Usando o Apache Iceberg em AWS

AWS Orientação prescritiva

AWS Orientação prescritiva: Usando o Apache Iceberg em AWS

Table of Contents

Usando o Apache Iceberg em AWS

Amazon Web Services (colaboradores)

Novembro de 2025 (histórico do documento)

O Apache Iceberg é um formato de tabela de código aberto que simplifica o gerenciamento de tabelas e melhora o desempenho. AWS serviços de análise, como Amazon EMR AWS Glue, Amazon Athena e Amazon Redshift, incluem suporte nativo para o Iceberg, para que você possa criar facilmente lagos de dados transacionais com base no Amazon Simple Storage Service (Amazon S3). AWS

Além disso, a próxima geração da Amazon SageMaker é construída em uma arquitetura de lakehouse aberta que unifica o acesso aos dados em lagos de dados, armazéns de AWS dados e fontes terceirizadas e federadas. O lakehouse é totalmente compatível com o Iceberg e oferece a flexibilidade de acessar e consultar dados no local usando a API REST do Iceberg.

Este guia técnico fornece orientação sobre como começar a usar o Iceberg em diferentes Serviços da AWSáreas e inclui as melhores práticas e recomendações para operar o Iceberg em grande escala e, AWS ao mesmo tempo, otimizar o custo e o desempenho.

Se você está apenas começando com o Iceberg ou é um usuário experiente que deseja otimizar suas cargas de trabalho existentes do Iceberg AWS, este guia oferece informações valiosas para cada estágio do seu projeto

Neste guia:

Lagos de dados modernos

Casos de uso avançados em lagos de dados modernos

A evolução do armazenamento de dados progrediu de bancos de dados para armazéns de dados e lagos de dados, onde cada tecnologia atende aos requisitos exclusivos de negócios e dados. Os bancos de dados tradicionais se destacaram no tratamento de dados estruturados e cargas de trabalho transacionais, mas enfrentaram desafios de desempenho à medida que os volumes de dados aumentavam. Os data warehouses surgiram para resolver problemas de desempenho e escalabilidade, mas, assim como os bancos de dados, eles dependiam de formatos proprietários em sistemas integrados verticalmente.

Os data lakes oferecem uma das melhores opções para armazenar dados em termos de custo, escalabilidade e flexibilidade. Você pode usar um data lake para reter grandes volumes de dados estruturados e não estruturados a um custo baixo e usar esses dados para diferentes tipos de cargas de trabalho de análise, desde relatórios de business intelligence até processamento de big data, análises em tempo real, aprendizado de máquina e inteligência artificial generativa (IA), para ajudar a orientar melhores decisões.

Apesar desses benefícios, os data lakes não foram projetados inicialmente com recursos semelhantes aos de bancos de dados. Um data lake não fornece suporte para semântica de processamento de atomicidade, consistência, isolamento e durabilidade (ACID), que você pode precisar para otimizar e gerenciar seus dados de forma eficaz em escala para centenas ou milhares de usuários usando muitas tecnologias diferentes. Os data lakes não fornecem suporte nativo para as seguintes funcionalidades:

- Realizar atualizações e exclusões eficientes em nível de registro à medida que os dados mudam em sua empresa
- Gerenciando o desempenho das consultas à medida que as tabelas crescem para milhões de arquivos e centenas de milhares de partições
- Garantindo a consistência dos dados entre vários escritores e leitores simultâneos
- Prevenir a corrupção de dados quando as operações de gravação falham no meio da operação
- Esquemas de tabela em evolução ao longo do tempo sem reescrever (parcialmente) conjuntos de dados

Esses desafios se tornaram particularmente predominantes em casos de uso, como lidar com captura de dados alterados (CDC) ou casos de uso relacionados à privacidade, exclusão de dados e ingestão de dados por streaming, o que pode resultar em tabelas abaixo do ideal.

Os data lakes que usam as tabelas tradicionais no formato Hive oferecem suporte a operações de gravação somente para arquivos inteiros. Isso torna as atualizações e exclusões difíceis de implementar, demoradas e caras. Além disso, controles e garantias de concorrência oferecidos em sistemas compatíveis com ACID são necessários para garantir a integridade e a consistência dos dados.

Esses desafios deixam os usuários com um dilema: escolher entre uma plataforma totalmente integrada, mas proprietária, ou optar por um data lake autoconstruído, independente do fornecedor, mas com uso intensivo de recursos, que requer manutenção e migração constantes para obter seu valor potencial.

Para ajudar a superar esses desafios, o Iceberg fornece uma funcionalidade adicional semelhante a um banco de dados que simplifica a otimização e a sobrecarga de gerenciamento dos lagos de dados, ao mesmo tempo em que oferece suporte ao armazenamento em sistemas econômicos, como o Amazon S3.

Introdução ao Apache Iceberg

O Apache Iceberg é um formato de tabela de código aberto que fornece recursos em tabelas de data lake que, historicamente, estavam disponíveis apenas em bancos de dados ou data warehouses. Ele foi projetado para oferecer escala e desempenho e é adequado para gerenciar tabelas com mais de centenas de gigabytes. Algumas das principais características das mesas Iceberg são:

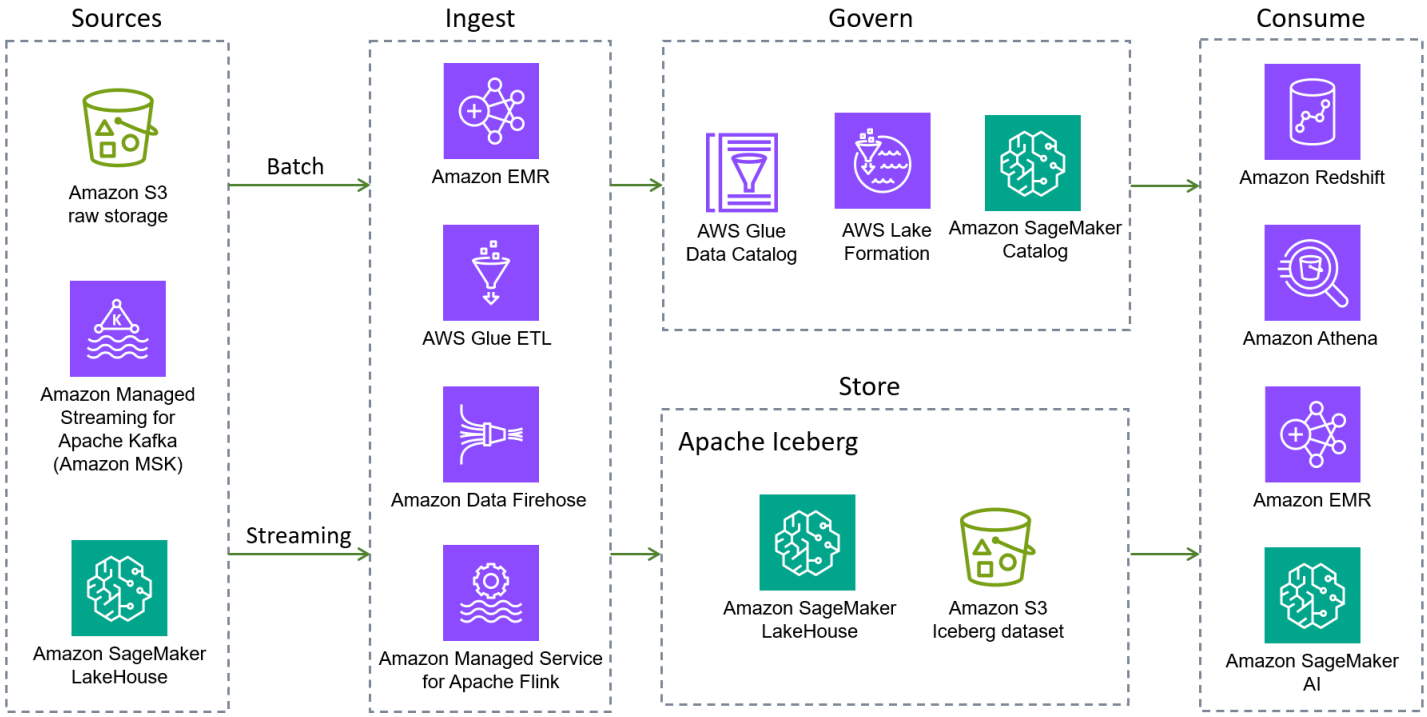
- Excluir, atualizar e mesclar.O Iceberg suporta comandos SQL padrão para armazenamento de dados para uso com tabelas de data lake.
- Planejamento rápido de escaneamento e filtragem avançada. O Iceberg armazena metadados, como estatísticas em nível de partição e coluna, que podem ser usados pelos mecanismos para acelerar o planejamento e a execução de consultas.
- Evolução completa do esquema. O Iceberg suporta adicionar, descartar, atualizar ou renomear colunas sem efeitos colaterais.
- Evolução da partição. Você pode atualizar o layout da partição de uma tabela à medida que o volume de dados ou os padrões de consulta mudam. O Iceberg suporta a alteração das colunas nas quais uma tabela é particionada, a adição ou remoção de colunas de partições compostas.

- Particionamento oculto.Esse recurso impede a leitura automática de partições desnecessárias. Isso elimina a necessidade de os usuários entenderem os detalhes de particionamento da tabela ou adicionarem filtros extras às consultas.
- Reversão da versão. Os usuários podem corrigir problemas rapidamente voltando ao estado anterior à transação.
- Viagem no tempo. Os usuários podem consultar uma versão anterior específica de uma tabela.
- Isolamento serializável. As alterações na tabela são atômicas, então os leitores nunca veem alterações parciais ou não confirmadas.
- Escritores simultâneos. O Iceberg usa simultaneidade otimista para permitir que várias transações sejam bem-sucedidas. Em caso de conflito, um dos redatores precisa repetir a transação.
- Formatos de arquivo abertos. O Iceberg suporta vários formatos de arquivo de código aberto, incluindo Apache Parquet, Apache Avro e Apache ORC.

Em resumo, os data lakes que usam o formato Iceberg se beneficiam da consistência transacional, da velocidade, da escala e da evolução do esquema. Para obter mais informações sobre esses e outros recursos do Iceberg, consulte a documentação do Apache Iceberg.

AWS suporte para Apache Iceberg

O Apache Iceberg é suportado por Serviços da AWS empresas como Amazon EMR, AmazonAthena, Amazon Redshift e Amazon. AWS Glue SageMaker O diagrama a seguir mostra uma arquitetura de referência simplificada de um data lake baseado no Iceberg.



O seguinte Serviços da AWS fornece integrações nativas do Iceberg. Existem outros Serviços da AWS que podem interagir com o Iceberg, indiretamente ou empacotando as bibliotecas do Iceberg.

