facturés au Compte B, le propriétaire de la ressource. Le coût des demandes de l'utilisateur qui a créé la ressource est facturé au compte du demandeur, le Compte A.

Pour plus d'informations sur la AWS Glue facturation et la tarification, consultez la section Fonctionnement de la AWS tarification.

Limitations des accès inter-comptes

AWS GlueL'accès intercompte à a les limitations suivantes :

- Permettre l'accès entre comptes à AWS Glue n'est pas autorisé si vous avez créé des bases de données et des tables à l'aide d'Amazon Athena ou d'Amazon Redshift Spectrum avant la prise en charge d'une région pour AWS Glue et le propriétaire de la ressource compte n'a pas migré le catalogue de données Amazon Athena vers AWS Glue. Vous pouvez trouver l'état actuel de la migration à l'aide de GetCatalogImportStatus (get_catalog_import_status). Pour plus d'informations sur la migration d'un catalogue Athena vers leAWS Glue, consultez la section Mise à niveau vers le AWS Glue Data Catalog step-by-step dans le guide de l'utilisateur d'Amazon Athena.
- L'accès intercompte est uniquement pris en charge pour les ressources du catalogue de données, y compris les bases de données, les tables, les fonctions définies par l'utilisateur et les connexions.
- L'accès entre comptes au catalogue de données depuis Athena nécessite que vous enregistrez le catalogue en tant que ressource `DataCatalog` d'Athena. Pour obtenir des instructions, veuillez consulter la rubrique Enregistrement d'un AWS Glue Data Catalog à partir d'un autre compte dans le Guide de l'utilisateur Amazon Athena.

Résolution des problèmes d'identité et d'accès à AWS Glue

Utilisez les informations suivantes pour diagnostiquer et résoudre les problèmes courants que vous pouvez rencontrer lorsque vous travaillez avec AWS Glue et IAM.

Rubriques

- Je ne suis pas autorisé à effectuer une action dans AWS Glue
- Je ne suis pas autorisé à effectuer iam : PassRole
- Je souhaite autoriser des personnes extérieures à moi Compte AWS à accéder à mes ressources AWS Glue

Je ne suis pas autorisé à effectuer une action dans AWS Glue

Si vous recevez une erreur qui indique que vous n'êtes pas autorisé à effectuer une action, vos politiques doivent être mises à jour afin de vous permettre d'effectuer l'action.

L'exemple d'erreur suivant se produit quand l'utilisateur IAM `mateojackson` tente d'utiliser la console pour afficher des informations détaillées sur une ressource *`my-example-widget`* fictive, mais ne dispose pas des autorisations `glue:`*`GetWidget`* fictives.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 glue:GetWidget on resource: my-example-widget
```

Dans ce cas, la politique qui s'applique à l'utilisateur `mateojackson` doit être mise à jour pour autoriser l'accès à la ressource *`my-example-widget`* à l'aide de l'action `glue:`*`GetWidget`*.

Si vous avez besoin d'aide, contactez votre AWS administrateur. Votre administrateur vous a fourni vos informations d'identification de connexion.

Je ne suis pas autorisé à effectuer iam : PassRole

Si vous recevez un message d'erreur indiquant que vous n'êtes pas autorisé à effectuer l'`iam:PassRole`action, vos politiques doivent être mises à jour pour vous permettre de transmettre un rôle à AWS Glue.

Certains vous Services AWS permettent de transmettre un rôle existant à ce service au lieu de créer un nouveau rôle de service ou un rôle lié à un service. Pour ce faire, vous devez disposer des autorisations nécessaires pour transmettre le rôle au service.

L'exemple d'erreur suivant se produit lorsqu'un utilisateur IAM nommé `marymajor` essaie d'utiliser la console pour effectuer une action dans AWS Glue. Toutefois, l'action nécessite que le service ait des autorisations accordées par un rôle de service. Mary n'est pas autorisée à transmettre le rôle au service.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

Dans ce cas, les politiques de Mary doivent être mises à jour pour lui permettre d'exécuter l'action `iam:PassRole`.

Si vous avez besoin d'aide, contactez votre AWS administrateur. Votre administrateur vous a fourni vos informations d'identification de connexion.

Je souhaite autoriser des personnes extérieures à moi Compte AWS à accéder à mes ressources AWS Glue

Vous pouvez créer un rôle que les utilisateurs provenant d'autres comptes ou les personnes extérieures à votre organisation pourront utiliser pour accéder à vos ressources. Vous pouvez spécifier qui est autorisé à assumer le rôle. Pour les services qui prennent en charge les politiques basées sur les ressources ou les listes de contrôle d'accès (ACLs), vous pouvez utiliser ces politiques pour autoriser les utilisateurs à accéder à vos ressources.

Pour plus d'informations, consultez les éléments suivants :

- Pour savoir si AWS Glue prend en charge ces fonctionnalités, consultezComment AWS Glue fonctionne avec IAM.
- Pour savoir comment fournir l'accès à vos ressources sur celles Comptes AWS que vous possédez, consultez la section Fournir l'accès à un utilisateur IAM dans un autre utilisateur Compte AWS que vous possédez dans le Guide de l'utilisateur IAM.
- Pour savoir comment fournir l'accès à vos ressources à des tiers Comptes AWS, consultez la section Fournir un accès à des ressources Comptes AWS détenues par des tiers dans le guide de l'utilisateur IAM.
- Pour savoir comment fournir un accès par le biais de la fédération d’identité, consultez Fournir un accès à des utilisateurs authentifiés en externe (fédération d’identité) dans le Guide de l’utilisateur IAM.
- Pour en savoir plus sur la différence entre l’utilisation des rôles et des politiques basées sur les ressources pour l’accès intercompte, consultez Accès intercompte aux ressources dans IAM dans le Guide de l’utilisateur IAM.

AWS Lake Formation modèles de contrôle d'accès

AWS Glue La version 5.0 et les versions ultérieures prennent en charge deux modèles d'accès aux données via AWS Lake Formation :

Rubriques

- Utilisation de AWS Glue with AWS Lake Formation pour un accès complet à la table
- Utiliser AWS Glue with AWS Lake Formation pour un contrôle d'accès précis

Utilisation de AWS Glue with AWS Lake Formation pour un accès complet à la table

Présentation de l’accès complet aux tables

AWS Glue Les versions 5.0 et ultérieures prennent en charge le contrôle de l'accès complet aux tables (FTA) dans Apache Spark en fonction de vos politiques définies dans AWS Lake Formation. Cette fonctionnalité permet d'effectuer des opérations de lecture et d'écriture à partir de vos tâches AWS Glue Spark sur des tables AWS Lake Formation enregistrées lorsque le rôle de tâche dispose d'un accès complet aux tables. FTA est idéal pour les cas d'utilisation qui doivent respecter les réglementations de sécurité au niveau des tables et prend en charge les fonctionnalités de Spark,

notamment les ensembles de données distribués résilients (RDDs), les bibliothèques personnalisées et les fonctions définies par l'utilisateur (UDFs) avec des AWS Lake Formation tables.

Lorsqu'une tâche AWS Glue Spark est configurée pour un accès complet aux tables (FTA), les AWS Lake Formation informations d'identification sont utilisées pour accéder aux données read/write Amazon S3 relatives aux tables AWS Lake Formation enregistrées, tandis que les informations d'identification du rôle d'exécution de la tâche sont utilisées pour read/write les tables non enregistrées auprès AWS Lake Formation de celles-ci. Cette fonctionnalité active les opérations de langage de manipulation de données (DML), notamment les instructions CREATE, ALTER, DELETE, UPDATE et MERGE INTO sur les tables Apache Hive et Iceberg.

> **Note**
>
> Examinez vos exigences et déterminez si le contrôle précis des accès (FGAC) ou l'accès complet aux tables (FTA) répondent à vos besoins. Une seule méthode AWS Lake Formation d'autorisation peut être activée pour une AWS Glue tâche donnée. Une tâche ne peut pas exécuter simultanément l'accès complet aux tables (FTA) et le contrôle précis des accès (FGAC)

Comment fonctionne Full-Table Access (FTA) sur AWS Glue

AWS Lake Formation propose deux approches pour le contrôle d'accès aux données : le contrôle d'accès détaillé (FGAC) et l'accès complet aux tables (FTA). Le FGAC assure une sécurité renforcée grâce à un filtrage au niveau des colonnes, des lignes et des cellules, l'idéal pour les scénarios nécessitant des autorisations précises. Le FTA est idéal pour les scénarios de contrôle d'accès simples où vous avez besoin d'autorisations au niveau de la table. Il simplifie la mise en œuvre en éliminant le besoin d'activer le mode d'accès précis, améliore les performances et réduit les coûts en évitant le pilote système et les exécuteurs système, et prend en charge les opérations de lecture et d'écriture (y compris les commandes CREATE, ALTER, DELETE, UPDATE et MERGE INTO).

Dans la AWS Glue version 4.0, l'accès aux données AWS Lake Formation basé sur la GlueContext classe s'effectuait par le biais de la classe utilitaire fournie par AWS Glue. Dans la AWS Glue version 5.0, l'accès aux données AWS Lake Formation basé est disponible via Spark SQL natif, Spark DataFrames, et continue d'être pris en charge par le biais GlueContext de classes.

Mise en œuvre de l'accès complet aux tables

Étape 1 : activer l'accès complet à la table dans AWS Lake Formation

Pour utiliser le mode Accès complet aux tables (FTA), vous devez autoriser les moteurs de requêtes tiers à accéder aux données sans que la balise de session IAM ne soit validée dans AWS Lake Formation. Pour l'activer, suivez les étapes de la section Intégration des applications pour un accès complet aux tables.

Étape 2 : configurer des autorisations IAM pour le rôle d'exécution des tâches

Pour accéder en lecture ou en écriture aux données sous-jacentes, en plus des AWS Lake Formation autorisations, un rôle d'exécution de tâche nécessite l'autorisation `lakeformation:GetDataAccess` IAM. Avec cette autorisation, AWS Lake Formation accorde la demande d'informations d'identification temporaires pour accéder aux données.

Voici un exemple de politique expliquant comment fournir les autorisations IAM pour accéder à un script dans Amazon S3, le chargement de journaux sur Amazon S3, les autorisations d'API AWS Glue et les autorisations d'accès à AWS Lake Formation.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "ScriptAccess",
      "Effect": "Allow",
      "Action": [
        "s3:GetObject"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/scripts/*"
      ]
    },
    {
      "Sid": "LoggingAccess",
      "Effect": "Allow",
      "Action": [
        "s3:PutObject"
      ],
```

```
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/logs/*"
      ]
    },
    {
      "Sid": "GlueCatalogAccess",
      "Effect": "Allow",
      "Action": [
        "glue:GetDatabase",
        "glue:GetDatabases",
        "glue:GetTable",
        "glue:GetTables",
        "glue:GetPartition",
        "glue:GetPartitions",
        "glue:CreateTable",
        "glue:UpdateTable"
      ],
      "Resource": [
        "arn:aws:glue:us-east-1:111122223333:catalog",
        "arn:aws:glue:us-east-1:111122223333:database/default",
        "arn:aws:glue:us-east-1:111122223333:table/default/*"
      ]
    },
    {
      "Sid": "LakeFormationAccess",
      "Effect": "Allow",
      "Action": [
        "lakeformation:GetDataAccess"
      ],
      "Resource": "*"
    }
  ]
}
```

Étape 2.1 Configuration AWS Lake Formation des autorisations

AWS Glue Les tâches Spark qui lisent des données depuis Amazon S3 nécessitent l'autorisation AWS Lake Formation SELECT.

AWS Glue Spark Jobs pour write/delete lesquels les données d'Amazon S3 nécessitent AWS Lake Formation TOUTES les autorisations.

AWS Glue Les tâches Spark qui interagissent avec le catalogue de AWS Glue données nécessitent les autorisations DESCRIBE, ALTER, DROP, selon le cas.

Étape 3 : Initialisation d'une session Spark pour un accès complet à la table à l'aide de AWS Lake Formation

Pour accéder aux tables enregistrées auprès de Spark AWS Lake Formation, les configurations suivantes doivent être définies lors de l'initialisation de Spark afin que Spark utilise les AWS Lake Formation informations d'identification.

Pour accéder aux tables enregistrées auprès de vous AWS Lake Formation, vous devez configurer explicitement votre session Spark pour utiliser les AWS Lake Formation informations d'identification. Ajoutez les configurations suivantes lors de l'initialisation de votre session Spark :

```
from pyspark.sql import SparkSession

# Initialize Spark session with Lake Formation configurations
spark = SparkSession.builder \
    .appName("Lake Formation Full Table Access") \
    .config("spark.sql.catalog.glue_catalog",
 "org.apache.spark.sql.catalog.hive.GlueCatalog") \
    .config("spark.sql.catalog.glue_catalog.glue.lakeformation-enabled", "true") \
    .config("spark.sql.defaultCatalog", "glue_catalog") \
    .getOrCreate()
```

Principales configurations :

- `spark.sql.catalog.glue_catalog`: enregistre un catalogue nommé « glue_catalog » qui utilise l'implémentation GlueCatalog
- `spark.sql.catalog.glue_catalog.glue.lakeformation-enabled`: active explicitement AWS Lake Formation l'intégration pour ce catalogue
- Le nom du catalogue (« glue_catalog » dans cet exemple) peut être personnalisé, mais il doit être cohérent dans les deux paramètres de configuration.

Hive

```
--conf spark.hadoop.fs.s3.credentialsResolverClass=com.amazonaws.AWS
 Glue.accesscontrol.AWSLakeFormationCredentialResolver
```

```
--conf spark.hadoop.fs.s3.useDirectoryHeaderAsFolderObject=true
--conf spark.hadoop.fs.s3.folderObject.autoAction.disabled=true
--conf spark.sql.catalog.skipLocationValidationOnCreateTable.enabled=true
--conf spark.sql.catalog.createDirectoryAfterTable.enabled=true
--conf spark.sql.catalog.dropDirectoryBeforeTable.enabled=true
```

Iceberg

```
--conf spark.hadoop.fs.s3.credentialsResolverClass=com.amazonaws.AWS
 Glue.accesscontrol.AWSLakeFormationCredentialResolver
--conf spark.hadoop.fs.s3.useDirectoryHeaderAsFolderObject=true
--conf spark.hadoop.fs.s3.folderObject.autoAction.disabled=true
--conf spark.sql.catalog.skipLocationValidationOnCreateTable.enabled=true
--conf spark.sql.catalog.createDirectoryAfterTable.enabled=true
--conf spark.sql.catalog.dropDirectoryBeforeTable.enabled=true
--conf spark.sql.catalog.<catalog>.AWS Glue.lakeformation-enabled=true
```

- `spark.hadoop.fs.s3.credentialsResolverClass=com.amazonaws.AWS Glue.accesscontrol.AWSLakeFormationCredentialResolver`: Configurez le système de fichiers EMR (EMRFS) pour utiliser les informations d'identification AWS Lake Formation S3 pour les tables enregistrées. AWS Lake Formation Si la table n'est pas enregistrée, utilisez les informations d'identification du rôle d'exécution de la tâche.
- `spark.hadoop.fs.s3.useDirectoryHeaderAsFolderObject=true` et `spark.hadoop.fs.s3.folderObject.autoAction.disabled=true` : configurez EMRFS pour utiliser l'en-tête de type de contenu application/x-directory au lieu du suffixe $folder$ lors de la création de dossiers S3. Cela est nécessaire lors de la lecture de AWS Lake Formation tables, car les AWS Lake Formation informations d'identification ne permettent pas de lire les dossiers de tables portant le suffixe $folder$.
- `spark.sql.catalog.skipLocationValidationOnCreateTable.enabled=true` : configurez Spark pour ignorer la validation de l'emplacement vide de la table avant sa création. Cela est nécessaire pour les tables AWS Lake Formation enregistrées, car les AWS Lake Formation informations d'identification permettant de vérifier l'emplacement vide ne sont disponibles qu'après la création de la table du catalogue de AWS Glue données. Sans cette configuration, les informations d'identification du rôle d'exécution de la tâche valideront l'emplacement de la table vide.

- `spark.sql.catalog.createDirectoryAfterTable.enabled=true` : configurez Spark pour créer le dossier Amazon S3 après la création de la table dans le métastore Hive. Cela est obligatoire pour les tables AWS Lake Formation enregistrées, car les AWS Lake Formation informations d'identification permettant de créer le dossier Amazon S3 ne sont disponibles qu'après la création de la table du catalogue de AWS Glue données.
- `spark.sql.catalog.dropDirectoryBeforeTable.enabled=true` : configurez Spark pour qu’il supprime le dossier Amazon S3 avant de supprimer la table dans le métastore Hive. Cela est nécessaire pour les tables AWS Lake Formation enregistrées, car les AWS Lake Formation informations d'identification permettant de supprimer le dossier S3 ne sont pas disponibles après la suppression des tables du catalogue de AWS Glue données.
- `spark.sql.catalog.<catalog>.AWS Glue.lakeformation-enabled=true`: configurez le catalogue Iceberg pour utiliser les informations d'identification AWS Lake Formation Amazon S3 pour les tables AWS Lake Formation enregistrées. Si la table n’est pas enregistrée, utilisez les informations d’identification d’environnement par défaut.

Modèles d’utilisation

Utiliser FTA avec DataFrames

Pour les utilisateurs habitués à Spark, il DataFrames peut être utilisé avec AWS Lake Formation Full Table Access.

AWS Glue La version 5.0 ajoute le support natif de Spark pour l'accès complet aux tables de Lake Formation, simplifiant ainsi la façon dont vous travaillez avec les tables protégées. Cette fonctionnalité permet aux tâches AWS Glue Spark AWS Glue 5.0 de lire et d'écrire directement des données lorsque l'accès complet à la table est accordé, supprimant ainsi les limitations qui limitaient auparavant certaines opérations d'extraction, de transformation et de chargement (ETL). Vous pouvez désormais tirer parti des fonctionnalités avancées de Spark, notamment des ensembles de données distribués résilients (RDDs), des bibliothèques personnalisées et des fonctions définies par l'utilisateur (UDFs) avec des AWS Lake Formation tables.

Native Spark FTA en version 5.0 AWS Glue

AWS Glue La version 5.0 prend en charge le contrôle d'accès aux tables complètes (FTA) dans Apache Spark en fonction de vos politiques définies dans. AWS Lake Formation Ce niveau de contrôle est idéal pour les cas d’utilisation qui doivent être conformes aux réglementations de sécurité au niveau de la table.

Exemple de table Apache Iceberg

```
from pyspark.sql import SparkSession

catalog_name = "spark_catalog"
aws_region = "us-east-1"
aws_account_id = "123456789012"
warehouse_path = "s3://amzn-s3-demo-bucket/warehouse/"

spark = SparkSession.builder \

 .config("spark.sql.extensions","org.apache.iceberg.spark.extensions.IcebergSparkSessionExtensi
 \
    .config(f"spark.sql.catalog.{catalog_name}",
 "org.apache.iceberg.spark.SparkSessionCatalog") \
    .config(f"spark.sql.catalog.{catalog_name}.warehouse", f"{warehouse_path}") \
    .config(f"spark.sql.catalog.{catalog_name}.client.region",f"{aws_region}") \
    .config(f"spark.sql.catalog.{catalog_name}.glue.account-id",f"{aws_account_id}") \
    .config(f"spark.sql.catalog.{catalog_name}.glue.lakeformation-enabled","true") \
    .config(f"spark.sql.catalog.dropDirectoryBeforeTable.enabled", "true") \
    .config(f"spark.sql.catalog.{catalog_name}.catalog-impl",
 "org.apache.iceberg.aws.glue.GlueCatalog") \
    .config(f"spark.sql.catalog.{catalog_name}.io-impl",
 "org.apache.iceberg.aws.s3.S3FileIO") \
    .config("spark.sql.defaultCatalog", catalog_name) \  # Add this line
    .getOrCreate()

database_name = "your_database"
table_name = "your_table"

df = spark.sql(f"select * from {database_name}.{table_name}")
df.show()
```

Autorisations IAM nécessaires

Votre AWS Glue rôle d'exécution de tâches doit avoir :

```
{
    "Action": "lakeformation:GetDataAccess",
    "Resource": "*",
    "Effect": "Allow"
}
```

Plus les autorisations d’accès S3 appropriées pour vos emplacements de vos données.

Configuration de Lake Formation

Avant d'utiliser Spark FTA natif dans la AWS Glue version 5.0 :

1. Autoriser les moteurs de requêtes tiers à accéder aux données sans validation des balises de session IAM dans AWS Lake Formation
2. Accordez les autorisations de table appropriées à votre AWS Glue rôle d'exécution de tâches via AWS Lake Formation la console
3. Configurez votre session Spark avec les paramètres requis indiqués dans l’exemple ci-dessus.

Utiliser FTA avec DynamicFrames

AWS Glue DynamicFrames peut être utilisé avec AWS Lake Formation Full Table Access pour des opérations ETL optimisées. L’accès complet aux tables (FTA) fournit un modèle de sécurité qui accorde des autorisations au niveau de la table, ce qui permet un traitement des données plus rapide que le contrôle précis des accès (FGAC), car il évite les surcharges liées aux vérifications d’autorisation au niveau des lignes et des colonnes. Cette approche est utile lorsque vous devez traiter des tables entières et que les autorisations au niveau des tables répondent à vos exigences de sécurité.

Dans la AWS Glue version 4.0, DynamicFrames FTA nécessitait une GlueContext configuration spécifique. Alors que le DynamicFrame code AWS Glue 4.0 existant avec FTA continuera de fonctionner dans la version AWS Glue 5.0, la nouvelle version offre également un support natif de Spark FTA avec une plus grande flexibilité. Pour les nouveaux développements, pensez à utiliser l'approche native de Spark décrite dans la DataFrames section, en particulier si vous avez besoin de fonctionnalités supplémentaires telles que des ensembles de données distribués résilients (RDDs), des bibliothèques personnalisées et des fonctions définies par l'utilisateur (UDFs) avec des AWS Lake Formation tables.

Autorisations nécessaires

Le rôle IAM exécutant votre tâche Glue doit disposer des éléments suivants :

- `lakeformation:GetDataAccess` autorisation
- Autorisations de tables Lake Formation appropriées accordées via la console Lake Formation

Exemple de DynamicFrame mise en œuvre dans la AWS Glue version 5.0

```
from awsglue.context import GlueContext
from pyspark.context import SparkContext

# Initialize Glue context
sc = SparkContext()
glueContext = GlueContext(sc)

# Configure catalog for Iceberg tables
catalog_name = "glue_catalog"
aws_region = "us-east-1"
aws_account_id = "123456789012"
warehouse_path = "s3://amzn-s3-demo-bucket/warehouse/"

spark = glueContext.spark_session
spark.conf.set(f"spark.sql.catalog.{catalog_name}",
 "org.apache.iceberg.spark.SparkCatalog")
spark.conf.set(f"spark.sql.catalog.{catalog_name}.warehouse", f"{warehouse_path}")
spark.conf.set(f"spark.sql.catalog.{catalog_name}.catalog-impl",
 "org.apache.iceberg.aws.glue.GlueCatalog")
spark.conf.set(f"spark.sql.catalog.{catalog_name}.io-impl",
 "org.apache.iceberg.aws.s3.S3FileIO")
spark.conf.set(f"spark.sql.catalog.{catalog_name}.glue.lakeformation-enabled","true")
spark.conf.set(f"spark.sql.catalog.{catalog_name}.client.region",f"{aws_region}")
spark.conf.set(f"spark.sql.catalog.{catalog_name}.glue.id", f"{aws_account_id}")

# Read Lake Formation-protected table with DynamicFrame
df = glueContext.create_data_frame.from_catalog(
    database="your_database",
    table_name="your_table"
)
```

Configuration supplémentaire

Configurer le mode d'accès complet aux tables dans les AWS Glue Studio blocs-notes

Pour accéder aux tables AWS Lake Formation enregistrées à partir de sessions Spark interactives dans des AWS Glue Studio blocs-notes, vous devez utiliser le mode d'autorisation de compatibilité. Utilisez la commande magique `%%configure` pour définir votre configuration Spark avant de démarrer votre session interactive. Cette configuration doit être la première commande de votre bloc-

notes, car elle ne peut pas être appliquée une fois la session démarrée. Choisissez la configuration en fonction de votre type de table :

Pour les tables Hive

```
%%configure
--conf
 spark.hadoop.fs.s3.credentialsResolverClass=com.amazonaws.glue.accesscontrol.AWSLakeFormationC
--conf spark.hadoop.fs.s3.useDirectoryHeaderAsFolderObject=true
--conf spark.hadoop.fs.s3.folderObject.autoAction.disabled=true
--conf spark.sql.catalog.skipLocationValidationOnCreateTable.enabled=true
--conf spark.sql.catalog.createDirectoryAfterTable.enabled=true
--conf spark.sql.catalog.dropDirectoryBeforeTable.enabled=true
```

Pour les tables Iceberg

```
%%configure
--conf
 spark.hadoop.fs.s3.credentialsResolverClass=com.amazonaws.glue.accesscontrol.AWSLakeFormationC
--conf spark.hadoop.fs.s3.useDirectoryHeaderAsFolderObject=true
--conf spark.hadoop.fs.s3.folderObject.autoAction.disabled=true
--conf spark.sql.catalog.skipLocationValidationOnCreateTable.enabled=true
--conf spark.sql.catalog.createDirectoryAfterTable.enabled=true
--conf spark.sql.catalog.dropDirectoryBeforeTable.enabled=true
--conf spark.sql.catalog.glue_catalog.glue.lakeformation-enabled=true
--conf spark.sql.catalog.glue_catalog.warehouse=s3://example-s3-bucket_DATA_LOCATION
--conf spark.sql.catalog.glue_catalog.catalog-
impl=org.apache.iceberg.aws.glue.GlueCatalog
--conf spark.sql.catalog.glue_catalog.io-impl=org.apache.iceberg.aws.s3.S3FileIO
--conf spark.sql.catalog.glue_catalog.glue.account-id=ACCOUNT_ID
--conf spark.sql.catalog.glue_catalog.glue.region=REGION
```

Remplacez les espaces réservés :

- S3_DATA_LOCATION : *s3://amzn-s3-demo-bucket*
- RÉGION : *AWS Region (e.g., us-east-1)*
- IDENTIFIANT DU COMPTE : *Your AWS Account ID*

> **Note**
>
> Vous devez définir ces configurations avant d'exécuter des opérations Spark dans votre bloc-notes.

Opérations prises en charge

Ces opérations utiliseront des AWS Lake Formation informations d'identification pour accéder aux données de la table.

> **Note**
>
> Lors de l'activation AWS Lake Formation :
>
> - Pour FTA : activer la configuration Spark `spark.sql.catalog.{catalog_name}.glue.lakeformation-enabled`

- CREATE TABLE
- ALTER TABLE
- INSERT INTO
- INSERT OVERWRITE
- SELECT
- UPDATE
- MERGE INTO
- DELETE FROM
- ANALYZE TABLE
- REPAIR TABLE
- DROP TABLE
- Requêtes de source de données Spark
- Écritures de source de données Spark

> **Note**
>
> Les opérations non répertoriées ci-dessus continueront à utiliser les autorisations IAM pour accéder aux données de la table.

Migration de la AWS Glue version 4.0 à la version AWS Glue 5.0 FTA

Lors de la migration de la AWS Glue version 4.0 GlueContext FTA vers la AWS Glue version 5.0 native de Spark FTA :

1. Autorisez les moteurs de requêtes tiers à accéder aux données sans validation des balises de session IAM dans AWS Lake Formation. Suivez Étape 1 : activer l'accès complet à la table dans AWS Lake Formation.
2. Vous n'avez pas besoin de modifier le rôle d'exécution des tâches. Vérifiez toutefois que le rôle AWS Glue d'exécution de la tâche dispose de l'autorisation lakeformation : GetDataAccess IAM.
3. Modifiez les configurations de session Spark dans le script. Assurez-vous que les configurations Spark suivantes sont présentes :

```
--conf spark.sql.catalog.spark_catalog=org.apache.iceberg.spark.SparkSessionCatalog
--conf spark.sql.catalog.spark_catalog.warehouse=s3://<bucket-name>/warehouse/
--conf spark.sql.catalog.spark_catalog.client.region=<REGION>
--conf spark.sql.catalog.spark_catalog.glue.account-id=ACCOUNT_ID
--conf spark.sql.catalog.spark_catalog.glue.lakeformation-enabled=true
--conf spark.sql.catalog.dropDirectoryBeforeTable.enabled=true
```

4. Mettez à jour le script tel qu' GlueContext DataFrames il est remplacé par Native Spark DataFrames.
5. Mettez à jour votre AWS Glue tâche pour utiliser la AWS Glue version 5.0

Considérations et restrictions

- Si une table Hive est créée à l'aide d'une tâche pour laquelle l'accès complet aux tables n'est pas activé et qu'aucun enregistrement n'est inséré, les lectures ou écritures suivantes à partir d'une tâche avec accès complet aux tables échoueront. Cela est dû au fait que AWS Glue Spark sans accès complet à la table ajoute le suffixe $folder$ au nom du dossier de la table. Pour résoudre cela, vous pouvez procéder comme suit :
  - Insérez au moins une ligne dans la table à partir d'une tâche pour laquelle FTA n'est pas activé.

  - Configurez la tâche pour laquelle FTA n'est pas activé afin de ne pas utiliser le suffixe $folder$ dans le nom du dossier dans S3. Cela peut être réalisé en configurant `spark.hadoop.fs.s3.useDirectoryHeaderAsFolderObject=true` pour Spark.
  - Créez un dossier Amazon S3 à l'emplacement de la table `s3://path/to/table/table_name` à l'aide de la console Amazon S3 ou de la CLI Amazon S3.
- L'accès complet aux tables fonctionne exclusivement avec EMR Filesystem (EMRFS). S3A Filesystem n'est pas compatible.
- L'accès complet aux tables est pris en charge pour les tables Hive et Iceberg. La prise en charge pour les tables Hudi et Delta n'a pas encore été ajoutée.
- Les tâches faisant référence à des tables dotées de règles de contrôle d' AWS Lake Formation accès fin (FGAC) ou à des vues de catalogue de AWS Glue données échoueront. Pour interroger une table avec des règles FGAC ou une vue de catalogue de AWS Glue données, vous devez utiliser le mode FGAC. Vous pouvez activer le mode FGAC en suivant les étapes décrites dans la AWS documentation : Utilisation AWS Glue avec AWS Lake Formation pour un contrôle d'accès précis.
- L'accès complet aux tables ne prend pas en charge Spark Streaming.
- Ne peut pas être utilisé simultanément avec FGAC.

Utiliser AWS Glue with AWS Lake Formation pour un contrôle d'accès précis

Présentation de

Avec AWS la version 5.0 et les versions ultérieures de Glue, vous pouvez AWS Lake Formation appliquer des contrôles d'accès précis aux tables du catalogue de données soutenues par S3. Cette fonctionnalité vous permet de configurer des contrôles d'accès au niveau des tables, des lignes, des colonnes et des cellules pour les requêtes read contenues dans vos tâches AWS Glue pour Apache Spark. Consultez les sections suivantes pour en savoir plus sur Lake Formation et sur son utilisation avec AWS Glue.

`GlueContext`le contrôle d'accès basé sur une table avec des AWS Lake Formation autorisations prises en charge dans Glue 4.0 ou antérieur n'est pas pris en charge dans Glue 5.0. Utilisez le nouveau contrôle précis des accès (FGAC) natif Spark de Glue 5.0. Prenez note des détails suivants :

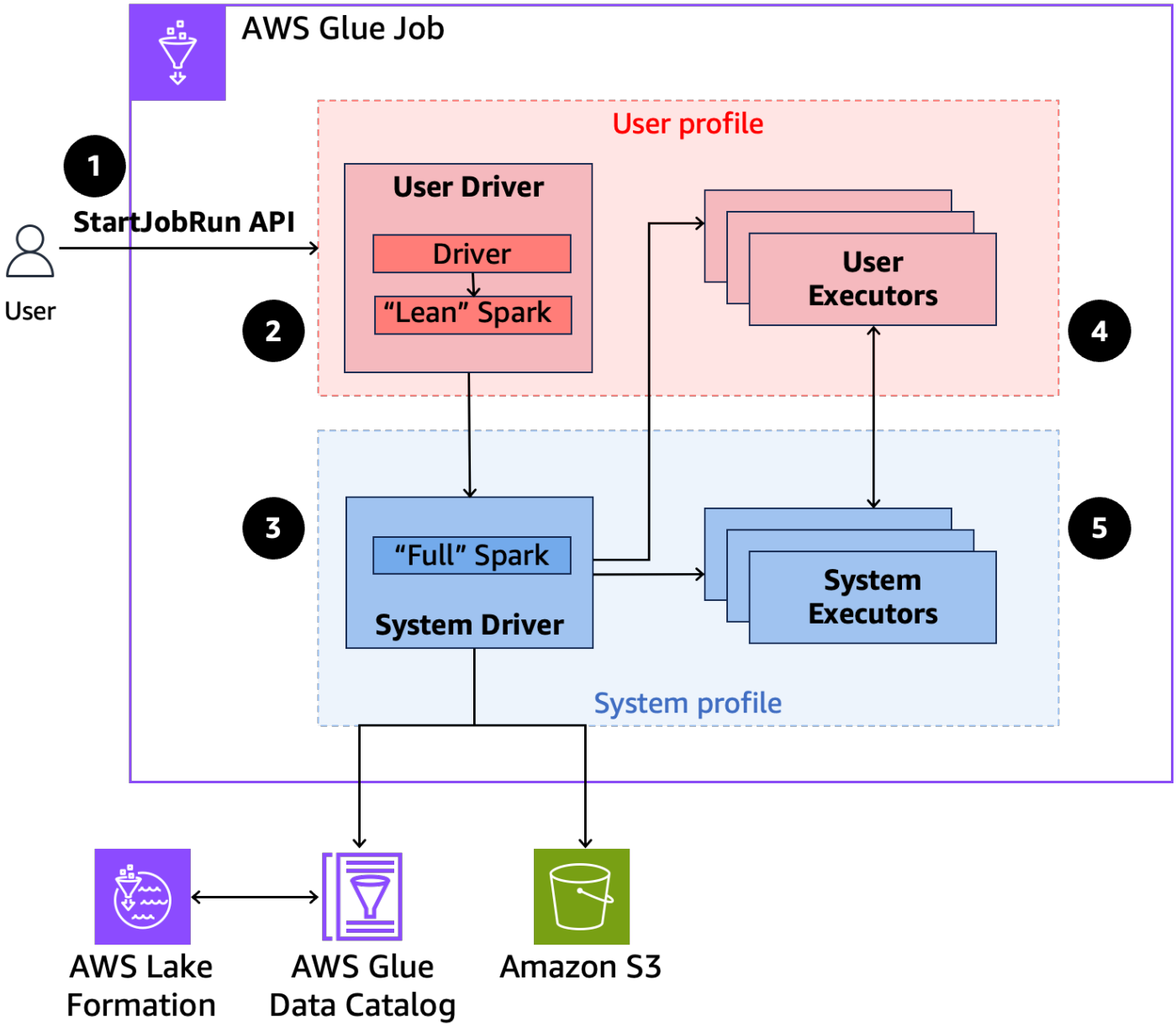
- Si vous avez besoin d'un contrôle d'accès détaillé (FGAC) pour le contrôle row/column/cell d'accès, vous devez migrer depuis `GlueContext` DynamicFrame /Glue dans Glue 4.0 et avant le dataframe Spark dans Glue 5.0. Pour obtenir des exemples, consultez Migration de Spark GlueContext/Glue DynamicFrame vers Spark DataFrame
- Si vous avez besoin d'un contrôle d'accès complet aux tables (FTA), vous pouvez tirer parti du FTA dans DynamicFrames AWS Glue 5.0. Vous pouvez également migrer vers l'approche native de Spark pour bénéficier de fonctionnalités supplémentaires telles que les ensembles de données distribués résilients (RDDs), les bibliothèques personnalisées et les fonctions définies par l'utilisateur (UDFs) avec des AWS Lake Formation tables. Pour des exemples, voir Migration de AWS Glue 4.0 vers AWS Glue 5.0.
- Si vous n'avez pas besoin de FGAC, aucune migration vers le dataframe Spark n'est nécessaire et les fonctionnalités `GlueContext` telles que les signets de tâches et le déploiement des prédicats continueront de fonctionner.
- Les tâches avec FGAC nécessitent un minimum de quatre travailleurs : un pilote utilisateur, un pilote système, un exécuteur système et un exécuteur utilisateur de secours.

L'utilisation AWS de Glue with AWS Lake Formation entraîne des frais supplémentaires.

Comment fonctionne AWS Glue avec AWS Lake Formation

L'utilisation de AWS Glue with Lake Formation vous permet d'appliquer une couche d'autorisations à chaque tâche Spark afin d'appliquer le contrôle des autorisations de Lake Formation lorsque AWS Glue exécute des tâches. AWS Glue utilise les profils de ressources Spark pour créer deux profils afin d'exécuter efficacement les tâches. Le profil utilisateur exécute le code fourni par l'utilisateur, tandis que le profil système applique les politiques Lake Formation. Pour plus d'informations, consultez What is AWS Lake Formation et Considerations and limitations.

Voici un aperçu général de la manière dont AWS Glue accède aux données protégées par les politiques de sécurité de Lake Formation.



1. Un utilisateur appelle l'`StartJobRun`API sur une tâche AWS Glue compatible avec AWS Lake Formation.
2. AWS Glue envoie la tâche à un pilote utilisateur et l'exécute dans le profil utilisateur. Le pilote utilisateur exécute une version allégée de Spark qui n'est pas en mesure de lancer des tâches, de demander des exécuteurs, d'accéder à S3 ni au catalogue Glue. Il crée un plan de tâche.
3. AWS Glue configure un deuxième pilote appelé pilote système et l'exécute dans le profil système (avec une identité privilégiée). AWS Glue met en place un canal TLS crypté entre les deux pilotes pour la communication. Le pilote utilisateur utilise le canal pour envoyer les plans de tâche au pilote système. Le pilote système n'exécute pas le code soumis par l'utilisateur. Il

exécute Spark dans son intégralité et communique avec S3 et le catalogue de données pour l'accès aux données. Il demande des exécuteurs et compile le plan de tâche en une séquence d'étapes d'exécution.

4. AWS Glue exécute ensuite les étapes sur les exécuteurs avec le pilote utilisateur ou le pilote système. À n'importe quelle étape, le code utilisateur est exécuté exclusivement sur les exécuteurs de profil utilisateur.
5. Les étapes qui lisent les données des tables du catalogue de données protégées par des filtres de sécurité AWS Lake Formation ou qui appliquent des filtres de sécurité sont déléguées aux exécuteurs du système.

Exigences de minimum de travailleurs

Une tâche compatible avec Lake Formation dans AWS Glue nécessite un minimum de 4 travailleurs : un pilote utilisateur, un pilote système, un exécuteur système et un exécuteur utilisateur de secours. Cela représente une augmentation par rapport au minimum de 2 travailleurs requis pour les travaux standard de AWS Glue.

Une tâche compatible avec Lake Formation dans AWS Glue utilise deux pilotes Spark, l'un pour le profil système et l'autre pour le profil utilisateur. De même, les exécuteurs sont également divisés en deux profils :

- Exécuteurs système : gèrent les tâches auxquelles les filtres de données de Lake Formation sont appliqués.
- Exécuteurs utilisateur : sont demandés par le pilote système selon les besoins.

Comme les jobs de Spark sont par nature paresseux, AWS Glue réserve 10 % du nombre total de travailleurs (minimum 1), après déduction des deux facteurs, aux exécuteurs utilisateurs.

L'autoscaling est activé pour toutes les tâches compatibles avec Lake Formation, ce qui signifie que les exécuteurs utilisateur ne démarrent qu'en cas de besoin.

Pour un exemple de configuration, consultez Considerations and limitations.

Autorisations IAM du rôle d'exécution des tâches

Les autorisations de Lake Formation contrôlent l'accès aux ressources du catalogue de données AWS Glue, aux sites Amazon S3 et aux données sous-jacentes de ces sites. Les autorisations

IAM contrôlent l'accès à la Lake Formation and AWS Glue APIs et aux ressources. Bien que vous ayez l'autorisation Lake Formation d'accéder à une table du catalogue de données (SELECT), votre opération échoue si vous ne disposez pas de l'autorisation IAM sur l'opération d'API `glue:Get*`.

Voici un exemple de politique expliquant comment fournir les autorisations IAM pour accéder à un script dans Amazon S3, le chargement de journaux sur S3, les autorisations d'API AWS Glue et les autorisations d'accès à Lake Formation.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "ScriptAccess",
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:ListBucket"
      ],
      "Resource": [
        "arn:aws:s3:::*.amzn-s3-demo-bucket/scripts",
        "arn:aws:s3:::*.amzn-s3-demo-bucket/*"
      ]
    },
    {
      "Sid": "LoggingAccess",
      "Effect": "Allow",
      "Action": [
        "s3:PutObject"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/logs/*"
      ]
    },
    {
      "Sid": "GlueCatalogAccess",
      "Effect": "Allow",
      "Action": [
        "glue:Get*",
        "glue:Create*",
        "glue:Update*"
```

```
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Sid": "LakeFormationAccess",
      "Effect": "Allow",
      "Action": [
        "lakeformation:GetDataAccess"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
```

Configuration des autorisations de Lake Formation pour le rôle d'exécution des tâches

Tout d'abord, enregistrez l'emplacement de votre table Hive avec Lake Formation. Créez ensuite des autorisations pour votre rôle d'exécution des tâches dans la table de votre choix. Pour plus de détails sur Lake Formation, voir Qu'est-ce que c'est AWS Lake Formation ? dans le Guide AWS Lake Formation du développeur.

Après avoir configuré les autorisations de Lake Formation, vous pouvez soumettre des tâches Spark sur AWS Glue.

Soumission d'une exécution de tâche

Une fois que vous aurez fini de configurer les subventions Lake Formation, vous pourrez soumettre des jobs Spark sur AWS Glue. Pour exécuter des tâches Iceberg, vous devez fournir les configurations Spark suivantes. Pour configurer via les paramètres de tâche Glue, entrez le paramètre suivant :

- Clé :

```
--conf
```

- Valeur :

```
spark.sql.catalog.spark_catalog=org.apache.iceberg.spark.SparkSessionCatalog
       --conf spark.sql.catalog.spark_catalog.warehouse=<S3_DATA_LOCATION>
       --conf spark.sql.catalog.spark_catalog.glue.account-id=<ACCOUNT_ID>
       --conf spark.sql.catalog.spark_catalog.client.region=<REGION>
       --conf spark.sql.catalog.spark_catalog.glue.endpoint=https://
glue.<REGION>.amazonaws.com
```

Utilisation d'une session interactive

Une fois que vous avez fini de configurer les AWS Lake Formation subventions, vous pouvez utiliser Interactive Sessions on AWS Glue. Vous devez fournir les configurations Spark suivantes via la commande magique `%%configure` avant d'exécuter le code.

```
%%configure
{
    "--enable-lakeformation-fine-grained-access": "true",
    "--conf":
 "spark.sql.catalog.spark_catalog=org.apache.iceberg.spark.SparkSessionCatalog
 --conf spark.sql.catalog.spark_catalog.warehouse=<S3_DATA_LOCATION> --conf
 spark.sql.catalog.spark_catalog.catalog-impl=org.apache.iceberg.aws.glue.GlueCatalog
 --conf spark.sql.catalog.spark_catalog.io-impl=org.apache.iceberg.aws.s3.S3FileIO --
conf
 spark.sql.extensions=org.apache.iceberg.spark.extensions.IcebergSparkSessionExtensions
 --conf spark.sql.catalog.spark_catalog.client.region=<REGION> --conf
 spark.sql.catalog.spark_catalog.glue.account-id=<ACCOUNT_ID> --conf
 spark.sql.catalog.spark_catalog.glue.endpoint=https://glue.<REGION>.amazonaws.com"
}
```

FGAC for AWS Glue 5.0 Notebook ou sessions interactives

Pour activer le contrôle d'accès fin (FGAC) dans AWS Glue, vous devez spécifier les configurations Spark requises pour Lake Formation dans le cadre de la magie de %%configure avant de créer la première cellule.

Une spécification ultérieure à l'aide des appels `SparkSession.builder().conf("").get()` ou ne `SparkSession.builder().conf("").create()` suffira pas. Il s'agit d'une modification par rapport au comportement de AWS Glue 4.0.

Prise en charge du format de tableau ouvert

AWS La version 5.0 ou ultérieure de Glue inclut la prise en charge d'un contrôle d'accès précis basé sur Lake Formation. AWS Glue est compatible avec les tables Hive et Iceberg. Le tableau suivant décrit toutes les options prises en charge.

Opérations	Hive	Iceberg
Commandes DDL	Avec des autorisations de rôle IAM uniquement	Avec des autorisations de rôle IAM uniquement
Requêtes progressives	Non applicable	Entièrement pris en charge
Requêtes Time Travel	Non applicable à ce format de table	Entièrement pris en charge
Tables de métadonnées	Non applicable à ce format de table	Pris en charge, mais certaines tables sont masquées Pour plus d'informations, consultez Considerations and limitations.
`DML INSERT`	Avec des autorisations IAM uniquement	Avec des autorisations IAM uniquement
DML UPDATE	Non applicable à ce format de table	Avec des autorisations IAM uniquement
`DML DELETE`	Non applicable à ce format de table	Avec des autorisations IAM uniquement
Opérations de lecture	Entièrement pris en charge	Entièrement pris en charge
Procédures stockées	Non applicable	Pris en charge à l'exception de `register_table` et de `migrate` Pour plus d'informations, consultez Considerations and limitations.

Migration de Spark GlueContext/Glue DynamicFrame vers Spark DataFrame

Vous trouverez ci-dessous des exemples Python et Scala de migration `GlueContext`/Glue `DynamicFrame` dans Glue 4.0 vers Spark `DataFrame` dans Glue 5.0.

Python

Avant :

```
escaped_table_name= ``<dbname>`.`<table_name>``

additional_options = {
  "query": f'select * from {escaped_table_name} WHERE column1 = 1 AND column7 = 7'
}

# DynamicFrame example
dataset = glueContext.create_data_frame_from_catalog(
    database="<dbname>",
    table_name=escaped_table_name,
    additional_options=additional_options)
```

Après :

```
table_identifier= ``<catalogname>`.`<dbname>`.`<table_name>`"' #catalogname is optional

# DataFrame example
dataset = spark.sql(f'select * from {table_identifier} WHERE column1 = 1 AND column7 =
 7')
```

Scala

Avant :

```
val escapedTableName = "`<dbname>`.`<table_name>`"

val additionalOptions = JsonOptions(Map(
    "query" -> s"select * from $escapedTableName WHERE column1 = 1 AND column7 = 7"
    )
)

# DynamicFrame example
val datasource0 = glueContext.getCatalogSource(
```

```
    database="<dbname>",
    tableName=escapedTableName,
    additionalOptions=additionalOptions).getDataFrame()
```

Après :

```
val tableIdentifier = "`<catalogname>`.`<dbname>`.`<table_name>`" //catalogname is
 optional

# DataFrame example
val datasource0 = spark.sql(s"select * from $tableIdentifier WHERE column1 = 1 AND
 column7 = 7")
```

Considérations et restrictions

Tenez compte des considérations et des restrictions suivantes lorsque vous utilisez Lake Formation avec AWS Glue.

AWS Glue with Lake Formation est disponible dans toutes les régions prises en charge, à l'exception de AWS GovCloud (USA Est) et AWS GovCloud (USA Ouest).

- AWS Glue permet un contrôle d'accès précis via Lake Formation uniquement pour les tables Apache Hive et Apache Iceberg. Les formats Apache Hive incluent Parquet, ORC et CSV.
- Vous ne pouvez utiliser Lake Formation qu'avec des tâches Spark.
- AWS Glue with Lake Formation ne prend en charge qu'une seule session Spark tout au long d'une tâche.
- Lorsque Lake Formation est activé,AWS Glue nécessite un plus grand nombre de travailleurs car il nécessite un pilote système, des exécuteurs système, un pilote utilisateur et éventuellement des exécuteurs utilisateur (requis lorsque votre tâche possède UDFs ou`spark.createDataFrame`).
- AWS Glue with Lake Formation ne prend en charge que les requêtes de table entre comptes partagées via des liens de ressources. Le lien de ressource doit être nommé de la même manière que la ressource du compte source.
- Pour activer un contrôle d'accès précis pour les tâches AWS Glue, transmettez le paramètre de la `--enable-lakeformation-fine-grained-access` tâche.
- Vous pouvez configurer vos jobs AWS Glue pour qu'ils fonctionnent avec la hiérarchie multi-catalogues AWS Glue. Pour plus d'informations sur les paramètres de configuration à utiliser avec l'`StartJobRun`API AWS Glue, voir Working with AWS Glue multi-catalog Hire on EMR Serverless.
- Les éléments suivants ne sont pas pris en charge :

  - Jeux de données distribués résilients (RDD)
  - Spark Streaming
  - Écriture avec les autorisations accordées par Lake Formation
  - Contrôle d'accès pour les colonnes imbriquées
- AWS Glue bloque les fonctionnalités susceptibles de compromettre l'isolation complète du pilote système, notamment les suivantes :
  - UDTs, Hive UDFs et toute fonction définie par l'utilisateur impliquant des classes personnalisées
  - Sources de données personnalisées
  - Fourniture de fichiers JAR supplémentaires pour l'extension, le connecteur ou le métastore Spark
  - Commande `ANALYZE TABLE`
- Pour appliquer les contrôles d'accès, les opérations `EXPLAIN PLAN` et DDL telles que `DESCRIBE TABLE` n'exposent pas les informations restreintes.
- AWS Glue restreint l'accès aux journaux Spark du pilote système sur les applications compatibles avec Lake Formation. Étant donné que le pilote système s'exécute avec plus d'accès, les événements et les journaux générés par le pilote système peuvent inclure des informations sensibles. Pour empêcher les utilisateurs ou le code non autorisés d'accéder à ces données sensibles, AWS Glue a désactivé l'accès aux journaux des pilotes du système. Pour le dépannage, contactez AWS le support.
- Si vous avez enregistré l'emplacement d'une table avec Lake Formation, le chemin d'accès aux données passe par les informations d'identification stockées dans Lake Formation, indépendamment de l'autorisation IAM pour le rôle d'exécution de la tâche AWS Glue. Si vous configurez mal le rôle enregistré avec l'emplacement de la table, les tâches soumises qui utilisent le rôle avec l'autorisation S3 IAM sur l'emplacement de la table échoueront.
- L'écriture dans une table Lake Formation utilise l'autorisation IAM plutôt que les autorisations accordées par Lake Formation. Si votre rôle d'exécution de tâches dispose des autorisations S3 nécessaires, vous pouvez l'utiliser pour exécuter des opérations d'écriture.

Voici les restrictions et les considérations à prendre en compte lors de l'utilisation d'Apache Iceberg :

- Vous ne pouvez utiliser Apache Iceberg qu'avec un catalogue de sessions et non avec des catalogues nommés arbitrairement.
- Les tables Iceberg enregistrées dans Lake Formation ne prennent en charge que les tables de métadonnées `history`metadata_log_entries,`snapshots`,, `files`manifests, et`refs`.

AWS Glue masque les colonnes susceptibles de contenir des données sensibles, telles que `partitionspath`, et`summaries`. Cette restriction ne s'applique pas aux tables Iceberg qui ne sont pas enregistrées dans Lake Formation.

- Les tables que vous n'enregistrez pas dans Lake Formation prennent en charge toutes les procédures stockées par Iceberg. Les procédures `register_table` et `migrate` ne sont prises en charge pour aucune table.
- Nous vous recommandons d'utiliser Iceberg DataFrameWriter V2 au lieu de V1.

Exemple d'allocation de travailleurs

Pour une tâche configurée avec les paramètres suivants :

```
--enable-lakeformation-fine-grained-access=true
--number-of-workers=20
```

L'allocation de travailleurs serait la suivante :

- Un travailleur pour le pilote utilisateur.
- Un travailleur pour le pilote système.
- 10 % des 18 travailleurs restants (soit 2 travailleurs) réservés aux exécuteurs utilisateur.
- Jusqu'à 16 travailleurs alloués aux exécuteurs système.

Lorsque autoscaling est activé, les exécuteurs utilisateur peuvent utiliser n'importe quelle capacité non allouée depuis les exécuteurs système, si nécessaire.

Contrôle de l'allocation d'exécuteurs utilisateurs

Vous pouvez ajuster le pourcentage de réservation pour les exécuteurs utilisateur à l'aide de la configuration suivante :

```
--conf spark.dynamicAllocation.maxExecutorsRatio=<value between 0 and 1>
```

Cette configuration permet de contrôler avec précision le nombre d'exécuteurs utilisateur réservés par rapport à la capacité totale disponible.

Résolution des problèmes

Consultez les sections suivantes pour obtenir des solutions de dépannage.

Logging

AWS Glue utilise les profils de ressources Spark pour diviser l'exécution des tâches. AWS Glue utilise le profil utilisateur pour exécuter le code que vous avez fourni, tandis que le profil système applique les politiques de Lake Formation. Vous pouvez accéder aux journaux des tâches exécutées en tant que profil utilisateur.

Interface utilisateur Live et serveur d'historique Spark

L'interface utilisateur Live et le serveur d'historique Spark contiennent tous les événements Spark générés à partir du profil utilisateur et les événements expurgés générés à partir du pilote système.

Vous pouvez voir toutes les tâches des pilotes utilisateur et système dans l'onglet Exécuteurs. Toutefois, les liens du journal ne sont disponibles que pour le profil utilisateur. Certaines informations sont également expurgées de l''interface utilisateur Live, comme le nombre d'enregistrements de sortie.

Échec de la tâche en raison d'autorisations insuffisantes pour Lake Formation

Assurez-vous que votre rôle d'exécution des tâches dispose des autorisations nécessaires pour exécuter SELECT et DESCRIBE sur la table à laquelle vous accédez.

Échec de l'exécution de la tâche avec RDD

AWS Glue ne prend actuellement pas en charge les opérations de jeu de données distribué résilient (RDD) sur les tâches compatibles avec Lake Formation.

Impossible d'accéder aux fichiers de données dans Amazon S3

Assurez-vous d'avoir enregistré l'emplacement du lac de données dans Lake Formation.

Exception de validation de sécurité

AWS Glue a détecté une erreur de validation de sécurité. Contactez le AWS support pour obtenir de l'aide.

Partage du catalogue de données et des tableaux AWS Glue entre les comptes

Vous pouvez partager des bases de données et des tables entre les comptes tout en continuant à utiliser Lake Formation. Pour plus d'informations, voir Partage de données entre comptes dans Lake Formation et How do I share AWS Glue Data Catalog and tables cross-account using ? .

Le tableau suivant présente un résumé de la manière de choisir entre le contrôle précis des accès (FGAC) et l'accès complet aux tables (FTA) pour votre charge de travail.

Fonctionnalité	Contrôle précis des accès (FGAC)	Accès complet aux tables (FTA)
Niveau d'accès	Niveau de colonne/ligne	Tableau complet
Cas d'utilisation	Requêtes et ETL avec autorisations limitées	ETL
Impact sur les performances	Nécessite des transitions system/user spatiales pour l'évaluation du contrôle d'accès, ce qui augmente la latence	Performances optimisées

Utilisation d'Amazon S3 Access Grants avec AWS Glue

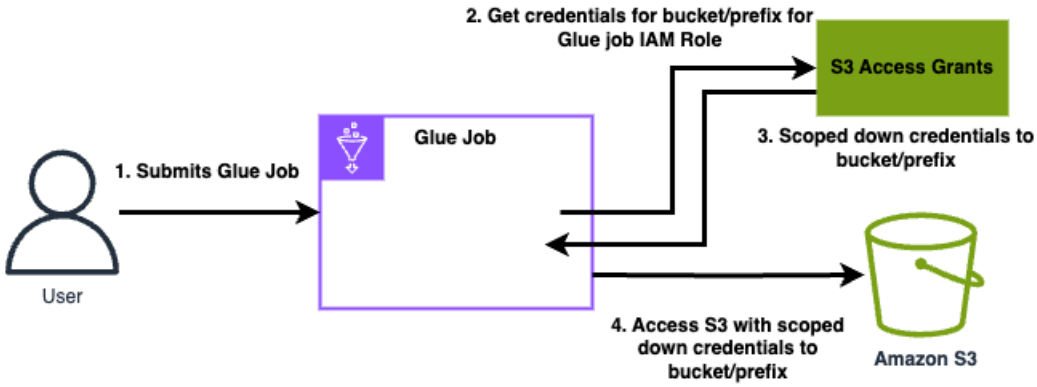
Avec la version 5.0 de Glue, Amazon S3 Access Grants fournit une solution de contrôle d'accès évolutive que vous pouvez utiliser pour augmenter l'accès à vos données Amazon S3 depuis AWS Glue. Si vos exigences en matière de données S3 nécessitent une configuration d'autorisations complexe ou importante, vous pouvez utiliser les autorisations d'accès S3 pour mettre à l'échelle les autorisations de données S3 pour les utilisateurs et les rôles.

Utilisez les subventions d'accès S3 pour augmenter l'accès aux données Amazon S3 au-delà des autorisations accordées par le rôle d'exécution ou des rôles IAM associés aux identités ayant accès à votre AWS Glue tâche. Pour plus d'informations, voir la rubrique Gestion des accès avec S3 Access Grants du Guide de l'utilisateur Amazon S3.

Comment AWS Glue fonctionne le S3 Access Grants

AWS Glue les versions 5.0 et supérieures fournissent une intégration native avec S3 Access Grants. Vous pouvez activer les subventions d'accès S3 sur AWS Glue et exécuter des tâches Spark. Lorsqu'une tâche Spark demande à accéder aux données S3, Amazon S3 fournit des informations d'identification temporaires limitées au compartiment, au préfixe ou à l'objet concerné.

Vous trouverez ci-dessous un aperçu général de la manière AWS Glue d'accéder aux données auxquelles S3 Access Grants gère l'accès.



1. Un utilisateur soumet une tâche AWS Glue Spark qui utilise des données stockées dans Amazon S3.
2. AWS Glue fait une demande de subventions d'accès S3 pour vendre des informations d'identification temporaires à l'utilisateur qui donnent accès au compartiment, au préfixe ou à l'objet.
3. AWS Glue renvoie des informations d'identification temporaires sous la forme d'un jeton STS (AWS Security Token Service) pour l'utilisateur. Le jeton permet uniquement d'accéder au compartiment, au préfixe ou à l'objet S3 concerné.
4. AWS Glue utilise le jeton STS pour récupérer les données de S3.
5. AWS Glue reçoit les données de S3 et renvoie les résultats à l'utilisateur.

Considérations relatives aux subventions d'accès S3 avec AWS Glue

Prenez note des comportements et des limites suivants lorsque vous utilisez S3 Access Grants avec AWS Glue.

Support des fonctionnalités

- S3 Access Grants est pris en charge avec AWS Glue les versions 5.0 et supérieures.
- Spark est le seul type de travail pris en charge lorsque vous utilisez S3 Access Grants avec AWS Glue.
- Delta Lake et Hudi sont les seuls formats de table ouverte pris en charge lorsque vous utilisez S3 Access Grants avec. AWS Glue

- Les fonctionnalités suivantes ne sont pas prises en charge pour une utilisation avec les autorisations d'accès S3 :
  - Tables Apache Iceberg
  - AWS Demandes CLI adressées à Amazon S3 qui utilisent des rôles IAM
  - Accès à S3 via le protocole open source S3A

Considérations comportementales

- AWS Glue fournit un cache d'informations d'identification pour garantir qu'un utilisateur n'a pas besoin de demander à plusieurs reprises les mêmes informations d'identification dans le cadre d'une tâche Spark. Par conséquent, demande AWS Glue toujours le privilège de niveau par défaut lorsqu'il demande des informations d'identification. Pour plus d'informations, voir la rubrique Demande d'accès aux données S3 du Guide de l'utilisateur Amazon S3.

Configurez les autorisations d'accès S3 avec AWS Glue

Conditions préalables

L'appelant ou l'administrateur a créé une instance d'autorisations d'accès S3.

Configuration des AWS Glue politiques et de la configuration des tâches

Pour configurer S3 Access Grants avec, AWS Glue vous devez configurer des politiques de confiance et IAM, et transmettre la configuration via les paramètres de tâche.

1. Configurez les politiques de confiance minimale et IAM suivantes sur le rôle utilisé pour les subventions (le AWS Glue rôle qui exécute les sessions ou les tâches).

   Stratégie d'approbation :

```
{
            "Sid": "Stmt1234567891011",
            "Action": [
                "sts:AssumeRole",
                "sts:SetSourceIdentity",
                "sts:SetContext"
            ],
            "Effect": "Allow",
            "Principal": {
```

```
                "Service": "access-grants.s3.amazonaws.com"
            },
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "123456789012",
                    "aws:SourceArn": "arn:aws:s3:<region>:123456789012:access-grants/
default"
                }
            }
        }
```

Politique IAM :

```
{
            "Sid": "S3Grants",
            "Effect": "Allow",
            "Action": [
                "s3:GetDataAccess",
                "s3:GetAccessGrantsInstanceForPrefix"
            ],
            "Resource": "arn:aws:s3:<region>:123456789012:access-grants/default"
        },
        {
            "Sid": "BucketLevelReadPermissions",
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:ResourceAccount": "123456789012"
                },
                "ArnEquals": {
                    "s3:AccessGrantsInstanceArn": [
                        "arn:aws:s3:<region>:123456789012:access-grants/default"
                    ]
                }
            }
        },
        {
```

```
            "Sid": "ObjectLevelReadPermissions",
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion",
                "s3:GetObjectAcl",
                "s3:GetObjectVersionAcl",
                "s3:ListMultipartUploadParts"
            ],
            "Resource": [
                "arn:aws:s3:::*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:ResourceAccount": "123456789012"
                },
                "ArnEquals": {
                    "s3:AccessGrantsInstanceArn": [
                        "arn:aws:s3:<region>:123456789012:access-grants/default"
                    ]
                }
            }
        }
```

2. Dans votre AWS Glue tâche, transmettez la configuration Spark suivante via les paramètres de la AWS Glue tâche ou`SparkConf`.

```
--conf spark.hadoop.fs.s3.s3AccessGrants.enabled=true \
--conf spark.hadoop.fs.s3.s3AccessGrants.fallbackToIAM=false
```

Propagation d'identité fiable avec AWS Glue ETL

Avec IAM Identity Center, vous pouvez vous connecter à des fournisseurs d'identité (IdPs) et gérer de manière centralisée l'accès des utilisateurs et des groupes à travers les services AWS d'analyse. Vous pouvez intégrer des fournisseurs d'identité tels qu'Okta, Ping et Microsoft Entra ID (anciennement Azure Active Directory) à IAM Identity Center pour que les utilisateurs de votre organisation puissent accéder aux données via une expérience d'authentification unique. IAM Identity Center prend également en charge la connexion à d'autres fournisseurs d'identité tiers.

Avec la AWS Glue version 5.0 et les versions ultérieures, vous pouvez propager les identités des utilisateurs depuis IAM Identity Center vers AWS Glue des sessions interactives. AWS Glue Les

sessions interactives propageront davantage l'identité fournie aux services en aval tels qu'Amazon S3 Access Grants et Amazon Redshift AWS Lake Formation, permettant ainsi un accès sécurisé aux données via l'identité des utilisateurs dans ces services en aval.

Présentation de

Identity Center est l'approche recommandée pour l'authentification et l'autorisation du personnel AWS pour les organisations de toutes tailles et de tous types. Avec Identity Center, vous pouvez créer et gérer les identités des utilisateurs dans AWS ou connecter votre source d'identité existante, notamment Microsoft Active Directory, Okta, Ping Identity JumpCloud, Google Workspace et Microsoft Entra ID (anciennement Azure AD).

La propagation fiable des identités est une fonctionnalité du centre d'identité IAM que les administrateurs de AWS services connectés peuvent utiliser pour accorder et auditer l'accès aux données de service. L'accès à ces données est basé sur les attributs utilisateur tels que les associations de groupe. La mise en place d'une propagation d'identité sécurisée nécessite une collaboration entre les administrateurs des AWS services connectés et les administrateurs de l'IAM Identity Center.

Fonctionnalités et avantages

L'intégration des sessions AWS Glue interactives à IAM Identity Center Trusted Identity Propagation offre les avantages suivants :

- Vous avez la possibilité d'appliquer des autorisations au niveau des tables et un contrôle d'accès précis avec les identités Identity Center sur les tables du catalogue de données AWS Glue gérées par Lake Formation.
- Vous avez la possibilité d'appliquer des autorisations avec les identités Identity Center sur les clusters Amazon Redshift.
- Permet le suivi de bout en bout des actions des utilisateurs à des fins d'audit.
- Vous avez la possibilité d'appliquer l'autorisation au niveau du préfixe Amazon S3 avec les identités Identity Center aux préfixes Amazon S3 gérés par les autorisations d'accès Amazon S3.

Cas d'utilisation

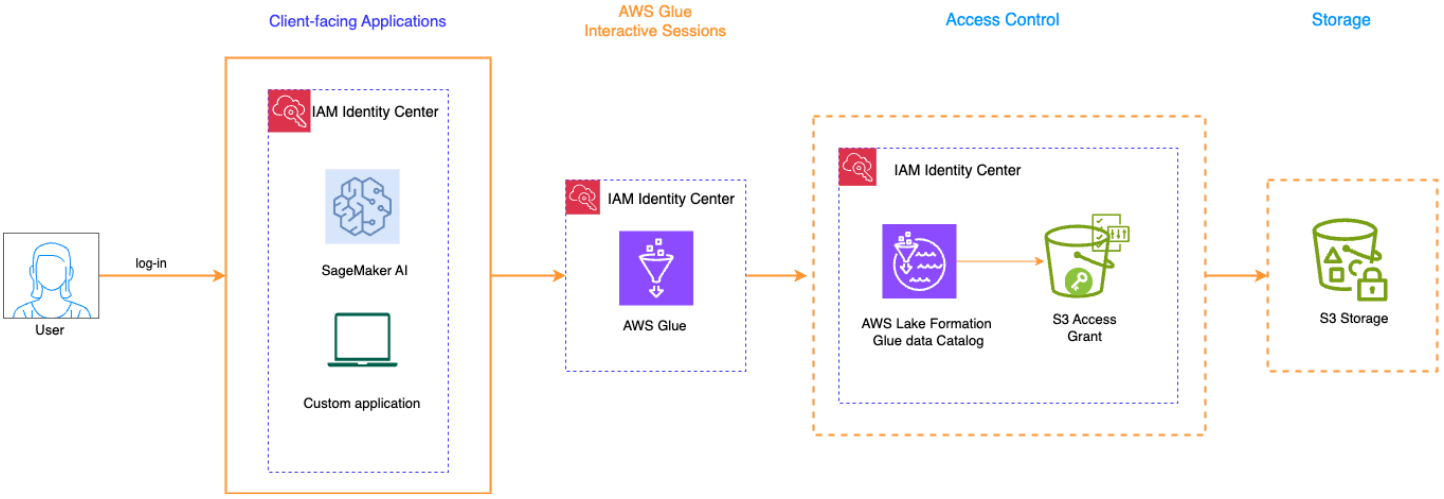
Exploration et analyse interactives des données

Les ingénieurs de données utilisent leur identité d'entreprise pour accéder et analyser facilement les données de plusieurs AWS comptes. Grâce à SageMaker Studio, ils lancent des sessions Spark interactives via AWS Glue ETL, en se connectant à diverses sources de données, notamment Amazon S3 et le catalogue de AWS Glue données. Lorsque les ingénieurs explorent les jeux de données, Spark applique des contrôle précis des accès définis dans Lake Formation en fonction de leur identité, garantissant ainsi qu'ils ne peuvent consulter que les données autorisées. Toutes les requêtes et transformations de données sont enregistrées avec l'identité de l'utilisateur, ce qui crée une piste d'audit claire. Cette approche rationalisée permet le prototypage rapide de nouveaux produits d'analytique tout en maintenant une gouvernance des données stricte dans les environnements client.

Préparation des données et ingénierie des fonctionnalités

Les scientifiques des données de plusieurs équipes de recherche collaborent sur des projets complexes en s'aidant d'une plateforme de données unifiée. Ils se connectent à SageMaker Studio avec leurs identifiants professionnels et accèdent immédiatement à un vaste lac de données partagé qui couvre plusieurs AWS comptes. Alors qu'ils commencent à concevoir des fonctionnalités pour de nouveaux modèles d'apprentissage automatique, les sessions Spark lancées via AWS Glue ETL appliquent les politiques de sécurité au niveau des colonnes et des lignes de Lake Formation en fonction de leurs identités propagées. Les scientifiques peuvent préparer efficacement les données et concevoir des fonctionnalités à l'aide d'outils bien connus, tandis que les équipes de conformité ont l'assurance que chaque interaction avec les données est automatiquement suivie et auditée. Cet environnement sécurisé et collaboratif accélère les pipelines de recherche tout en respectant les normes strictes de protection des données requises dans les secteurs réglementés.

Comment ça marche



Un utilisateur se connecte à des applications orientées client (SageMaker IA ou applications personnalisées) en utilisant son identité d'entreprise via IAM Identity Center. Cette identité est ensuite propagée via l'ensemble du pipeline d'accès aux données.

L'utilisateur authentifié lance des sessions AWS AWS Glue interactives, qui servent de moteur de calcul pour le traitement des données. Ces sessions conservent le contexte d'identité de l'utilisateur tout au long du flux de travail.

AWS Lake Formation et le catalogue de AWS Glue données fonctionnent ensemble pour appliquer des contrôles d'accès précis. Lake Formation applique des politiques de sécurité basées sur l'identité propagée de l'utilisateur, tandis que les autorisations d'accès Amazon S3 fournissent des couches d'autorisation supplémentaires, garantissant que les utilisateurs ne peuvent accéder qu'aux données qu'ils sont autorisés à consulter.

Enfin, le système se connecte au stockage Amazon S3 où résident les données réelles. Tous les accès sont régis par les politiques de sécurité combinées, ce qui permet d'assurer la gouvernance des données tout en permettant une exploration et une analyse interactives des données. Cette architecture permet un accès aux données sécurisé et basé sur l'identité à travers plusieurs AWS services tout en garantissant une expérience utilisateur fluide aux data scientists et aux ingénieurs travaillant avec de grands ensembles de données.

Intégrations

AWS environnement de développement géré

Les applications AWS gérées destinées aux clients suivantes prennent en charge la propagation d'identités fiables grâce à des sessions AWS Glue interactives :

- Sagemaker Unified Studio
- Amazon SageMaker AI

Sagemaker Unified Studio

Pour utiliser la propagation d'identité fiable avec Sagemaker Unified Studio :

1. Configurez le projet Sagemaker Unified Studio avec la propagation d'identité fiable activée en tant qu'environnement de développement orienté client.
2. Configurez Lake Formation pour permettre un contrôle d'accès précis aux AWS Glue tables en fonction de l'utilisateur ou du groupe dans IAM Identity Center.

3. Configurez les autorisations d'accès Amazon S3 pour permettre un accès temporaire aux emplacements de données sous-jacents dans Amazon S3.
4. Ouvrez l'espace JupyterLab IDE Sagemaker Unified Studio et sélectionnez AWS Glue comme calcul pour l'exécution du bloc-notes.

Environnement de bloc-notes auto-hébergé géré par le client

Pour permettre la propagation d'identités fiables pour les utilisateurs d'applications développées sur mesure, consultez la section Accès aux AWS services par programmation à l'aide de la propagation d'identité fiable dans le blog sur la AWS sécurité.

Commencer à utiliser la propagation fiable des identités dans l' AWS Glue ETL

Cette section vous aide à configurer AWS Glue l'application avec des sessions interactives afin de l'intégrer à IAM Identity Center et de permettre la propagation d'identités fiables.

Conditions préalables

- Une instance d'Identity Center située dans la AWS région où vous souhaitez créer des sessions AWS Glue interactives activées par la propagation d'identités fiables. Une instance d'Identity Center ne peut exister que dans une seule région pour un AWS compte. Pour plus d'informations, consultez Enable IAM Identity Center et Provision the users and groups from your source of identities into IAM Identity Center.
- Activez la propagation d'identité fiable pour les services en aval tels que Lake Formation ou les autorisations d'accès Amazon S3 ou le cluster Amazon Redshift avec lequel une charge de travail interactive interagit pour accéder aux données.

Autorisations nécessaires pour connecter l' AWS Glue ETL à IAM Identity Center

Créer un rôle IAM

Le rôle qui crée la connexion à IAM Identity Center nécessite des autorisations pour créer et modifier la configuration de l'application dans AWS Glue et IAM Identity Center, conformément à la politique en ligne suivante.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:CreateGlueIdentityCenterConfiguration",
                "sso:CreateApplication",
                "sso:PutApplicationAssignmentConfiguration",
                "sso:PutApplicationAuthenticationMethod",
                "sso:PutApplicationGrant",
                "sso:PutApplicationAccessScope",
                "sso:ListInstances"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

Les politiques en ligne suivantes contiennent des autorisations spécifiques requises pour afficher, mettre à jour et supprimer les propriétés d'intégration d' AWS Glue avec IAM Identity Center.

Utilisez la politique intégrée suivante pour autoriser un rôle IAM à visualiser une AWS Glue intégration avec IAM Identity Center.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:GetGlueIdentityCenterConfiguration"
            ],
            "Resource": [
```

```
                "*"
            ]
        }
    ]
}
```