- O Amazon S3 é o melhor lugar para criar lagos de dados devido à sua durabilidade, disponibilidade, escalabilidade, segurança, conformidade e recursos de auditoria. O Iceberg foi projetado e construído para interagir perfeitamente com o Amazon S3 e fornece suporte para muitos recursos do Amazon S3, conforme listado na documentação do Iceberg. Além disso, as tabelas do Amazon S3 oferecem o primeiro armazenamento de objetos em nuvem com suporte integrado ao Iceberg e simplificam o armazenamento de dados tabulares em grande escala. Com o suporte do S3 Tables para o Iceberg, você pode consultar facilmente seus dados tabulares usando mecanismos de consulta populares AWS e de terceiros.
- A próxima geração do SageMaker é baseada em uma arquitetura de lakehouse aberta que unifica o acesso aos dados em lagos de dados do Amazon S3, armazéns de dados do Amazon Redshift e fontes de dados federadas e terceirizadas. Esses recursos ajudam você a criar análises e AI/ML aplicativos poderosos em uma única cópia dos dados. O lakehouse é totalmente compatível com o Iceberg, então você tem a flexibilidade de acessar e consultar dados no local usando a API REST do Iceberg.
- O Amazon EMR é uma solução de big data para processamento de dados em escala de petabytes, análise interativa e aprendizado de máquina usando estruturas de código aberto, como Apache Spark, Flink, Trino e Hive. O Amazon EMR pode ser executado em clusters

personalizados do Amazon Elastic Compute Cloud (Amazon EC2), no Amazon Elastic Kubernetes Service (Amazon EKS) AWS Outposts ou no Amazon EMR Serverless.

- O Amazon Athena é um serviço de análise interativo e sem servidor baseado em estruturas de código aberto. Ele suporta formatos de tabela aberta e de arquivo e fornece uma maneira simplificada e flexível de analisar petabytes de dados onde eles estão. O Athena fornece suporte nativo para consultas de leitura, viagem no tempo, gravação e DDL para o Iceberg e usa o AWS Glue Data Catalog metastore for Iceberg.
- O Amazon Redshift é um data warehouse em nuvem na escala de petabytes que oferece suporte a opções de implantação baseadas em cluster e sem servidor. O Amazon Redshift Spectrum pode consultar tabelas externas registradas e armazenadas no Amazon S3. AWS Glue Data Catalog O Redshift Spectrum também fornece suporte para o formato de armazenamento Iceberg.
- AWS Glueé um serviço de integração de dados sem servidor que facilita a descoberta, a preparação, a movimentação e a integração de dados de várias fontes para análise, aprendizado de máquina (ML) e desenvolvimento de aplicativos. É totalmente integrado ao Iceberg. Especificamente, você pode realizar operações de leitura e gravação em tabelas do Iceberg usando AWS Glue trabalhos, gerenciar tabelas por meio do AWS Glue Data Catalog(compatível com a metástore do Hive), descobrir e registrar tabelas automaticamente usando AWS Glue rastreadores e avaliar a qualidade dos dados nas tabelas do Iceberg por meio do recurso Qualidade de dados. AWS Glue O AWS Glue Data Catalog também suporta a coleta de estatísticas de colunas, o cálculo e a atualização do número de valores distintos (NDVs) para cada coluna nas tabelas do Iceberg e otimizações automáticas de tabelas (compactação, retenção de instantâneos, exclusão de arquivos órfãos). AWS Glue também oferece suporte a integrações sem ETL de uma lista de aplicativos Serviços da AWS e de terceiros em tabelas Iceberg.
- O Amazon Data Firehose é um serviço totalmente gerenciado para fornecer dados de streaming em tempo real para destinos como Amazon S3, Amazon Redshift, Amazon OpenSearch Service, OpenSearch Amazon Serverless, Splunk, Apache Iceberg e quaisquer endpoints HTTP ou HTTP personalizados de propriedade de provedores de serviços terceirizados compatíveis, incluindo Datadog, Dynatrace, MongoDB, New Relic, Coralogix, e elástico. LogicMonitor Com o Firehose, você não precisa criar aplicativos nem gerenciar recursos. Configure os produtores de dados para enviar dados ao Firehose e ele entregará automaticamente os dados ao destino especificado. Você também pode configurar o Firehose para transformar os dados antes de entregá-los.
- O Amazon Managed Service para Apache Flink é um serviço totalmente gerenciado da Amazon que permite usar um aplicativo Apache Flink para processar dados de streaming. Ele suporta leitura e gravação em tabelas Iceberg e permite processamento e análise de dados em tempo real.

- O Amazon SageMaker AI suporta o armazenamento de conjuntos de recursos na Amazon SageMaker AI Feature Store usando o formato Iceberg.
- AWS Lake Formationfornece permissões de controle de acesso gerais e refinadas para acessar dados, incluindo tabelas Iceberg consumidas pelo Athena ou pelo Amazon Redshift. Para saber mais sobre o suporte de permissões para tabelas Iceberg, consulte a documentação do Lake Formation.

AWS tem uma ampla variedade de serviços que oferecem suporte ao Iceberg, mas cobrir todos esses serviços está além do escopo deste guia. As seções a seguir abordam o Spark (streaming estruturado e em lote) no Amazon EMR AWS Glue e também no Athena SQL. A seção a seguir fornece uma visão rápida do suporte do Iceberg no Athena SQL.

Introdução às tabelas Iceberg no Amazon Athena SQL

O Amazon Athena fornece suporte integrado para o Iceberg. Você pode usar o Iceberg sem nenhuma etapa ou configuração adicional, exceto para definir os pré-requisitos de serviço detalhados na seção Introdução da documentação do Athena. Esta seção fornece uma breve introdução à criação de tabelas no Athena. Para obter mais informações, consulte Trabalhando com tabelas do Iceberg usando o Athena SQL mais adiante neste guia.

Você pode criar tabelas Iceberg AWS usando mecanismos diferentes. Essas tabelas funcionam perfeitamente em todas Serviços da AWS as partes. Para criar suas primeiras tabelas Iceberg com o Athena SQL, você pode usar o seguinte código clichê.

```
CREATE TABLE <table_name> (
    col_1 string,
    col_2 string,
    col_3 bigint,
    col_ts timestamp)
PARTITIONED BY (col_1, <<<partition_transform>>>(col_ts))
LOCATION 's3://<bucket>/<folder>/<table_name>/'
TBLPROPERTIES (
    'table_type' ='ICEBERG'
)
```

As seções a seguir fornecem exemplos de criação de tabelas Iceberg particionadas e não particionadas no Athena. Para obter mais informações, consulte a sintaxe do Iceberg detalhada na documentação do Athena.

Criação de uma tabela não particionada

O exemplo de instrução a seguir personaliza o código SQL padronizado para criar uma tabela Iceberg não particionada no Athena. Você pode adicionar essa instrução ao editor de consultas no console do Athena para criar a tabela.

```
CREATE TABLE athena_iceberg_table (
    color string,
    date string,
    name string,
    price bigint,
```

```
    product string,
    ts timestamp)
LOCATION 's3://DOC_EXAMPLE_BUCKET/ice_warehouse/iceberg_db/athena_iceberg_table/'
TBLPROPERTIES (
    'table_type' ='ICEBERG'
)
```

Para step-by-step obter instruções sobre como usar o editor de consultas, consulte Introdução na documentação do Athena.

Criar uma tabela particionada

A declaração a seguir cria uma tabela particionada com base na data usando o conceito de particionamento oculto do Iceberg. Ele usa a `day()` transformação para derivar partições diárias, usando o `dd-mm-yyyy` formato, de uma coluna de carimbo de data/hora. O Iceberg não armazena esse valor como uma nova coluna no conjunto de dados. Em vez disso, o valor é derivado dinamicamente quando você grava ou consulta dados.

```
CREATE TABLE athena_iceberg_table_partitioned (
    color string,
    date string,
    name string,
    price bigint,
    product string,
    ts timestamp)
PARTITIONED BY (day(ts))
LOCATION 's3://DOC_EXAMPLE_BUCKET/ice_warehouse/iceberg_db/athena_iceberg_table/'
TBLPROPERTIES (
    'table_type' ='ICEBERG'
)
```

Criando uma tabela e carregando dados com uma única instrução CTAS

Nos exemplos particionados e não particionados das seções anteriores, as tabelas Iceberg são criadas como tabelas vazias. Você pode carregar dados nas tabelas usando a `MERGE` instrução `INSERT` or. Como alternativa, você pode usar uma `CREATE TABLE AS SELECT (CTAS)` instrução para criar e carregar dados em uma tabela Iceberg em uma única etapa.

O CTAS é a melhor maneira no Athena de criar uma tabela e carregar dados em uma única instrução. O exemplo a seguir ilustra como usar o CTAS para criar uma tabela Iceberg () a partir de uma Hive/Parquet tabela existente (`iceberg_ctas_table`) `hive_table` no Athena.

```
CREATE TABLE iceberg_ctas_table WITH (
  table_type = 'ICEBERG',
  is_external = false,
  location = 's3://DOC_EXAMPLE_BUCKET/ice_warehouse/iceberg_db/iceberg_ctas_table/'
) AS
SELECT * FROM "iceberg_db"."hive_table" limit 20
---
SELECT * FROM "iceberg_db"."iceberg_ctas_table" limit 20
```

Para saber mais sobre o CTAS, consulte a documentação do Athena CTAS.

Inserindo, atualizando e excluindo dados

O Athena oferece suporte a diferentes formas de gravar dados em uma tabela Iceberg usando as `INSERT INTO` instruções, `UPDATEMERGE INTO`, e M. `DELETE FRO`

> **Note**
>
> Atualmente, o Athena SQL não oferece suporte à copy-on-write abordagem. `UPDATE`,`MERGE INTO`, e `DELETE FROM` as operações sempre usam a merge-on-read abordagem com exclusões posicionais, independentemente das propriedades especificadas da tabela. Se você configurar propriedades de tabela como`write.update.mode`, e `write.delete.mode` para usar `write.merge.mode` copy-on-write, suas consultas não falharão, mas o Athena as ignorará e continuará usando. merge-on-read

A declaração a seguir é usada `INSERT INTO` para adicionar dados a uma tabela do Iceberg:

```
INSERT INTO "iceberg_db"."ice_table" VALUES (
    'red', '222022-07-19T03:47:29', 'PersonNew', 178, 'Tuna', now()
)

SELECT * FROM "iceberg_db"."ice_table"
where color = 'red' limit 10;
```

Exemplo de saída:

Results (1)

Copy | Download results

Search rows

1

#	color	date	name	price	product	ts
1	red	222022-07-19T03:47:29	PersonNew	178	Tuna	2023-10-11 11:35:01.298000 UTC

Para obter mais informações, consulte a documentação do Athena.