Utilisez la politique intégrée suivante pour autoriser un rôle IAM à mettre à jour AWS Glue l'intégration avec IAM Identity Center.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:UpdateGlueIdentityCenterConfiguration",
                "sso:PutApplicationAccessScope",
                "sso:DeleteApplicationAccessScope"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

Utilisez la politique intégrée suivante pour autoriser un rôle IAM à supprimer une AWS Glue intégration avec IAM Identity Center.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
```

```
                "glue:DeleteGlueIdentityCenterConfiguration",
                "sso:DeleteApplication"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

Description des autorisations

- `glue:CreateGlueIdentityCenterConfiguration`— Accorde l'autorisation de créer la configuration AWS Glue iDC.
- `glue:GetGlueIdentityCenterConfiguration` : octroie l'autorisation d'obtenir une configurations IdC existante.
- `glue:DeleteGlueIdentityCenterConfiguration`— Accorde l'autorisation de supprimer une configuration AWS Glue iDC existante.
- `glue:UpdateGlueIdentityCenterConfiguration`— Autorise la mise à jour d'une configuration AWS Glue iDC existante.
- `sso:CreateApplication`— Accorde l'autorisation de créer une application IAM Identity Center AWS Glue gérée.
- `sso:DescribeApplication`- Accorde l'autorisation de décrire une application IAM Identity Center AWS Glue gérée.
- `sso:DeleteApplication`— Accorde l'autorisation de supprimer une application IAM Identity Center AWS Glue gérée.
- `sso:UpdateApplication`— Autorise la mise à jour d'une application IAM Identity Center AWS Glue gérée.
- `sso:PutApplicationGrant`— Accorde l'autorisation d'appliquer l'échange de jetons, IntrospectToken, RefreshToken et des subventions sur l'application iDC. RevokeToken
- `sso:PutApplicationAuthenticationMethod`— Accorde l'autorisation de mettre AuthenticationMethod sur une application iDC AWS Glue gérée qui permet au principal de AWS Glue service d'interagir avec l'application iDC.
- `sso:PutApplicationAccessScope`— Autorise l'ajout ou la mise à jour de la liste des étendues de service en aval autorisées sur l'application iDC AWS Glue gérée.

- `sso:DeleteApplicationAccessScope`- Accorde l'autorisation de supprimer des étendues en aval si une étendue est supprimée pour l'application iDC AWS Glue gérée.
- `sso:PutApplicationAssignmentConfiguration`— Accorde l'autorisation de définir le paramètre « U ser-assignment-not-required » sur l'application iDC.
- `sso:ListInstances`— Accorde l'autorisation de répertorier les instances et de valider l'IDC InstanceArn que vous spécifiez en identity-center-configuration paramètre.

Connexion AWS Glue à IAM Identity Center

Lorsqu'il AWS Glue est connecté à IAM Identity Center, il crée une application iDC gérée par un singleton par compte. L'exemple suivant montre comment vous connecter AWS Glue à IAM Identity Center :

```
aws glue create-glue-identity-center-configuration \
--instance-arn arn:aws:sso:::instance/ssoins-123456789 \
--scopes '["s3:access_grants:read_write", "redshift:connect","lakeformation:query"]'
```

Pour mettre à jour les portées de l'application gérée (généralement pour se propager à un plus grand nombre de services en aval), vous pouvez utiliser :

```
aws glue update-glue-identity-center-configuration \
--scopes '["s3:access_grants:read_write", "redshift:connect","lakeformation:query"]'
```

Le paramètre Portées est facultatif et toutes les portées seront ajoutées s'il n'est pas fourni. Les valeurs prises en charge sont `s3:access_grants:read_write`, `redshift:connect` et `lakeformation:query`.

Pour obtenir les détails de la configuration, vous pouvez utiliser :

```
aws glue get-glue-identity-center-configuration
```

Vous pouvez supprimer la connexion entre AWS Glue IAM Identity Center à l'aide de la commande suivante :

```
aws glue delete-glue-identity-center-configuration
```

> **Note**
>
> AWS Glue crée une application Identity Center gérée par un service dans votre compte que le service utilise pour les validations d'identité et la propagation des identités vers les services en aval. AWS Glue L'application Identity Center gérée créée est partagée entre toutes les trusted-identity-propagation sessions de votre compte.
> Avertissement : ne modifiez pas manuellement les paramètres de l'application Identity Center gérée. Toute modification peut affecter toutes les sessions AWS Glue interactives trusted-identity-propagation activées dans votre compte.

Création d'une session AWS Glue interactive avec activation de la propagation d'identité sécurisée

Une fois connecté AWS Glue à IAM Identity Center, vous pouvez utiliser les informations d'identification de rôle améliorées pour créer une AWS Glue session interactive. Il n'est pas nécessaire de transmettre des paramètres supplémentaires lors de la création d'une AWS Glue session 5.0. Comme il AWS Glue est connecté au centre d'identité IAM, s'il est AWS Glue détecté identity-enhanced-role-credentials, il propagera automatiquement les informations d'identité aux services en aval appelés dans le cadre de vos déclarations. Cependant, le rôle d'exécution de la session doit disposer de l'autorisation `sts:SetContext`, comme décrit ci-dessous.

Autorisations du rôle d'exécution pour propager l'identité

Comme les AWS Glue sessions utilisent les informations d'identification améliorées par Identity pour propager l'identité aux AWS services en aval, la politique de confiance associée à son rôle d'exécution doit disposer d'une autorisation supplémentaire pour permettre la propagation de l'identité `sts:SetContext` vers les services en aval (Amazon S3 access-grant, Lake Formation, Amazon Redshift). Pour en savoir plus sur la création d'un rôle d'exécution, consultez Setting up a runtime role.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
```

```
      "Principal": {
        "Service": "glue.amazonaws.com"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:SetContext"
      ]
    }
  ]
}
```

De plus, le rôle Runtime aurait besoin d'autorisations pour les AWS services en aval que job-run invoquerait pour récupérer des données en utilisant l'identité de l'utilisateur. Veuillez consulter les liens suivants pour configurer les autorisations d'accès Amazon S3 et Lake Formation :

- Utilisation de Lake Formation avec AWS Glue
- Utilisation d'Amazon S3 Access Grants avec AWS Glue

Considérations et limites relatives à l'intégration d' AWS Glue ETL Trusted Identity Propagation

> **Important**
>
> Par défaut, les sessions ne sont pas privées, ce qui signifie qu'un utilisateur IdC peut accéder à la session d'un autre utilisateur IdC. Vous pouvez l'utiliser tagOnCreatepour rendre vos sessions privées. Par exemple, la session peut être étiquetée avec une balise de propriétaire et sa valeur en tant qu'ID utilisateur IDC, puis dans la politique, vous pouvez utiliser une clé de condition globale, telle que pour valider par rapport `identitystore:UserId`à la balise propriétaire dans la politique de principal/runtime rôle du client pour toutes les opérations d'API de session afin de garantir qu'un utilisateur IdC ne puisse pas accéder à la session d'un autre utilisateur IdC.

Tenez compte des points suivants lorsque vous utilisez IAM Identity Center Trusted Identity Propagation avec une AWS Glue application :

- La propagation d'identités fiables via Identity Center est prise en charge sur les AWS Glue versions 5.0 et supérieures, et uniquement avec les sessions AWS Glue interactives.
- AWS Glue le catalogue de données est couvert par l'intégration du centre d'identité de Lake Formation.
- Trusted Identity Propagation est limitée aux sessions interactives dans AWS Glue, à l'exclusion d'autres entités de traitement de données telles que les tâches, les déclencheurs, les flux de travail et les tâches ML. Cependant AWS Glue APIs, tous enregistrent l'identité des utilisateurs à des AWS CloudTrail fins d'audit.
- AWS Glue prend actuellement en charge l'intégration avec IAM Identity Center exclusivement via les interfaces API et CLI, et non via la console.
- Une fois qu'une application est activée sur le AWS Glue site, assurez-vous de créer des sessions 5.0 avec des informations d'identification iDC, mais ne créez pas de session 4.0 avec des informations d'identification iDC.
- La propagation d'identités fiables avec AWS Glue est prise en charge dans les AWS régions suivantes :
  - af-south-1 – Afrique (Le Cap)
  - ap-east-1 – Asie-Pacifique (Hong Kong)
  - ap-northeast-1 – Asie-Pacifique (Tokyo)
  - ap-northeast-2, Asie-Pacifique (Séoul)
  - ap-northeast-3 – Asie-Pacifique (Osaka)
  - ap-south-1, Asie-Pacifique (Mumbai)
  - ap-southeast-1 – Asie-Pacifique (Singapour)
  - ap-southeast-2 – Asie-Pacifique (Sydney)
  - ap-southeast-3 – Asie-Pacifique (Jakarta)
  - ca-central-1, Canada (Centre)
  - eu-central-1 – Europe (Francfort)
  - eu-north-1, Europe (Stockholm)
  - eu-south-1 – Europe (Milan)
  - eu-west-1 – Europe (Irlande)
  - eu-west-2, Europe (Londres)
  - eu-west-3, Europe (Paris)
  - me-south-1 – Moyen-Orient (Bahreïn)

- sa-east-1, Amérique du Sud (São Paulo)
- us-east-1 – USA Est (Virginie du Nord)
- us-east-2 – USA Est (Ohio)
- us-west-1, USA Ouest (Californie du Nord)
- us-west-2 – USA Ouest (Oregon)

Sessions d'arrière-plan utilisateur pour AWS Glue ETL

Les sessions d'arrière-plan des utilisateurs permettent de poursuivre les tâches d'analyse et d'apprentissage automatique de longue durée même après que l'utilisateur se soit déconnecté de l'interface de son bloc-notes. Cette fonctionnalité est mise en œuvre par le biais AWS Glue de la fonctionnalité de propagation d'identité sécurisée. La page suivante explique les options de configuration et les comportements des sessions d'arrière-plan pour les utilisateurs.

> **Note**
>
> Les sessions d'arrière-plan utilisateur s'appliquent aux sessions AWS Glue interactives initiées via des interfaces de bloc-notes telles que SageMaker Unified Studio. L'activation ou la désactivation de cette fonctionnalité n'affecte que les nouvelles sessions interactives ; les sessions actives existantes ne sont pas affectées.

Configurer les sessions d'arrière-plan des utilisateurs

Les sessions d'arrière-plan de l'utilisateur doivent être activées à deux niveaux pour fonctionner correctement :

1. Niveau d'instance IAM Identity Center (configuré par les administrateurs iDC)
2. AWS Glue Niveau de configuration d'Identity Center (configuré par AWS Glue les administrateurs)

Activez les sessions d'arrière-plan des utilisateurs pour AWS Glue

Pour activer les sessions utilisateur en arrière-plan pour AWS Glue, vous devez définir le `userBackgroundSessionsEnabled` paramètre sur `true` dans la configuration Identity Center lors de la création ou de la mise à jour de la configuration.

Conditions préalables

- Votre rôle IAM utilisé pour create/update la configuration de l' AWS Glue Identity Center doit disposer de cette `sso:PutApplicationSessionConfiguration` autorisation. Cette autorisation permet d' AWS Glue activer les sessions d'arrière-plan des utilisateurs au niveau de l'application iDC AWS Glue gérée.
- Vos sessions AWS Glue interactives doivent utiliser AWS Glue la version 5.0 ou ultérieure et la propagation sécurisée des identités doit être activée.

Pour activer les sessions d'arrière-plan des utilisateurs à l'aide de AWS CLI :

```
aws glue create-glue-identity-center-configuration \
    --instance-arn "arn:aws:sso:::instance/ssoins-1234567890abcdef" \
    --user-background-sessions-enabled
```

Pour mettre à jour une configuration existante :

```
aws glue update-glue-identity-center-configuration \
    --user-background-sessions-enabled
```

Matrice de configuration

La configuration effective de la session utilisateur en arrière-plan dépend à la fois du paramètre de AWS Glue configuration et des paramètres au niveau de l'instance d'IAM Identity Center :

Le centre d'identité IAM est-il userBackgroundSession activé ?	AWS Glue userBackgroundSessionsActivé	Comportement
Oui	TRUE	Sessions d'arrière-plan utilisateur activées
Oui	FALSE	La session expire avec la déconnexion de l'utilisateur
Non	TRUE	La création de session échoue avec une exception
Non	FALSE	La session expire avec la déconnexion de l'utilisateur

Durée par défaut de la session d'arrière-plan pour les utilisateurs

Par défaut, toutes les sessions d'arrière-plan des utilisateurs ont une durée maximale de 7 jours dans IAM Identity Center. Les administrateurs peuvent modifier cette durée dans la console IAM Identity Center. Ce paramètre s'applique au niveau de l'instance IAM Identity Center, affectant toutes les applications IAM Identity Center prises en charge au sein de cette instance.

- La durée peut être réglée sur n'importe quelle valeur comprise entre 15 minutes et 90 jours
- Ce paramètre est configuré dans la console IAM Identity Center sous Paramètres → Authentification → Configurer (section Tâches non interactives)

> **Note**
>
> AWS Glue les sessions interactives ont un délai d'inactivité distinct de 48 heures par défaut. Les sessions se terminent lorsque le délai d'inactivité de la AWS Glue session ou que la durée de la session d'arrière-plan de l'utilisateur est atteinte, selon la première éventualité.

Impact de la désactivation des sessions d'arrière-plan des utilisateurs

Lorsque les sessions d'arrière-plan des utilisateurs sont désactivées au niveau AWS Glue de la configuration :

- Sessions interactives existantes : continuez à fonctionner sans interruption si elles ont été démarrées avec les sessions utilisateur en arrière-plan activées. Ces sessions continueront à utiliser leurs jetons de session d'arrière-plan existants jusqu'à ce qu'elles se terminent naturellement ou soient explicitement arrêtées.
- Nouvelles sessions interactives : utiliseront le flux standard de propagation des identités fiables et s'arrêteront lorsque l'utilisateur se déconnectera ou que sa session interactive expirera (par exemple lors de la fermeture d'un JupyterLab bloc-notes SageMaker Unified Studio).

Modification de la durée des sessions d'arrière-plan des utilisateurs

Lorsque le paramètre de durée des sessions d'arrière-plan des utilisateurs est modifié dans IAM Identity Center :

- Sessions interactives existantes : continuez à fonctionner avec la même durée de session en arrière-plan que celle avec laquelle elles ont été démarrées

- Nouvelles sessions interactives : utilisera la nouvelle durée de session pour les sessions d'arrière-plan

Considérations relatives à l’exécution

Conditions de fin de session

Lorsque vous utilisez des sessions utilisateur en arrière-plan, une session AWS Glue interactive continue de fonctionner jusqu'à ce que l'un des événements suivants se produise :

- La session d'arrière-plan de l'utilisateur expire (selon la configuration iDC, jusqu'à 90 jours)
- la session d’arrière-plan pour les utilisateurs est révoquée manuellement par un administrateur.
- La session AWS Glue interactive atteint son délai d'inactivité (par défaut : 48 heures après la dernière instruction exécutée)
- L'utilisateur arrête ou redémarre explicitement le noyau du bloc-notes

Persistance des données

Lorsque vous utilisez des sessions d'arrière-plan utilisateur :

- Les utilisateurs ne peuvent pas se reconnecter à l'interface de leur bloc-notes pour voir les résultats une fois déconnectés
- Configurez vos instructions Spark pour écrire les résultats dans un stockage persistant (tel qu'Amazon S3) avant la fin de l'exécution

Implication financière

- Les tâches continueront de s'exécuter jusqu'à leur fin même une fois que les utilisateurs auront mis fin à leur JupyterLab session SageMaker Unified Studio et seront facturées pendant toute la durée de l'exécution terminée.
- Surveillez vos sessions d'arrière-plan actives pour éviter les coûts inutiles liés à des sessions oubliées ou abandonnées

Disponibilité des fonctions

Les sessions d'arrière-plan des utilisateurs pour AWS Glue sont disponibles pour :

- AWS Glue sessions interactives uniquement (les AWS Glue jobs et les jobs de streaming ne sont pas pris en charge)
- AWS Glue version 5.0 et versions ultérieures
- Configurations activées pour la propagation des identités fiables uniquement

Connexion et surveillance AWS Glue

Vous pouvez automatiser l'exécution de vos tâches d'extraction, de transformation et de chargement (ETL). AWS Glue fournit les métriques des crawlers et des tâches que vous pouvez surveiller. Une fois que vous avez configuré l' AWS Glue Data Catalog avec les métadonnées requises, AWS Glue fournit des statistiques sur l'état de votre environnement. Vous pouvez automatiser l'appel des crawlers et des tâches avec une planification temporelle basée sur cron. Vous pouvez également déclencher des tâches lorsqu'un déclencheur basé sur un événement s'exécute.

AWS Glueest intégré à AWS CloudTrail un service qui fournit un enregistrement des actions entreprises par un utilisateur, un rôle ou un AWS service dansAWS Glue. Si vous créez un suivi, vous pouvez activer la diffusion continue des CloudTrail événements vers un bucket Amazon Simple Storage Service (Amazon S3), Amazon Logs et CloudWatch Amazon CloudWatch Events. Chaque événement ou entrée de journal contient des informations sur la personne ayant initié la demande. Avec AWS Glue la version 5.1, le contexte d'audit AWS Glue et les appels d' AWS Lake Formation API seront disponibles dans les AWS CloudTrail journaux.

Utilisez Amazon CloudWatch Events pour automatiser vos AWS services et répondre automatiquement aux événements du système tels que les problèmes de disponibilité des applications ou les modifications des ressources. Les événements issus AWS des services sont transmis à CloudWatch Events en temps quasi réel. Vous pouvez écrire des règles simples pour préciser les événements qui vous intéressent et les actions automatisées à effectuer quand un événement correspond à une règle.

Consultez aussi

- Automatiser AWS Glue avec EventBridge
- Journalisation entre comptes CloudTrail

La journalisation constitue un aspect important de la sécurité dans le cloud. Vous devez configurer la journalisation de manière à ne pas capturer les secrets et le matériel confidentiel tout en capturant

les informations nécessaires au débogage et à la sécurisation de votre infrastructure cloud. Assurez-vous de vous familiariser avec ce qui est enregistré.

Validation de conformité pour AWS Glue

Pour savoir si un programme Services AWS de conformité Service AWS s'inscrit dans le champ d'application de programmes de conformité spécifiques, consultez Services AWS la section de conformité et sélectionnez le programme de conformité qui vous intéresse. Pour des informations générales, voir Programmes de AWS conformité Programmes AWS de .

Vous pouvez télécharger des rapports d'audit tiers à l'aide de AWS Artifact. Pour plus d'informations, voir Téléchargement de rapports dans AWS Artifact .

Votre responsabilité en matière de conformité lors de l'utilisation Services AWS est déterminée par la sensibilité de vos données, les objectifs de conformité de votre entreprise et les lois et réglementations applicables. Pour plus d'informations sur votre responsabilité en matière de conformité lors de l'utilisation Services AWS, consultez AWS la documentation de sécurité.

Résilience dans AWS Glue

L'infrastructure AWS mondiale est construite autour des AWS régions et des zones de disponibilité. AWS Les régions fournissent plusieurs zones de disponibilité physiquement séparées et isolées, connectées par un réseau à faible latence, à haut débit et hautement redondant. Avec les zones de disponibilité, vous pouvez concevoir et exploiter des applications et des bases de données qui basculent automatiquement d'une zone de disponibilité à l'autre sans interruption. Les zones de disponibilité sont plus hautement disponibles, tolérantes aux pannes et évolutives que les infrastructures traditionnelles à un ou plusieurs centres de données.

Pour plus d'informations sur AWS les régions et les zones de disponibilité, consultez la section Infrastructure AWS mondiale.

Pour plus d'informations sur la résilience des AWS Glue tâches, voir Erreur : comportement de basculement entre VPCs entrées. AWS Glue

Sécurité de l'infrastructure dans AWS Glue

En tant que service géré, AWS Glue il est protégé par les procédures de sécurité du réseau AWS mondial décrites dans le livre blanc Amazon Web Services : présentation des processus de sécurité.

Vous utilisez des appels d'API AWS publiés pour accéder AWS Glue via le réseau. Les clients doivent supporter le protocole TLS (Sécurité de la couche transport) 1.0 ou une version ultérieure. Nous recommandons TLS 1.2 ou version ultérieure. Les clients doivent aussi prendre en charge les suites de chiffrement PFS (Perfect Forward Secrecy) comme Ephemeral Diffie-Hellman (DHE) ou Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). La plupart des systèmes modernes tels que Java 7 et les versions ultérieures prennent en charge ces modes.

En outre, les demandes doivent être signées à l'aide d'un ID de clé d'accès et d'une clé d'accès secrète associée à un principal IAM. Vous pouvez également utiliser AWS Security Token Service (AWS STS) pour générer des informations d'identification de sécurité temporaires et signer les demandes.

Rubriques

- Configuration de points de terminaison d'un VPC d'interface (AWS PrivateLink) pour AWS Glue (AWS PrivateLink)
- Configuration d'Amazon partagé VPCs

Configuration de points de terminaison d'un VPC d'interface (AWS PrivateLink) pour AWS Glue (AWS PrivateLink)

Vous pouvez établir une connexion privée entre votre VPC et AWS Glue en créant un point de terminaison de VPC d'interface. Les points de terminaison de l'interface sont alimentés par AWS PrivateLinkune technologie qui vous permet d'accéder en privé AWS Glue APIs sans passerelle Internet, appareil NAT, connexion VPN ou connexion AWS Direct Connect. Les instances de votre VPC ne nécessitent pas d'adresses IP publiques pour communiquer avec AWS Glue APIs. Le trafic entre votre VPC et AWS Glue ne quitte pas le réseau Amazon.

Chaque point de terminaison d'interface est représenté par une ou plusieurs interfaces réseau Elastic dans vos sous-réseaux.

Pour de plus amples informations, consultez Points de terminaison VPC (AWS PrivateLink) dans le Guide de l'utilisateur Amazon VPC.

Considérations relatives aux points de terminaison de VPC AWS Glue

Avant de configurer un point de terminaison de VPC d'interface pour AWS Glue, assurez-vous de vérifier les Propriétés et limitations du point de terminaison d'interface dans le Guide de l'utilisateur Amazon VPC.

AWS Glue prend en charge l'exécution d'appels en direction de toutes ses actions d'API depuis votre VPC.

Création d'un point de terminaison de VPC d'interface pour AWS Glue

Vous pouvez créer un point de terminaison VPC pour le AWS Glue service à l'aide de la console Amazon VPC ou du (). AWS Command Line Interface AWS CLI Pour de plus amples informations, veuillez consulter Création d'un point de terminaison d'interface dans le Guide de l'utilisateur Amazon VPC.

Pour créer un point de terminaison de VPC pour AWS Glue, utilisez le nom de service suivant :

- com.amazonaws. *region*.colle

Si vous activez le DNS privé pour le point de terminaison, vous pouvez faire des demandes d'API à AWS Glue en utilisant son nom DNS par défaut pour la région, par exemple `glue.us-east-1.amazonaws.com`.

Pour plus d'informations, consultez Accès à un service via un point de terminaison d'interface dans le Guide de l'utilisateur Amazon VPC.

Création d'une stratégie de point de terminaison d'un VPC pour AWS Glue

Vous pouvez attacher une stratégie de point de terminaison à votre point de terminaison d'un VPC qui contrôle l'accès à AWS Glue. La politique spécifie les informations suivantes :

- Le principal qui peut exécuter des actions.
- Les actions qui peuvent être effectuées.
- Les ressources sur lesquelles les actions peuvent être exécutées.

Pour plus d'informations, consultez Contrôle de l'accès aux services avec points de terminaison d'un VPC dans le Guide de l'utilisateur Amazon VPC.

Exemple : politique du point de terminaison de VPC pour AWS Glue afin de permettre la création et la mise à jour de tâches

Voici un exemple de stratégie de point de terminaison pour AWS Glue. Lorsqu'elle est attachée à un point de terminaison, cette politique accorde l'accès aux actions AWS Glue répertoriées pour tous les principaux sur toutes les ressources.

```
{
  "Statement": [
    {
      "Sid": "RestrictPassRole",
      "Effect": "Allow",
      "Action": "iam:PassRole",
      "Resource": "arn:aws:iam::123456789012:role/GlueServiceRole*",
      "Condition": {
        "StringEquals": {
          "iam:PassedToService": "glue.amazonaws.com"
        }
      }
    }
  ]
}
```

Exemple : politique de point de terminaison d'un VPC autorisant l'accès au catalogue de données en lecture seule

Voici un exemple de stratégie de point de terminaison pour AWS Glue. Lorsqu'elle est attachée à un point de terminaison, cette politique accorde l'accès aux actions AWS Glue répertoriées pour tous les principaux sur toutes les ressources.

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:GetDatabase",
        "glue:GetDatabases",
        "glue:GetTable",
        "glue:GetTables",
        "glue:GetTableVersion",
        "glue:GetTableVersions",
        "glue:GetPartition",
        "glue:GetPartitions",
        "glue:BatchGetPartition",
        "glue:SearchTables"
      ],
      "Resource": "*"
    }
  ]
```

```
}
```

Configuration d'Amazon partagé VPCs

AWS Glueprend en charge les clouds privés virtuels partagés (VPCs) dans Amazon Virtual Private Cloud. Le partage Amazon VPC permet à plusieurs AWS comptes de créer leurs ressources applicatives, telles que des EC2 instances Amazon et des bases de données (Amazon Relational Database Service Amazon RDS), dans un Amazon partagé et géré de manière centralisée. VPCs Dans ce modèle, le compte propriétaire du VPC (propriétaire) partage un ou plusieurs sous-réseaux avec d'autres comptes (participants) appartenant à la même organisation. AWS Organizations Une fois un sous-réseau partagé, les participants peuvent afficher, créer, modifier et supprimer leurs ressources d'application contenues dans les sous-réseaux partagés avec eux.

Dans AWS Glue, pour créer une connexion avec un sous-réseau partagé, vous devez créer un groupe de sécurité au sein de votre compte et associer le groupe de sécurité au sous-réseau partagé.

Pour plus d'nformations, consultez les rubriques suivantes :

- Utilisation de Shared VPCs dans le guide de l'utilisateur Amazon VPC
- Qu'est-ce que c'est AWS Organizations ? dans le guide de AWS Organizations l'utilisateur

Résolution des problèmes de AWS Glue

Si vous rencontrez des problèmes lors de l' AWS Glue utilisation, consultez les rubriques de cette section.

Pour le dépannage d'Apache Spark dans AWS Glue, voir Résolution des problèmes liés à l'IA générative pour Apache Spark AWS Glue, qui fournit une assistance basée sur l'IA pour diagnostiquer et résoudre les problèmes liés aux tâches Spark.

Transformez votre résolution des problèmes liés à Apache Spark grâce à notre agent de dépannage basé sur l'IA, qui prend désormais en charge tous les principaux modes de déploiement, notamment AWS Glue Amazon EMR-EC2, Amazon EMR-Serverless et Amazon AI Notebooks . SageMaker Ce puissant agent élimine les processus de débogage complexes en combinant des interactions en langage naturel, une analyse de la charge de travail en temps réel et des recommandations de code intelligent dans une expérience fluide. Pour plus de détails sur l'implémentation, reportez-vous à la section Qu'est-ce que l'agent de résolution des problèmes Apache Spark pour Amazon EMR ?

Rubriques

- Collecte d'informations AWS Glue de dépannage
- Résolution des erreurs de configuration courantes de Glue
- Erreurs du robot lorsqu'il utilise les autorisations de Lake Formation
- Résolution AWS Glue des erreurs Ray liées aux journaux
- Exceptions AWS Glue liées au machine learning
- AWS Glue quotas

Collecte d'informations AWS Glue de dépannage

Si vous rencontrez des erreurs ou un comportement inattendu AWS Glue et que vous devez les contacter AWS Support, vous devez d'abord recueillir des informations sur IDs les noms et les journaux associés à l'action ayant échoué. La disponibilité de ces informations peut vous aider Support à résoudre les problèmes que vous rencontrez.

En plus de votre ID de compte, collectez les informations suivantes pour chacun de ces types de défaillances :

Lorsqu'un crawler échoue, collectez les informations suivantes :

- Nom du crawler

  Les journaux des exécutions des robots d'exploration se trouvent dans CloudWatch Logs sous `/aws-glue/crawlers`.

Lorsqu'une connexion de test échoue, collectez les informations suivantes :

- Connection name (Nom de la connexion)
- ID de connexion
- La chaîne de connexion JDBC au format `jdbc:protocol://host:port/database-name`.

  Les journaux des connexions de test se trouvent dans CloudWatch Logs sous `/aws-glue/testconnection`.

Lorsqu'une tâche échoue, collectez les informations suivantes :

- Nom du job
- ID d'exécution de la tâche au format `jr_xxxxx`.

  Les journaux des exécutions de tâches se trouvent dans CloudWatch Logs sous `/aws-glue/jobs`.

Résolution des erreurs de configuration courantes de Glue

Si vous rencontrez des erreurs AWS Glue, utilisez les informations suivantes pour vous aider à trouver la source des problèmes et à les résoudre.

> **Note**
>
> Le AWS Glue GitHub référentiel contient des conseils de dépannage supplémentaires dans AWS Glue les questions fréquemment posées.

Rubriques

- Erreur : Ressource non disponible

- Erreur : Impossible de trouver le point de terminaison S3 ou la passerelle NAT du subnetId dans VPC
- Erreur : Règle de trafic entrant obligatoire dans le groupe de sécurité
- Erreur : Règle de trafic sortant obligatoire dans le groupe de sécurité
- Erreur : L'exécution du Job a échoué car le rôle transmis doit être autorisé à assumer le rôle pour le AWS Glue service
- Erreur : l' DescribeVpcEndpoints action n'est pas autorisée. impossible de valider l'ID VPC vpc-id
- Erreur : l' DescribeRouteTables action n'est pas autorisée. impossible de valider l'identifiant du sous-réseau : ID du sous-réseau dans l'identifiant du VPC : vpc-id
- Erreur : Impossible d'appeler ec2 : DescribeSubnets
- Erreur : Impossible d'appeler ec2 : DescribeSecurityGroups
- Erreur : Impossible de trouver le sous-réseau pour la zone de disponibilité
- Erreur : Exception d'exécution de tâche lors d'écriture sur une cible JDBC
- Erreur : Amazon S3 : l'opération n'est pas valide pour la classe de stockage de l'objet
- Erreur : Délai Amazon S3
- Erreur : Accès à Amazon S3 refusé
- Erreur : L'identifiant de la clé d'accès Amazon S3 n'existe pas
- Erreur : l'exécution d'une tâche échoue lors de l'accès à Amazon S3 avec un URI s3a://
- Erreur : Le jeton de service Amazon S3 a expiré
- Erreur : Aucun DNS privé trouvé pour l'interface réseau
- Erreur : Échec de l'allocation du point de terminaison de développement
- Erreur : Serveur de bloc-notes CREATE_FAILED
- Erreur : Échec du démarrage du bloc-notes local
- Erreur : Échec de l'exécution du crawler
- Erreur : les partitions n'ont pas été mises à jour
- Erreur : la mise à jour du signet de tâche a échoué en raison d'une incompatibilité de version
- Erreur : Une tâche retraite des données lorsque les signets de tâche sont activés
- Erreur : comportement de basculement entre VPCs AWS Glue

Erreur : Ressource non disponible

En cas de AWS Glue renvoi d'un message indiquant que la ressource n'est pas disponible, vous pouvez consulter les messages d'erreur ou les journaux pour en savoir plus sur le problème. Les tâches suivantes décrivent des méthodes générales permettant de résoudre des problèmes.

- Pour tous les points de terminaison de connexion et de développement que vous utilisez, vérifiez que votre cluster n'est pas à court d'interfaces réseau Elastic.

Erreur : Impossible de trouver le point de terminaison S3 ou la passerelle NAT du subnetId dans VPC

Vérifiez l'ID du sous-réseau et l'ID du VPC dans le message afin de diagnostiquer le problème rencontré.

- Vérifiez que vous disposez d'un point de terminaison d'un VPC Amazon S3 configuré, ce qui est obligatoire avec AWS Glue. En outre, vérifiez votre passerelle NAT, si celle-ci fait partie de votre configuration. Pour de plus amples informations, veuillez consulter Types de points de terminaison d'un VPC pour Amazon S3.

Erreur : Règle de trafic entrant obligatoire dans le groupe de sécurité

Au moins un groupe de sécurité doit ouvrir tous les ports d'entrée. Pour limiter le trafic, le groupe de sécurité source de votre règle de trafic entrant peut être limité au même groupe de sécurité.

- Pour toutes les connexions que vous utilisez, vérifiez que votre groupe de sécurité possède une règle de trafic entrant avec référence circulaire. Pour de plus amples informations, veuillez consulter Configuration de l'accès réseau aux magasins de données.
- Lorsque vous utilisez un point de terminaison de développement, vérifiez que votre groupe de sécurité possède une règle de trafic entrant avec référence circulaire. Pour de plus amples informations, veuillez consulter Configuration de l'accès réseau aux magasins de données.

Erreur : Règle de trafic sortant obligatoire dans le groupe de sécurité

Au moins un groupe de sécurité doit ouvrir tous les ports de sortie. Pour limiter le trafic, le groupe de sécurité source de votre règle de trafic sortant peut être limité au même groupe de sécurité.

- Pour toutes les connexions que vous utilisez, vérifiez que votre groupe de sécurité possède une règle de trafic sortant avec référence circulaire. Pour de plus amples informations, veuillez consulter Configuration de l'accès réseau aux magasins de données.
- Lorsque vous utilisez un point de terminaison de développement, vérifier que votre groupe de sécurité possède une règle de trafic sortant avec référence circulaire. Pour de plus amples informations, veuillez consulter Configuration de l'accès réseau aux magasins de données.

Erreur : L'exécution du Job a échoué car le rôle transmis doit être autorisé à assumer le rôle pour le AWS Glue service

L'utilisateur qui définit une tâche doit avoir l'autorisation pour `iam:PassRole` pour AWS Glue.

- Lorsqu'un utilisateur crée une AWS Glue tâche, vérifiez que le rôle de l'utilisateur contient une politique contenant `iam:PassRole` pour AWS Glue. Pour de plus amples informations, veuillez consulter Étape 3 : attacher une politique aux utilisateurs ou aux groupes accédant à AWS Glue.

Erreur : l' DescribeVpcEndpoints action n'est pas autorisée. impossible de valider l'ID VPC vpc-id

- Vérifiez la politique transmise AWS Glue pour `ec2:DescribeVpcEndpoints` obtenir l'autorisation.

Erreur : l' DescribeRouteTables action n'est pas autorisée. impossible de valider l'identifiant du sous-réseau : ID du sous-réseau dans l'identifiant du VPC : vpc-id

- Vérifiez la politique transmise AWS Glue pour `ec2:DescribeRouteTables` obtenir l'autorisation.

Erreur : Impossible d'appeler ec2 : DescribeSubnets

- Vérifiez la politique transmise AWS Glue pour `ec2:DescribeSubnets` obtenir l'autorisation.

Erreur : Impossible d'appeler ec2 : DescribeSecurityGroups

- Vérifiez la politique transmise AWS Glue pour `ec2:DescribeSecurityGroups` obtenir l'autorisation.

Erreur : Impossible de trouver le sous-réseau pour la zone de disponibilité

- La zone de disponibilité n'est peut-être pas disponible pour AWS Glue. Créez et utilisez un sous-réseau dans une autre zone de disponibilité que celle indiquée dans le message.

Erreur : Exception d'exécution de tâche lors d'écriture sur une cible JDBC

Lorsque vous exécutez une tâche qui écrit sur une cible JDBC, la tâche peut rencontrer des erreurs dans les scénarios suivants :

- Si votre tâche écrit sur une table Microsoft SQL Server, que la table comporte des colonnes définies de type `Boolean`, alors que la table doit être prédéfinie dans la base de données SQL Server. Lorsque vous définissez la tâche sur la AWS Glue console à l'aide d'une cible SQL Server avec l'option Créer des tables dans votre cible de données, ne mappez aucune colonne source à une colonne cible contenant un type de données`Boolean`. Vous pourriez rencontrer une erreur lors de l'exécution de la tâche.

  Vous pouvez éviter cette erreur en procédant comme suit :

  - Choisissez une table existante avec la colonne Boolean (Booléen).
  - Modifiez la transformation `ApplyMapping` et mappez la colonne Boolean (Booléen) dans la source à un nombre ou à une chaîne de la cible.
  - Modifiez la transformation `ApplyMapping` pour supprimer la colonne Boolean (Booléen) de la source.
- Si votre tâche écrit sur une table Oracle, vous devrez peut-être ajuster la longueur des noms des objets Oracle. Dans certaines versions d'Oracle, la longueur maximale d'un identifiant est limitée à 30 octets ou 128 octets. Cette limite affecte les noms de tables et les noms de colonnes des magasins de données cibles Oracle.

  Vous pouvez éviter cette erreur en procédant comme suit :

  - Nommez les tables cibles Oracle en respectant la dans la limite de votre version.

- Les noms de colonnes par défaut sont générés à partir des noms de champs dans les données. Dans les situations où les noms des colonnes excèdent la limite, utilisez les transformations `ApplyMapping` ou `RenameField` pour modifier le nom de la colonne afin de respecter de la limite.

Erreur : Amazon S3 : l'opération n'est pas valide pour la classe de stockage de l'objet

Si cette erreur est AWS Glue renvoyée, votre AWS Glue tâche a peut-être consisté à lire des données à partir de tables comportant des partitions entre les différents niveaux de classe de stockage Amazon S3.

- En utilisant des exclusions de classes de stockage, vous pouvez vous assurer que vos AWS Glue tâches fonctionneront sur des tables comportant des partitions entre ces niveaux de classe de stockage. Sans exclusion, les tâches qui lisent les données de ces niveaux échouent avec l'erreur suivante : `AmazonS3Exception: The operation is not valid for the object's storage class`.

  Pour de plus amples informations, veuillez consulter Exclusion des classes de stockage Amazon S3.

Erreur : Délai Amazon S3

S'il AWS Glue renvoie une erreur d'expiration du délai de connexion, cela peut être dû au fait qu'il tente d'accéder à un compartiment Amazon S3 dans une autre AWS région.

- Un point de terminaison Amazon S3 VPC peut uniquement acheminer le trafic vers des compartiments au sein d'une région. AWS Si vous avez besoin de vous connecter à des compartiments dans d'autres régions, vous pouvez utiliser une passerelle NAT pour contourner ce problème. Pour plus d'informations, consultez Passerelles NAT.

Erreur : Accès à Amazon S3 refusé

Si AWS Glue un message d'erreur de refus d'accès est renvoyé à un compartiment ou à un objet Amazon S3, cela peut être dû au fait que le rôle IAM fourni ne dispose pas d'une politique autorisant l'accès à votre magasin de données.

- Une tâche ETL doit avoir accès à un magasin de données Amazon S3 utilisé comme source ou cible. Un crawler doit avoir accès à un magasin de données Amazon S3 qu'il analyse. Pour de plus amples informations, veuillez consulter Étape 2 : créer un rôle IAM pour AWS Glue.

Erreur : L'identifiant de la clé d'accès Amazon S3 n'existe pas

Si une erreur AWS Glue renvoie une clé d'accès ID n'existe pas lors de l'exécution d'une tâche, cela peut être dû à l'une des raisons suivantes :

- Une tâche ETL utilise un rôle IAM pour accéder aux banques de données, confirmez que le rôle IAM de votre tâche n'a pas été supprimé avant que la tâche commence.
- Un rôle IAM contient des autorisations permettant d'accéder à vos magasins de données, confirmez que la stratégie Amazon S3 attachée contenant `s3:ListBucket` est correcte.

Erreur : l'exécution d'une tâche échoue lors de l'accès à Amazon S3 avec un URI **`s3a://`**

Si une tâche renvoie une erreur similaire à Échec de l'analyse d'un document XML avec une classe de gestionnaire , la raison peut en être une défaillance lors de l'affichage de centaines de fichiers à l'aide d'un URI `s3a://`. Accédez plutôt à votre magasin de données à l'aide d'un URI `s3://`. Le suivi de l'exception suivant met en évidence les erreurs à rechercher :

```
1. com.amazonaws.SdkClientException: Failed to parse XML document with handler class
 com.amazonaws.services.s3.model.transform.XmlResponsesSaxParser$ListBucketHandler
2. at
 com.amazonaws.services.s3.model.transform.XmlResponsesSaxParser.parseXmlInputStream(XmlRespons
3. at
 com.amazonaws.services.s3.model.transform.XmlResponsesSaxParser.parseListBucketObjectsResponse
4. at com.amazonaws.services.s3.model.transform.Unmarshallers
$ListObjectsUnmarshaller.unmarshall(Unmarshallers.java:70)
5. at com.amazonaws.services.s3.model.transform.Unmarshallers
$ListObjectsUnmarshaller.unmarshall(Unmarshallers.java:59)
6. at
 com.amazonaws.services.s3.internal.S3XmlResponseHandler.handle(S3XmlResponseHandler.java:62)
7. at
 com.amazonaws.services.s3.internal.S3XmlResponseHandler.handle(S3XmlResponseHandler.java:31)
8. at
 com.amazonaws.http.response.AwsResponseHandlerAdapter.handle(AwsResponseHandlerAdapter.java:70
```

```
9. at com.amazonaws.http.AmazonHttpClient
$RequestExecutor.handleResponse(AmazonHttpClient.java:1554)
10. at com.amazonaws.http.AmazonHttpClient
$RequestExecutor.executeOneRequest(AmazonHttpClient.java:1272)
11. at com.amazonaws.http.AmazonHttpClient
$RequestExecutor.executeHelper(AmazonHttpClient.java:1056)
12. at com.amazonaws.http.AmazonHttpClient
$RequestExecutor.doExecute(AmazonHttpClient.java:743)
13. at com.amazonaws.http.AmazonHttpClient
$RequestExecutor.executeWithTimer(AmazonHttpClient.java:717)
14. at com.amazonaws.http.AmazonHttpClient
$RequestExecutor.execute(AmazonHttpClient.java:699)
15. at com.amazonaws.http.AmazonHttpClient$RequestExecutor.access
$500(AmazonHttpClient.java:667)
16. at com.amazonaws.http.AmazonHttpClient
$RequestExecutionBuilderImpl.execute(AmazonHttpClient.java:649)
17. at com.amazonaws.http.AmazonHttpClient.execute(AmazonHttpClient.java:513)
18. at com.amazonaws.services.s3.AmazonS3Client.invoke(AmazonS3Client.java:4325)
19. at com.amazonaws.services.s3.AmazonS3Client.invoke(AmazonS3Client.java:4272)
20. at com.amazonaws.services.s3.AmazonS3Client.invoke(AmazonS3Client.java:4266)
21. at com.amazonaws.services.s3.AmazonS3Client.listObjects(AmazonS3Client.java:834)
22. at org.apache.hadoop.fs.s3a.S3AFileSystem.getFileStatus(S3AFileSystem.java:971)
23. at
 org.apache.hadoop.fs.s3a.S3AFileSystem.deleteUnnecessaryFakeDirectories(S3AFileSystem.java:115
24. at org.apache.hadoop.fs.s3a.S3AFileSystem.finishedWrite(S3AFileSystem.java:1144)
25. at org.apache.hadoop.fs.s3a.S3AOutputStream.close(S3AOutputStream.java:142)
26. at org.apache.hadoop.fs.FSDataOutputStream
$PositionCache.close(FSDataOutputStream.java:74)
27. at org.apache.hadoop.fs.FSDataOutputStream.close(FSDataOutputStream.java:108)
28. at org.apache.parquet.hadoop.ParquetFileWriter.end(ParquetFileWriter.java:467)
29. at
 org.apache.parquet.hadoop.InternalParquetRecordWriter.close(InternalParquetRecordWriter.java:1
30. at
 org.apache.parquet.hadoop.ParquetRecordWriter.close(ParquetRecordWriter.java:112)
31. at
 org.apache.spark.sql.execution.datasources.parquet.ParquetOutputWriter.close(ParquetOutputWrit
32. at org.apache.spark.sql.execution.datasources.FileFormatWriter
$SingleDirectoryWriteTask.releaseResources(FileFormatWriter.scala:252)
33. at org.apache.spark.sql.execution.datasources.FileFormatWriter$$anonfun
$org$apache$spark$sql$execution$datasources$FileFormatWriter$$executeTask
$3.apply(FileFormatWriter.scala:191)
34. at org.apache.spark.sql.execution.datasources.FileFormatWriter$$anonfun
$org$apache$spark$sql$execution$datasources$FileFormatWriter$$executeTask
$3.apply(FileFormatWriter.scala:188)
```

```
35. at org.apache.spark.util.Utils
$.tryWithSafeFinallyAndFailureCallbacks(Utils.scala:1341)
36. at org.apache.spark.sql.execution.datasources.FileFormatWriter$.org$apache$spark
$sql$execution$datasources$FileFormatWriter$$executeTask(FileFormatWriter.scala:193)
37. at org.apache.spark.sql.execution.datasources.FileFormatWriter$$anonfun$write$1$
$anonfun$3.apply(FileFormatWriter.scala:129)
38. at org.apache.spark.sql.execution.datasources.FileFormatWriter$$anonfun$write$1$
$anonfun$3.apply(FileFormatWriter.scala:128)
39. at org.apache.spark.scheduler.ResultTask.runTask(ResultTask.scala:87)
40. at org.apache.spark.scheduler.Task.run(Task.scala:99)
41. at org.apache.spark.executor.Executor$TaskRunner.run(Executor.scala:282)
42. at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1149)
43. at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:624)
44. at java.lang.Thread.run(Thread.java:748)
```

Erreur : Le jeton de service Amazon S3 a expiré

Lors du transfert de données vers et depuis Amazon Redshift, des informations d'identification Amazon S3 temporaires qui expirent au bout d'une heure sont utilisées. Si vous avez une tâche de longue durée, elle peut échouer. Pour plus d'informations sur la façon de configurer vos tâches de longue durée pour déplacer des données vers et à partir d'Amazon Redshift, veuillez consulter aws-glue-programming-etl-connect-redshift-home.

Erreur : Aucun DNS privé trouvé pour l'interface réseau

Si une tâche échoue ou si l'allocation d'un point de terminaison échoue, cela peut être dû à un problème de configuration réseau.

- Si vous utilisez le DNS fourni par Amazon, la valeur de `enableDnsHostnames` doit être défini sur true. Pour plus d'informations, consultez DNS.

Erreur : Échec de l'allocation du point de terminaison de développement

Si le provisionnement d'un point de terminaison de développement AWS Glue échoue, cela peut être dû à un problème de configuration réseau.

- Lorsque vous définissez un point de terminaison de développement, le VPC, le sous-réseau et les groupes de sécurité sont validés pour confirmer qu'ils répondent à certaines exigences.
- Si vous avez fourni la clé publique SSH facultative, vérifiez qu'il s'agit d'une clé publique SSH valide.

- Vérifiez dans la console VPC que votre VPC utilise DHCP option set (Jeu d'options DHCP) valide. Pour plus d'informations, consultez Jeux d'options DHCP.
- Si le cluster reste dans l'état PROVISIONING (APPROVISIONNEMENT EN COURS), contactez AWS Support.

Erreur : Serveur de bloc-notes CREATE_FAILED

Si vous AWS Glue ne parvenez pas à créer le serveur bloc-notes pour un point de terminaison de développement, cela peut être dû à l'un des problèmes suivants :

- AWS Glue transmet un rôle IAM à Amazon EC2 lors de la configuration du serveur de blocs-notes. Le rôle IAM doit avoir une relation d'approbation avec Amazon EC2.
- Le rôle IAM doit avoir un profil d'instance du même nom. Lorsque vous créez le rôle pour Amazon EC2 avec la console IAM, le profil d'instance avec le même nom est automatiquement créé. Recherchez un message d'erreur dans le journal concernant un nom de profil d'instance `iamInstanceProfile.name` non valide. Pour plus d'informations, consultez Using Instance Profiles.
- Vérifiez que votre rôle est autorisé à accéder aux compartiments `aws-glue*` dans la stratégie que vous transmettez pour créer le serveur de bloc-notes.

Erreur : Échec du démarrage du bloc-notes local

Si votre bloc-notes local ne démarre pas et renvoie des messages d'erreur indiquant qu'un répertoire ou dossier est introuvable, cela peut être dû à l'un des problèmes suivants :

- Si vous travaillez sous Microsoft Windows, assurez-vous que la variable d'environnement `JAVA_HOME` pointe vers le répertoire Java correct. Il est possible de mettre à jour Java sans mettre à jour cette variable, et si celle-ci pointe vers un dossier qui n'existe plus, les blocs-notes Jupyter échouent à démarrer.

Erreur : Échec de l'exécution du crawler

Si vous AWS Glue ne parvenez pas à exécuter un robot d'exploration pour cataloguer vos données, cela peut être dû à l'une des raisons suivantes. Commencez par vérifier si une erreur figure dans la liste des crawlers de la console AWS Glue . Vérifiez s'il y a une icône de point d'exclamation en regard du nom du crawler et passez la souris sur l'icône pour affichez les messages associés.

- Consultez les journaux du robot d'exploration exécuté dans CloudWatch Logs under`/aws-glue/crawlers`.

Erreur : les partitions n'ont pas été mises à jour

Si vos partitions n'ont pas été mises à jour dans le catalogue de données lorsque vous avez exécuté une tâche ETL, ces instructions de journal provenant de la `DataSink` classe figurant dans les CloudWatch journaux peuvent être utiles :

- « `Attempting to fast-forward updates to the Catalog - nameSpace:` » – Indique la base de données, la table et le catalogID que la tâche a tenté de modifier. Si cette instruction n'est pas ici, vérifiez si `enableUpdateCatalog` a la valeur true et qu'elle est correctement passée en tant que paramètre `getSink()` ou dans `additional_options`.
- « `Schema change policy behavior:` » – Affiche la valeur du schéma `updateBehavior` que vous avez transmise.
- « `Schemas qualify (schema compare):` » – Sera vrai ou faux.
- « `Schemas qualify (case-insensitive compare):` » – Sera vrai ou faux.
- Si les deux valeurs sont fausses et `updateBehavior` que votre valeur n'est pas définie sur `UPDATE_IN_DATABASE`, votre DynamicFrame schéma doit être identique ou contenir un sous-ensemble des colonnes figurant dans le schéma de table du catalogue de données.

Pour en savoir plus sur la mise à jour des partitions, consultez Mise à jour du schéma et ajout de nouvelles partitions dans le catalogue de données à l'aide des tâches AWS Glue ETL.

Erreur : la mise à jour du signet de tâche a échoué en raison d'une incompatibilité de version

Vous essayez peut-être de paramétrer des AWS Glue tâches pour les appliquer à différents ensembles transformation/logic de données dans Amazon S3. Vous voulez suivre les fichiers traités sur les emplacements fournis. Lorsque vous exécutez la même tâche sur le même compartiment source et que vous écrivez simultanément sur la same/different destination (simultanéité >1), la tâche échoue avec cette erreur :

```
py4j.protocol.Py4JJavaError: An error occurred while
 callingz:com.amazonaws.services.glue.util.Job.commit.:com.amazonaws.services.gluejobexecutor.m
```

```
Continuation update failed due to version mismatch. Expected version 2 but found
version 3
```

Solution : définissez la concurrence sur 1 ou n'exécutez pas la tâche simultanément.

Actuellement, AWS Glue les favoris ne prennent pas en charge les exécutions de tâches simultanées et les validations échoueront.

Erreur : Une tâche retraite des données lorsque les signets de tâche sont activés

Dans certains cas, vous avez activé les signets de AWS Glue tâches, mais votre tâche ETL retraite des données qui ont déjà été traitées lors d'une exécution précédente. Vérifiez si l'origine de l'erreur est l'une des causes suivantes :

Simultanéité max.

Le fait de définir un nombre maximal d'exécutions simultanées pour la tâche supérieur à la valeur par défaut de 1 peut interférer avec les signets des tâches. Cela peut se produire lorsque les signets de tâche vérifient l'heure de la dernière modification des objets afin de contrôler quels objets doivent être traités à nouveau. Pour plus d'informations, consultez la discussion sur la simultanéité maximale dans Configuration des propriétés des tâches pour les tâches Spark dans AWS Glue.

Objet de tâche manquant

Assurez-vous que votre script d'exécution de tâche se termine par les éléments suivants :

```
job.commit()
```

Lorsque vous incluez cet objet, AWS Glue enregistre l'horodatage et le chemin de la tâche exécutée. Si vous réexécutez la tâche avec le même chemin, seuls AWS Glue les nouveaux fichiers seront traités. Si vous n'incluez pas cet objet et si les signets de tâche sont activés, la tâche retraite les fichiers déjà traités en même temps que les nouveaux fichiers et crée une redondance dans le magasin de données cible de la tâche.

Paramètre de contexte de transformation manquant

Le contexte de transformation est un paramètre facultatif dans la classe `GlueContext`, mais les signets de tâche ne fonctionnent pas si vous ne l'incluez pas. Pour résoudre cette erreur, ajoutez le paramètre de contexte de transformation lorsque vous créez le DynamicFrame, comme indiqué ci-dessous :

```
sample_dynF=create_dynamic_frame_from_catalog(database,
 table_name,transformation_ctx="sample_dynF")
```

Source d'entrée

Si vous utilisez une base de données relationnelle (une connexion JDBC) pour la source d'entrée, les signets de travail fonctionnent uniquement si les clés primaires de la table sont classées par ordre séquentiel. Les signets de tâche fonctionnent pour les nouvelles lignes, mais pas pour les lignes mises à jour. En effet, les signets de tâche recherchent les clés primaires, qui existent déjà. Cela ne s'applique pas si la source des entrées est Amazon Simple Storage Service (Amazon S3).

Heure de la dernière modification

Pour les sources d'entrée Amazon S3, les signets de tâche vérifient l'heure de la dernière modification des objets, au lieu des noms de fichier, afin de contrôler quels objets doivent être retraités. Si les données de la source d'entrée ont été modifiées depuis votre dernière exécution de tâche, les fichiers sont traités de nouveau lorsque vous exécutez la tâche à nouveau.

Erreur : comportement de basculement entre VPCs AWS Glue

Le processus suivant est utilisé pour le basculement des tâches dans les versions AWS Glue 5.1 et antérieures.

Résumé : une AWS Glue connexion est sélectionnée au moment de l'envoi d'une tâche. Si l'exécution de la tâche rencontre des problèmes (manque d'adresses IP, connectivité à la source, problème de routage), l'exécution de la tâche échouera. Si de nouvelles tentatives sont configurées, je AWS Glue réessaierai avec la même connexion.

1. À chaque tentative d'exécution, AWS Glue vérifie l'état des connexions dans l'ordre indiqué dans la configuration de la tâche, jusqu'à ce qu'il en trouve une utilisable. En cas de défaillance d'une zone de disponibilité (AZ), les connexions provenant de cette zone échoueront à la vérification et seront ignorées.
2. AWS Glue valide la connexion avec ce qui suit :
   - vérification de la validité de l'identifiant et du sous-réseau Amazon VPC ;
   - vérification de l'existence d'une passerelle NAT ou d'un point de terminaison Amazon VPC ;
   - vérification que le sous-réseau dispose de plus de zéro adresse IP allouée ;
   - vérification que la zone de disponibilité est saine.

AWS Glue Impossible de vérifier la connectivité au moment de la soumission de l'exécution de la tâche.

3. Pour les tâches utilisant Amazon VPC, tous les pilotes et exécuteurs seront créés dans la même zone de disponibilité avec la connexion sélectionnée au moment de la soumission de l'exécution de la tâche.
4. Si de nouvelles tentatives sont configurées, je AWS Glue réessaierai avec la même connexion. En effet, nous ne pouvons pas garantir que les problèmes liés à cette connexion dureront longtemps. Si une zone de disponibilité échoue, les exécutions de tâches existantes (en fonction de l'étape de l'exécution de la tâche) dans cette zone de disponibilité peuvent échouer. Une nouvelle tentative devrait détecter un échec de la zone de disponibilité et choisir une autre zone pour la nouvelle exécution.

Erreurs du robot lorsqu'il utilise les autorisations de Lake Formation

Utilisez les informations ci-dessous pour diagnostiquer et résoudre divers problèmes lors de la configuration du Crawler à l'aide des informations d'identification de Lake Formation.

Erreur : l'emplacement S3 : s3://examplepath n'est pas enregistré

Pour qu'un Crawler puisse s'exécuter à l'aide des informations d'identification de Lake Formation, vous devez d'abord configurer les autorisations Lake Formation. Pour résoudre cette erreur, veuillez enregistrer l'emplacement Amazon S3 cible dans Lake Formation. Pour plus d'informations, consultez la rubrique Enregistrement d'un emplacement Amazon S3.

Erreur : n' User/Role est pas autorisé à exécuter : lakeformation : GetDataAccess on resource

Veuillez ajouter la permission `lakeformation:GetDataAccess` au rôle crawler à l'aide de la console IAM ou AWS CLI. Avec cette autorisation, Lake Formation accède à la demande d'informations d'identification temporaires pour accéder aux données. Consultez la politique ci-dessous :

JSON

```
{
  "Version":"2012-10-17",
```

```
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "lakeformation:GetDataAccess"
    ],
    "Resource": "*"
  }
}
```

Erreur : autorisation(s) insuffisante(s) sur Lake Formation (nom de la base de données : ExampleDatabase, nom de la table : ExampleTable)

Dans la console Lake Formation (https://console.aws.amazon.com/lakeformation/), accordez au rôle d'explorateur des autorisations d'accès (`Create`,`Describe`,`Alter`) sur la base de données, qui est spécifiée comme base de données de sortie. Vous pouvez également accorder des autorisations sur le tableau. Pour plus d'informations, consultez la rubrique Octroi d'autorisations de base de données via la méthode de ressource nommée.

Erreur : autorisation(s) insuffisante(s) pour Lake Formation sur s3://examplepath

1. Indexation de site web intercompte
   a. Connectez-vous à la console Lake Formation (https://console.aws.amazon.com/lakeformation/) à l'aide du compte sur lequel le bucket Amazon S3 est enregistré (compte B). Accordez des autorisations d'emplacement des données au compte sur lequel le crawler sera exécuté. Cela permettra au crawler de lire les données depuis l'emplacement Amazon S3 cible.
   b. Dans le compte où le crawler est créé (compte A), accordez des autorisations emplacement des données sur l'emplacement Amazon S3 cible au rôle IAM utilisé pour l'exécution du crawler afin que le crawler puisse lire les données depuis la destination dans Lake Formation. Pour plus d'informations, consultez la rubrique Octroi d'autorisations d'emplacement de données (compte externe).
2. Indexation de site web dans le compte (le crawler et l'emplacement Amazon S3 enregistré sont dans le même compte) : accordez des autorisations d'emplacement de données au rôle IAM utilisé pour l'exécution du crawler à l'emplacement Amazon S3 pour que le crawler puisse lire les données de la cible dans Lake Formation. Pour plus d'informations, consultez la rubrique Octroi d'autorisations d'emplacement de données (même compte).

Questions fréquemment posées sur la configuration du crawler à l'aide des informations d'identification Lake Formation

1. Comment configurer un crawler pour qu'il fonctionne à l'aide des informations d'identification Lake Formation en utilisant la AWS Console ?

   Dans la AWS Glue console (https://console.aws.amazon.com/glue/), lors de la configuration du robot d'exploration, sélectionnez l'option Utiliser les informations d'identification de Lake Formation pour analyser la source de données Amazon S3. Pour l'exploration entre comptes, spécifiez l' Compte AWS ID auquel l'emplacement Amazon S3 cible est enregistré auprès de Lake Formation. Le champ accountId est facultatif pour l'indexation de site web intégrée au compte.

2. Comment configurer un crawler pour qu'il fonctionne à l'aide des informations d'identification Lake Formation en utilisant AWS CLI ?

   Lors de l'appel de l'API `CreateCrawler`, ajoutez `LakeFormationConfiguration` :

   ```
   "LakeFormationConfiguration": {
       "UseLakeFormationCredentials": true,
       "AccountId": "111111111111" (AWS account ID where the target Amazon S3 location
    is registered with Lake Formation)
     }
   ```

3. Quelles sont les cibles prises en charge pour Crawler à l'aide des informations d'identification Lake Formation ?

   Un Crawler utilisant les informations d'identification Lake Formation n'est pris en charge que pour Amazon S3 (indexation de site web dans le compte et entre comptes) et pour les cibles de catalogue de données dans le compte (où l'emplacement sous-jacent est Amazon S3), et dans les cibles Apache Iceberg.

4. Puis-je crawler plusieurs compartiments Amazon S3 au sein crawler à l'aide des informations d'identification Lake Formation ?

   Non. Pour les cibles d'indexation de site web utilisant le distributeur d'informations d'identification Lake Formation, les emplacements Amazon S3 sous-jacents doivent appartenir au même compartiment. Par exemple, les clients peuvent utiliser plusieurs sites cibles (`s3://bucket1/folder1, s3://bucket1/folder2`) s'ils se trouvent sous le même compartiment (bucket1). La spécification de différents compartiments (s3://bucket1/folder1, s3://bucket2/folder2) n'est pas prise en charge.

Résolution AWS Glue des erreurs Ray liées aux journaux

> **Important**
>
> AWS Glue for Ray ne sera plus ouvert aux nouveaux clients à compter du 30 avril 2026. Si vous souhaitez utiliser AWS Glue for Ray, inscrivez-vous avant cette date. Les clients existants peuvent continuer à utiliser le service normalement. Pour des fonctionnalités similaires à celles de AWS Glue for Ray, explorez Amazon EKS. Pour plus d'informations, consultez la section AWS Glue relative à la fin du support de Ray.

AWS Glue fournit un accès aux journaux émis par les processus Ray pendant l'exécution du job. En cas d'erreurs ou de comportements inattendus dans les tâches Ray, collectez d'abord des informations à partir des journaux pour déterminer la cause de l'échec. Nous fournissons également des journaux similaires pour les sessions interactives. Les journaux de sessions sont identifiables par le préfixe `/aws-glue/ray/sessions`.

Les lignes de journal sont envoyées CloudWatch en temps réel, au fur et à mesure de l'exécution de votre tâche. Les instructions d'impression sont ajoutées aux CloudWatch journaux une fois l'exécution terminée. Les journaux sont conservés pendant deux semaines après l'exécution d'une tâche.

Inspection des journaux de tâches Ray

Lorsqu'une tâche échoue, collectez le nom et l'ID d'exécution de celle-ci. Vous les trouverez dans la AWS Glue console. Accédez à la page de la tâche, puis à l'onglet Runs (Exécutions). Les journaux des tâches Ray sont stockés dans les groupes de CloudWatch journaux dédiés suivants.

- `/aws-glue/ray/jobs/script-log/` : stocke les journaux émis par votre script Ray principal.
- `/aws-glue/ray/jobs/ray-monitor-log/` : stocke les journaux émis par le processus de scalabilité automatique Ray. Ces journaux sont générés pour le nœud principal et non pour les autres composants master.
- `/aws-glue/ray/jobs/ray-gcs-logs/` : stocke les journaux émis par le processus GCS (global control store). Ces journaux sont générés pour le nœud principal et non pour les autres composants master.
- `/aws-glue/ray/jobs/ray-process-logs/` : stocke les journaux émis par d'autres processus Ray (principalement l'agent du tableau de bord) exécutés sur le nœud principal. Ces journaux sont générés pour le nœud principal et non pour les autres composants master.