Consultando tabelas Iceberg

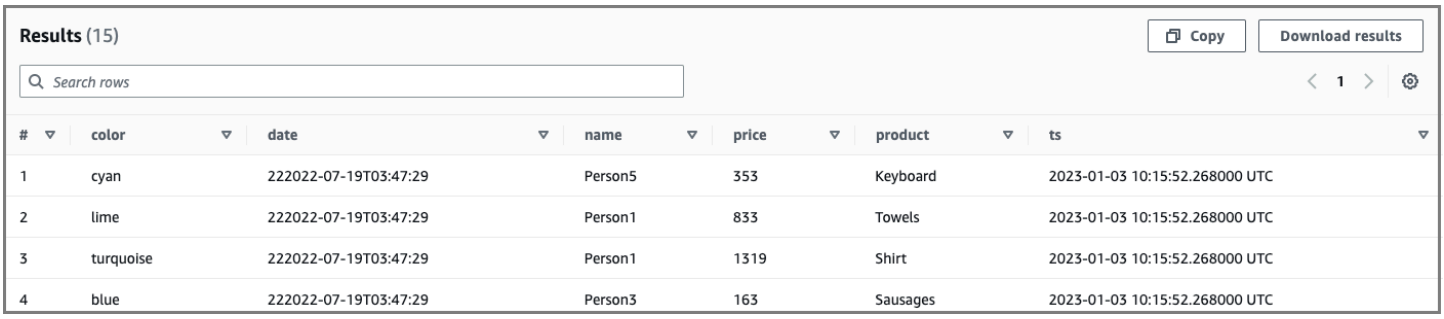
Você pode executar consultas SQL regulares em suas tabelas do Iceberg usando o Athena SQL, conforme ilustrado no exemplo anterior.

Além das consultas usuais, o Athena também oferece suporte a consultas de viagem no tempo para tabelas Iceberg. Conforme discutido anteriormente, você pode alterar os registros existentes por meio de atualizações ou exclusões em uma tabela do Iceberg, portanto, é conveniente usar consultas de viagem no tempo para examinar as versões mais antigas da sua tabela com base em um carimbo de data/hora ou em um ID de instantâneo.

Por exemplo, a instrução a seguir atualiza um valor de cor para e`Person5`, em seguida, exibe um valor anterior de 4 de janeiro de 2023:

```
UPDATE ice_table SET color='new_color' WHERE name='Person5'

SELECT * FROM "iceberg_db"."ice_table" FOR TIMESTAMP AS OF TIMESTAMP '2023-01-04
 12:00:00 UTC'
```

Exemplo de saída:



Results (15)

Copy | Download results

Search rows

1

#	color	date	name	price	product	ts
1	cyan	222022-07-19T03:47:29	Person5	353	Keyboard	2023-01-03 10:15:52.268000 UTC
2	lime	222022-07-19T03:47:29	Person1	833	Towels	2023-01-03 10:15:52.268000 UTC
3	turquoise	222022-07-19T03:47:29	Person1	1319	Shirt	2023-01-03 10:15:52.268000 UTC
4	blue	222022-07-19T03:47:29	Person3	163	Sausages	2023-01-03 10:15:52.268000 UTC

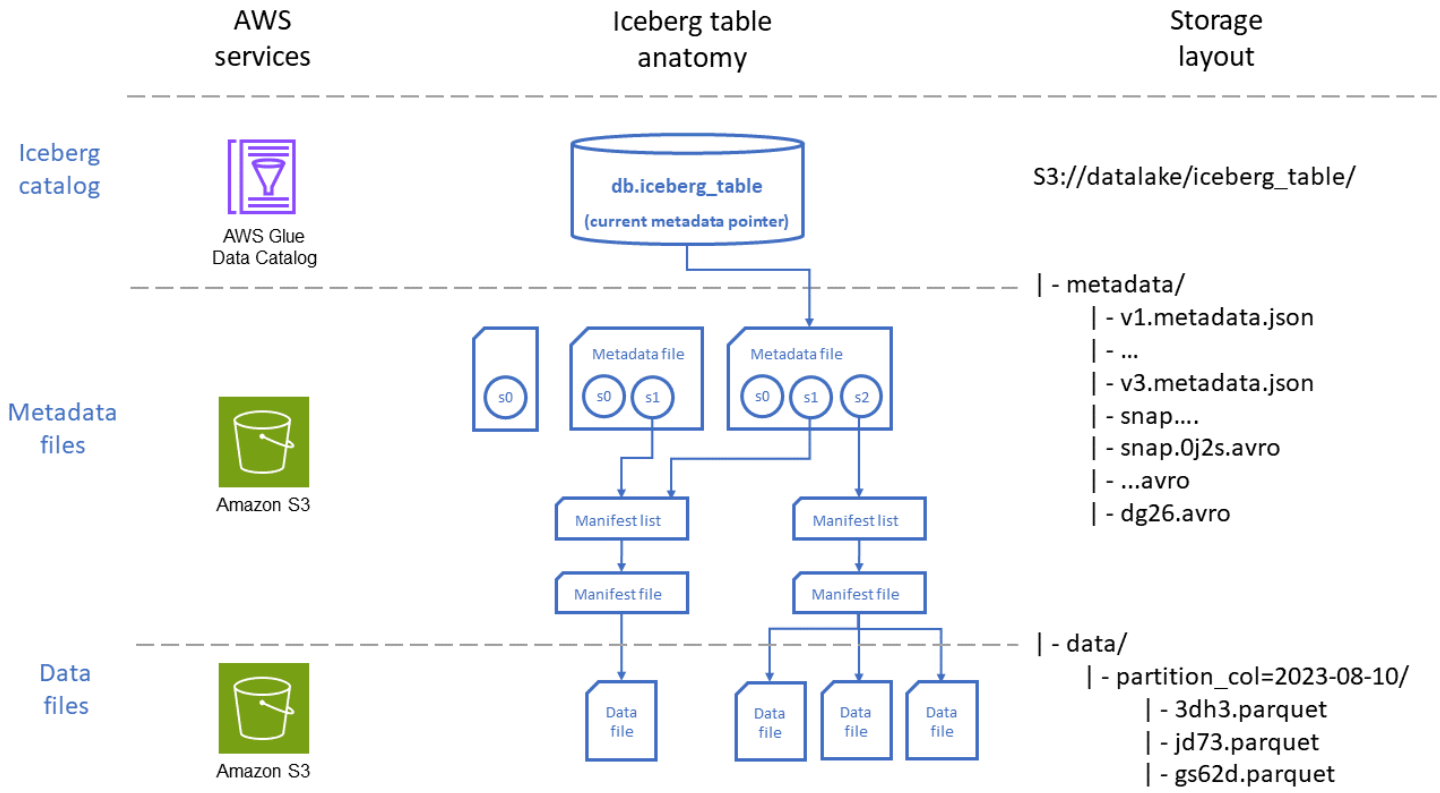
Para ver exemplos adicionais e de sintaxe de viagens no tempo, consulte a documentação do Athena.

Anatomia da mesa de iceberg

Agora que abordamos as etapas básicas do trabalho com mesas Iceberg, vamos nos aprofundar nos detalhes intrincados e no design de uma mesa Iceberg.

Para habilitar os recursos descritos anteriormente neste guia, o Iceberg foi projetado com camadas hierárquicas de dados e arquivos de metadados. Essas camadas gerenciam os metadados de forma inteligente para otimizar o planejamento e a execução de consultas.

O diagrama a seguir retrata a organização de uma tabela Iceberg por meio de duas perspectivas: a Serviços da AWS usada para armazenar a tabela e a colocação do arquivo no Amazon S3.



Conforme mostrado no diagrama, uma tabela Iceberg consiste em três camadas principais:

- Catálogo Iceberg: AWS Glue Data Catalog integra-se nativamente ao Iceberg e é, para a maioria dos casos de uso, a melhor opção para cargas de trabalho executadas em. AWS Os serviços que interagem com as tabelas do Iceberg (por exemplo, Athena) usam o catálogo para encontrar a versão instantânea atual da tabela, seja para ler ou gravar dados.
- Camada de metadados: os arquivos de metadados, ou seja, os arquivos de manifesto e os arquivos de lista de manifestos, controlam informações como o esquema das tabelas, a estratégia de partição e a localização dos arquivos de dados, bem como estatísticas em nível de coluna,

como intervalos mínimos e máximos dos registros armazenados em cada arquivo de dados. Esses arquivos de metadados são armazenados no Amazon S3 dentro do caminho da tabela.

  - Os arquivos de manifesto contêm um registro para cada arquivo de dados, incluindo sua localização, formato, tamanho, soma de verificação e outras informações relevantes.
  - As listas de manifestos fornecem um índice dos arquivos de manifesto. À medida que o número de arquivos de manifesto aumenta em uma tabela, dividir essas informações em subseções menores ajuda a reduzir o número de arquivos de manifesto que precisam ser verificados por consultas.
  - Os arquivos de metadados contêm informações sobre toda a tabela Iceberg, incluindo as listas de manifestos, os esquemas, os metadados da partição, os arquivos de instantâneos e outros arquivos usados para gerenciar os metadados da tabela.
- Camada de dados: essa camada contém os arquivos que têm os registros de dados nos quais as consultas serão executadas. Esses arquivos podem ser armazenados em diferentes formatos, incluindo Apache Parquet, ApacheAvro e Apache ORC.
  - Os arquivos de dados contêm os registros de dados de uma tabela.
  - Os arquivos de exclusão codificam operações de exclusão e atualização em nível de linha em uma tabela Iceberg. O Iceberg tem dois tipos de arquivos de exclusão, conforme descrito na documentação do Iceberg. Esses arquivos são criados por operações usando o merge-on-read modo.

Trabalhando com a Iceberg no Amazon EMR

O Amazon EMR fornece processamento de dados em escala de petabytes, análises interativas e aprendizado de máquina na nuvem usando estruturas de código aberto, como Apache Spark, Apache Hive, Flink e Trino.

> **Note**
>
> Este guia usa o Apache Spark como exemplos.

O Amazon EMR oferece suporte a várias opções de implantação: Amazon EMR ativado, Amazon EMR no EKS, EC2 Amazon EMR Serverless e Amazon EMR ativado. AWS Outposts Para escolher uma opção de implantação para sua carga de trabalho, consulte as perguntas frequentes do Amazon EMR.

Compatibilidade de versões e recursos

O Amazon EMR versão 6.5.0 e versões posteriores oferecem suporte nativo ao Apache Iceberg. Para obter uma lista das versões compatíveis do Iceberg para cada versão do Amazon EMR, consulte o histórico de lançamentos do Iceberg na documentação do Amazon EMR. Leia também as seções em Use um cluster com o Iceberg para ver quais recursos do Iceberg são compatíveis com o Amazon EMR em diferentes estruturas.

Recomendamos que você use a versão mais recente do Amazon EMR para se beneficiar da versão mais recente compatível do Iceberg. Os exemplos de código e as configurações nesta seção pressupõem que você esteja usando o Amazon EMR versão emr-7.8.0.