- `/aws-glue/ray/jobs/ray-raylet-logs/` : stocke les journaux émis par chaque processus raylet. Ces journaux sont collectés dans un flux unique pour chaque composant master, y compris le nœud principal.
- `/aws-glue/ray/jobs/ray-worker-out-logs/` : stocke les journaux `stdout` pour chaque travail du cluster. Ces journaux sont générés pour chaque composant master, y compris le nœud principal.
- `/aws-glue/ray/jobs/ray-worker-err-logs/` : stocke les journaux `stderr` pour chaque travail du cluster. Ces journaux sont générés pour chaque composant master, y compris le nœud principal.
- `/aws-glue/ray/jobs/ray-runtime-env-log/` : stocke les journaux relatifs au processus de configuration Ray. Ces journaux sont générés pour chaque composant master, y compris le nœud principal.

Résolution des erreurs liées aux tâches Ray

Pour comprendre l'organisation des groupes de journaux Ray et identifier les groupes de journaux qui vous aideront à résoudre vos erreurs, vous devez disposer d'informations de base sur l'architecture Ray.

Dans l' AWS Glue ETL, un worker correspond à une instance. Lorsque vous configurez des travailleurs pour une AWS Glue tâche, vous définissez le type et le nombre d'instances dédiées à la tâche. Ray utilise le terme travail de différentes manières.

Ray utilise le nœud principal et le composant master pour distinguer les responsabilités d'une instance au sein d'un cluster Ray. Un composant master Ray peut héberger plusieurs processus acteurs qui effectuent des calculs afin d'obtenir le résultat de vos calculs distribués. Les acteurs qui exécutent une réplique d'une fonction sont appelés réplicas. Les réplicas d'acteurs peuvent également être appelés processus de travail. Les réplicas peuvent également s'exécuter sur le nœud principal, connu sous le nom de tête, car il exécute des processus supplémentaires pour coordonner le cluster.

Chaque acteur qui contribue à votre calcul génère son propre flux de journaux. Ce scénario nous donne quelques informations :

- Le nombre de processus qui émettent des journaux peut être supérieur au nombre de travaux qui sont alloués à la tâche. Souvent, chaque cœur de chaque instance comporte un acteur.

- Les nœuds principaux Ray émettent des journaux de gestion et de démarrage du cluster. En revanche, les composants master Ray émettent uniquement des journaux pour le travail qui les concerne.

Pour plus d'informations sur l'architecture Ray, consultez Architecture Whitepapers (Livres blancs sur l'architecture) dans la documentation Ray.

Problème : accès à Amazon S3

Vérifiez le message d'échec de la tâche exécutée. Si ce dernier ne fournit pas suffisamment d'informations, vérifiez `/aws-glue/ray/jobs/script-log/`.

Problème : gestion de la dépendance PIP

Vérifiez `/aws-glue/ray/jobs/ray-runtime-env-log/`.

Problème : inspection des valeurs intermédiaires dans le processus principal

Écrivez dans `stderr` ou `stdout` depuis votre script principal, et récupérez les journaux dans `/aws-glue/ray/jobs/script-log/`.

Problème : inspection des valeurs intermédiaires dans un processus enfant

Écrivez dans `stderr` ou `stdout` depuis votre fonction `remote`. Récupérez ensuite les journaux depuis `/aws-glue/ray/jobs/ray-worker-out-logs/` ou `/aws-glue/ray/jobs/ray-worker-err-logs/`. Votre fonction a pu être exécutée sur n'importe quel réplica. Vous pouvez donc être amené à examiner plusieurs journaux pour identifier la sortie souhaitée.

Problème : interprétation des adresses IP dans les messages d'erreur

Dans certaines situations d'erreur, votre tâche peut émettre un message d'erreur contenant une adresse IP. Ces adresses IP sont éphémères et sont utilisées par le cluster pour identifier les nœuds et communiquer entre eux. Les journaux d'un nœud seront publiés dans un flux de journaux avec un suffixe unique basé sur l'adresse IP.

Dans CloudWatch, vous pouvez filtrer vos journaux pour inspecter ceux spécifiques à cette adresse IP en identifiant ce suffixe. Par exemple, étant donné *`FAILED_IP`* et*`JOB_RUN_ID`*, vous pouvez identifier le suffixe avec :

```
filter @logStream like /JOB_RUN_ID/
| filter @message like /IP-/
| parse @message "IP-[*]" as ip
| filter ip like /FAILED_IP/
| fields replace(ip, ":", "_") as uIP
| stats count_distinct by uIP as logStreamSuffix
| display logStreamSuffix
```

Exceptions AWS Glue liées au machine learning

Cette rubrique décrit les chaînes et les codes d'erreur HTTP des exceptions AWS Glue liées au machine learning. Les codes d'erreur et les chaînes d'erreur sont fournis pour chaque activité de machine learning susceptible de se produire lorsque vous effectuez une opération. Vous pouvez également déterminer s'il est possible de retenter l'opération qui a entraîné l'erreur.

Annuler MLTask RunActivity

Cette activité comporte les exceptions suivantes :

- EntityNotFoundException (400)
  - « Impossible de trouver MLTransform dans le compte [AccountID] avec le descripteur [TransformName]. »
  - « Aucune exécution de tâche ML n'a été trouvée pour [taskRunId] : dans le compte [AccountID] pour transform [TransformName]. »

  Nouvelle tentative possible : non.

Créez MLTask RunActivity

Cette activité comporte les exceptions suivantes :

- InvalidInputException (400)
  - Internal service failure due to unexpected input (Échec du service interne en raison d'une entrée inattendue)
  - « A AWS Glue Table input source should be specified in transform. » (Une source d'entrée Glue Table doit être spécifiée dans la transformation.)

- Input source column [columnName] has an invalid data type defined in the catalog (Le type de données de la colonne de source d'entrée [Nom de la colonne] défini dans le catalogue n'est pas valide)
- Exactly one input record table must be provided (Une seule table d'enregistrement en entrée doit être fournie)
- Should specify database name (Doit spécifier le nom de la base de données)
- Should specify table name (Doit spécifier le nom de la table)
- Schema is not defined on the transform (Le schéma n'est pas défini sur la transformation)
- Schema should contain given primary key: [primaryKey] (Le schéma doit contenir une clé primaire donnée : [Clé primaire])
- Problem fetching the data catalog schema: [message] (Problème de récupération du schéma de catalogue de données : [message])
- « Impossible de définir la capacité maximale et le Num/Type nombre de travailleurs en même temps. »
- « Les deux WorkerType et NumberOfWorkers doivent être réglés. »
- « MaxCapacity devrait être >= [MaxCapacity]. »
- « NumberOfWorkers devrait être >= [MaxCapacity]. »
- Max retries should be non-negative (Les tentatives maximales ne peuvent pas être négatives)
- Find Matches parameters have not been set (Les paramètres de recherche de correspondances n'ont pas été définis)
- A primary key must be specified in Find Matches parameters (Une clé primaire doit être spécifiée dans les paramètres de recherche de correspondances)

Nouvelle tentative possible : non.

- AlreadyExistsException (400)
  - Transform with name [transformName] already exists (Une transformation appelée [Nom de la transformation] existe déjà)

  Nouvelle tentative possible : non.
- IdempotentParameterMismatchException (400)
  - Idempotent create request for transform [transformName] had mismatching parameters (Les paramètres de la demande de création idempotente ne correspondent pas)

  Nouvelle tentative possible : non.

- InternalServiceException (500)
  - Dependency failure (Échec de la dépendance)

  Nouvelle tentative possible : oui.
- ResourceNumberLimitExceededException (400)
  - ML Transforms count ([count]) has exceeded the limit of [limit] transforms (Le nombre de transformations ML [nombre] a dépassé la limite de [nombre limite] transformations)

  Nouvelle tentative possible : oui, une fois que vous aurez supprimé une transformation pour faire de la place à cette nouvelle transformation.

Supprimer MLTransform l'activité

Cette activité comporte les exceptions suivantes :

- EntityNotFoundException (400)
  - « Impossible de trouver MLTransform dans le compte [AccountID] avec le handle [TransformName] »

  Nouvelle tentative possible : non.

Obtenez MLTask RunActivity

Cette activité comporte les exceptions suivantes :

- EntityNotFoundException (400)
  - « Impossible de trouver MLTransform dans le compte [AccountID] avec le descripteur [TransformName]. »
  - « Aucune exécution de tâche ML n'a été trouvée pour [taskRunId] : dans le compte [AccountID] pour transform [TransformName]. »

  Nouvelle tentative possible : non.

Obtenez MLTask RunsActivity

Cette activité comporte les exceptions suivantes :

- EntityNotFoundException (400)
  - « Impossible de trouver MLTransform dans le compte [AccountID] avec le descripteur [TransformName]. »
  - « Aucune exécution de tâche ML n'a été trouvée pour [taskRunId] : dans le compte [AccountID] pour transform [TransformName]. »

  Nouvelle tentative possible : non.

Obtenir de MLTransform l'activité

Cette activité comporte les exceptions suivantes :

- EntityNotFoundException (400)
  - « Impossible de trouver MLTransform dans le compte [AccountID] avec le descripteur [TransformName]. »

  Nouvelle tentative possible : non.

Obtenir de MLTransforms l'activité

Cette activité comporte les exceptions suivantes :

- EntityNotFoundException (400)
  - « Impossible de trouver MLTransform dans le compte [AccountID] avec le descripteur [TransformName]. »

  Nouvelle tentative possible : non.
- InvalidInputException (400)
  - Account ID can't be blank (L'ID de compte doit être renseigné)
  - Sorting not supported for column [column] (Le tri n'est pas pris en charge pour la colonne [colonne])
  - [column] can't be blank ([colonne] ne peut pas être vide)
  - Internal service failure due to unexpected input (Échec du service interne en raison d'une entrée inattendue)

  Nouvelle tentative possible : non.

GetSaveLocationForTransformArtifactActivity

Cette activité comporte les exceptions suivantes :

- EntityNotFoundException (400)
  - « Impossible de trouver MLTransform dans le compte [AccountID] avec le descripteur [TransformName]. »

  Nouvelle tentative possible : non.
- InvalidInputException (400)
  - Unsupported artifact type [artifactType] (Type d'artefact [Type d'artefact] non pris en charge)
  - Internal service failure due to unexpected input (Échec du service interne en raison d'une entrée inattendue)

  Nouvelle tentative possible : non.

GetTaskRunArtifactActivity

Cette activité comporte les exceptions suivantes :

- EntityNotFoundException (400)
  - « Impossible de trouver MLTransform dans le compte [AccountID] avec le descripteur [TransformName]. »
  - « Aucune exécution de tâche ML n'a été trouvée pour [taskRunId] : dans le compte [AccountID] pour transform [TransformName]. »

  Nouvelle tentative possible : non.
- InvalidInputException (400)
  - File name [fileName] is invalid for publish (Le nom de fichier [Nom de fichier] n'est pas valide pour la publication)
  - Cannot retrieve artifact for [taskType] task type (Impossible de récupérer l'artefact pour le type de tâche [Type de tâche])
  - Cannot retrieve artifact for [artifactType] (Impossible de récupérer l'artefact pour [Type d'artefact]
  - Internal service failure due to unexpected input (Échec du service interne en raison d'une entrée inattendue)

Nouvelle tentative possible : non.

Publier MLTransform ModelActivity

Cette activité comporte les exceptions suivantes :

- EntityNotFoundException (400)
  - « Impossible de trouver MLTransform dans le compte [AccountID] avec le descripteur [TransformName]. »
  - An existing model with version - [version] cannot be found for account id - [accountId] - and transform id - [transformId] (Impossible de trouver un modèle existant avec la version [version] pour l'ID de compte [ID de compte] et l'ID de transformation [ID de transformation])

  Nouvelle tentative possible : non.
- InvalidInputException (400)
  - File name [fileName] is invalid for publish (Le nom de fichier [Nom de fichier] n'est pas valide pour la publication)
  - Illegal leading minus sign on unsigned string [string] (Signe moins de début non valide sur une chaîne non signée [chaîne])
  - Bad digit at end of [string] (Chiffre incorrect à la fin de [chaîne])
  - String value [string] exceeds range of unsigned long (La valeur de chaîne [chaîne] dépasse considérablement la plage des chaînes non signées)
  - Internal service failure due to unexpected input (Échec du service interne en raison d'une entrée inattendue)

  Nouvelle tentative possible : non.

PullLatestMLTransformModelActivity

Cette activité comporte les exceptions suivantes :

- EntityNotFoundException (400)
  - « Impossible de trouver MLTransform dans le compte [AccountID] avec le descripteur [TransformName]. »

Nouvelle tentative possible : non.

- InvalidInputException (400)
  - Internal service failure due to unexpected input (Échec du service interne en raison d'une entrée inattendue)

  Nouvelle tentative possible : non.
- ConcurrentModificationException (400)
  - Cannot create model version to train due to racing inserts with mismatching parameters (Impossible de créer une version de modèle pour l'entraînement en raison d'insertions de course avec des paramètres incorrects)
  - The ML Transform model for transform id [transformId] is stale or being updated by another process; Please retry (Le modèle de transformation ML pour l'ID [ID de transformation] est obsolète ou en cours de mise à jour par un autre processus. Veuillez réessayer)

  Nouvelle tentative possible : oui.

PutJobMetadataForMLTransformActivité

Cette activité comporte les exceptions suivantes :

- EntityNotFoundException (400)
  - « Impossible de trouver MLTransform dans le compte [AccountID] avec le descripteur [TransformName]. »
  - « Aucune exécution de tâche ML n'a été trouvée pour [taskRunId] : dans le compte [AccountID] pour transform [TransformName]. »

  Nouvelle tentative possible : non.
- InvalidInputException (400)
  - Internal service failure due to unexpected input (Échec du service interne en raison d'une entrée inattendue)
  - Unknown job metadata type [jobType] (Type de métadonnées inconnu pour la tâche [Type de tâche])
  - Must provide a task run ID to update (Doit fournir un ID d'exécution de tâche à mettre à jour)

  Nouvelle tentative possible : non.

StartExportLabelsTaskRunActivity

Cette activité comporte les exceptions suivantes :

- EntityNotFoundException (400)
  - « Impossible de trouver MLTransform dans le compte [AccountID] avec le descripteur [TransformName]. »
  - No labelset exists for transformId [transformId] in account id [accountId] (Aucun jeu d'étiquettes n'existe pour l'ID de transformation [ID de transformations] dans l'ID de compte [ID de compte])

  Nouvelle tentative possible : non.
- InvalidInputException (400)
  - [message]
  - S3 path provided is not in the same region as transform Expecting region - [region], but got - [region] (Le chemin S3 fourni n'est pas dans la même région que la transformation. La région attendue est [région], mais la région obtenue correspond à [région])

  Nouvelle tentative possible : non.

StartImportLabelsTaskRunActivity

Cette activité comporte les exceptions suivantes :

- EntityNotFoundException (400)
  - « Impossible de trouver MLTransform dans le compte [AccountID] avec le descripteur [TransformName]. »

  Nouvelle tentative possible : non.
- InvalidInputException (400)
  - [message]
  - Invalid label file path (Chemin d'accès non valide au fichier d'étiquettes)
  - Cannot access the label file at [labelPath]. [message] (Impossible d'accéder au fichier d'étiquettes sous [Chemin]. [message])
  - Cannot use IAM role provided in the transform Role: [role] (Impossible d'utiliser le rôle IAM fourni dans la transformation. Rôle : [rôle])
  - Invalid label file of size 0 (Fichier d'étiquettes non valide avec la taille 0)

- S3 path provided is not in the same region as transform Expecting region - [region], but got - [region] (Le chemin S3 fourni n'est pas dans la même région que la transformation. La région attendue est [région], mais la région obtenue correspond à [région])

Nouvelle tentative possible : non.

- ResourceNumberLimitExceededException (400)
  - Label file has exceeded the limit of [limit] MB (Le fichier d'étiquettes a dépassé la limite de [nombre] Mo)

Nouvelle tentative possible : non. Envisagez de fragmenter votre fichier d'étiquettes en plusieurs fichiers de petite taille.

Démarrer MLEvaluation TaskRunActivity

Cette activité comporte les exceptions suivantes :

- EntityNotFoundException (400)
  - « Impossible de trouver MLTransform dans le compte [AccountID] avec le descripteur [TransformName]. »

Nouvelle tentative possible : non.

- InvalidInputException (400)
  - Exactly one input record table must be provided (Une seule table d'enregistrement en entrée doit être fournie)
  - Should specify database name (Doit spécifier le nom de la base de données)
  - Should specify table name (Doit spécifier le nom de la table)
  - Find Matches parameters have not been set (Les paramètres de recherche de correspondances n'ont pas été définis)
  - A primary key must be specified in Find Matches parameters (Une clé primaire doit être spécifiée dans les paramètres de recherche de correspondances)

Nouvelle tentative possible : non.

- MLTransformNotReadyException (400)
  - This operation can only be applied to a transform that is in a READY state (Cette opération ne peut être appliquée qu'à une transformation dont l'état est READY)

Nouvelle tentative possible : non.

- InternalServiceException (500)
  - Dependency failure (Échec de la dépendance)

  Nouvelle tentative possible : oui.
- ConcurrentRunsExceededException (400)
  - ML Task Runs count [count] has exceeded the transform limit of [limit] task runs (Le nombre d'exécutions de tâches ML [nombre limite] a dépassé la limite de transformations de [nombre limite] exécutions de tâches)
  - ML Task Runs count [count] has exceeded the limit of [limit] task runs (Le nombre d'exécutions de tâches ML [nombre] a dépassé la limite de [nombre limite] exécutions de tâches)

  Nouvelle tentative possible : oui, une fois l'exécution de la tâche terminée.

Démarrer MLLabeling SetGenerationTaskRunActivity

Cette activité comporte les exceptions suivantes :

- EntityNotFoundException (400)
  - « Impossible de trouver MLTransform dans le compte [AccountID] avec le descripteur [TransformName]. »

  Nouvelle tentative possible : non.
- InvalidInputException (400)
  - Exactly one input record table must be provided (Une seule table d'enregistrement en entrée doit être fournie)
  - Should specify database name (Doit spécifier le nom de la base de données)
  - Should specify table name (Doit spécifier le nom de la table)
  - Find Matches parameters have not been set (Les paramètres de recherche de correspondances n'ont pas été définis)
  - A primary key must be specified in Find Matches parameters (Une clé primaire doit être spécifiée dans les paramètres de recherche de correspondances)

  Nouvelle tentative possible : non.
- InternalServiceException (500)

- Dependency failure (Échec de la dépendance)

Nouvelle tentative possible : oui.

- ConcurrentRunsExceededException (400)
  - ML Task Runs count [count] has exceeded the transform limit of [limit] task runs (Le nombre d'exécutions de tâches ML [nombre limite] a dépassé la limite de transformations de [nombre limite] exécutions de tâches)

Nouvelle tentative possible : oui, une fois les exécutions de tâches terminées.

MLTransformActivité de mise à jour

Cette activité comporte les exceptions suivantes :

- EntityNotFoundException (400)
  - « Impossible de trouver MLTransform dans le compte [AccountID] avec le descripteur [TransformName]. »

Nouvelle tentative possible : non.

- InvalidInputException (400)
  - Another transform with name [transformName] already exists (Une autre transformation appelée [Nom de la transformation] existe déjà)
  - [message]
  - Transform name cannot be blank (Le nom de la transformation doit être renseigné)
  - « Impossible de définir la capacité maximale et le Num/Type nombre de travailleurs en même temps. »
  - « Les deux WorkerType et NumberOfWorkers doivent être réglés. »
  - « MaxCapacity devrait être >= [minMaxCapacity]. »
  - « NumberOfWorkers devrait être >= [minNumWorkers]. »
  - Max retries should be non-negative (Les tentatives maximales ne peuvent pas être négatives)
  - Internal service failure due to unexpected input (Échec du service interne en raison d'une entrée inattendue)
  - Find Matches parameters have not been set (Les paramètres de recherche de correspondances n'ont pas été définis)

  - A primary key must be specified in Find Matches parameters (Une clé primaire doit être spécifiée dans les paramètres de recherche de correspondances)

  Nouvelle tentative possible : non.

- AlreadyExistsException (400)
  - Transform with name [transformName] already exists (Une transformation appelée [Nom de la transformation] existe déjà)

  Nouvelle tentative possible : non.

- IdempotentParameterMismatchException (400)
  - Idempotent create request for transform [transformName] had mismatching parameters (Les paramètres de la demande de création idempotente ne correspondent pas)

  Nouvelle tentative possible : non.

AWS Glue quotas

Vous pouvez nous contacter AWS Support pour demander une augmentation des quotas de service répertoriés dans le Références générales AWS. Sauf indication contraire, chaque quota est spécifique à une région. Pour plus d'informations, consultez Points de terminaison et quotas AWS Glue.

Améliorer les AWS Glue performances

Stratégie de base pour le réglage des performances

Afin d'améliorer les AWS Glue performances, vous pouvez envisager de mettre à jour certains AWS Glue paramètres liés aux performances. Lors de la préparation du réglage des paramètres, tenez compte des bonnes pratiques suivantes :

- Déterminez vos objectifs de performance avant de commencer à identifier les problèmes.
- Utilisez des métriques pour identifier les problèmes avant de tenter de modifier les paramètres de réglage.

Pour obtenir des résultats plus cohérents lors du réglage d'une tâche, élaborez une stratégie de base pour votre travail de réglage.

En général, le réglage des performances est effectué dans le flux de travail suivant :

1. Déterminez les objectifs de performance.
2. Mesurez les métriques.
3. Identifiez les goulots d'étranglement.
4. Réduisez l'impact des goulots d'étranglement.
5. Répétez les étapes 2 à 4 jusqu'à ce que vous atteigniez l'objectif prévu.

Réglage de stratégies pour votre type de tâche

Jobs Spark : suivez les instructions de la section Meilleures pratiques pour le réglage AWS Glue des performances des tâches Apache Spark sur AWS Prescriptive Guidance.

Autres tâches : vous pouvez optimiser AWS Glue les tâches shell Ray et AWS Glue Python en adaptant les stratégies disponibles dans d'autres environnements d'exécution.

Amélioration des performances AWS Glue pour les tâches Apache Spark

Afin d'améliorer les performances AWS Glue de Spark, vous pouvez envisager de mettre à jour certains paramètres liés aux performances AWS Glue et Spark.

Pour plus d'informations sur les stratégies spécifiques permettant d'identifier les goulots d'étranglement à l'aide de métriques et de réduire leur impact, consultez la section Meilleures pratiques en matière de réglage AWS Glue des performances pour les tâches Apache Spark sur AWS Prescriptive Guidance. Ce guide présente les principaux sujets applicables à Apache Spark dans tous les environnements d'exécution, comme l'architecture Spark et les jeux de données distribués résilients. À l'aide de ces rubriques, le guide vous explique comment mettre en œuvre des stratégies spécifiques d'optimisation des performances, telles que l'optimisation des réorganisations et la mise en parallèle des tâches.

Vous pouvez identifier les goulots d'étranglement en configurant AWS Glue pour afficher l'interface utilisateur de Spark. Pour de plus amples informations, veuillez consulter the section called "Surveillance à l'aide de l'interface utilisateur Spark".

En outre, AWS Glue fournit des fonctionnalités de performance qui peuvent être applicables au type spécifique de magasin de données auquel votre tâche se connecte. Vous trouverez des informations de référence sur les paramètres de performance pour les entrepôts de données dans the section called "Paramètres de connexion".

Optimisation des lectures avec pushdown dans AWS Glue ETL

Le pushdown est une technique d'optimisation qui rapproche la logique de récupération des données de la source de vos données. La source peut être une base de données ou un système de fichiers tel qu'Amazon S3. Lorsque vous exécutez certaines opérations directement sur la source, vous pouvez économiser du temps et de la puissance de traitement en ne transférant pas toutes les données via le réseau vers le moteur Spark géré par AWS Glue.

Autrement dit, la poussée réduit l'analyse des données. Pour plus d'informations sur le processus permettant d'identifier le moment où cette technique est appropriée, consultez la section Réduire la quantité de données analysées dans le guide des meilleures pratiques pour le réglage des performances de AWS Glue pour les tâches Apache Spark sur le guide AWS prescriptif.

Pushdown de prédicat sur les fichiers stockés dans Amazon S3

Lorsque vous travaillez sur des fichiers organisés par préfixe sur Amazon S3, vous pouvez filtrer vos chemins Amazon S3 cibles en définissant un pushdown de prédicat. Plutôt que de lire le jeu de données complet et d'appliquer des filtres dans un `DynamicFrame`, vous pouvez appliquer directement le filtre aux métadonnées de partition stockées dans le catalogue de données AWS . Cette approche vous permet de lister et de lire de manière sélective uniquement les données

nécessaires. Pour plus d'informations sur ce processus, y compris l'écriture dans un compartiment par partitions, consultez the section called "Gestion des partitions".

Le pushdown de prédicats dans Amazon S3 s'effectue à l'aide du paramètre `push_down_predicate`. Prenons l'exemple d'un compartiment dans Amazon S3 que vous avez partitionné par année, mois et jour. Si vous souhaitez récupérer les données clients pour juin 2022, vous pouvez demander à AWS Glue de ne lire que les chemins Amazon S3 pertinents. Dans ce cas, le `push_down_predicate` est `year='2022' and month='06'`. En mettant tout cela ensemble, l'opération de lecture peut être réalisée comme suit :

Python

```
customer_records = glueContext.create_dynamic_frame.from_catalog(
    database = "customer_db",
    table_name = "customer_tbl",
    push_down_predicate = "year='2022' and month='06'"
)
```

Scala

```
val customer_records = glueContext.getCatalogSource(
database="customer_db",
tableName="customer_tbl",
pushDownPredicate="year='2022' and month='06'"
).getDynamicFrame()
```

Dans le scénario précédent, `push_down_predicate` récupère une liste de toutes les partitions du catalogue de données AWS Glue et les filtre avant de lire les fichiers Amazon S3 sous-jacents. Bien que cela soit utile dans la plupart des cas, lorsque vous travaillez avec des jeux de données contenant des millions de partitions, le processus de liste des partitions peut prendre beaucoup de temps. Pour résoudre ce problème, le nettoyage des partitions côté serveur peut être utilisé pour améliorer les performances. Cela se fait en créant un index de partition pour vos données dans le catalogue de données AWS Glue. Pour plus d'informations sur les index de partition, consultez the section called "Création d'index de partition". Vous pouvez ensuite utiliser l'option `catalogPartitionPredicate` pour référencer l'index. Pour un exemple de récupération de partitions avec `catalogPartitionPredicate`, consultez the section called "Prédicats de partition de catalogue".

Pushdown lors de l'utilisation de sources JDBC

Le lecteur JDBC AWS Glue utilisé dans le pushdown prend en `GlueContext` charge les bases de données prises en charge en fournissant des requêtes SQL personnalisées qui peuvent être exécutées directement sur la source. Cela peut être réalisé en configurant le paramètre `sampleQuery`. Votre exemple de requête peut spécifier les colonnes à sélectionner et fournir un pushdown de prédicat pour limiter les données transférées vers le moteur Spark.

Par défaut, les exemples de requêtes fonctionnent sur un seul nœud, ce qui peut entraîner des échecs lors du traitement de volumes de données importants. Pour utiliser cette fonctionnalité afin d'interroger des données à grande échelle, vous devez configurer le partitionnement des requêtes en définissant `enablePartitioningForSampleQuery` sur true, ce qui distribue la requête à plusieurs nœuds via une clé de votre choix. Le partitionnement des requêtes nécessite également quelques autres paramètres de configuration. Pour plus d'informations sur le partitionnement des requêtes, consultez the section called "Lecture en parallèle à partir de JDBC".

Lors de la configuration`enablePartitioningForSampleQuery`, AWS Glue combinera votre prédicat pushdown avec un prédicat de partitionnement lors de l'interrogation de votre base de données. Vous `sampleQuery` devez terminer par un `AND` for AWS Glue pour ajouter des conditions de partitionnement. (Si vous ne fournissez pas de pushdown de prédicat, `sampleQuery` doit se terminer par un `WHERE`). Vous trouverez ci-dessous un exemple dans lequel nous poussons un prédicat vers le bas pour récupérer uniquement les lignes dont `id` est supérieur à 1 000. Cette `sampleQuery` ne renvoie que les colonnes de nom et d'emplacement pour les lignes où `id` est supérieur à la valeur spécifiée :

Python

```
sample_query = "select name, location from customer_tbl WHERE id>=1000 AND"
customer_records = glueContext.create_dynamic_frame.from_catalog(
    database="customer_db",
    table_name="customer_tbl",
    sample_query = "select name, location from customer_tbl WHERE id>=1000 AND",

    additional_options = {
                           "hashpartitions": 36 ,
                           "hashfield":"id",
                           "enablePartitioningForSampleQuery":True,
                           "sampleQuery":sample_query
                          }
)
```

Scala

```
val additionalOptions = Map(
        "hashpartitions" -> "36",
        "hashfield" -> "id",
        "enablePartitioningForSampleQuery" -> "true",
        "sampleQuery" -> "select name, location from customer_tbl WHERE id >= 1000
 AND"
        )

    val customer_records = glueContext.getCatalogSource(
        database="customer_db",
        tableName="customer_tbl").getDynamicFrame()
```

> **Note**
>
> Si `customer_tbl` porte un nom différent dans votre catalogue de données et votre entrepôt de données sous-jacent, vous devez fournir le nom de la table sous-jacente dans sample_query, car la requête est transmise à l'entrepôt de données sous-jacent.

Vous pouvez également effectuer des requêtes sur des tables JDBC sans intégrer le AWS Glue Data Catalog. Au lieu de fournir un nom d'utilisateur et le mot de passe comme paramètres de la méthode, vous pouvez réutiliser les informations d'identification d'une connexion préexistante en fournissant `useConnectionProperties` et `connectionName`. Dans cet exemple, nous récupérons les informations d'identification à partir d'une connexion appelée `my_postgre_connection`.

Python

```
connection_options_dict = {
    "useConnectionProperties": True,
    "connectionName": "my_postgre_connection",
    "dbtable":"customer_tbl",
    "sampleQuery":"select name, location from customer_tbl WHERE id>=1000 AND",
    "enablePartitioningForSampleQuery":True,
    "hashfield":"id",
    "hashpartitions":36
    }
```

```
customer_records = glueContext.create_dynamic_frame.from_options(
    connection_type="postgresql",
    connection_options=connection_options_dict
    )
```

Scala

```
val connectionOptionsJson = """
      {
        "useConnectionProperties": true,
        "connectionName": "my_postgre_connection",
        "dbtable": "customer_tbl",
        "sampleQuery": "select name, location from customer_tbl WHERE id>=1000 AND",
        "enablePartitioningForSampleQuery" : true,
        "hashfield" : "id",
        "hashpartitions" : 36
      }
    """

    val connectionOptions = new JsonOptions(connectionOptionsJson)

    val dyf = glueContext.getSource("postgresql",
 connectionOptions).getDynamicFrame()
```

Remarques et limites relatives au pushdown dans Glue AWS

Le Pushdown, en tant que concept, est applicable lors de la lecture à partir de sources autres que le streaming. AWS Glue prend en charge une variété de sources. La capacité à appuyer vers le bas dépend de la source et du connecteur.

- Lorsque vous vous connectez à Snowflake, vous pouvez utiliser l'option `query`. Des fonctionnalités similaires existent dans le connecteur Redshift de AWS Glue 4.0 et des versions ultérieures. Pour de plus amples informations sur la lecture à partir de Snowflake avec `query`, consultez the section called "Lire dans Snowflake".
- Le lecteur ETL DynamoDB ne prend pas en charge les filtres ou les prédicats pushdown. MongoDB et DocumentDB ne prennent pas non plus en charge ce type de fonctionnalité.
- Lors de la lecture de données stockées dans Amazon S3 dans des formats de table ouverts, la méthode de partitionnement des fichiers dans Amazon S3 n'est plus suffisante. Pour lire et écrire

à partir de partitions utilisant des formats de table ouverts, consultez la documentation relative au format.

- DynamicFrame les méthodes n'exécutent pas le pushdown de la projection Amazon S3. Toutes les colonnes seront lues à partir des fichiers qui satisfont au filtre du prédicat.
- Lorsque vous travaillez avec des `custom.jdbc` connecteurs dans AWS Glue, la possibilité d'appuyer vers le bas dépend de la source et du connecteur. Consultez la documentation du connecteur approprié pour vérifier si et comment il prend en charge le pushdown dans AWS Glue.

Utilisation d'Auto Scaling pour AWS Glue

Autoscaling est disponible pour vos tâches ETL, de sessions interactives et de streaming AWS Glue avec la version 3.0 d'AWS Glue ou une version plus récente.

Lorsque Auto Scaling est activé, vous bénéficiez des avantages suivants :

- AWS Glue ajoute et supprime automatiquement les employés du cluster en fonction du parallélisme à chaque étape ou micro-lot de la tâche exécutée.
- Il réduit le besoin pour vous d'expérimenter et de décider du nombre d'employés à affecter à vos tâches AWS Glue ETL.
- Avec le nombre maximal de travailleurs donné, AWS Glue choisira les ressources de taille appropriée pour la charge de travail.
- Vous pouvez voir comment la taille du cluster change pendant l'exécution de la tâche en consultant les CloudWatch indicateurs sur la page de détails de l'exécution de la tâche dans AWS Glue Studio.

Autoscaling pour les tâches AWS Glue ETL et de streaming permet une augmentation et une réduction horizontales à la demande des ressources de calcul de vos tâches AWS Glue. L'augmentation d'échelle à la demande vous aide à allouer uniquement les ressources de calcul requises initialement au démarrage de l'exécution de la tâche, ainsi qu'à allouer les ressources requises en fonction de la demande pendant la tâche.

Autoscaling prend également en charge la réduction horizontale dynamique des ressources de la tâche AWS Glue pendant le déroulement d'une tâche. Au cours de l'exécution d'une tâche, lorsque plus d'exécuteurs sont demandés par votre application Spark, plus d'employés seront ajoutés au cluster. Lorsque l'exécuteur est inactif sans tâches de calcul actives, il sera supprimé de même que l'employé correspondant.

Les scénarios courants dans lesquels Autoscaling contribue à réduire les coûts et l'utilisation de vos applications Spark incluent :

- un pilote Spark répertoriant un grand nombre de fichiers dans Amazon S3 ou effectuant un chargement alors que les exécuteurs sont inactifs ;
- des étapes Spark exécutées avec seulement quelques exécuteurs en raison d'un surprovisionnement ;
- des distorsions de données ou une demande de calcul inégale entre les étapes Spark.

Exigences

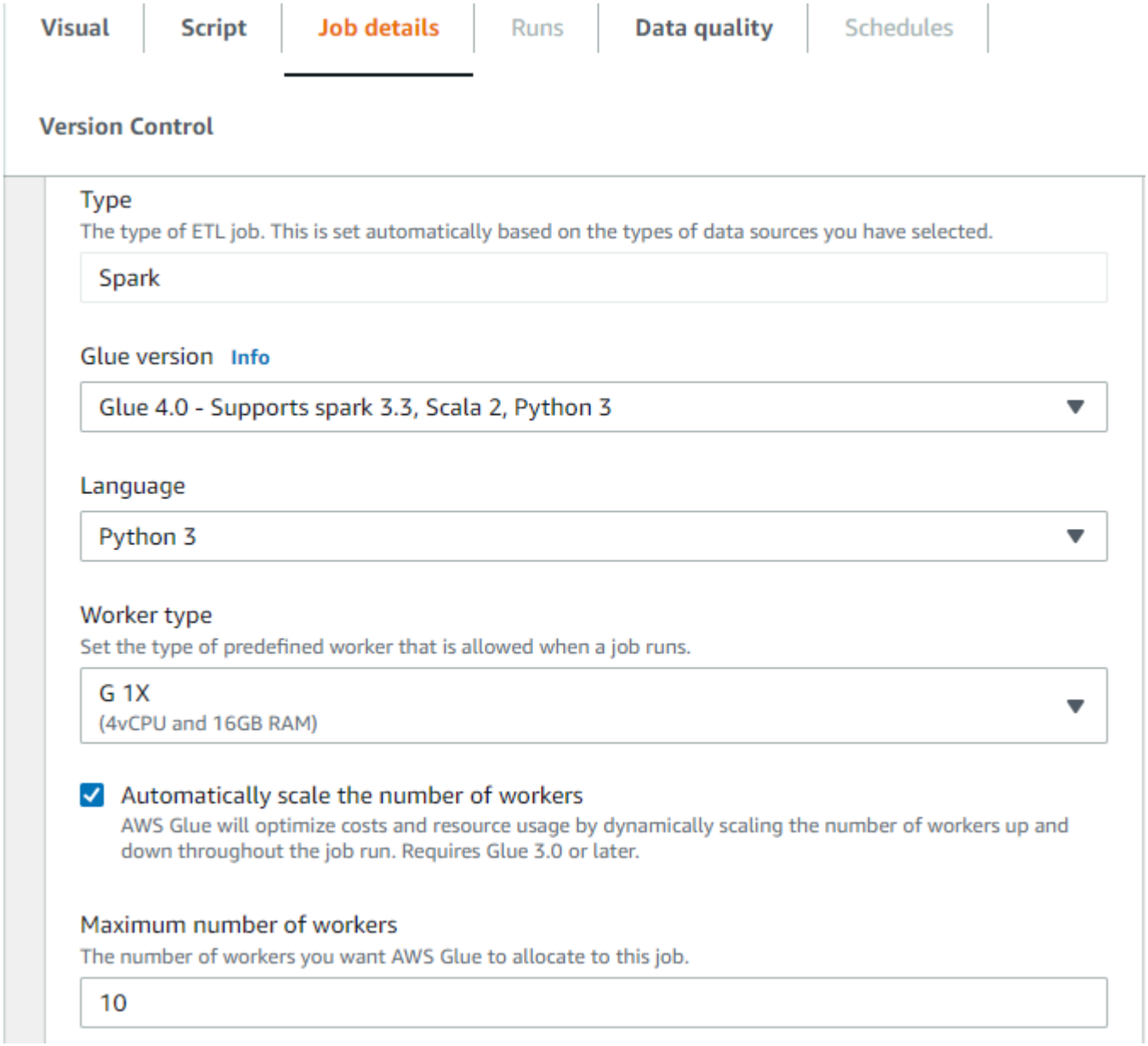
Auto Scaling n'est disponible que pour la version 3.0 ou ultérieure de AWS Glue. Pour utiliser Auto Scaling, vous pouvez suivre le guide de migration afin de migrer vos tâches existantes vers la version 3.0 ou ultérieure de AWS Glue ou de créer de nouvelles tâches avec la version 3.0 ou ultérieure de AWS Glue.

Autoscaling est disponible pour les tâches AWS Glue ayant les types de travailleurs `G.1X`, `G.2X`, `G.4X`, `G.8X`, `G.12X`, `G.16X`, `R.1X`, `R.2X`, `R.4X`, `R.8X` ou `G.025X` (uniquement pour les tâches Streaming). DPUs Les standards ne sont pas pris en charge pour Auto Scaling.

Activation d'Auto Scaling dans AWS Glue Studio

Dans l'onglet Informations de la tâche d'AWS Glue Studio, choisissez le type Spark ou Spark Streaming, et dans le champ Version de Glue, choisissez **`Glue 3.0`** ou version ultérieure. Ensuite, une case à cocher apparaît sous Type de travailleur.

- Sélectionnez l'option Automatically scale the number of workers (Mise à l'échelle automatique du nombre d'employés).
- Configurez le Nombre maximal d'employés pour définir le nombre maximal d'employés pouvant être transférés à l'exécution de la tâche.



Activation de l'Auto Scaling avec la AWS CLI ou le SDK

Pour activer Auto Scaling à partir de la AWS CLI pour l'exécution de votre tâche, exécutez `start-job-run` la configuration suivante :

```
{
    "JobName": "<your job name>",
    "Arguments": {
        "--enable-auto-scaling": "true"
    },
```

```
    "WorkerType": "G.2X", // G.1X, G.2X, G.4X, G.8X, G.12X, G.16X, R.1X, R.2X, R.4X,
 and R.8X are supported for Auto Scaling Jobs
    "NumberOfWorkers": 20, // represents Maximum number of workers
    ...other job run configurations...
}
```

Une fois l'exécution de la tâche ETL terminée, vous pouvez également appeler `get-job-run` pour vérifier l'utilisation réelle des ressources de la tâche exécutée en secondes DPU. Remarque : le nouveau champ n'DPUSecondsapparaîtra que pour vos tâches par lots dans la AWS Glue version 4.0 ou ultérieure activée avec Auto Scaling. Ce champ n'est pas pris en charge pour les tâches de streaming.

```
$ aws glue get-job-run --job-name your-job-name --run-id jr_xx --endpoint https://
glue.us-east-1.amazonaws.com --region us-east-1
{
    "JobRun": {
        ...
        "GlueVersion": "3.0",
        "DPUSeconds": 386.0
    }
}
```

Vous pouvez également configurer des exécutions de tâches avec Auto Scaling à l'aide du kit SDK AWS Glue en suivant la même configuration.

Activation d'Autoscaling avec des sessions interactives

Pour activer Auto Scaling lors de la création de AWS Glue tâches avec des sessions interactives, consultez la section Configuration des sessions AWS Glue interactives.

Conseils et considérations

Conseils et points à prendre en compte pour peaufiner AWS Glue Auto Scaling :

- Si vous n'avez aucune idée de la valeur initiale du nombre maximum de travailleurs, vous pouvez commencer par le calcul approximatif expliqué dans Estimation du AWS Glue DPU. Vous ne devez pas configurer de valeur extrêmement élevée dans le nombre maximum de travailleurs pour de très faibles volumes de données.
- AWS Glue Auto Scaling configure `spark.sql.shuffle.partitions` et `spark.default.parallelism` se base sur le nombre maximum de DPU (calculé en fonction du

nombre maximum de travailleurs et du type de travailleur) configurés sur la tâche. Si vous préférez la valeur fixe pour ces configurations, vous pouvez remplacer ces paramètres par les paramètres de tâche suivants :

- Clé : `--conf`
- Value (Valeur) : `spark.sql.shuffle.partitions=200 --conf spark.default.parallelism=200`

- Pour les tâches de streaming, par défaut, la mise à l'échelle automatique AWS Glue ne s'effectue pas au sein de microlots et plusieurs microlots sont nécessaires pour lancer la mise à l'échelle automatique. Si vous souhaitez activer l'autoscaling dans le cadre de microlots, indiquez `--auto-scale-within-microbatch`. Pour plus d'informations, consultez Job parameter reference.

Surveillance d'Auto Scaling à l'aide CloudWatch des métriques Amazon

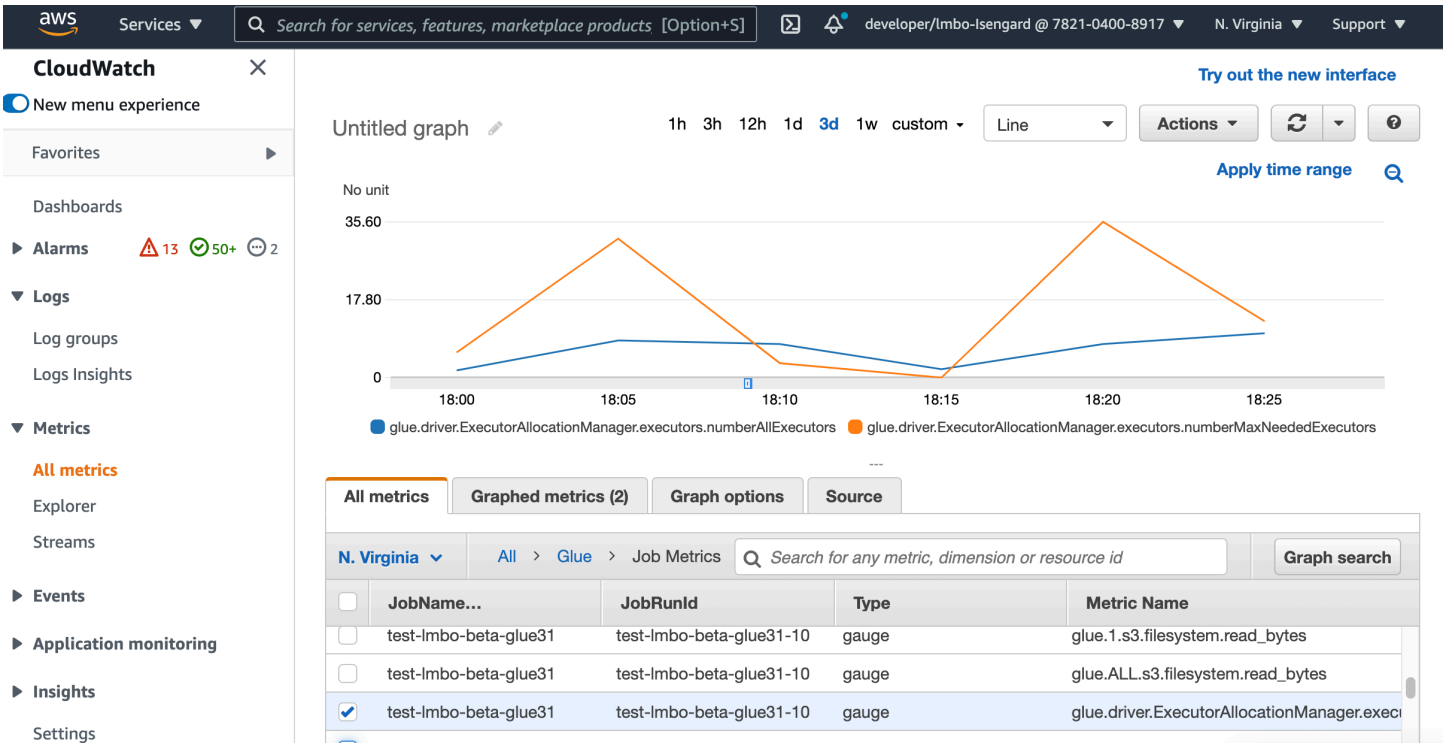
Les métriques de l' CloudWatch exécuteur sont disponibles pour vos tâches AWS Glue 3.0 ou ultérieures si vous activez Auto Scaling. Les métriques peuvent être utilisées pour contrôler la demande et l'utilisation optimisée des exécuteurs dans leurs applications Spark activées avec Auto Scaling. Pour de plus amples informations, veuillez consulter Surveillance AWS Glue à l'aide CloudWatch des métriques Amazon.

Vous pouvez également utiliser des indicateurs AWS Glue d'observabilité pour obtenir des informations sur l'utilisation des ressources. Par exemple, en surveillant `glue.driver.workerUtilization`, vous pouvez contrôler la quantité de ressources réellement utilisée avec ou sans autoscaling. Autre exemple, en surveillant `glue.driver.skewness.job` et `glue.driver.skewness.stage`, vous pouvez voir à quel point les données sont faussées. Ces informations vous aideront à décider d'activer l'autoscaling et d'optimiser les configurations. Pour plus d'informations, consultez la section Surveillance avecSurveillance à l'aide de métriques d'observabilité AWS Glue.

- colle.driver. ExecutorAllocationManager.exécuteurs. numberAllExecutors
- colle.driver. ExecutorAllocationManager.exécuteurs. numberMaxNeededExécuteurs

Pour en savoir plus sur ces métriques, consultez Surveillance de la planification des capacités de DPU.

> **Note**
>
> CloudWatch les métriques de l'exécuteur ne sont pas disponibles pour les sessions interactives.

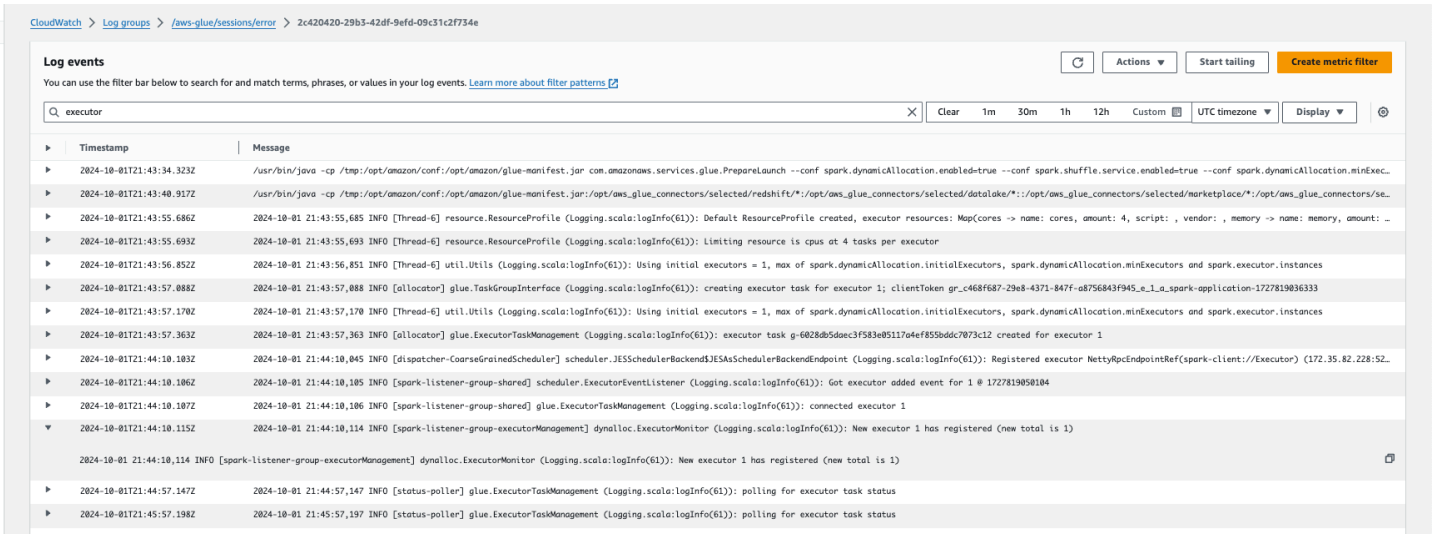


Surveillance de l'Auto Scaling avec Amazon CloudWatch Logs

Si vous utilisez des sessions interactives, vous pouvez surveiller le nombre d'exécuteurs en activant Amazon CloudWatch Logs en continu et en recherchant le terme « exécuteur » dans les journaux, ou en utilisant l'interface utilisateur Spark. Pour ce faire, utilisez la commande magique `%%configure` pour activer la journalisation continue avec `enable auto scaling`.

```
%%configure{
    "--enable-continuous-cloudwatch-log": "true",
    "--enable-auto-scaling": "true"
}
```

Dans les CloudWatch événements Amazon Logs, recherchez « exécuteur » dans les journaux :

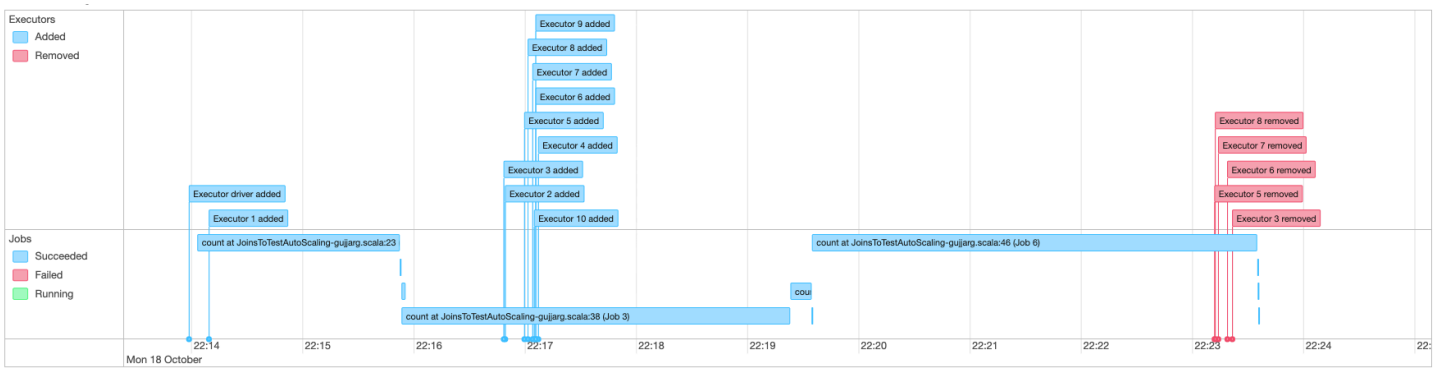


Surveillance de Auto Scaling avec Spark UI

Lorsqu'Auto Scaling est activé, vous pouvez également surveiller l'ajout et la suppression d'exécuteurs à l'aide de l'augmentation et de la réduction en fonction de la demande de vos tâches AWS Glue à l'aide de la Glue Spark UI. Pour de plus amples informations, veuillez consulter Activation de l'interface utilisateur web Apache Spark pour les tâches AWS Glue.

Lorsque vous utilisez des sessions interactives depuis le bloc-notes Jupyter, vous pouvez exécuter la commande magique suivante pour activer l'autoscaling avec l'interface utilisateur Spark :

```
%%configure{
    "--enable-auto-scaling": "true",
    "--enable-continuous-cloudwatch-log": "true"
}
```



Surveillance de l'utilisation du DPU d'exécution de la tâche Auto Scaling

Vous pouvez utiliser AWS Glue Studio Job run view pour vérifier l'utilisation du DPU de vos tâches Auto Scaling.

1. Choisissez Monitoring dans le volet AWS Glue Studio de navigation. La page Surveillance apparaît.
2. Faites défiler la page jusqu'à atteindre le graphique Exécutions de tâche.
3. Accédez à l'exécution de la tâche qui vous intéresse et faites défiler la page jusqu'à atteindre la colonne « heures » du DPU pour vérifier l'utilisation de l'exécution de la tâche spécifique.

Limitations

AWS Gluestreaming Auto Scaling ne prend actuellement pas en charge une DataFrame jointure de streaming avec une statique DataFrame créée en dehors de`ForEachBatch`. Une statique DataFrame créée à l'intérieur du `ForEachBatch` fonctionnera comme prévu.

Partitionnement de la charge de travail avec exécution limitée

Les erreurs dans les applications Spark proviennent généralement de scripts Spark inefficaces, d'exécution distribuée en mémoire de transformations à grande échelle et d'anomalies des jeux de données. Il existe de nombreuses raisons qui peuvent causer des problèmes de mémoire de pilote ou de l'exécuteur, par exemple une asymétrie de données, l'énumération d'un trop grand nombre d'objets ou de grands brassages de données. Ces problèmes apparaissent souvent lorsque vous traitez d'énormes quantités de données en attente avec Spark.

AWS Glue vous permet de résoudre les problèmes d'OOM (Out Of Memory) et de faciliter le traitement de votre ETL grâce au partitionnement de la charge de travail. Lorsque le partitionnement de la charge de travail est activé, chaque exécution de travail ETL ne sélectionne que les données non traitées, avec une limite supérieure sur la taille du jeu de données ou le nombre de fichiers à traiter avec cette exécution de tâche. Les futures exécutions de tâches traiteront les données restantes. Par exemple, si 1 000 fichiers doivent être traités, vous pouvez définir le nombre de fichiers à 500 et les séparer en deux exécutions de tâche.

Le partitionnement de charge de travail est pris en charge uniquement pour les sources de données Amazon S3.

Activation du partitionnement de la charge de travail

Vous pouvez activer l'exécution limitée en définissant manuellement les options dans votre script ou en ajoutant des propriétés de table de catalogue.

Pour activer le partitionnement de la charge de travail avec une exécution limitée dans votre script :

1. Pour éviter le retraitement des données, activez les signets de tâche dans la nouvelle tâche ou celle existante. Pour plus d'informations, veuillez consulter la rubrique Suivi des données traitées à l'aide des signets de tâches.
2. Modifiez votre script et définissez la limite limitée dans les options supplémentaires de l'API AWS Glue `getSource`. Vous devez également définir le contexte de transformation du signet de tâche pour stocker l'élément `state`. Par exemple :

   Python

   ```
   glueContext.create_dynamic_frame.from_catalog(
       database = "database",
       table_name = "table_name",
       redshift_tmp_dir = "",
       transformation_ctx = "datasource0",
       additional_options = {
           "boundedFiles" : "500", # need to be string
         # "boundedSize" : "1000000000" unit is byte
       }
   )
   ```

   Scala

   ```
   val datasource0 = glueContext.getCatalogSource(
       database = "database", tableName = "table_name", redshiftTmpDir = "",
       transformationContext = "datasource0",
       additionalOptions = JsonOptions(
           Map("boundedFiles" -> "500") // need to be string
         //"boundedSize" -> "1000000000" unit is byte
       )
   ).getDynamicFrame()
   ```

   ```
   val connectionOptions = JsonOptions(
       Map("paths" -> List(baseLocation), "boundedFiles" -> "30")
   ```

```
)
val source = glueContext.getSource("s3", connectionOptions, "datasource0", "")
```

Pour activer le partitionnement de la charge de travail avec une exécution limitée dans votre table de catalogue de données :

1. Définissez les paires clé-valeur dans le champ `parameters` de votre structure de table dans le catalogue des données. Pour en savoir plus, veuillez consulter la rubrique Affichage et modification des détails de table.
2. Définir la limite supérieure de la taille du jeu de données ou du nombre de fichiers traités :
   - Définir `boundedSize` à la taille cible du jeu de données, en octets. L'exécution de la tâche s'arrêtera après avoir atteint la taille cible de la table.
   - Définir `boundedFiles` au nombre cible de fichiers. L'exécution de la tâche s'arrêtera après le traitement du nombre cible de fichiers.

> **Note**
>
> Vous ne devez définir que `boundedSize` ou `boundedFiles`, car une seule limite est prise en charge.

Configuration d'un déclencheur AWS Glue pour exécuter automatiquement la tâche

Une fois que vous avez activé les limites d'exécution, vous pouvez configurer un déclencheur AWS Glue pour exécuter automatiquement la tâche et charger de manière incrémentielle les données dans des exécutions séquentielles. Accédez à la console AWS Glue et créez un déclencheur, configurez l'heure de planification et attachez-le à votre tâche. Ensuite, il déclenchera automatiquement l'exécution de la tâche suivante et traitera le nouveau lot de données.

Vous pouvez également utiliser les flux de travail AWS Glue pour orchestrer plusieurs tâches afin de traiter les données de différentes partitions en parallèle. Pour de plus amples informations, veuillez consulter les rubriques Déclencheurs AWS Glue et Flux de travail AWS Glue.

Pour plus d'informations sur les cas d'utilisation et les options, veuillez consulter le blog Optimisation des applications Spark avec partitionnement de la charge de travail dans AWS Glue.

Problèmes connus liés à AWS Glue

Prenez en considération les problèmes connus suivants pour AWS Glue.

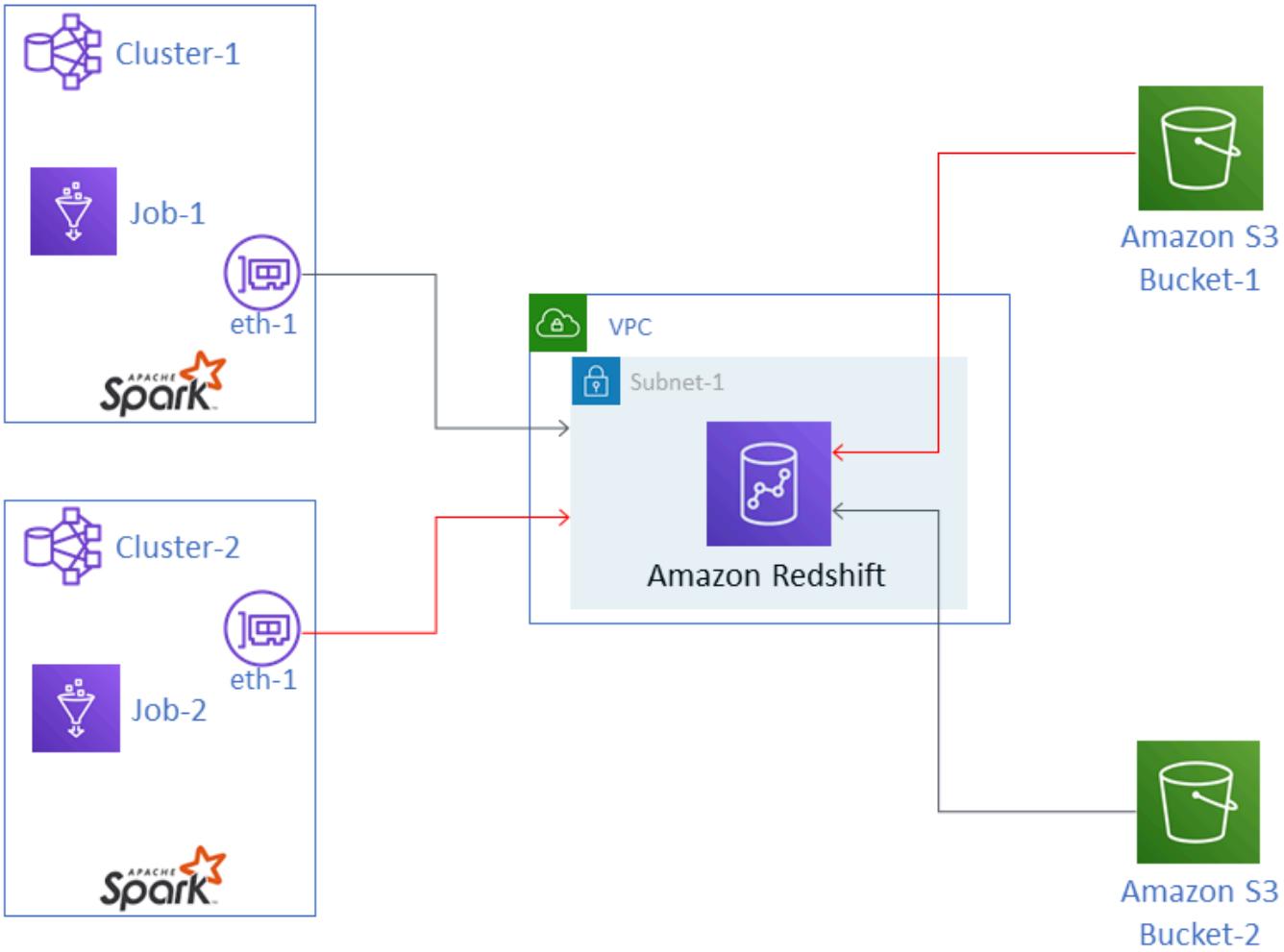
Rubriques

- Interdiction d'accès aux données inter-tâches

Interdiction d'accès aux données inter-tâches

Imaginons le cas où vous avez deux tâches AWS Glue Spark dans un même AWS compte, chacune exécutée dans un cluster AWS Glue Spark distinct. Les tâches utilisent des connexions AWS Glue pour accéder aux ressources dans le même VPC (Virtual Private Cloud). Dans ce cas, une tâche s'exécutant dans un cluster peut accéder aux données à partir de la tâche s'exécutant dans l'autre cluster.

Le schéma suivant illustre une telle situation.



Dans le schéma, AWS Glue `Job-1` s'exécute dans `Cluster-1` et Job-2 s'exécute dans `Cluster-2`. Les deux tâches fonctionnent avec la même instance Amazon Redshift, qui réside dans le sous-réseau `Subnet-1` d'un VPC. `Subnet-1` peut être un sous-réseau public ou privé.

`Job-1`transforme les données d'Amazon Simple Storage Service (Amazon `Bucket-1` S3) et les écrit sur Amazon Redshift. `Job-2`fait de même avec les données en entrée`Bucket-2`. `Job-1`utilise le rôle Gestion des identités et des accès AWS (IAM) `Role-1` (non illustré), qui donne accès à`Bucket-1`. `Job-2`utilise `Role-2` (non illustré), qui donne accès à`Bucket-2`.

Ces tâches comportent des chemins réseau qui leur permettent de communiquer avec les clusters des autres tâches et donc d'accéder aux données de ces dernières. Par exemple, `Job-2` peut accéder aux données dans `Bucket-1`. Dans le schéma, le chemin en rouge illustre cet accès.

Pour éviter cette situation, nous vous recommandons d'attacher des configurations de sécurité différentes à `Job-1` et `Job-2`. Lorsque vous attachez les configurations de sécurité, l'accès inter-tâches aux données est bloqué grâce aux certificats créés par AWS Glue. Les configurations de sécurité peuvent être des configurations factices. En d'autres termes, vous pouvez créer les configurations de sécurité sans activer le chiffrement des données Amazon S3, des CloudWatch données Amazon ou des signets de tâches. Les trois options de chiffrement peuvent être désactivées.

Pour de plus amples informations sur les configurations de sécurité, veuillez consulter the section called “Chiffrement des données écrites par AWS Glue”.

Pour attacher une configuration de sécurité à une tâche

1. Ouvrez la AWS Glue console à l'adresse https://console.aws.amazon.com/glue/.
2. Sur la page Configure the job properties (Configurer les propriétés de la tâche) de la tâche, développez la section Paramètres de configuration de sécurité, des bibliothèques de scripts et des tâches.
3. Sélectionnez une configuration de sécurité dans la liste.