Criação de um cluster do Amazon EMR com o Iceberg

Para criar um cluster do Amazon EMR na Amazon EC2 com o Iceberg instalado, siga as instruções na documentação do Amazon EMR.

Especificamente, seu cluster deve ser configurado com a seguinte classificação:

```
[{
    "Classification": "iceberg-defaults",
    "Properties": {
```

```
        "iceberg.enabled": "true"
    }
}]
```

Você também pode optar por usar o Amazon EMR Serverless ou o Amazon EMR no EKS como opções de implantação para suas cargas de trabalho do Iceberg, a partir do Amazon EMR 6.6.0.

Desenvolvendo aplicativos Iceberg no Amazon EMR

Para desenvolver o código Spark para seus aplicativos Iceberg, você pode usar o Amazon EMR Studio, que é um ambiente de desenvolvimento integrado (IDE) baseado na web para notebooks Jupyter totalmente gerenciados que são executados em clusters do Amazon EMR.

Usando notebooks Amazon EMR Studio

Você pode desenvolver aplicativos Spark de forma interativa em notebooks Amazon EMR Studio Workspace e conectar esses notebooks ao Amazon EMR em clusters EC2 ou ao Amazon EMR em endpoints gerenciados pelo EKS. Consulte a AWS service (Serviço da AWS) documentação para obter instruções sobre como configurar um EMR Studio para Amazon EMR on EC2 EKS.

Para usar o Iceberg no EMR Studio, siga estas etapas:

1. Inicie um cluster do Amazon EMR com o Iceberg ativado, conforme as instruções em Use um cluster com o Iceberg instalado.
2. Configure um EMR Studio. Para obter instruções, consulte Configurar um Amazon EMR Studio.
3. Abra um notebook do EMR Studio Workspace e execute o código a seguir como a primeira célula no notebook para configurar sua sessão do Spark para usar o Iceberg:

```
%%configure -f
{
    "conf": {
        "spark.sql.catalog.<catalog_name>": "org.apache.iceberg.spark.SparkCatalog",
        "spark.sql.catalog.<catalog_name>.warehouse": "s3://YOUR-BUCKET-NAME/YOUR-
FOLDER-NAME/",
        "spark.sql.catalog.<catalog_name>.type": "glue",
        "spark.sql.extensions":
 "org.apache.iceberg.spark.extensions.IcebergSparkSessionExtensions"
    }
}
```

em que:

- `<catalog_name>`é o nome do catálogo de sessões do Iceberg Spark. Substitua-o por um nome de sua escolha e lembre-se de alterar as referências em todas as configurações associadas a esse catálogo. No seu código, você pode se referir às suas tabelas do Iceberg usando o nome da tabela totalmente qualificado, incluindo o nome do catálogo da sessão do Spark, da seguinte forma:

  ```
  <catalog_name>.<database_name>.<table_name>
  ```

  Como alternativa, você pode alterar o catálogo padrão para o catálogo Iceberg que você definiu configurando `spark.sql.defaultCatalog` o nome do seu catálogo. Essa segunda abordagem permite que você faça referência a tabelas sem o prefixo do catálogo, o que pode simplificar suas consultas.

- `<catalog_name>.warehouse`aponta para o caminho do Amazon S3 onde você deseja armazenar seus dados e metadados.
- Para tornar o catálogo um AWS Glue Data Catalog, `spark.sql.catalog.<catalog_name>.type` defina como`glue`. Essa chave é necessária para apontar para uma classe de implementação para qualquer implementação de catálogo personalizado. A seção de melhores práticas gerais, mais adiante neste guia, descreve os diferentes catálogos compatíveis com o Iceberg.

4. Agora você pode começar a desenvolver interativamente seu aplicativo Spark para o Iceberg no notebook, como faria com qualquer outro aplicativo Spark.

Para obter mais informações sobre como configurar o Spark para o Apache Iceberg usando o Amazon EMR Studio, consulte a postagem do blog Crie um data lake evolutivo de alto desempenho, compatível com ACID, usando o Apache Iceberg no Amazon EMR.

Executando trabalhos do Iceberg no Amazon EMR

Depois de desenvolver o código do aplicativo Spark para sua carga de trabalho do Iceberg, você pode executá-lo em qualquer opção de implantação do Amazon EMR que suporte o Iceberg (consulte as perguntas frequentes do Amazon EMR).

Assim como em outros trabalhos do Spark, você pode enviar trabalhos para um Amazon EMR EC2 em cluster adicionando etapas ou enviando interativamente trabalhos do Spark para o nó principal.

Para executar um trabalho do Spark, consulte as seguintes páginas de documentação do Amazon EMR:

- Para obter uma visão geral das diferentes opções de envio de trabalho para um Amazon EMR EC2 em cluster e instruções detalhadas para cada opção, consulte Enviar trabalho para um cluster.
- Para o Amazon EMR no EKS, consulte Executando trabalhos do Spark com. StartJobRun
- Para o EMR Serverless, consulte Executando trabalhos.

As seções a seguir fornecem um exemplo para cada opção de implantação do Amazon EMR.

Amazon EMR ativado EC2

Você pode usar essas etapas para enviar a tarefa do Iceberg Spark:

1. Crie o arquivo `emr_step_iceberg.json` com o seguinte conteúdo em sua estação de trabalho:

```
[{
    "Name": "iceberg-test-job",
    "Type": "spark",
    "ActionOnFailure": "CONTINUE",
    "Args": [
        "--deploy-mode",
        "client",
        "--conf",

  "spark.sql.extensions=org.apache.iceberg.spark.extensions.IcebergSparkSessionExtensions",
        "--conf",
        "spark.sql.catalog.<catalog_name>=org.apache.iceberg.spark.SparkCatalog",
        "--conf",
        "spark.sql.catalog.<catalog_name>.type=glue",
        "--conf",
        "spark.sql.catalog.<catalog_name>.warehouse=s3://YOUR-BUCKET-NAME/YOUR-
FOLDER-NAME/",
        "s3://YOUR-BUCKET-NAME/code/iceberg-job.py"
    ]
}]
```

2. Modifique o arquivo de configuração para sua tarefa específica do Spark personalizando as opções de configuração do Iceberg destacadas em negrito.
3. Envie a etapa usando o AWS Command Line Interface (AWS CLI). Execute o comando no diretório em que o `emr_step_iceberg.json` arquivo está localizado.

```
aws emr add-steps --cluster-id <cluster_id> --steps file://emr_step_iceberg.json
```

Amazon EMR Sem Servidor

Para enviar uma tarefa do Iceberg Spark para o EMR Serverless usando: AWS CLI

1. Crie o arquivo e`mr_serverless_iceberg.json` com o seguinte conteúdo em sua estação de trabalho:

```
{
    "applicationId": "<APPLICATION_ID>",
    "executionRoleArn": "<ROLE_ARN>",
    "name": "iceberg-test-job",
    "jobDriver": {
        "sparkSubmit": {
            "entryPoint": "s3://YOUR-BUCKET-NAME/code/iceberg-job.py",
            "entryPointArguments": []
        }
    },
    "configurationOverrides": {
        "applicationConfiguration": [{
            "classification": "spark-defaults",
            "properties": {
                "spark.sql.extensions":
 "org.apache.iceberg.spark.extensions.IcebergSparkSessionExtensions",
                "spark.sql.catalog.<catalog_name>":
 "org.apache.iceberg.spark.SparkCatalog",
                "spark.sql.catalog.<catalog_name>.type": "glue",
                "spark.sql.catalog.<catalog_name>.warehouse": "s3://YOUR-BUCKET-NAME/
YOUR-FOLDER-NAME/",
                "spark.jars":"/usr/share/aws/iceberg/lib/iceberg-spark3-runtime.jar",

 "spark.hadoop.hive.metastore.client.factory.class":"com.amazonaws.glue.catalog.metastore.AW
            }
        }],
        "monitoringConfiguration": {
            "s3MonitoringConfiguration": {
                "logUri": "s3://YOUR-BUCKET-NAME/emr-serverless/logs/"
            }
        }
    }
```

```
}
```

2. Modifique o arquivo de configuração para sua tarefa específica do Spark personalizando as opções de configuração do Iceberg destacadas em negrito.
3. Envie o trabalho usando AWS CLI o. Execute o comando no diretório em que o `emr_serverless_iceberg.json` arquivo está localizado:

```
aws emr-serverless start-job-run --cli-input-json file://emr_serverless_iceberg.json
```

Para enviar uma tarefa do Iceberg Spark para o EMR Serverless usando o console do EMR Studio:

1. Siga as instruções na documentação do EMR Serverless.
2. Para a configuração do Job, use a configuração do Iceberg para o Spark fornecida para o AWS CLI e personalize os campos destacados do Iceberg. Para obter instruções detalhadas, consulte Usando o Apache Iceberg com o EMR Serverless na documentação do Amazon EMR.

Amazon EMR no EKS

Para enviar uma tarefa do Iceberg Spark para o Amazon EMR no EKS usando: AWS CLI

1. Crie o arquivo `emr_eks_iceberg.json` com o seguinte conteúdo em sua estação de trabalho:

```
{
    "name": "iceberg-test-job",
    "virtualClusterId": "<VIRTUAL_CLUSTER_ID>",
    "executionRoleArn": "<ROLE_ARN>",
    "releaseLabel": "emr-6.9.0-latest",
    "jobDriver": {
        "sparkSubmitJobDriver": {
            "entryPoint": "s3://YOUR-BUCKET-NAME/code/iceberg-job.py",
            "entryPointArguments": [],
            "sparkSubmitParameters": "--jars local:///usr/share/aws/iceberg/lib/
iceberg-spark3-runtime.jar"
        }
    },
    "configurationOverrides": {
        "applicationConfiguration": [{
            "classification": "spark-defaults",
            "properties": {
```

```
                "spark.sql.extensions":
 "org.apache.iceberg.spark.extensions.IcebergSparkSessionExtensions",
                "spark.sql.catalog.<catalog_name>":
 "org.apache.iceberg.spark.SparkCatalog",
                "spark.sql.catalog.<catalog_name>.type": "glue",
                "spark.sql.catalog.<catalog_name>.warehouse": "s3://YOUR-BUCKET-NAME/
YOUR-FOLDER-NAME/",
                "spark.hadoop.hive.metastore.client.factory.class":
 "com.amazonaws.glue.catalog.metastore.AWSGlueDataCatalogHiveClientFactory"
            }
        }],
        "monitoringConfiguration": {
            "persistentAppUI": "ENABLED",
            "s3MonitoringConfiguration": {
                "logUri": "s3://YOUR-BUCKET-NAME/emr-serverless/logs/"
            }
        }
    }
}
```

2. Modifique o arquivo de configuração da sua tarefa do Spark personalizando as opções de configuração do Iceberg destacadas em negrito.
3. Envie o trabalho usando AWS CLI o. Execute o seguinte comando no diretório em que o `emr_eks_iceberg.json` arquivo está localizado:

```
aws emr-containers start-job-run --cli-input-json file://emr_eks_iceberg.json
```

Para obter instruções detalhadas, consulte Como usar o Apache Iceberg com o Amazon EMR no EKS na documentação do Amazon EMR no EKS.

Melhores práticas para o Amazon EMR

Esta seção fornece diretrizes gerais para ajustar trabalhos do Spark no Amazon EMR para otimizar a leitura e gravação de dados nas tabelas do Iceberg. Para conhecer as melhores práticas específicas do Iceberg, consulte a seção Melhores práticas mais adiante neste guia.

- Use a versão mais recente do Amazon EMR — O Amazon EMR fornece otimizações do Spark prontas para usar com o tempo de execução do Amazon EMR Spark. AWS melhora o desempenho do mecanismo de tempo de execução do Spark a cada nova versão.

- Determine a infraestrutura ideal para suas cargas de trabalho do Spark — As cargas de trabalho do Spark podem exigir diferentes tipos de hardware para diferentes características de trabalho para garantir o desempenho ideal. O Amazon EMR oferece suporte a vários tipos de instância (como otimizado para computação, otimizado para memória, uso geral e otimizado para armazenamento) para cobrir todos os tipos de requisitos de processamento. Ao integrar novas cargas de trabalho, recomendamos que você faça o benchmark com tipos gerais de instância, como M5 ou M6g. Monitore o sistema operacional (OS) e as métricas do YARN da Ganglia e da Amazon CloudWatch para determinar os gargalos do sistema (CPU, memória, armazenamento e E/S) no pico de carga e escolha o hardware apropriado.
- Ajuste `spark.sql.shuffle.partitions` — defina a `spark.sql.shuffle.partitions` propriedade como o número total de núcleos virtuais (vCores) em seu cluster ou como um múltiplo desse valor (normalmente, 1 a 2 vezes o número total de vCores). Essa configuração afeta o paralelismo do Spark quando você usa particionamento de hash e intervalo como modo de distribuição de gravação. Ele solicita uma reprodução aleatória antes de gravar para organizar os dados, o que garante o alinhamento das partições.
- Ative a escalabilidade gerenciada — Para quase todos os casos de uso, recomendamos que você habilite a escalabilidade gerenciada e a alocação dinâmica. No entanto, se você tiver uma carga de trabalho com um padrão previsível, sugerimos que você desative o escalonamento automático e a alocação dinâmica. Quando a escalabilidade gerenciada estiver habilitada, recomendamos que você use instâncias spot para reduzir custos. Use instâncias spot para nós de tarefas em vez de nós principais ou principais. Ao usar instâncias spot, use frotas de instâncias com vários tipos de instância por frota para garantir a disponibilidade spot.
- Use a junção de transmissão quando possível — A junção de transmissão (no lado do mapa) é a junção ideal, desde que uma de suas tabelas seja pequena o suficiente para caber na memória do seu menor nó (na ordem de MBs) e você esteja realizando uma junção equi (=). Todos os tipos de junção, exceto as junções externas completas, são suportados. Uma junção de transmissão transmite a tabela menor como uma tabela de hash em todos os nós de trabalho na memória. Depois que a pequena tabela for transmitida, você não poderá fazer alterações nela. Como a tabela de hash está localmente na máquina virtual Java (JVM), ela pode ser mesclada facilmente com a tabela grande com base na condição de junção usando uma junção de hash. As junções de transmissão oferecem alto desempenho devido à sobrecarga mínima de reprodução aleatória.
- Ajuste o coletor de lixo — Se os ciclos de coleta de lixo (GC) forem lentos, considere mudar do coletor de lixo paralelo padrão para o G1GC para melhorar o desempenho. Para otimizar o desempenho do GC, você pode ajustar os parâmetros do GC. Para monitorar o desempenho do GC, você pode monitorá-lo usando a interface do usuário do Spark. Idealmente, o tempo de GC deve ser menor ou igual a 1% do tempo de execução total da tarefa.

Trabalhando com a Iceberg em AWS Glue

AWS Glueé um serviço de integração de dados sem servidor que facilita a descoberta, a preparação, a movimentação e a integração de dados de várias fontes para análise, aprendizado de máquina (ML) e desenvolvimento de aplicativos. Um dos principais recursos do AWS Glue é a capacidade de realizar operações de extração, transformação e carregamento (ETL) de maneira simples e econômica. Isso ajuda a categorizar seus dados, limpá-los, enriquecê-los e movê-los de forma confiável entre vários armazenamentos de dados e fluxos de dados.

AWS Glue trabalhos encapsulam scripts que definem a lógica de transformação usando um tempo de execução Apache Spark ou Python. AWS Glue os trabalhos podem ser executados tanto no modo batch quanto no modo de streaming.

Ao criar trabalhos do Iceberg em AWS Glue, dependendo da versão do AWS Glue, você pode usar a integração nativa do Iceberg ou uma versão personalizada do Iceberg para anexar dependências do Iceberg ao trabalho.

Usando a integração nativa do Iceberg

AWS Glue as versões 3.0, 4.0 e 5.0 oferecem suporte nativo a formatos de data lake transacionais, como Apache Iceberg, Apache Hudi e Linux Foundation Delta Lake in for Spark. AWS Glue Esse recurso de integração simplifica as etapas de configuração necessárias para começar a usar essas estruturas no. AWS Glue

Para ativar o suporte do Iceberg para seu AWS Glue trabalho, defina o trabalho: escolha a guia Detalhes do trabalho para seu AWS Glue trabalho, vá até Parâmetros do trabalho em Propriedades avançadas e defina a chave `--datalake-formats` e seu valor como`iceberg`.

Se você estiver criando um trabalho usando um notebook, você pode configurar o parâmetro na primeira célula do notebook usando a `%%configure` mágica da seguinte forma:

```
%%configure
{
  "--conf" : <job-specific Spark configuration discussed later>,
  "--datalake-formats" : "iceberg"
}
```

A `iceberg` configuração para `--datalake-formats` in AWS Glue corresponde a versões específicas do Iceberg com base na sua AWS Glue versão:

AWS Glue versão	Versão padrão do Iceberg
5,0	1.7.1
4,0	1.0.0
3.0	0.13.1

Usando uma versão personalizada do Iceberg

Em algumas situações, talvez você queira manter o controle sobre a versão do Iceberg para o trabalho e atualizá-la no seu próprio ritmo. Por exemplo, a atualização para uma versão posterior pode desbloquear o acesso a novos recursos e aprimoramentos de desempenho. Para usar uma versão específica do Iceberg AWS Glue, você pode fornecer seus próprios arquivos JAR.

Antes de implementar uma versão personalizada do Iceberg, verifique a compatibilidade com seu AWS Glue ambiente consultando a seção de AWS Glue versões da AWS Glue documentação. Por exemplo, o AWS Glue 5.0 requer compatibilidade com o Spark 3.5.4.

Como exemplo, para executar AWS Glue trabalhos que usam a versão 1.9.1 do Iceberg, siga estas etapas:

1. Adquira e faça o upload dos arquivos JAR necessários para o Amazon S3:
   a. Baixe iceberg-spark-runtime-3.5_2.12-1.9.1.jar e -1.9.1.jar do repositório Apache Maven. iceberg-aws-bundle
   b. Faça o upload desses arquivos para o local designado do bucket do S3 (por exemplo,`s3://your-bucket-name/jars/`).
2. Configure os parâmetros do trabalho para seu AWS Glue trabalho da seguinte forma:
   a. Especifique o caminho completo do S3 para os dois arquivos JAR no `--extra-jars` parâmetro, separando-os com uma vírgula (por exemplo,). `s3://your-bucket-name/jars/iceberg-spark-runtime-3.5_2.12-1.9.1.jar,s3://your-bucket-name/jars/iceberg-aws-bundle-1.9.1.jar`
   b. Não inclua `iceberg` como um valor para o parâmetro `--datalake-formats`.
   c. Se você usar AWS Glue 5.0, deverá definir o `--user-jars-first` parâmetro como`true`.

Configurações do Spark para Iceberg em AWS Glue

Esta seção discute as configurações do Spark necessárias para criar uma tarefa de AWS Glue ETL para um conjunto de dados do Iceberg. Você pode definir essas configurações usando a chave `--conf` Spark com uma lista separada por vírgulas de todas as chaves e valores de configuração do Spark. Você pode usar a `%%configure` mágica em um notebook ou na seção Job parameters do AWS Glue Studio console.

```
%glue_version 5.0

%%configure
{
  "--conf" : "spark.sql.extensions=org.apache.iceberg.spark.extensions...",
  "--datalake-formats" : "iceberg"
}
```

Configure a sessão do Spark com as seguintes propriedades:

- `<catalog_name>`é o nome do seu catálogo de sessões do Iceberg Spark. Substitua-o por um nome de sua escolha e lembre-se de alterar as referências em todas as configurações associadas a esse catálogo. No seu código, você pode se referir às suas tabelas do Iceberg usando o nome da tabela totalmente qualificado, incluindo o nome do catálogo da sessão do Spark, da seguinte forma:

  `<catalog_name>.<database_name>.<table_name>`

  Como alternativa, você pode alterar o catálogo padrão para o catálogo Iceberg que você definiu configurando `spark.sql.defaultCatalog` o nome do seu catálogo. Você pode usar essa segunda abordagem para se referir a tabelas sem o prefixo do catálogo, o que pode simplificar suas consultas.
- `<catalog_name>.<warehouse>`aponta para o caminho do Amazon S3 onde você deseja armazenar seus dados e metadados.
- Para tornar o catálogo um AWS Glue Data Catalog, `spark.sql.catalog.<catalog_name>.type` defina como`glue`. Essa chave é necessária para apontar para uma classe de implementação para qualquer implementação de catálogo personalizado. Para catálogos compatíveis com o Iceberg, consulte a seção de melhores práticas gerais mais adiante neste guia.

Por exemplo, se você tiver um catálogo chamado`glue_iceberg`, poderá configurar seu trabalho usando várias `--conf` chaves da seguinte forma:

```
%%configure
{
  "--datalake-formats" : "iceberg",
  "--conf" :
 "spark.sql.extensions=org.apache.iceberg.spark.extensions.IcebergSparkSessionExtensions
 --conf spark.sql.catalog.glue_iceberg=org.apache.iceberg.spark.SparkCatalog --
conf spark.sql.catalog.glue_iceberg.warehouse=s3://<your-warehouse-dir>/ --conf
 spark.sql.catalog.glue_iceberg.type=glue"
}
```

Como alternativa, você pode usar o código para adicionar as configurações acima ao seu script do Spark da seguinte forma:

```
spark = SparkSession.builder\

 .config("spark.sql.extensions","org.apache.iceberg.spark.extensions.IcebergSparkSessionExtensi
                    .config("spark.sql.catalog.glue_iceberg",
 "org.apache.iceberg.spark.SparkCatalog")\
                    .config("spark.sql.catalog.glue_iceberg.warehouse","s3://<your-
warehouse-dir>/")\
                    .config("spark.sql.catalog.glue_iceberg.type", "glue") \
                    .getOrCreate()
```

Melhores práticas para AWS Glue empregos

Esta seção fornece diretrizes gerais para ajustar as tarefas do Spark para otimizar AWS Glue a leitura e gravação de dados nas tabelas do Iceberg. Para conhecer as melhores práticas específicas do Iceberg, consulte a seção Melhores práticas mais adiante neste guia.