Historique de la documentation pour AWS Glue

Modification	Description	Date
AWS Glue for Ray ne sera plus ouvert aux nouveaux clients à compter du 30 avril 2026.	AWS Glue for Ray ne sera plus ouvert aux nouveaux clients à compter du 30 avril 2026. Si vous souhaitez utiliser AWS Glue for Ray, inscrivez-vous avant cette date. Les clients existants peuvent continuer à utiliser le service normalement. Pour plus d'informations, consultez la section AWS Glue relative à la fin du support de Ray.	31 mars 2026
Support pour AWS Glue la version 5.1	Ajout d'informations sur le support de AWS Glue la version 5.1. Les fonctionn alités incluent la mise à jour d'Apache Spark vers la version 3.5.6, les mises à jour du format de table ouvert, les vues matérialisées Iceberg, la version 3 du format Iceberg, les lectures et écritures avec accès complet aux tables Hudi, le contrôle d'accès fin natif de Spark DDL/DML sur les tables Hive, Iceberg et Delta Lake, etc. Pour plus d'informations, consultez les notes AWS Glue de publicati on et la migration des AWS	26 novembre 2025

	Glue tâches vers la AWS Glue version 5.1.	
AWS Glue Data Quality prend désormais en charge l'étiquetage des règles pour améliorer les rapports	AWS Glue Data Quality prend désormais en charge l'étiquetage des règles, ce qui vous permet d'organiser et d'analyser les résultats de qualité des données de manière plus efficace. Vous pouvez interroger les résultats à l'aide d'étiquettes spécifiques pour identifier les règles défaillantes au sein de catégories particulières, compter les résultats des règles par équipe ou par domaine et créer des rapports ciblés pour les différentes parties prenantes. Pour plus d'informations, consultez la section Étiquettes.	21 novembre 2025
AWS Glue Data Quality prend désormais en charge les requêtes de prétraitement	AWS Glue Data Quality prend désormais en charge les constantes dans le DQDL, ce qui vous permet de définir des valeurs constantes et de les référencer tout au long de votre script. Cela permet d'éviter les problèmes liés aux limites de taille des requêtes lors de l'utilisation d'instructions SQL volumineuses. Pour plus d'informations, consultez la section Constantes.	21 novembre 2025

Prise en charge des nouveaux types de travailleurs G.12X, G.16X et des types de travailleurs R optimisés pour la mémoire	Ajout de la prise en charge des nouveaux types de travailleurs G.12X, G.16X et des types de travailleurs R.1X, R.2X, R.4X, R.8X optimisés pour la mémoire pour les tâches. AWS Glue Ces nouveaux types de travailleurs fournissent des options de calcul et de mémoire supplémentaires pour répondre aux différentes exigences de charge de travail. Pour plus d'informations, consultez la section Ajout de tâches AWS Glue.	30 janvier 2025
Support pour 14 nouveaux connecteurs natifs SaaS supplémentaires pour AWS Glue	Quatorze connecteurs natifs SaaS supplémentaires ont été ajoutés AWS Glue. Pour plus d'informations, consultez la section Ajout d'une AWS Glue connexion.	30 janvier 2025
Support pour 16 nouveaux connecteurs natifs SaaS supplémentaires pour AWS Glue	Seize connecteurs natifs SaaS supplémentaires ont été ajoutés AWS Glue. Pour plus d'informations, consultez la section Ajout d'une AWS Glue connexion.	17 décembre 2024

Génération automatique de statistiques de colonne	AWS Glue Data Quality prend désormais en charge Amazon SageMaker AI LakeHouse les tables et les tables Iceberg, Delta et HUDI AWS Lake Formation gérées dans Data Catalog et ETL. Pour plus d’informations, consultez AWS Glue Data Quality.	6 décembre 2024
Prise en charge des intégrations zéro ETL	Zero-ETL est un ensemble d'intégrations entièrement gérées AWS qui minimise le besoin de créer des pipelines de données ETL. Pour plus d’informations, consultez Intégrations zéro ETL.	3 décembre 2024
Prise en charge des connexions réutilisables	Un nouveau schéma de AWS Glue connexions fournit un moyen unifié de gérer les connexions de données entre les AWS services et les applications AWS Glue, tels qu'Amazon Athena et Amazon SageMaker Unified Studio. Pour plus d’informations, consultez Connecting to data.	3 décembre 2024

Support pour AWS Glue la version 5.0.	Ajout d'informations sur le support de AWS Glue la version 5.0. Les fonctionn alités incluent une mise à jour d'Apache Spark vers la version 3.52, une mise à jour de Java vers la version 17, des mises à jour du format de table ouvert, le contrôle précis des accès natif de Spark, l'intégration de Sagemaker Lakehouse et de l'abstrac tion de l'entrepôt de données, la prise en charge dans Sagemaker Unified Studio, etc. Pour plus d'informations, consultez les notes AWS Glue de publication et la migration des AWS Glue tâches vers la AWS Glue version 5.0.	3 décembre 2024
Connexion à l' AWS Glue Data Catalog aide du point de AWS Glue terminaison Iceberg REST	Le point de terminaison REST Iceberg d'AWS Glue prend en charge les opération s API spécifiées dans la spécification REST Apache Iceberg. À l'aide d'un client REST Iceberg, vous pouvez connecter votre applicati on exécutée sur un moteur analytique au catalogue REST hébergé dans le catalogue de données. Pour plus d'informa tions, consultez Accessing the Data Catalog.	3 décembre 2024

Génération automatique de statistiques de colonne	Générez automatiquement des statistiques de colonne pour les nouvelles tables dans AWS Glue Data Catalog. Pour plus d'informations, consultez Automatic column statistics generation.	3 décembre 2024
Support pour les mises à niveau de l'IA générative pour Apache Spark dans AWS Glue	Spark Upgrades in AWS Glue permet aux ingénieurs de données et aux développeurs de mettre à niveau et de migrer, de mettre à niveau et de migrer leurs tâches AWS Glue Spark existantes vers les dernières versions de Spark à l'aide de l'IA générative. Pour plus d'informations, consultez Upgrade analysis with AI.	22 novembre 2024
Support pour la résolution des problèmes liés à l'IA générative pour Apache Spark dans AWS Glue	La résolution des problèmes liés à l'IA générative pour les tâches Apache Spark AWS Glue permet aux ingénieurs de données et aux scientifiques de diagnostiquer et de résoudre facilement les problèmes liés à leurs applications Spark. Pour plus d'informations, consultez Troubleshooting Spark jobs with AI.	22 novembre 2024

Prise en charge des optimiseurs Iceberg pour accéder aux compartiments Amazon S3 dans un VPC	AWS Glue Data Catalog prend en charge les optimiseurs de table Iceberg pour accéder aux compartiments Amazon S3 depuis un Virtual Private Cloud (AWS Glue VPC) spécifique via une connexion réseau. Pour plus d’informations, consultez Optimisation des tables Iceberg.	20 novembre 2024
Support pour neuf nouveaux connecteurs natifs SaaS supplémentaires pour AWS Glue	Neuf autres connecteurs natifs SaaS ont été ajoutés AWS Glue. Pour plus d'informations, consultez la section Ajout d'une AWS Glue connexion.	19 novembre 2024
Support de dix nouveaux connecteurs natifs SaaS pour AWS Glue	Dix connecteurs natifs SaaS ont été ajoutés AWS Glue. Pour plus d'informations, consultez la section Ajout d'une AWS Glue connexion.	15 novembre 2024
Support pour la mise en file d'attente pour AWS Glue les tâches ETL	Vous pouvez activer la mise en file d’attente pour exécuter des tâches ultérieurement, lorsqu’elles ne peuvent pas être exécutées immédiatement en raison de Service Quotas. Pour plus d'informations, consultez la section Configuration des propriétés des tâches Spark dans AWS Glue.	3 septembre 2024

Mise à jour des modifications de politique	Modifications documentées des AwsGlueSessionUser RestrictedNotebookServiceRo lepolitiques AwsGlueSe ssionUserRestrictedNotebook Policyet, nécessaires à la prise en charge tag-on-cr eate des sessions avec clé de balise propriétaire. Pour plus d'informations, voir les AWS Glue mises à jour des politique s AWS gérées.	30 août 2024
La détection des anomalies et les règles dynamiques sont désormais généralement disponibles	AWS Glue Data Quality utilise un algorithme d'apprent issage automatique pour tirer des leçons des tendances passées, puis prévoir les valeurs futures afin de détecter les anomalies. Les règles dynamiques vous permettent d'indiquer des seuils dynamiques. Pour plus d'informations, consultez Optimizing query performance for Iceberg tables.	7 août 2024

Mise à jour des modifications de politique	Modifications documentées des AwsGlueSessionUser RestrictedServiceRolepolitiques AwsGlueSessionUser RestrictedPolicyet, nécessaires à la prise en charge tag-on-create des sessions avec clé de balise propriétaire. Pour plus d'informations, voir les AWS Glue mises à jour des politiques AWS gérées.	5 août 2024
La génération de statistiques de colonne pour les tables Iceberg est désormais généralement disponible	AWS Glue prend en charge le calcul et la mise à jour du nombre de valeurs distinctes (NDVs) pour chaque colonne des tables Iceberg. Pour plus d'informations, consultez la section Détection des anomalies dans la section Qualité AWS Glue des données et Règles dynamiques.	9 juillet 2024

Support pour les profils AWS Glue d'utilisation	Les administrateurs peuvent créer des profils AWS Glue d'utilisation pour différent es catégories d'utilisateurs au sein du compte, telles que les développeurs, les testeurs et les équipes produit. Cette flexibilité permet aux administrateurs d'appliquer différents contrôles d'utilisa tion et de coûts pour chaque catégorie d'utilisateurs. Pour plus d'informations, consultez la section Configuration des profils AWS Glue d'utilisation.	18 juin 2024
Support pour un connecteu r Salesforce AWS Glue pour Spark	Ajout d'informations sur un nouveau AWS Glue connecteu r pour Salesforce. Cette fonctionnalité vous permet d'utiliser Spark AWS Glue pour lire et écrire dans Salesforc e dans les versions AWS Glue 4.0 et ultérieures. Pour plus d'informations, consultez Connecting to Salesforce.	22 mai 2024

Intégration des données Amazon Q dans AWS Glue (GA)	L'intégration de données dans Amazon Q AWS Glue est une nouvelle fonctionnalité d'IA générative AWS Glue qui permet aux ingénieurs de données et aux développeurs ETL de créer des tâches d'intégration de données en langage naturel. Les ingénieurs et les développeurs peuvent demander à Q de créer des tâches, de résoudre des problèmes et de répondre à des questions concernant AWS Glue l'intégration des données. Pour plus d'informations, consultez la rubrique Intégration des données Amazon Q dans AWS Glue. Cette fonctionnalité inclut une mise à jour des politiques `AwsGlueSessionUserRestrictedPolicy` `AwsGlueSessionUserRestrictedNotebookServiceRole` , et `AwsGlueSessionUserRestrictedServiceRole` AWS gérées. Pour plus d'informations, voir les AWS Glue mises à jour des politiques AWS gérées.	30 avril 2024

Intégration des données Amazon Q dans AWS Glue (version préliminaire)	L'intégration de données dans Amazon Q AWS Glue est une nouvelle fonctionnalité d'IA générative AWS Glue qui permet aux ingénieurs de données et aux développeurs ETL de créer des tâches d'intégration de données en langage naturel. Les ingénieurs et les développeurs peuvent demander à Q de créer des tâches, de résoudre des problèmes et de répondre à des questions concernant AWS Glue l'intégration des données. Pour plus d'informations, consultez la rubrique Intégration des données Amazon Q dans AWS Glue. Cette fonctionnalité inclut une mise à jour de la politique `AwsGlueSessionUserRestrictedNotebookPolicy` AWS gérée. Pour plus d'informations, voir les AWS Glue mises à jour des politiques AWS gérées.	30 janvier 2024

Mise à jour de la documentation pour le AWS Glue streaming	Ajout d'un nouveau chapitre avec du contenu nouveau et réorganisé pour le AWS Glue streaming. Ce contenu décrit le fonctionnement du streaming AWS Glue, les caractéristiques du traitement des données en temps réel et comment surveiller vos tâches de streaming. Pour plus d’informations, veuillez consulter AWS Glue Streaming.	27 décembre 2023
Prise en charge de la détection détaillée des données sensibles	La transformation Détecter les données sensibles permet de détecter, masquer ou supprimer des entités que vous définissez ou sont prédéfinies par AWS Glue. Les actions détaillées vous permettent en outre d'appliquer une action spécifique par entité. Pour plus d’informations, consultez la rubrique Utilisation d’une détection détaillée des données sensibles.	26 novembre 2023

Support pour le suivi des tâches à l'aide de métriques AWS Glue d'observabilité	Utilisez les métriques AWS Glue d'observabilité pour obtenir des informations sur ce qui se passe dans vos tâches AWS Glue pour Apache Spark afin d'améliorer le triage et l'analyse des problèmes. Pour plus d'informations, consultez la section Surveillance à l'aide de métriques AWS Glue d'observabilité.	26 novembre 2023
Support pour la détection des anomalies dans le domaine de la qualité AWS Glue des données	AWS Glue La détection des anomalies de qualité des données applique des algorithmes d'apprentissage automatique (ML) aux statistiques des données au fil du temps afin de détecter les modèles anormaux et les problèmes cachés de qualité des données difficiles à détecter par le biais de règles. Pour plus d'informations, consultez la section Détection des anomalies dans AWS Glue Data Quality.	26 novembre 2023

Mise à jour du comportement par défaut de journalisation de l’interface utilisateur Spark	Les tâches Spark générant des journaux de l'interface utilisateur Spark seront désormais écrites avec un modèle de nom de fichier différent pour prendre en charge l'interface utilisateur Spark dans la AWS Glue console. Cela ne modifie pas le comportement du CloudWatch journal. Vous pouvez revenir à l’ancien comportement en mettant à jour la configuration de votre tâche. Pour plus d’informations, consultez la rubrique Surveillance des tâches à l’aide de l’interface web d’Apache Spark.	17 novembre 2023

Support pour les nouvelles sources de données dans AWS Glue Spark	Les connexions à Amazon OpenSearch Service, Azure SQL, Azure Cosmos pour NoSQL, SAP HANA, Teradata Vantage et Vertica sont désormais prises en charge de manière native au sein de ce service. AWS Glue En outre, les connexions à ces sources de données, ainsi qu'à MongoDB, peuvent désormais être utilisées dans l'éditeur visuel de AWS Glue Studio. Pour plus d'informations, consultez Types et options de connexion pour ETL dans Spark AWS Glue pour plus d'informations sur AWS Glue le support de Spark et Ajout d'une AWS Glue connexion pour des informations sur l'utilisation dans l'éditeur visuel AWS Glue Studio.	17 novembre 2023
Prise en charge de la génération de statistiques de colonne	Vous pouvez calculer des statistiques au niveau des colonnes pour AWS Glue Data Catalog des tables dans des formats de données tels que Parquet, ORC, JSON, ION, CSV et XML sans configure r de pipelines de données supplémentaires. Pour plus d'informations, consultez Utilisation des statistiques de colonne.	16 novembre 2023

Prise en charge du compactage des données des tables Iceberg	Pour améliorer les performances de lecture des services AWS d'analyse tels qu'Amazon Athena et Amazon EMR, ainsi que pour les tâches AWS Glue ETL, Data Catalog propose un compactage géré (un processus qui compacte de petits objets Amazon S3 en objets plus grands) pour les tables Iceberg dans Data Catalog. Pour plus d'informations, consultez Optimisation des tables Iceberg.	13 novembre 2023
Mise à jour du comportement d'attente lors de l'exécution des tâches	Les exécutions de tâches standard du shell Spark et Python seront désormais transférées vers `WAITING` dans certaines situations, au lieu de passer immédiatement à `FAILED`. Pour plus d'informations, consultez la rubrique États d'exécution des tâches AWS Glue.	8 novembre 2023
AWS Glue Studio guide de l'utilisateur consolidé dans le guide AWS Glue du développeur	Le guide de AWS Glue Studio l'utilisateur a été transféré dans le guide du développeur afin de créer un guide utilisateur unifié unique pour AWS Glue Studio la AWS Glue console et l'accès AWS Glue Studio par programmation.	25 octobre 2023

Mise à jour de la politique AWSGlue ServiceNotebookRole AWS gérée	Ajout d'informations concernant une mise à jour mineure de la politique AWSGlue ServiceNotebookRole AWS gérée. Pour plus d'informations, consultez la section AWS Glue Mises à jour des politiques AWS gérées.	9 octobre 2023
AWS Glue Studio prend en charge cinq nouvelles transformations intégrées	AWS Glue Studio prend en charge les cinq nouvelles transformations intégrées suivantes : correspondance des enregistrements, suppression des lignes nulles, analyse de la colonne JSON, extraction du chemin JSON et extracteur Regex. Pour plus d'informations, consultez la section Modification des nœuds de transformation de données AWS Glue gérés.	11 août 2023
Mise à jour de la politique AWSGlue ServiceRole AWS gérée	Ajout d'informations concernant une mise à jour mineure de la politique AWSGlue ServiceRole AWS gérée. Pour plus d'informations, consultez la section AWS Glue Mises à jour des politiques AWS gérées.	4 août 2023

Prise en charge pour l'indexation de tables Apache Hudi	Ajout d'informations sur l'utilisation AWS Glue pour explorer les tables Hudi dans les compartiments Amazon S3 et sur l'enregistrement des tables Hudi dans le. AWS Glue Data Catalog Pour plus d'informations, consultez Which data stores can I crawl?, et Crawler properties.	21 juillet 2023
Mise à jour de la politique AWSGlue ConsoleFullAccess AWS gérée	Ajout d'informations concernant une mise à jour mineure de la politique AWSGlue ConsoleFullAccess AWS gérée. Pour plus d'informations, consultez la section AWS Glue Mises à jour des politiques AWS gérées.	14 juillet 2023
Prise en charge pour l'indexation de tables Apache Iceberg	Ajout d'informations sur l'utilisation AWS Glue pour explorer les tables Iceberg dans les compartiments Amazon S3 et sur l'enregistrement des tables Iceberg dans le. AWS Glue Data Catalog Pour plus d'informations, consultez Which data stores can I crawl?, et Crawler properties.	7 juillet 2023

Support pour « AWS Glue with Ray »	Ajout d'informations sur AWS Glue with Ray, un nouveau moteur qui permet de sauvegarder AWS Glue des jobs. Réorganisé l'existant AWS Glue avec du contenu Spark pour lever toute ambiguïté.	30 mai 2023
Support pour la qualité AWS Glue des données (GA)	AWS Glue La qualité des données est désormais généralement disponible. AWS Glue La qualité des données vous aide à évaluer et à contrôler la qualité de vos données. Pour plus d'informations sur l'utilisation de la qualité AWS Glue des données avec le catalogue de données, consultez la section QualitéAWS Glue des données. Pour en savoir plus sur la qualité AWS Glue des données pour AWS Glue Studio, consultez la section Évaluation de la qualité des données avec AWS Glue Studio.	24 mai 2023

Prise en charge de types de travailleurs plus importants pour les tâches Apache Spark	La prise en charge est désormais disponible pour l'utilisation des types de travailleurs `G.4X` et `G.8X` pour les tâches Apache Spark. Ces types de travailleurs sont appropriés pour les tâches dont les charges de travail contiennent les transform ations, les agrégations, les jointures et les requêtes les plus exigeantes. Pour plus d'informations, consultez la section Ajout de tâches AWS Glue.	8 mai 2023
Prise en charge de la création d'index de partition lors de l'indexation de tables	Ajout d'informations sur la manière dont les Crawlers prennent en charge la création d'index de partition pour les tables détectées par le Crawler. Pour plus d'informa tions, consultez Setting the partition index crawler configuration option.	24 avril 2023
Prise en charge des mesures d'utilisation des ressources	Ajout d'informations sur l'affichage de l'utilisation des ressources du service et la configuration des alarmes dans Amazon CloudWatc h. Pour plus d'informations, consulter AWS Glue resource monitoring.	7 avril 2023

Mise à jour de la politique AWSGlue ConsoleFullAccess AWS gérée	Ajout d'informations concernant une mise à jour mineure de la politique AWSGlue ConsoleFullAccess AWS gérée. Pour plus d'informations, consultez la section AWS Glue Mises à jour des politiques AWS gérées.	28 mars 2023
Ajout de conseils d'utilisation AWS Glue avec un AWS SDK avec des exemples	Le guide du AWS Glue développeur comporte deux nouvelles sections qui fournissent des informations pour vous aider AWS Glue à utiliser un AWS SDK. Pour plus d'informations, consultez les sections Utilisation AWS Glue avec un AWS SDK et Exemples de code d' AWS Glue utilisation AWS SDKs.	23 février 2023
Mise à jour de la documentation pour IAM avec AWS Glue	Informations réorganisées et ajoutées sur l'utilisation d'IAM avec. AWS Glue Pour plus d'informations, veuillez consulter la rubrique Gestion des identités et des accès pour AWS Glue.	15 février 2023

Support pour l'exécution de tâches ETL en streaming dans AWS Glue la version 4.0	Ajout d'informations concernant la prise en charge de l'exécution de tâches ETL en streaming dans Glue version 4.0, et de nouvelles options de connexion à un cluster Kafka ou à un cluster Amazon Managed Streaming for Apache Kafka et Amazon Kinesis Data Streams. Pour plus d'informations, veuillez consulter les rubriques Ajout de tâches ETL en streaming dans AWS Glue et Types et options de connexion pour ETL dans AWS Glue.	8 février 2023
Prise en charge de l'analyse des sources de données MongoDB Atlas	Ajout d'informations sur l'utilisation AWS Glue pour explorer les sources de données MongoDB Atlas. Pour plus d'informations, consultez Which data stores can I crawl?, MongoDB and MongoDB Atlas connection properties et Using a MongoDB or MongoDB Atlas connection.	6 février 2023

Prise en charge de l'analyse des tables Delta Lake à l'aide d'un connecteur Delta Lake natif	Ajout d'informations sur l'utilisation AWS Glue pour parcourir les tables de Delta Lake à l'aide d'un connecteur Delta Lake natif. Cette fonctionnalité vous permet d'utiliser des moteurs de AWS requête pour interroger directement le journal des transactions Delta et d'utiliser des fonctionnalités telles que le voyage dans le temps et les garanties ACID, et de synchroniser vos métadonnées Delta Lake à partir des fichiers de transactions Amazon S3 dans le catalogue de données afin d'activer les autorisations de colonne sur vos requêtes dans Lake Formation. Pour plus d'informations, veuillez consulter les rubriques Comment préciser les options de configuration pour un magasin de données Delta Lake et Interrogation des tables Delta Lake.	15 décembre 2022

Support pour la qualité AWS Glue des données (version préliminaire)	Support est désormais disponible pour AWS Glue Data Quality (version préliminaire). AWS Glue La qualité des données vous permet d'évaluer et de contrôler la qualité de vos données lorsque vous utilisez la AWS Glue version 3.0. Pour plus d'informations sur l'utilisation de la qualité AWS Glue des données avec le catalogue de données, voir QualitéAWS Glue des données (version préliminaire). Pour en savoir plus sur la qualité AWS Glue des données pour AWS Glue Studio, consultez la section Évaluation de la qualité des données avec AWS Glue Studio.	30 novembre 2022

Prise en charge d'un nouveau connecteur Amazon Redshift Spark avec de nouvelles fonctionnalités et des améliorations de performances	Support est désormais disponible pour un nouveau connecteur Amazon Redshift Spark doté d'un nouveau pilote JDBC à utiliser avec des tâches AWS Glue ETL afin de créer des applications Apache Spark capables de lire et d'écrire sur des données dans Amazon Redshift dans le cadre de vos pipelines d'ingestion et de transform ation des données. Pour plus d'informations, consultez la rubrique Moving data to and from Amazon Redshift (Déplacement de données vers et depuis Amazon Redshift).	29 novembre 2022

Support pour AWS Glue la version 4.0.	Ajout d'informations sur le support de AWS Glue la version 4.0. Les fonctionn alités incluent la prise en charge native des infrastru ctures de lacs de données ouvertes avec Apache Hudi, Delta Lake et Apache Iceberg, ainsi que la prise en charge native du plug-in Cloud Shuffle Storage basé sur Amazon S3 (un plug-in Apache Spark) qui utilise Amazon S3 pour le brassage et une capacité de stockage élastique. Pour plus d'informations, consultez les notes AWS Glue de publicati on et la migration des AWS Glue tâches vers la AWS Glue version 4.0.	28 novembre 2022
AWS Glue Studio propose désormais des transformations visuelles personnalisées	Les transformations visuelles personnalisées permettent aux clients de définir, de réutilise r et de partager une logique ETL spécifique à l'entrepr ise avec leurs équipes. Pour plus d'informations, consultez Custom visual transforms (Transformations visuelles personnalisées).	28 novembre 2022

Support pour l'utilisation du AWS Glue robot d'exploration pour publier des métadonnées pour les magasins de données JDBC	Support est désormais disponible pour l'utilisation du AWS Glue robot d'exploration pour publier des métadonnées telles que des commentaires et des types bruts dans le catalogue de données pour les magasins de données JDBC. Pour plus d'informations, voir Paramètres définis sur les tables du catalogue de données par robot, propriétés du robot et JdbcTarget structure.	18 novembre 2022

Prise en charge de l'indexation des magasins de données Snowflake	Support est désormais disponible AWS Glue pour explorer les tables et les vues Snowflake et pour publier les métadonnées dans le catalogue de données sous forme d'entrée de table. Pour les tables externes Snowflake dans Amazon S3, le crawler analyse également l'emplacement Amazon S3 et le type de format de fichier de la table externe et les renseigne en tant que paramètres de table. Pour plus d'informations, consultez Which data stores can I crawl? (Quels magasins de données puis-je analyser ?), les propriétés de connexion AWS Glue et les paramètres définis sur les tables du catalogue de données par un crawler.	18 novembre 2022
Prise en charge d'une meilleure gestion du brassage de vos applications Spark	Désormais, la prise en charge d'un nouveau plug-in Cloud Shuffle Storage pour Apache Spark est disponible. Pour plus d'informations, consultez AWS Glue Spark shuffle plugin with Amazon S3 et Cloud Shuffle Storage Plugin for Apache Spark.	15 novembre 2022

Ajout de la prise en charge des cibles du catalogue de données lors de l’accélération des notifications d’événements Amazon S3 d’indexation	Outre la prise en charge existante des cibles Amazon S3, la prise en charge est désormais disponible pour accélérer l’indexation de site web des cibles du catalogue de données à l’aide des notifications d’événements Amazon S3. Pour en savoir plus, consultez Accélération des analyses à l'aide des notifications d'événements Amazon S3.	13 octobre 2022
Support permettant de spécifier le nombre maximum de tables qu'un crawler peut créer	Le support est désormais disponible pour spécifier le nombre maximum de tables que le crawler est autorisé à créer. Pour plus d'informations, consultez la rubrique Comment spécifier le nombre maximum de tables que le crawler est autorisé à créer.	6 septembre 2022
Support de Python 3.9 dans les jobs shell Python dans AWS Glue	Support est désormais disponible pour l'exécution de scripts compatibles avec Python 3.9 dans les jobs shell Python et pour le choix d'utiliser des ensembles de bibliothèques préemballés. AWS Glue Pour en savoir plus, consultez Tâches Python shell dans AWS Glue.	11 août 2022

Support pour exécuter des AWS Glue tâches non urgentes ou non urgentes avec des capacités inutilisées	La prise en charge est désormais disponible pour la configuration d'exécutions de tâches flexibles pour les tâches non urgentes telles que les tâches de pré-production, les tests et les chargements de données ponctuels. Pour plus d'informations, consultez la section Ajout de tâches AWS Glue.	9 août 2022
Prise en charge d'un nouveau type d'employé pour les tâches de streaming	Une prise en charge est désormais disponible pour une utilisation du type d'employé `G.025X` pour les tâches de streaming à faible volume. Pour plus d'informations, consultez la section Ajout de tâches AWS Glue.	14 juillet 2022
Support pour l'utilisation de Kafka SASL dans les connexions AWS Glue	Support est désormais disponible pour l'utilisation de Kafka SASL dans les AWS Glue connexions. Pour plus d'informations, consultez Propriétés de connexion Kafka AWS Glue pour l'authentification du client.	5 juillet 2022
Prise en charge du connecteur Apache Kafka pour schémas Protobuf	La prise en charge du connecteur Apache Kafka est désormais disponible pour schémas Protobuf. Pour plus d'informations, consultez Registre de schémas AWS Glue.	9 juin 2022

Support pour Auto Scaling for AWS Glue jobs (GA)	Ajout d'informations sur l'utilisation d'Auto Scaling pour les tâches dans AWS Glue la version 3.0 afin de dimensionner dynamiquement les ressources de calcul. Pour plus d'informations, consultez Utilisation d'Auto Scaling pour AWS Glue.	14 avril 2022
Mise à jour de la documentation pour le AWS Glue développement et le test de scripts de AWS Glue travail	Informations réorganisées et ajoutées sur les méthodes de développement et de test disponibles pour AWS Glue, y compris les instructions pour le développement avec Docker. Pour plus d'informations, consultez la section Développement et test de scripts de AWS Glue travail.	14 mars 2022
Ajout de tampons de protocole (protobuf) en tant que format de données pris en charge pour le registre de schémas AWS Glue	Ajout d'informations sur Protobuf en tant que format de données pris en charge (en plus d'AVRO et de JSON). Pour plus d'informations, consultez Registre de schémas AWS Glue.	25 février 2022
Prise en charge des tables rampantes Delta Lake	Ajout d'informations sur l'utilisation AWS Glue pour parcourir les tables de Delta Lake. Pour plus d'informations, consultez Comment préciser les options de configuration pour un magasin de données Delta Lake.	24 février 2022

Support pour obtenir des informations sur les AWS Glue offres d'emploi	Ajout d'informations sur l'utilisation AWS Glue des informations sur les tâches pour simplifier le débogage des tâches et l'optimisation de vos AWS Glue tâches. Pour plus d'informations, consultez la section Surveillance à l'aide des informations sur les AWS Glue offres d'emploi.	8 février 2022
Prise en charge de l'analyse des tables du catalogue de données basées sur Amazon S3 à l'aide d'un point de terminaison d'un VPC	En plus des magasins de données Amazon S3, vous pouvez configurer vos tables du catalogue de données basées sur Amazon S3 pour qu'ils soient accessibles uniquement par un environnement Amazon Virtual Private Cloud (Amazon VPC), à des fins de sécurité, d'audit ou de contrôle. Pour plus d'informations, consultez Analyse d'un magasin de données Amazon S3 ou tables du catalogue de données basées sur Amazon S3 à l'aide d'un point de terminaison d'un VPC.	3 février 2022

Support des tables régies de Lake Formation	Ajout d'informations sur la AWS Glue prise en charge des tables gouvernées par Lake Formation, qui prennent en charge les transactions ACID, le compactage automatique des données et les requêtes de voyage dans le temps. Pour en savoir plus, consultez API AWS Glue et le guide du développeur AWS Lake Formation.	30 novembre 2021
Nouvelles politiques AWS gérées ajoutées pour les sessions interactives et les blocs-notes	Les nouvelles politiques gérées pour IAM ont renforcé la sécurité lors de l'utilisa tion AWS Glue des sessions interactives et des blocs-not es. Pour plus d'informations, veuillez consulter la rubrique Politiques gérées par AWS pour AWS Glue.	30 novembre 2021
Registre de schémas Glue désormais supportés avec les tâches de streaming	Vous pouvez créer des tâches de streaming qui accèdent aux tables qui font partie du registre de schéma Glue. Pour en savoir plus, consultez Registre de schémas AWS Glue et Ajouter des tâches ETL de streaming dans AWS Glue.	15 novembre 2021

Support des nouvelles fonctions de machine learning	Ajout d'informations sur les nouvelles fonctions de transformation machine learning Rechercher de correspondances, y compris la correspondance progressive et la notation des correspondances. Pour en savoir plus, consultez Recherche de correspondances progressives et Estimation de la qualité des correspondances à l'aide des notes de confiance des correspondances.	31 octobre 2021
(Aperçu privé) Support pour les jobs AWS Glue flexibles	Ajout d'informations sur la configuration des tâches AWS Glue Spark avec une classe d'exécution flexible, adaptée aux tâches non urgentes dont les heures de début et de fin peuvent varier. Pour plus d'informations, consultez la section Ajout de tâches AWS Glue.	29 octobre 2021
Support d'accélération des explorations à l'aide des notifications d'événements Amazon S3	Ajout d'informations sur l'accélération des explorations à l'aide des notifications d'événements Amazon S3. Pour en savoir plus, consultez Accélération des explorations à l'aide des notifications d'événements Amazon S3.	15 octobre 2021

Options de configuration de sécurité supplémentaires liées au contrôle d'accès et VPCs	Ajout d'informations sur la façon dont vous pouvez configurer les nouvelles autorisations de contrôle d'accès AWS Glue et la configuration de VPCs. Pour plus d'informations, consultez les sections AWS Balises AWS Glue, politiques basées sur l'identité (politiques IAM) qui contrôlent les paramètres à l'aide de clés de condition ou de clés contextuelles, et Configuration de tous les AWS appels pour qu'ils passent par votre VPC.	13 octobre 2021
Support des politiques de point de terminaison d'un VPC	Ajout d'informations sur le support des politiques de point de terminaison Virtual Private Cloud (VPC) dans AWS Glue. Pour en savoir plus, consultez la section relative à AWS Glue et points de terminaison VPC d'interface (AWS PrivateLink).	11 octobre 2021
Glue Studio est désormais disponible en Chine	AWS Glue Studio est désormais disponible dans les régions de Pékin et de Ningxia en Chine.	11 octobre 2021

AWS Glue Studio propose la création de blocs-notes, pour une édition interactive des tâches	Les blocs-notes vous permettent d'écrire et d'exécuter du code, de visualiser les résultats et de partager des informations. En général, les spécialistes des données utilisent des blocs-notes pour des expérienc es et des tâches d'explora tion de données. Pour plus d'informations, consultez Using Notebooks (Utilisation des blocs-notes).	1er octobre 2021
Un accès direct aux sources de streaming est désormais disponible	Lorsque vous ajoutez des sources de données à votre tâche ETL dans l'éditeur visuel, vous pouvez fournir des informations pour accéder au flux de données au lieu d'utiliser une base de données et une table du catalogue de données.	30 septembre 2021
A documenté la politique de support des AWS Glue versions	Ajout d'informations sur la politique de support des AWS Glue versions et les phases de fin de vie de certaines AWS Glue versions. Pour en savoir plus, consultez politique de prise en charge de version AWS Glue.	24 septembre 2021

Les connecteurs personnalisés peuvent désormais être utilisés avec des prévisualisations de données	Lorsque vous modifiez un nœud de source de données à l'aide d'un connecteur personnalisé, vous pouvez prévisualiser le jeu de données en cliquant sur l'onglet Aperçu des données. Pour plus d'informations, consultez Custom Connectors (Connecteurs personnalisés).	24 septembre 2021
Support pour les sessions AWS Glue interactives (aperçu privé)	(Aperçu privé) Ajout d'informations sur l'utilisation de sessions AWS Glue interactives pour exécuter des charges de travail Spark dans le cloud à partir de n'importe quel bloc-notes Jupyter. Les sessions interactives constituent la méthode préférée pour développer votre code AWS Glue d'extraction, de transformation et de chargement (ETL) lorsque vous utilisez la AWS Glue version 2.0 ou une version ultérieure. Pour plus d'informations, voir Configuration et exécution de sessions AWS Glue interactives pour Jupyter Notebook.	24 août 2021

Prise en charge de la création de flux de travail à partir de plans (GA)	Ajout d'informations sur le codage des cas d'utilisation courants Extract-transform-load (ETL) dans les modèles, puis sur la création de flux de travail à partir de modèles. Permet aux analystes de données de créer et d'exécuter facilement des processus ETL complexes. Pour plus d'informations, consultez Exécution d'activités ETL complexes à l'aide de plans et de flux de travail dans AWS Glue.	23 août 2021

Support pour AWS Glue la version 3.0.	Ajout d'informations sur la prise en charge de la AWS Glue version 3.0, qui prend en charge la mise à niveau du moteur Apache Spark 3.0 pour l'exécution des tâches ETL Apache Spark, ainsi que d'autres optimisations et mises à niveau. Pour plus d'informations, consultez les notes AWS Glue de publication et la migration des AWS Glue tâches vers la AWS Glue version 3.0. Les autres fonctionnalités de cette version incluent le gestionnaire de AWS Glue shuffle, un lecteur CSV vectorisé SIMD et des prédicats de partition de catalogue. Pour plus d'informations, consultez le gestionnaire de shuffle AWS Glue Spark avec Amazon S3, les options de format pour les entrées et sorties ETL AWS Glue, et le filtrage côté serveur à l'aide de prédicats de partition de catalogue.	18 août 2021
AWS GovCloud (US) Region	AWS Glue Studio est désormais disponible dans AWS GovCloud (US) Region	18 août 2021

Création de shell Python disponible dans AWS Glue Studio	Lors de la création d'une nouvelle tâche, vous pouvez maintenant choisir de créer une tâche shell Python. Pour en savoir plus, consultez Start the job creation process et Editing Python shell jobs in AWS Glue Studio.	13 août 2021
Support pour démarrer un flux de travail avec un EventBridge événement Amazon	Ajout d'informations sur la manière d' AWS Glue être un consommateur d'événements dans une architecture axée sur les événements. Pour plus d'informations, consultez Démarrer un AWS Glue flux de travail avec un EventBridge événement Amazon et Afficher les EventBridge événement s qui ont démarré un flux de travail.	14 juillet 2021
Ajout de JSON en tant que format de données pris en charge pour le registre de AWS Glue schémas	Ajout d'informations sur JSON en tant que format de données pris en charge (en plus d'AVRO). Pour de plus amples informations, veuillez consulter Registre de schémas AWS Glue.	30 Juin 2021