- Use a versão mais recente AWS Glue e atualize sempre que possível — novas versões AWS Glue oferecem melhorias de desempenho, tempos de inicialização reduzidos e novos recursos. Eles também oferecem suporte às versões mais recentes do Spark, que podem ser necessárias para as versões mais recentes do Iceberg. Para ver uma lista das AWS Glue versões disponíveis e das versões do Spark compatíveis com elas, consulte a AWS Glue documentação.
- Otimize a memória do AWS Glue trabalho — siga as recomendações na postagem do AWS blog Otimize o gerenciamento de memória em AWS Glue.

- Use o AWS Glue Auto Scaling — Quando você ativa o Auto Scaling AWS Glue , ajusta automaticamente o número de trabalhadores dinamicamente com base na sua AWS Glue carga de trabalho. Isso ajuda a reduzir o custo do seu AWS Glue trabalho durante picos de carga, porque AWS Glue reduz o número de trabalhadores quando a carga de trabalho é pequena e os trabalhadores estão ociosos. Para usar o AWS Glue Auto Scaling, você especifica um número máximo de trabalhadores para os quais seu AWS Glue trabalho pode ser escalado. Para obter mais informações, consulte Como usar o escalonamento automáticoAWS Glue na AWS Glue documentação.
- Use a versão desejada do Iceberg — a integração AWS Glue nativa com o Iceberg é melhor para começar a usar o Iceberg. No entanto, para cargas de trabalho de produção, recomendamos que você adicione dependências de biblioteca (conforme discutido anteriormente neste guia) para ter controle total sobre a versão do Iceberg. Essa abordagem ajuda você a se beneficiar dos recursos mais recentes do Iceberg e das melhorias de desempenho em seus AWS Glue trabalhos.
- Ative a interface do Spark para monitoramento e depuração — Você também pode usar a interface do Spark para AWS Glue inspecionar sua tarefa do Iceberg, visualizando os diferentes estágios de uma tarefa do Spark em um gráfico acíclico direcionado (DAG) e monitorando as tarefas em detalhes. A interface do usuário do Spark fornece uma maneira eficaz de solucionar problemas e otimizar as tarefas do Iceberg. Por exemplo, você pode identificar estágios de gargalo que apresentam grandes confusões ou vazamento de disco para identificar oportunidades de ajuste. Para obter mais informações, consulte Monitoramento de trabalhos usando a interface web do Apache Spark na AWS Glue documentação.

Trabalhando com tabelas Iceberg usando o Apache Spark

Esta seção fornece uma visão geral do uso do Apache Spark para interagir com tabelas Iceberg. Os exemplos são códigos padronizados que podem ser executados no Amazon EMR ou. AWS Glue

Observação: a interface principal para interagir com as tabelas do Iceberg é o SQL, então a maioria dos exemplos combinará o Spark SQL com a API. DataFrames

Criando e escrevendo tabelas Iceberg

Você pode usar o Spark SQL e o Spark DataFrames para criar e adicionar dados às tabelas do Iceberg.

Usando o Spark SQL

Para escrever um conjunto de dados do Iceberg, use instruções SQL padrão do Spark, como e. `CREATE TABLE` `INSERT INTO`

Tabelas não particionadas

Aqui está um exemplo de criação de uma tabela Iceberg não particionada com o Spark SQL:

```
spark.sql(f"""
    CREATE TABLE IF NOT EXISTS {CATALOG_NAME}.{DB_NAME}.{TABLE_NAME}_nopartitions (
        c_customer_sk             int,
        c_customer_id             string,
        c_first_name              string,
        c_last_name               string,
        c_birth_country           string,
        c_email_address           string)
    USING iceberg
    OPTIONS ('format-version'='2')
""")
```

Para inserir dados em uma tabela não particionada, use uma instrução padrão`INSERT INTO`:

```
spark.sql(f"""
INSERT INTO {CATALOG_NAME}.{DB_NAME}.{TABLE_NAME}_nopartitions
SELECT c_customer_sk, c_customer_id, c_first_name, c_last_name, c_birth_country,
 c_email_address
```

```
FROM another_table
""")
```

Tabelas particionadas

Aqui está um exemplo de criação de uma tabela Iceberg particionada com o Spark SQL:

```
spark.sql(f"""
    CREATE TABLE IF NOT EXISTS {CATALOG_NAME}.{DB_NAME}.{TABLE_NAME}_withpartitions (
        c_customer_sk             int,
        c_customer_id             string,
        c_first_name              string,
        c_last_name               string,
        c_birth_country           string,
        c_email_address           string)
    USING iceberg
    PARTITIONED BY (c_birth_country)
    OPTIONS ('format-version'='2')
""")
```

Para inserir dados em uma tabela Iceberg particionada com o Spark SQL, use uma instrução padrão: `INSERT INTO`

```
spark.sql(f"""
INSERT INTO {CATALOG_NAME}.{DB_NAME}.{TABLE_NAME}_withpartitions
SELECT c_customer_sk, c_customer_id, c_first_name, c_last_name, c_birth_country,
 c_email_address
FROM another_table
""")
```

> **Note**
>
> A partir do Iceberg 1.5.0, o modo de distribuição de `hash` gravação é o padrão quando você insere dados em tabelas particionadas. Para obter mais informações, consulte Escrevendo modos de distribuição na documentação do Iceberg.

Usando a DataFrames API

Para escrever um conjunto de dados do Iceberg, você pode usar a `DataFrameWriterV2` API.

Para criar uma tabela Iceberg e gravar dados nela, use a função `df.writeTo( t)`. Se a tabela existir, use a `.append()` função. Se isso não acontecer, use `.create().` Os exemplos a seguir usam `.createOrReplace()`, que é uma variação do `.create()` que é equivalente `CREATE OR REPLACE TABLE AS SELECT` a.

Tabelas não particionadas

Para criar e preencher uma tabela Iceberg não particionada usando a API: `DataFrameWriterV2`

```
input_data.writeTo(f"{CATALOG_NAME}.{DB_NAME}.{TABLE_NAME}_nopartitions") \
    .tableProperty("format-version", "2") \
    .createOrReplace()
```

Para inserir dados em uma tabela Iceberg não particionada existente usando a API: `DataFrameWriterV2`

```
input_data.writeTo(f"{CATALOG_NAME}.{DB_NAME}.{TABLE_NAME}_nopartitions") \
    .append()
```

Tabelas particionadas

Para criar e preencher uma tabela Iceberg particionada usando a API: `DataFrameWriterV2`

```
input_data.writeTo(f"{CATALOG_NAME}.{DB_NAME}.{TABLE_NAME}_withpartitions") \
    .tableProperty("format-version", "2") \
    .partitionedBy("c_birth_country") \
    .createOrReplace()
```

Para inserir dados em uma tabela Iceberg particionada usando a `DataFrameWriterV2` API:

```
input_data.writeTo(f"{CATALOG_NAME}.{DB_NAME}.{TABLE_NAME}_withpartitions") \
    .append()
```

Atualização de dados em tabelas Iceberg

O exemplo a seguir mostra como atualizar dados em uma tabela Iceberg. Este exemplo modifica todas as linhas que têm um número par na `c_customer_sk` coluna.

```
spark.sql(f"""
```

```
UPDATE {CATALOG_NAME}.{db.name}.{table.name}
SET c_email_address = 'even_row'
WHERE c_customer_sk % 2 == 0
""")
```

Essa operação usa a copy-on-write estratégia padrão e, portanto, reescreve todos os arquivos de dados afetados.

Atualizando dados em tabelas Iceberg

A atualização de dados se refere à inserção de novos registros de dados e à atualização dos registros de dados existentes em uma única transação. Para inserir dados em uma tabela Iceberg, você usa a declaração. `SQL MERGE INTO`

O exemplo a seguir altera o conteúdo da tabela`{UPSERT_TABLE_NAME}` dentro da tabela: `{TABLE_NAME}`

```
spark.sql(f"""
    MERGE INTO {CATALOG_NAME}.{DB_NAME}.{TABLE_NAME} t
    USING {UPSERT_TABLE_NAME} s
        ON t.c_customer_id = s.c_customer_id
    WHEN MATCHED THEN UPDATE SET t.c_email_address = s.c_email_address
    WHEN NOT MATCHED THEN INSERT *
""")
```

- Se um registro de cliente que está em `{UPSERT_TABLE_NAME}` já existir `{TABLE_NAME}` com o mesmo`c_customer_id`, o `c_email_address` valor do `{UPSERT_TABLE_NAME}` registro substituirá o valor existente (operação de atualização).
- Se um registro de cliente que está em `{UPSERT_TABLE_NAME}` não existir em`{TABLE_NAME}`, o `{UPSERT_TABLE_NAME}` registro será adicionado à `{TABLE_NAME}` (operação de inserção).

Excluindo dados em tabelas Iceberg

Para excluir dados de uma tabela do Iceberg, use a `DELETE FROM` expressão e especifique um filtro que corresponda às linhas a serem excluídas.

```
spark.sql(f"""
DELETE FROM {CATALOG_NAME}.{db.name}.{table.name}
WHERE c_customer_sk % 2 != 0
```

```
""")
```

Se o filtro corresponder a uma partição inteira, o Iceberg executará uma exclusão somente de metadados e deixará os arquivos de dados no lugar. Caso contrário, ele reescreve somente os arquivos de dados afetados.

O método delete pega os arquivos de dados afetados pela `WHERE` cláusula e cria uma cópia deles sem os registros excluídos. Em seguida, ele cria um novo instantâneo da tabela que aponta para os novos arquivos de dados. Portanto, os registros excluídos ainda estão presentes nos instantâneos mais antigos da tabela. Por exemplo, se você recuperar o instantâneo anterior da tabela, verá os dados que acabou de excluir. Para obter informações sobre como remover instantâneos antigos desnecessários com os arquivos de dados relacionados para fins de limpeza, consulte a seção Manutenção de arquivos usando compactação, mais adiante neste guia.

Leitura de dados

Você pode ler o status mais recente de suas tabelas Iceberg no Spark com o Spark SQL e. DataFrames

Exemplo usando o Spark SQL:

```
spark.sql(f"""
SELECT * FROM {CATALOG_NAME}.{db.name}.{table.name} LIMIT 5
""")
```

Exemplo de uso da DataFrames API:

```
df = spark.table(f"{CATALOG_NAME}.{DB_NAME}.{TABLE_NAME}").limit(5)
```

Usando a viagem no tempo

Cada operação de gravação (inserção, atualização, upsert, delete) em uma tabela do Iceberg cria um novo instantâneo. Em seguida, você pode usar esses instantâneos para viajar no tempo, para voltar no tempo e verificar o status de uma tabela no passado.

Para obter informações sobre como recuperar o histórico de instantâneos para tabelas usando `snapshot-id` e cronometrando valores, consulte a seção Acessando metadados mais adiante neste guia.

A consulta de viagem no tempo a seguir exibe o status de uma tabela com base em uma tabela específica`snapshot-id`.

Usar o Spark SQL:

```
spark.sql(f"""
SELECT * FROM {CATALOG_NAME}.{DB_NAME}.{TABLE_NAME} VERSION AS OF {snapshot_id}
""")
```

Usando a DataFrames API:

```
df_1st_snapshot_id = spark.read.option("snapshot-id", snapshot_id) \
     .format("iceberg") \
     .load(f"{CATALOG_NAME}.{DB_NAME}.{TABLE_NAME}") \
     .limit(5)
```

A consulta de viagem no tempo a seguir exibe o status de uma tabela com base no último instantâneo criado antes de um timestamp específico, em milissegundos (). `as-of-timestamp`

Usar o Spark SQL:

```
spark.sql(f"""
SELECT * FROM dev.{db.name}.{table.name} TIMESTAMP AS OF '{snapshot_ts}'
""")
```

Usando a DataFrames API:

```
df_1st_snapshot_ts = spark.read.option("as-of-timestamp", snapshot_ts) \
                          .format("iceberg") \
                          .load(f"dev.{DB_NAME}.{TABLE_NAME}") \
                          .limit(5)
```

Usando consultas incrementais

Você também pode usar instantâneos do Iceberg para ler dados anexados de forma incremental.

Nota: Atualmente, essa operação oferece suporte à leitura de dados de `append` instantâneos. Ele não suporta a busca de dados de operações como `replace`, `overwrite`, ou`delete`. Além disso, as operações de leitura incremental não são suportadas na sintaxe do Spark SQL.

O exemplo a seguir recupera todos os registros anexados a uma tabela Iceberg entre o instantâneo `start-snapshot-id` (exclusivo) e `end-snapshot-id` (inclusive).

```
df_incremental = (spark.read.format("iceberg")
    .option("start-snapshot-id", snapshot_id_start)
    .option("end-snapshot-id", snapshot_id_end)
    .load(f"glue_catalog.{DB_NAME}.{TABLE_NAME}")
)
```

Acessando metadados

O Iceberg fornece acesso aos seus metadados por meio de SQL. Você pode acessar os metadados de qualquer tabela (`<table_name>`) consultando o namespace. `<table_name>.<metadata_table>` Para obter uma lista completa das tabelas de metadados, consulte Como inspecionar tabelas na documentação do Iceberg.

O exemplo a seguir mostra como acessar a tabela de metadados do histórico do Iceberg, que mostra o histórico de confirmações (alterações) de uma tabela do Iceberg.

Usando o Spark SQL (com a `%%sql` mágica) de um notebook Amazon EMR Studio:

```
Spark.sql(f"""
SELECT * FROM {CATALOG_NAME}.{DB_NAME}.{TABLE_NAME}.history LIMIT 5
""")
```

Usando a DataFrames API:

```
spark.read.format("iceberg").load("{CATALOG_NAME}.{DB_NAME}.
{TABLE_NAME}.history").show(5,False)
```

Exemplo de saída:

Type: Table Pie Scatter Line Area Bar

made_current_at	snapshot_id	parent_id	is_current_ancestor
2023-01-09 02:50:17.547000+00:00	7501027970051178613	6598755163776233735	True
2023-01-12 05:39:29.567000+00:00	7069175828427777019	7501027970051178613	True
2023-01-12 05:39:58.807000+00:00	5173022175861138222	7069175828427777019	True
2023-01-12 05:40:18.499000+00:00	3703414997660223390	5173022175861138222	True
2023-01-12 05:40:41.827000+00:00	3807904412292252460	3703414997660223390	True

Trabalhando com tabelas Iceberg usando o Trino

Esta seção descreve como configurar e operar tabelas Iceberg usando o Trino no Amazon EMR. Os exemplos são códigos padronizados que você pode executar em um cluster do Amazon EMR no EC2. Os exemplos de código e as configurações nesta seção pressupõem que você esteja usando a versão emr-7.9.0 do Amazon EMR.

Configuração do Amazon EMR no EC2

1. Crie um arquivo `iceberg.properties` com o seguinte conteúdo. A `iceberg.file-format=parquet` configuração determina o formato de armazenamento padrão para novas tabelas se o formato não for especificado explicitamente na `CREATE TABLE` instrução.

```
connector.name=iceberg
iceberg.catalog.type=glue
iceberg.file-format=parquet
fs.native-s3.enabled=true
```

2. Faça upload do arquivo `iceberg.properties` no bucket do S3.
3. Crie uma ação de bootstrap que copie o `iceberg.properties` arquivo do seu bucket do S3 e o armazene como um arquivo de configuração Trino no cluster do Amazon EMR que você criará. Certifique-se de `<S3-bucket-name>` substituir pelo nome do bucket do S3.

```
#!/bin/bash
set -ex
sudo aws s3 cp s3://<S3-bucket-name>/iceberg.properties /etc/trino/conf/catalog/
iceberg.properties
```

4. Crie um cluster do Amazon EMR com o Trino instalado e especifique a execução do script anterior como uma ação de bootstrap. Aqui está um exemplo de comando AWS Command Line Interface (AWS CLI) para criar o cluster:

```
aws emr create-cluster --release-label emr-7.9.0 \
--applications Name=Trino \
--region <region> \
--name Trino_Iceberg_Cluster \
--bootstrap-actions '[{"Path":"s3://<S3-bucket-name>/bootstrap.sh","Name":"Add
 iceberg.properties"}]' \
```

```
--instance-groups
 '[{"InstanceGroupType":"MASTER","InstanceCount":1,"InstanceType":"m5.xlarge"},
{"InstanceGroupType":"CORE","InstanceCount":3,"InstanceType":"m5.xlarge"}]' \
--service-role "<IAM-service-role>" \
--ec2-attributes '{"KeyName":"<key-name>","InstanceProfile":"<EMR-EC2-instance-
profile>"}'
```

onde você substitui:

- `<S3-bucket-name>`com o nome do seu bucket S3
- `<region>`com seu específico Região da AWS
- `<key-name>`com seu key pair. Se o key pair não existir, ele será criado.
- `<IAM-service-role>`com sua função de serviço do Amazon EMR que segue o princípio do menor privilégio.
- `<EMR-EC2-instance-profile>`com seu perfil de instância.

5. Quando o cluster do Amazon EMR for inicializado, você poderá inicializar uma sessão do Trino executando o seguinte comando:

```
trino-cli
```

6. Na CLI do Trino, você pode visualizar os catálogos executando:

```
SHOW CATALOGS;
```

Criar tabelas Iceberg

Para criar uma tabela Iceberg, você pode usar a `CREATE TABLE` declaração. Aqui está um exemplo de criação de uma tabela particionada que usa o particionamento oculto do Iceberg:

```
CREATE TABLE iceberg.iceberg_db.iceberg_table (
             userid int,
             firstname varchar,
             city varchar)
       WITH (
             format = 'PARQUET',
             partitioning = ARRAY['city', 'bucket(userid, 16)'],
             location = 's3://<S3-bucket>/<prefix>');
```

> **Note**
>
> Se você não especificar o formato, o `iceberg.file-format` valor que você configurou na seção anterior será usado.

Para inserir dados, use o `INSERT INTO` comando. Veja um exemplo abaixo:

```
INSERT INTO iceberg.iceberg_db.iceberg_table (userid, firstname, city)
VALUES
    (1001, 'John', 'New York'),
    (1002, 'Mary', 'Los Angeles'),
    (1003, 'Mateo', 'Chicago'),
    (1004, 'Shirley', 'Houston'),
    (1005, 'Diego', 'Miami'),
    (1006, 'Nikki', 'Seattle'),
    (1007, 'Pat', 'Boston'),
    (1008, 'Terry', 'San Francisco'),
    (1009, 'Richard', 'Denver'),
    (1010, 'Pat', 'Phoenix');
```

Lendo as tabelas do Iceberg

Você pode ler o status mais recente da sua tabela Iceberg usando uma `SELECT` declaração, da seguinte forma:

```
SELECT * FROM iceberg.iceberg_db.iceberg_table;
```

Inserindo dados em tabelas Iceberg

Você pode executar uma operação de upsert (inserir simultaneamente novos registros e atualizar os existentes) usando a `MERGE INTO` instrução. Veja um exemplo abaixo:

```
MERGE INTO iceberg.iceberg_db.iceberg_table target
USING (
      VALUES
        (1001, 'John Updated', 'Boston'),       -- Update existing user
        (1002, 'Mary Updated', 'Seattle'),      -- Update existing user
        (1011, 'Martha', 'Portland'),           -- Insert new user
```

```
        (1012, 'Paulo', 'Austin')               -- Insert new user
    ) AS source (userid, firstname, city)
  ON target.userid = source.userid
WHEN MATCHED THEN
     UPDATE SET
       firstname = source.firstname,
       city = source.city
WHEN NOT MATCHED THEN
     INSERT (userid, firstname, city)
     VALUES (source.userid, source.firstname, source.city);
```

Excluindo registros das tabelas do Iceberg

Para excluir dados de uma tabela do Iceberg, use a `DELETE FROM` expressão e especifique um filtro que corresponda às linhas a serem excluídas. Veja um exemplo abaixo:

```
DELETE FROM iceberg.iceberg_db.iceberg_table WHERE userid IN (1003, 1004);
```

Consultar metadados de tabela do Iceberg

O Iceberg fornece acesso aos seus metadados por meio de SQL. Você pode acessar os metadados de qualquer tabela (`<table_name>`) consultando o namespace. `"<table_name>.$<metadata_table>"` Para obter uma lista completa das tabelas de metadados, consulte Como inspecionar tabelas na documentação do Iceberg.