Créez des tâches de AWS Glue streaming sans table de catalogue de données	La fonction Python `create_data_frame_from_options` ou `getSource` pour les scripts Scala prennent en charge la création de tâches ETL en streaming qui référencent directement les flux de données au lieu de nécessiter une table Data Catalog.	15 juin 2021
AWS Glue l'apprentissage automatique transforme désormais les AWS Key Management Service clés de support	Vous pouvez spécifier une configuration ou une AWS KMS clé de sécurité lorsque vous configurez des transformations de AWS Glue Machine Learning à l'aide de la console, de la CLI ou du AWS Glue APIs. Pour de plus amples informations, veuillez consulter Utilisation du chiffrement des données avec les transformations de machine learning et API de machine learning AWS Glue.	15 juin 2021
Mise à jour de la politique AWSGlue ConsoleFullAccess AWS gérée	Ajout d'informations concernant une mise à jour mineure de la politique AWSGlue ConsoleFullAccess AWS gérée. Pour plus d'informations, consultez la section AWS Glue Mises à jour des politiques AWS gérées.	10 juin 2021

Afficher le jeu de données de votre tâche lors de la création et de la modification des tâches	Vous pouvez utiliser le nouveau module Data preview (Prévisualisation des données) d'un nœud dans votre diagramme de tâche pour afficher un échantillon des données traitées par ce nœud. Pour de plus amples informations, veuillez consulter la rubrique Utilisation des prévisualisations de données dans l'éditeur de tâches visuel.	7 juin 2021
Prise en charge de la spécification d'une valeur qui indique l'emplacement de la table pour la sortie du crawler.	Ajout d'informations sur la spécification d'une valeur indiquant l'emplacement de la table lors de la configuration de la sortie du crawler. Pour de plus amples informations, veuillez consulter Procédure pour spécifier l'emplacement de la table.	4 juin 2021
Prise en charge de l'analyse d'un échantillon de fichiers dans un jeu de données lors de l'exploration d'un magasin de données Amazon S3	Ajout d'informations sur l'analyse d'un échantillon de fichiers lors de l'analyse d'Amazon S3. Pour plus d'informations, consultez Propriétés de l'crawler.	10 mai 2021

Support pour le parquettiste AWS Glue optimisé	Ajout d'informations sur l'utilisation du rédacteur de parquet AWS Glue optimisé DynamicFrames pour créer ou mettre à jour des tableaux avec la `parquet` classific ation. Pour plus d'informations, voir Création de tables, mise à jour du schéma et ajout de nouvelles partitions dans le catalogue de données à partir de tâches AWS Glue ETL et Options de format pour les entrées et sorties ETL dans AWS Glue.	4 mai 2021
Prise en charge des mots de passe pour l'authentification du client Kafka	Ajout d'informations sur la façon dont le streaming de tâches ETL AWS Glue prend en charge l'authentification par certificat client SSL auprès des producteurs de flux Apache Kafka. Vous pouvez désormais fournir un certificat personnalisé lors de la définitio n d'une AWS Glue connexion à un cluster Apache Kafka, qui AWS Glue sera utilisé lors de l'authentification auprès de celui-ci. Pour plus d'informa tions, consultez Propriétés de connexion AWS Glue et API de connexion.	28 avril 2021

Prise en charge de la consommation de données d'Amazon Kinesis Data Streams dans un autre compte dans les tâches ETL de streaming	Ajout d'informations sur la création d'une tâche ETL de streaming pour consommer les données d'Amazon Kinesis Data Streams dans un autre compte. Pour plus d'informations, consultez la section Ajout de tâches ETL en streaming dans AWS Glue.	30 mars 2021
Transformation SQL disponible	Vous pouvez utiliser un nœud de transformation SQL pour écrire votre propre transformation sous la forme d'une requête SQL. Pour de plus amples informations, veuillez consulter la rubrique Utilisation d'une requête SQL pour transformer des données.	23 mars 2021
Prise en charge de la création de flux de travail à partir de plans (version préliminaire publique)	(Version préliminaire publique) Ajout d'informations sur le codage des cas d'utilisation courants Extract-transform-load (ETL) dans les modèles, puis sur la création de flux de travail à partir de modèles. Permet aux analystes de données de créer et d'exécuter facilement des processus ETL complexes. Pour plus d'informations, consultez Exécution d'activités ETL complexes à l'aide de plans et de flux de travail dans AWS Glue.	22 mars 2021

Les connecteurs peuvent être utilisés pour les cibles de données	L'utilisation d'un AWS Marketplace connecteur ou d'un personnalisé pour votre cible de données est désormais prise en charge. Pour de plus amples informations, veuillez consulter la rubrique Création de tâches avec des connecteurs personnalisés.	15 mars 2021
Support des métriques d'importance des colonnes pour les transformations du AWS Glue machine learning	Ajout d'informations sur l'affichage des indicateurs d'importance des colonnes lors de l'utilisation de transformations de AWS Glue machine learning. Pour plus d'informations, voir Utilisation des transformations du Machine Learning sur la AWS Glue console	5 février 2021
La planification des tâches est désormais disponible dans AWS Glue Studio	Vous pouvez définir une planification temporelle pour l'exécution de vos tâches dans AWS Glue Studio. Vous pouvez utiliser la console pour créer une planification de base, ou définir une planification plus complexe à l'aide de la syntaxe de type Unix cron. Pour de plus amples informations, veuillez consulter la rubrique Planification des exécutions de tâches.	21 décembre 2020

AWS Glue Connecteurs personnalisés publiés	AWS Glue Les connecteurs personnalisés vous permettent de découvrir et de vous abonner à des connecteurs dans AWS Marketplace. Nous avons également publié des interfaces d'exécution AWS Glue Spark pour intégrer des connecteurs conçus pour Apache Spark Datasource, Athena federated Query et JDBC. APIs Pour plus d'informations, consultez la section Utilisation de connecteurs et de connexions avec AWS Glue Studio.	21 décembre 2020
Support pour l'exécution de tâches ETL en streaming dans AWS Glue la version 2.0	Ajout d'informations sur la prise en charge de l'exécution de tâches ETL en streaming dans Glue version 2.0. Pour plus d'informations, consultez la section Ajout de tâches ETL en streaming dans AWS Glue.	18 décembre 2020

Prise en charge du partitionnement de la charge de travail avec exécution limitée	Ajout d'informations sur l'activation du partitionnement de la charge de travail pour configurer les limites supérieures de la taille du jeu de données ou du nombre de fichiers traités lors des exécutions de tâches ETL. Pour de plus amples informations, veuillez consulter Partitionnement de la charge de travail avec exécution limitée.	23 novembre 2020
Prise en charge de la gestion améliorée des partitions	Ajout d'informations sur la façon d'utiliser new APIs pour ajouter ou supprimer une partition, indexer to/from une table existante. Pour plus d'informations, consultez Utilisation des index de partition.	23 novembre 2020
Support pour le registre des AWS Glue schémas	Ajout d'informations sur l'utilisation du registre des AWS Glue schémas pour découvrir, contrôler et faire évoluer les schémas de manière centralisée. Pour plus d'informations, consultez la section Registre des AWS Glue schémas.	19 novembre 2020

Prise en charge du format de saisie de Grok dans les tâches ETL de streaming	Ajout d'informations sur l'application de modèles Grok aux sources de streaming telles que les fichiers journaux. Pour de plus amples informations, veuillez consulter Application de modèles Grok à des sources de streaming.	17 novembre 2020
Support pour l'ajout de balises aux flux de travail sur la AWS Glue console	Ajout d'informations sur l'ajout de balises lors de la création d'un flux de travail à l'aide de la console AWS Glue . Pour plus d'informations, voir Création et développement d'un flux de travail à l'aide de la AWS Glue console.	27 octobre 2020
Prise en charge des exécutions progressives du crawler	Ajout d'informations sur la prise en charge des exécutions incrémentielles du crawler, qui n'analysent que les dossiers Amazon S3 ajoutés depuis la dernière exécution. Pour de plus amples informations, veuillez consulter Analyses incrémentielles.	21 octobre 2020

Prise en charge de la détection de schéma pour les sources de données ETL en streaming. Prise en charge des sources de données ETL en streaming Avro et Kafka autogéré	Les tâches d'extraction, de transformation et de chargement (ETL) en streaming AWS Glue peuvent désormais détecter automatiquement le schéma des enregistrements entrants et gérer les modifications de schéma par enregistrement. Les sources de données Kafka autogérées sont désormais prises en charge. Les tâches ETL en streaming prennent désormais en charge le format Avro dans les sources de données. Pour plus d'informations, consultez Streaming ETL in AWS Glue, Defining Job Properties for a Streaming ETL Job et Remarques et restrictions pour les sources de streaming Avro.	7 octobre 2020
Prise en charge de l'analyse des sources de données MongoDB et DocumentDB	Ajout d'informations sur la prise en charge de l'analyse des sources de données MongoDB et Amazon DocumentDB (avec compatibilité MongoDB). Pour plus d'informations, consultez Définition des crawlers.	5 octobre 2020

Prise en charge de la conformité FIPS	Ajout d'informations sur les points de terminaison FIPS pour les clients qui ont besoin de modules cryptographiques validés FIPS 140-2 lors de l'accès aux données à l'aide de AWS Glue. Pour plus d'informations, consultez Conformité FIPS.	23 septembre 2020
AWS Glue Studio fournit une interface visuelle facile à utiliser pour la création et le suivi des emplois	Vous pouvez désormais utiliser une interface graphique simple pour composer des tâches qui déplacent et transforment les données et les exécutent sur AWS Glue. Vous pouvez ensuite utiliser le tableau de bord d'exécution des tâches dans AWS Glue Studio pour contrôler l'exécuti on ETL et vous assurer que vos tâches fonctionnent comme prévu. Pour plus d’informations, consultez le AWS Glue Studio guide de l’utilisateur.	23 septembre 2020
Prise en charge de la création d'index de tableau pour améliorer les performances des requêtes	Ajout d'informations sur la création d'index de table pour vous permettre de récupérer un sous-ensemble des partitions d'une table. Pour plus d'informations, consultez Utilisation des index de partition.	9 septembre 2020

Support pour réduire les temps de démarrage lors de l'exécution de tâches ETL Apache Spark dans AWS Glue la version 2.0.	Ajout d'informations sur la prise en charge de la AWS Glue version 2.0, qui fournit une infrastructure améliorée pour exécuter les tâches ETL Apache Spark avec des temps de démarrage réduits, des modifications dans la journalisation et la prise en charge de la spécification de modules Python supplémentaires au niveau des tâches. Pour de plus amples informations, veuillez consulter Notes de mise à jour AWS Glue et Exécution de tâches ETL Spark avec des temps de démarrage réduits.	10 août 2020
Prise en charge de la limitation du nombre d'exécutions de flux de travail simultanées.	Ajout d'informations sur la manière de limiter le nombre d'exécutions de flux de travail simultanées pour un flux de travail particulier. Pour plus d'informations, voir Création et développement d'un flux de travail à l'aide de la AWS Glue console.	10 août 2020

Prise en charge de l'analyse d'un stock de données Amazon S3 à l'aide d'un point de terminaison d'un VPC	Ajout d'informations sur la configuration de votre magasin de données Amazon S3 pour qu'il soit accessible uniquement par un environnement Amazon Virtual Private Cloud (Amazon VPC), à des fins de sécurité, d'audit ou de contrôle. Pour de plus amples informations, veuillez consulter Analyse d'un magasin de données Amazon S3 à l'aide d'un point de terminaison VPC.	7 août 2020
Prise en charge de la reprise des exécutions de flux de travail	Ajout d'informations sur la façon de reprendre les exécutions de flux de travail qui ne se sont que partiellement terminées, car un ou plusieurs nœuds (tâches ou crawlers) ne se sont pas terminés avec succès. Pour plus d'informations, consultez Réparer et reprendre l'exécution d'un flux de travail.	27 juillet 2020

Prise en charge de l'activation des certificats d'autorité de certification privés dans les connexions Kafka au format AWS Glue.	Ajout d'informations sur les nouvelles options de connexion qui prennent en charge l'activation des certificats d'autorité de certification privés pour les connexions Kafka dans AWS Glue. Pour plus d'informations, voir Types de connexion et options pour l'ETL dans AWS Glue et Paramètres spéciaux utilisés par AWS Glue.	20 juillet 2020
Prise en charge de la lecture des données DynamoDB dans un autre compte	Ajout d'informations sur la AWS Glue prise en charge de la lecture de données depuis la table DynamoDB d'un autre AWS compte Pour plus d'informations, voir Lecture à partir de données DynamoDB dans un autre compte.	17 juillet 2020
Support pour une connexion d'enregistreur DynamoDB AWS Glue dans la version 1.0 ou ultérieure	Ajout d'informations sur la prise en charge de l'écriture DynamoDB et des options de connexion nouvelles ou mises à jour pour la lecture ou l'écriture de DynamoDB. Pour plus d'informations, veuillez consulter Types de connexion et options de connexion pour ETL dans AWS Glue.	17 juillet 2020

Support pour les liens vers les ressources et pour le contrôle d'accès entre comptes à l'aide des deux AWS Glue et de Lake Formation	Ajout de contenu sur les nouveaux objets du catalogue de données appelés liens de ressources, et sur la façon de gérer le partage des ressources du catalogue de données entre les comptes avec AWS Glue les deux AWS Lake Formation. Pour de plus amples informations, veuillez consulter Octroi d'un accès intercompte et Liens de ressources de table.	7 juillet 2020
Prise en charge des registres d'échantillonnage lors de l'analyse des stocks de données DynamoDB	Ajout d'informations sur les nouvelles propriétés que vous pouvez configurer lors de l'analyse d'un magasin de données DynamoDB. Pour plus d'informations, consultez Propriétés de l'crawler.	12 juin 2020
Prise en charge de l'arrêt d'une exécution de flux de travail.	Ajout d'informations sur la façon d'arrêter le cycle d'un flux de travail spécifique. Pour plus d'informations, consultez Arrêt d'un cycle de flux de travail.	14 mai 2020

Prise en charge des tâches ETL Spark Streaming	Ajout d'informations sur la création de tâches d'extraction, de transformation et de chargement (ETL) avec des sources de données en streaming. Pour de plus amples informations, consultez Ajout de tâches ETL en streaming dans AWS Glue.	27 avril 2020
Prise en charge de la création de tableaux, de la mise à jour du schéma et de l'ajout de nouvelles partitions dans le catalogue de données après l'exécution d'une tâche ETL	Ajout d'informations sur la façon dont vous pouvez activer la création de tables, la mise à jour du schéma et l'ajout de nouvelles partitions pour afficher les résultats de votre tâche ETL dans le catalogue de données. Pour plus d'informations, voir Création de tables, mise à jour du schéma et ajout de nouvelles partitions dans le catalogue de données à partir de tâches AWS Glue ETL.	2 avril 2020

Support pour spécifier une version pour le format de données Apache Avro en tant qu'entrée et sortie ETL dans AWS Glue	Ajout d'informations sur la spécification d'une version pour le format de données Apache Avro en tant qu'entrée et sortie ETL dans AWS Glue. Version par défaut 1.7. Vous pouvez utiliser l'option de format `version` pour spécifier à Avro 1.8 d'activer la lecture/écriture logique. Pour plus d'informations, consultez la section Options de format pour les entrées et sorties ETL dans AWS Glue.	31 mars 2020
Prise en charge du validateur EMRFS optimisé pour S3 dans le cadre de l'écriture de données Parquet dans Amazon S3	Ajout d'informations sur la façon de définir un nouvel indicateur pour activer le validateur EMRFR optimisé pour S3 dans le cadre de l'écriture de données Parquet sur Amazon S3 lors de la création ou de la mise à jour d'une tâche AWS Glue . Pour plus d'informations, consultez la section Paramètres spéciaux utilisés par AWS Glue.	30 mars 2020

Support à l'apprentissage automatique : transformation en tant que ressource gérée par AWS des balises de ressources	Des informations supplémentaires sur l'utilisation de balises de AWS ressources pour gérer et contrôler l'accès à votre machine learning se transforment AWS Glue. Vous pouvez attribuer AWS des balises de ressources aux tâches, aux déclencheurs, aux points de terminaison, aux robots d'exploration et aux transformations de machine learning. AWS Glue Pour plus d'informations, consultez la section AWS Tags in AWS Glue.	2 mars 2020
Prise en charge des arguments de tâche non substituables	Ajout d'informations sur la prise en charge des paramètres de tâche spéciaux qui ne peuvent pas être remplacés dans les déclencheurs ou lorsque vous exécutez la tâche. Pour plus d'informations, consultez Ajout de tâches dans AWS Glue.	12 février 2020

Prise en charge des nouvelles transformations pour l'utilisation des jeux de données dans Amazon S3	Ajout d'informations sur les nouvelles transformations (Fusion, Purge et Transition) et sur les exclusions de classe de stockage Amazon S3 pour les applications Apache Spark dans le cadre de l'utilisation des ensembles de données dans Amazon S3. Pour plus d'informations sur la prise en charge de ces transformations pour Python, consultez la section Utilisation mergeDynamicFramedes ensembles de données dans Amazon S3. Pour Scala, voir mergeDynamicFrameset AWS Glue Scala GlueContext APIs.	16 janvier 2020
Prise en charge de la mise à jour du catalogue de données avec les informations des nouvelles partitions à partir d'une tâche ETL	Ajout d'informations sur la façon de coder un script d'extraction, de transformation et de chargement (ETL) pour mettre à jour les informations de partition AWS Glue Data Catalog avec de nouvelles informations. Avec cette fonctionnalité, vous n'avez plus besoin de réexécuter le crawler à la fin de la tâche pour afficher les nouvelles partitions. Pour plus d'informations, consultez Mise à jour de Data Catalog avec de nouvelles partitions.	15 janvier 2020

Nouveau tutoriel : Utilisation d'un bloc-notes basé sur SageMaker l'IA	Ajout d'un didacticiel expliquant comment utiliser un SageMaker bloc-notes Amazon pour développer vos scripts ETL et d'apprentissage automatique. Consultez le didacticiel : utilisez un SageMaker bloc-notes Amazon avec votre terminal de développement.	3 janvier 2020
Prise en charge de la lecture depuis MongoDB et Amazon DocumentDB (avec compatibilité MongoDB)	Ajout d'informations sur les nouveaux types de connexion et options de connexion pour la lecture et l'écriture dans MongoDB et Amazon DocumentDB (avec compatibilité MongoDB). Pour plus d'informations, veuillez consulter Types de connexion et options de connexion pour ETL dans AWS Glue.	17 décembre 2019

Corrections et clarifications diverses	Des corrections et des clarifications ont été ajoutées. Des entrées ont été supprimées du chapitre Problèmes connus. Ajout d'avertissements qui ne prennent AWS Glue en charge que les clés principales symétriques du client (CMKs) lors de la spécification des paramètres de chiffrement du catalogue de données et de la création de configurations de sécurité. Ajout d'une note qui AWS Glue ne prend pas en charge l'écriture sur Amazon DynamoDB.	9 décembre 2019
Prise en charge des pilotes JDBC personnalisés	Ajout d'informations sur la connexion aux sources de données et aux cibles à l'aide de pilotes JDBC qui AWS Glue ne sont pas pris en charge de manière native, tels que MySQL version 8 et Oracle Database version 18. Pour plus d'informations, consultez JDBC connectionType Values.	25 novembre 2019

Support pour connecter des ordinateurs portables SageMaker dotés d'intelligence artificielle à différents terminaux de développement	Ajout d'informations sur la façon dont vous pouvez connecter un bloc-notes SageMaker AI à différents terminaux de développement. Mises à jour pour décrire la nouvelle action de console pour passer à un nouveau point de terminaison de développement, ainsi que la nouvelle politique SageMaker AI IAM. Pour plus d'informations, consultez les sections Utilisation des blocs-notes sur la AWS Glue console et Création d'une politique IAM pour les blocs-notes Amazon SageMaker AI.	21 novembre 2019
Support des AWS Glue versions lors des transformations liées à l'apprentissage automatique	Ajout d'informations sur la définition de la AWS Glue version dans une transformation d'apprentissage automatique pour indiquer la version compatible avec laquelle AWS Glue une transformation d'apprentissage automatique est compatible. Pour plus d'informations, voir Travailler avec les transformations du Machine Learning sur la AWS Glue console.	21 novembre 2019

Prise en charge de la restauration de vos favoris de tâche	Ajout d'informations sur la restauration de vos marque-pages de tâche sur une exécution de tâche précédente, ce qui entraîne le retraitement des données de l'exécution de tâche suivante uniquement à partir de l'exécution de travail marquée en favori. Décrit deux nouvelles sous-options pour l'option `job-bookmark-pause` , qui vous permettent d'exécuter une tâche entre deux favoris. Pour plus d'informations, voir Suivi des données traitées à l'aide des signets de tâches et des paramètres spéciaux utilisés par AWS Glue.	22 octobre 2019
Prise en charge des certificats JDBC personnalisés pour la connexion à un stock de données	Ajout d'informations sur la AWS Glue prise en charge des certificats JDBC personnalisés pour les connexions SSL aux sources de AWS Glue données ou aux cibles. Pour plus d'informations, consultez Utilisation des connexions sur la console AWS Glue.	10 octobre 2019

Prise en charge de Python wheel	Ajout d'informations sur la prise en AWS Glue charge des fichiers wheel (ainsi que des fichiers egg) en tant que dépendances pour les tâches shell Python. Pour plus d'informations, consultez Ajout de votre propre bibliothèque Python.	26 septembre 2019
Support pour le versionnement des terminaux de développement dans AWS Glue	Ajout d'informations sur la définition des points de terminaison `Glue version` en cours de développement. `Glue version`détermine les versions d'Apache Spark et de Python prises AWS Glue en charge. Pour plus d'informations, consultez Ajout d'un point de terminaison de développement.	19 septembre 2019
Support pour la surveillance à AWS Glue l'aide de l'interface utilisateur Spark	Ajout d'informations sur l'utilisation de l'interface utilisateur d'Apache Spark pour surveiller et déboguer les tâches AWS Glue ETL exécutées sur le système de AWS Glue tâches, ainsi que sur les applications Spark sur les terminaux de AWS Glue développement. Pour plus d'informations, consultez la section Surveillance à AWS Glue l'aide de l'interface utilisateur Spark.	19 septembre 2019

Amélioration de la prise en charge du développement de scripts ETL locaux à l'aide de la bibliothèque AWS Glue ETL publique	Le contenu de la bibliothè que AWS Glue ETL a été mis à jour pour indiquer que AWS Glue la version 1.0 est désormais prise en charge. Pour plus d'informations, voir Développement et test de scripts ETL localement à l'aide de la bibliothèque AWS Glue ETL.	18 septembre 2019
Prise en charge de l'exclusi on des classes de stockage Amazon S3 lors de l'exécution de tâches	Ajout d'informations sur l'exclusion des classes de stockage Amazon S3 lors de l'exécution de tâches AWS Glue ETL qui lisent des fichiers ou des partition s depuis Amazon S3. Pour de plus amples informations, veuillez consulter Exclusion des classes de stockage Amazon S3.	29 août 2019
Support pour le développe ment de scripts ETL locaux à l'aide de la bibliothèque AWS Glue ETL publique	Ajout d'informations sur la façon de développer et de tester des scripts ETL Python et Scala localement sans avoir besoin d'une connexion réseau. Pour plus d'informa tions, voir Développement et test de scripts ETL localemen t à l'aide de la bibliothèque AWS Glue ETL.	28 août 2019

Problèmes connus	Ajout d'informations sur les problèmes connus liés à AWS Glue. Pour plus d'informations, consultez Problèmes connus liés à AWS Glue.	28 août 2019
Support à l'apprentissage automatique se transforme en AWS Glue	Ajout d'informations sur les fonctionnalités d'apprentissage automatique fournies par AWS Glue pour créer des transformations personnalisées. Vous pouvez créer ces transformations lorsque vous créez une tâche. Pour plus d'informations, consultez Machine Learning Transforms in AWS Glue.	8 août 2019
Prise en charge du cloud privé virtuel partagé d'Amazon	Ajout d'informations sur la AWS Glue prise en charge du partage d'Amazon Virtual Private Cloud. Pour plus d'informations, consultez Shared Amazon VPCs.	6 août 2019
Support pour la gestion des versions dans AWS Glue	Ajout d'informations sur la définition `Glue version` des propriétés de la tâche. AWS Glue version détermine les versions d'Apache Spark et de Python prises AWS Glue en charge. Pour plus d'informations, consultez la section Ajout de tâches AWS Glue.	24 juillet 2019

Prise en charge d'options de configuration supplémentaires pour les points de terminaison de développement	Ajout d'informations sur les options de configuration pour les points de terminaison de développement ayant des charges de travail gourmandes en mémoire. Vous avez le choix entre deux nouvelles configurations qui offrent une plus grande capacité de mémoire par exécuteur. Pour plus d'informations, consultez la section Utilisation des points de terminaison de développement sur la AWS Glue console.	24 juillet 2019
Prise en charge de l'exécution d'activités d'extraction, de transfert et de chargement (ETL) à l'aide de flux de travail	Ajout d'informations sur l'utilisation d'une nouvelle structure appelée flux de travail pour concevoir une activité complexe d'extraction, de transformation et de chargement (ETL) multi-tâches AWS Glue pouvant être exécutée et suivie en tant qu'entité unique. Pour plus d'informations, voir Exécution d'activités ETL complexes à l'aide de flux de travail dans AWS Glue.	20 juin 2019

Prise en charge de Python 3.6 dans les tâches shell Python	Ajout d'informations sur la prise en charge de Python 3.6 dans les tâches shell Python. Vous pouvez spécifier Python 2.7 ou Python 3.6 en tant que propriété de tâche. Pour en savoir plus, consultez Ajout de tâches shell Python dans AWS Glue.	5 juin 2019
Prise en charge des points de terminaison du Virtual Private Cloud (VPC)	Ajout d'informations sur la connexion directe AWS Glue via un point de terminaison d'interface dans votre VPC. Lorsque vous utilisez un point de terminaison d'interface VPC, la communication entre votre VPC et celui-ci AWS Glue s'effectue entièrement et en toute sécurité au sein du réseau. AWS Pour plus d'informations, consultez la section Utilisation AWS Glue avec des points de terminaison VPC.	4 juin 2019

Support pour l'enregistrement continu et en temps réel des AWS Glue tâches.	Ajout d'informations sur l'activation et l'affichage des journaux des tâches Apache Spark en temps réel, CloudWatch notamment les journaux des pilotes, les journaux de chacun des exécuteurs et une barre de progression des tâches Spark. Pour plus d'informations, consultez Journalisation continue des tâches AWS Glue.	28 mai 2019
Prise en charge des tableaux du catalogue de données existantes en tant que sources du crawler	Ajout d'informations sur la spécification d'une liste des tables Data Catalog existantes en tant que sources du crawler. Les crawlers peuvent ensuite détecter des modifications apportées aux schémas de table, mettre à jour les définitions de table et enregistrer de nouvelles partitions au fur et à mesure que de nouvelles données sont disponibles. Pour plus d'informations, consultez Propriétés de l'crawler.	10 mai 2019

Prise en charge des options de configuration supplémentaires pour les tâches gourmandes en mémoire	Ajout d'informations sur les options de configuration pour les tâches Apache Spark avec des charges de travail gourmandes en mémoire. Vous avez le choix entre deux nouvelles configurations qui offrent une plus grande capacité de mémoire par exécuteur. Pour plus d'informations, consultez la section Ajout de tâches AWS Glue.	5 avril 2019
Prise en charge de classifieurs personnalisés CSV	Ajout d'informations sur l'utilisation d'un classifieur CSV personnalisé pour déduire le schéma de différents types de données CSV. Pour plus d'informations, consultez Écriture de classifieurs personnalisés.	26 mars 2019
Support pour les balises AWS de ressources	Ajout d'informations sur l'utilisation AWS des balises de ressources pour vous aider à gérer et à contrôler l'accès à vos AWS Glue ressources. Vous pouvez attribuer AWS des balises de ressources aux tâches, aux déclencheurs, aux points de terminaison et aux robots d'exploration dans. AWS Glue Pour plus d'informations, consultez la section AWS Tags in AWS Glue.	20 mars 2019

Prise en charge du catalogue de données pour les tâches Spark SQL	Ajout d'informations sur la configuration de vos AWS Glue tâches et de vos points de terminaison de développement pour les utiliser AWS Glue Data Catalog en tant que métastore Apache Hive externe. Ceci permet aux tâches et points de terminaison de développement d'exécuter directement des requêtes Apache Spark SQL sur les tables stockées dans le catalogue de données AWS Glue Data Catalog. Pour plus d'informations, consultez Prise en charge de AWS Glue Data Catalog des tâches Spark SQL.	14 mars 2019
Prise en charge des tâches shell Python	Ajout d'informations sur les tâches shell Python et le nouveau champ Maximum capacity (Capacité maximum). Pour en savoir plus, consultez Ajout de tâches shell Python dans AWS Glue.	18 janvier 2019

Prise en charge des notifications en cas de modification des base de données et des tableaux	Ajout d'informations sur les événements qui sont générés en cas de modification au niveau des appels d'API des bases de données, tables et partitions. Vous pouvez configurer des actions dans CloudWatch Events pour répondre à ces événements. Pour plus d'informations, consultez Automatisation à l' AWS Glue aide d' CloudWatch événements.	16 janvier 2019
Prise en charge du chiffrage de mots de passe de connexion	Ajout d'informations concernant le chiffrement de mots de passe utilisés dans les objets de connexion. Pour plus d'informations, consultez Chiffrement des mots de passe de connexion.	11 décembre 2018
Prise en charge des autorisations au niveau des ressources et des politiques basées sur les ressources	Ajout d'informations sur l'utilisation des autorisations au niveau des ressources et des politiques basées sur les ressources avec. AWS Glue Pour plus d'informations, consultez les rubriques répertoriées dans Sécurité dans AWS Glue.	15 octobre 2018

Support pour les ordinateurs portables SageMaker dotés d'intelligence artificielle	Ajout d'informations sur l'utilisation de blocs-notes SageMaker IA avec des terminaux AWS Glue de développement. Pour plus d'informations, consultez Gestion des blocs-notes.	5 octobre 2018
Prise en charge du chiffrement	Ajout d'informations sur l'utilisation du chiffrement avec AWS Glue. Pour de plus amples informations, veuillez consulter Chiffrement au repos, Chiffrement en transit et Configuration du chiffrement dans AWS Glue.	24 août 2018

Prise en charge des métriques de tâches Apache Spark	Informations supplémentaires sur l'utilisation des métriques Apache Spark pour un meilleur débogage et un meilleur profilage des tâches ETL. Vous pouvez facilement suivre les indicateurs d'exécution tels que les octets lus et écrits, l'utilisation de la mémoire et la charge du processeur du pilote et des exécuteurs, ainsi que les transferts de données entre les exécuteurs depuis la console. AWS Glue Pour plus d'informations, consultez les sections Surveillance AWS Glue à l'aide de CloudWatch métriques, Surveillance et débogage des tâches, et Utilisation des tâches sur la AWS Glue console.	13 juillet 2018
Prise en charge de DynamoDB en tant que source de données	Informations supplémentaires sur l'indexation de DynamoDB et sur son utilisation en tant que source de données de tâches ETL. Pour plus d'informations, consultez Catalogage de tables avec un crawler et Paramètres de connexion.	10 juillet 2018

Mises à jour pour créer une procédure de serveur de bloc-notes	Informations actualisées sur la façon de créer un serveur de bloc-notes sur une instance Amazon EC2 associée à un point de terminaison de développement. Pour plus d'informations, consultez Création d'un serveur de bloc-notes associé à un point de terminaison de développement.	9 juillet 2018
Mises à jour disponibles sur RSS	Vous pouvez à présent vous abonner à un flux RSS pour recevoir les notifications des mises à jour du Guide du développeur AWS Glue .	25 juin 2018
Prise en charge des notifications de délai d'attente pour les tâches	Ajout d'informations sur la configuration d'un seuil de délai d'attente lorsqu'une tâche est exécutée. Pour plus d'informations, consultez Ajout de tâches dans AWS Glue.	25 mai 2018
Configuration d'un crawler pour ajouter de nouvelles colonnes	Ajout d'informations sur la nouvelle option de configuration pour les robots d'exploration, MergeNewColumns. Pour plus d'informations, consultez Configuration d'un crawler.	7 mai 2018

Prise en charge du délai d'expiration des tâches	Ajout d'informations sur la définition d'un seuil de délai d'expiration lorsqu'une tâche est exécutée. Pour plus d'informations, consultez Ajout de tâches dans AWS Glue.	10 avril 2018
Prise en charge du script ETL Scala et déclenchement des tâches en fonction des statuts d'exécution	Informations supplémentaires sur l'utilisation de Scala comme langage de programmation ETL. De plus, l'API de déclenchement prend désormais en charge le déclenchement lorsque l'une des conditions est remplie (en plus de toutes les conditions). En outre, les tâches peuvent être déclenchées en fonction de l'« échec » ou de l'« arrêt » d'une tâche exécutée (en plus d'une exécution de tâche « réussie »).	12 janvier 2018

Mises à jour antérieures

Le tableau ci-après décrit les modifications importantes apportées dans chaque version du Manuel du développeur AWS Glue avant janvier 2018.

Modifier	Description	Date
Prise en charge des sources de données XML et nouvelle option de configuration du crawler	Ajout d'informations sur la classification des sources de données XML et nouvelle option crawler pour les modifications de partition.	16 novembre 2017

Modifier	Description	Date
Prise en charge de nouvelles transformations pour les moteurs de base de données Amazon RDS supplémentaires et améliorations des points de terminaison de développement	Ajout d'informations sur les transformations de filtrage et de mappage, prise en charge de Microsoft SQL Server sur Amazon RDS et Oracle sur Amazon RDS, et nouvelles fonctions pour les points de terminaison de développement.	29 septembre 2017
AWS Glue publication initiale	Il s'agit de la première version du Guide du développeur AWS Glue .	14 août 2017

AWS Glossaire

Pour la AWS terminologie la plus récente, consultez le AWS glossaire dans la Glossaire AWS référence.

Les traductions sont fournies par des outils de traduction automatique. En cas de conflit entre le contenu d'une traduction et celui de la version originale en anglais, la version anglaise prévaudra.