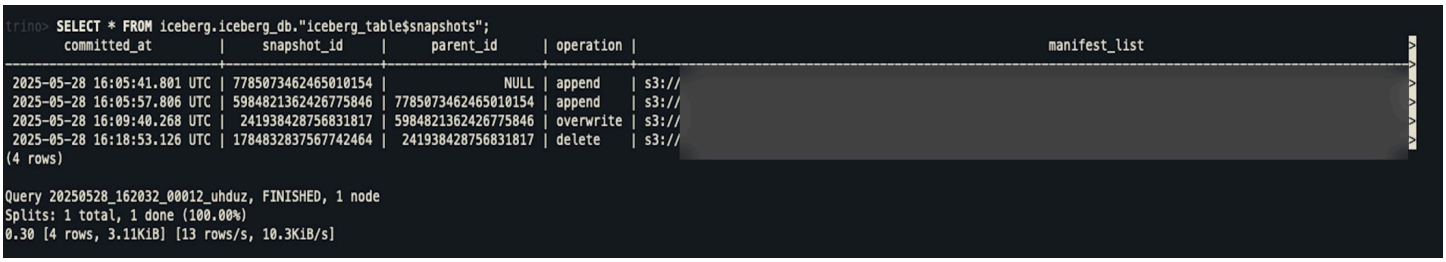
Aqui está um exemplo de lista de consultas para inspecionar os metadados do Iceberg:

```
SELECT  FROM iceberg.iceberg_db."iceberg_table$snapshots";
SELECT  FROM iceberg.iceberg_db."iceberg_table$history";
SELECT  FROM iceberg.iceberg_db."iceberg_table$partitions";
SELECT  FROM iceberg.iceberg_db."iceberg_table$files";
SELECT  FROM iceberg.iceberg_db."iceberg_table$manifests";
SELECT  FROM iceberg.iceberg_db."iceberg_table$refs";
SELECT * FROM iceberg.iceberg_db."iceberg_table$metadata_log_entries";
```

Por exemplo, esta consulta:

```
 SELECT * FROM iceberg.iceberg_db."iceberg_table$snapshots";
```

fornece a saída:



Usando a viagem no tempo

Cada operação de gravação (inserção, atualização, atualização ou exclusão) em uma tabela do Iceberg cria um novo instantâneo. Em seguida, você pode usar esses instantâneos para viajar no tempo, para voltar no tempo e verificar o status de uma tabela no passado.

A consulta de viagem no tempo a seguir exibe o status de uma tabela com base em uma tabela específica`snapshot_id`:

```
SELECT *
FROM iceberg.iceberg_db.iceberg_table FOR VERSION AS OF 241938428756831817;
```

A consulta de viagem no tempo a seguir exibe o status de uma tabela com base em um timestamp específico:

```
SELECT *
FROM iceberg.iceberg_db.iceberg_table FOR TIMESTAMP AS OF TIMESTAMP '2025-05-28
 16:09:40.268 UTC'
```

Considerações ao usar o Iceberg com Trino

As operações de gravação do Trino nas tabelas do Iceberg seguem o merge-on-readdesign, então elas criam arquivos de exclusão posicional em vez de reescrever arquivos de dados inteiros que são afetados por atualizações ou exclusões. Se você quiser usar a copy-on-write abordagem, considere usar o Spark para operações de gravação.

Trabalhando com tabelas Iceberg usando o Amazon Data Firehose

O Amazon Data Firehose é um serviço sem servidor e sem código para fornecer fluxos de dados de mais de 20 fontes, como logs AWS WAF , Amazon Logs, Amazon Kinesis AWS IoT Data Streams e Amazon CloudWatch Managed Streaming for Apache Kafka (Amazon MSK) para destinos como Amazon S3, Amazon Redshift, Snowflake e Splunk.

Você pode usar o Firehose para entregar diretamente dados de streaming às tabelas do Apache Iceberg no Amazon S3. Usando o Firehose, você pode rotear registros de um único stream para diferentes tabelas do Apache Iceberg e aplicar automaticamente operações de inserção, atualização e exclusão aos registros nas tabelas. O Firehose garante a entrega exata nas mesas Iceberg. Esse atributo requer o uso do AWS Glue Data Catalog.

O Firehose também pode entregar dados de streaming diretamente às tabelas do Amazon S3. Essas tabelas fornecem armazenamento otimizado para cargas de trabalho de análise em grande escala e incluem recursos que melhoram continuamente o desempenho das consultas e reduzem os custos de armazenamento de dados tabulares.

Para obter informações sobre como configurar um stream do Firehose para entregar dados às tabelas do Apache Iceberg, consulte Configurar o stream do Firehose na documentação do Firehose ou a postagem do blog Transmitir dados em tempo real para tabelas do Apache Iceberg no Amazon S3 usando o Amazon Data Firehose.

Trabalhando com tabelas Iceberg usando o Athena SQL

O Amazon Athena fornece suporte integrado para o Apache Iceberg e não exige etapas ou configurações adicionais. Esta seção fornece uma visão geral detalhada dos recursos suportados e orientações de alto nível sobre o uso do Athena para interagir com as tabelas Iceberg.

Compatibilidade de versões e recursos

Suporte à especificação da tabela Iceberg

A especificação da tabela Apache Iceberg especifica como as tabelas Iceberg devem se comportar. O Athena oferece suporte ao formato de tabela versão 2, portanto, qualquer tabela Iceberg que você criar com o console, a CLI ou o SDK usa essa versão inerentemente.

Se você usa uma tabela Iceberg criada com outro mecanismo, como o Apache Spark no Amazon EMR ou AWS Glue, certifique-se de definir a versão do formato da tabela usando as propriedades da tabela. Como referência, consulte a seção Criando e escrevendo tabelas Iceberg anteriormente neste guia.

Suporte ao recurso Iceberg

Você pode usar o Athena para ler e gravar em tabelas Iceberg. Quando você altera os dados usando as `DELETE FROM` instruçõesUPDATE, e`MERGE INTO`, o Athena suporta somente o merge-on-read modo. Essa propriedade não pode ser alterada. Para atualizar ou excluir dados com copy-on-write, você precisa usar outros mecanismos, como o Apache Spark no Amazon EMR ou. AWS Glue A tabela a seguir resume o suporte aos recursos do Iceberg no Athena.

		Suporte DDL		Suporte a DML		AWS Lake Formation para segurança (opcional)
	Formato da tabela	Create table	Evolução do esquema	Leitura de dados	Gravação de dados	Controle de acesso

		Suporte DDL		Suporte a DML		AWS Lake Formation para segurança (opcional)
						de linha/coluna
Amazon Athena	Versão 2	✓	✓	✓	XC opy-on-write	✓
					✓ M erge-on-read	✓

> **Note**
>
> - O Athena não oferece suporte a consultas incrementais.
> - No Athena, as operações de atualização, exclusão e mesclagem sempre usam como padrão a mesclagem na leitura (MoR), independentemente de qualquer configuração de cópia na gravação (CoW) nas propriedades da tabela, porque o CoW não é suportado.

Trabalhando com mesas Iceberg

Para começar rapidamente a usar o Iceberg no Athena, consulte a seção Introdução às tabelas do Iceberg no Athena SQL, anteriormente neste guia.

A tabela a seguir lista as limitações e recomendações.

Cenário	Limitação	Recomendação
Tabela de geração de DDL	As tabelas de iceberg criadas com outros mecanismos podem ter propriedades que não estão expostas no	Use a instrução equivalente no mecanismo que criou a tabela (por exemplo, a `SHOW CREATE TABLE` instrução do Spark).

Cenário	Limitação	Recomendação
	Athena. Para essas tabelas, não é possível gerar o DDL.	
Prefixos aleatórios do Amazon S3 em objetos que são gravados em uma tabela Iceberg	Por padrão, as tabelas do Iceberg criadas com o Athena têm `write.object-storage.enabled` a propriedade ativada.	Para desativar esse comportamento e obter controle total sobre as propriedades da tabela Iceberg, crie uma tabela Iceberg com outro mecanismo, como o Spark no Amazon EMR ou. AWS Glue
Consultas incrementais	Atualmente, não há suporte no Athena.	Para usar consultas incrementais para permitir pipelines incrementais de ingestão de dados, use o Spark no Amazon EMR ou. AWS Glue

Trabalhando com tabelas Iceberg usando PyIceberg

Esta seção explica como você pode interagir com as tabelas Iceberg usando PyIceberg. Os exemplos fornecidos são códigos padronizados que você pode executar em EC2 instâncias, AWS Lambdafunções do Amazon Linux 2023 ou em qualquer ambiente Python com credenciais configuradas corretamente.AWS

Pré-requisitos

> **Note**
>
> Esses exemplos usam PyIceberg 1.9.1.

Para trabalhar com PyIceberg, você precisa PyIceberg e AWS SDK para Python (Boto3) instalou. Aqui está um exemplo de como você pode configurar um ambiente virtual Python para trabalhar com PyIceberg e: AWS Glue Data Catalog

1. Faça o download PyIcebergusando o instalador do pacote pip python. Você também precisa do Boto3 para interagir. Serviços da AWS Você pode configurar um ambiente virtual Python local para testar usando estes comandos:

```
python3 -m venv my_env
cd my_env/bin/
source activate
pip install "pyiceberg[pyarrow,pandas,glue]"
pip install boto3
```

2. Execute `python` para abrir o shell do Python e testar os comandos.

Conectando-se ao catálogo de dados

Para começar a trabalhar com tabelas Iceberg em AWS Glue, primeiro você precisa se conectar ao AWS Glue Data Catalog.

A `load_catalog` função inicializa uma conexão com o Catálogo de Dados criando um objeto de catálogo que serve como sua interface principal para todas as operações do Iceberg:

```
from pyiceberg.catalog import load_catalog
region = "us-east-1"

glue_catalog = load_catalog(
    'default',
    **{
        'client.region': region
    },
    type='glue'
)
```

Listando e criando bancos de dados

Para listar bancos de dados existentes, use a `list_namespaces` função:

```
databases = glue_catalog.list_namespaces()
print(databases)
```

Para criar um novo banco de dados, use a `create_namespace` função:

```
database_name="mydb"
s3_db_path=f"s3://amzn-s3-demo-bucket/{database_name}"

glue_catalog.create_namespace(database_name, properties={"location": s3_db_path})
```

Criando e escrevendo tabelas Iceberg

Tabelas não particionadas

Aqui está um exemplo de criação de uma tabela Iceberg não particionada usando a função: `create_table`

```
from pyiceberg.schema import Schema
from pyiceberg.types import NestedField, StringType, DoubleType

database_name="mydb"
table_name="pyiceberg_table"
s3_table_path=f"s3://amzn-s3-demo-bucket/{database_name}/{table_name}"
```

```
schema = Schema(
    NestedField(1, "city", StringType(), required=False),
    NestedField(2, "lat", DoubleType(), required=False),
    NestedField(3, "long", DoubleType(), required=False),
)

glue_catalog.create_table(f"{database_name}.{table_name}", schema=schema,
 location=s3_table_path)
```

Você pode usar a `list_tables` função para verificar a lista de tabelas dentro de um banco de dados:

```
tables = glue_catalog.list_tables(namespace=database_name)
print(tables)
```

Você pode usar a `append` função e PyArrow inserir dados dentro de uma tabela Iceberg:

```
import pyarrow as pa
df = pa.Table.from_pylist(
    [
        {"city": "Amsterdam", "lat": 52.371807, "long": 4.896029},
        {"city": "San Francisco", "lat": 37.773972, "long": -122.431297},
        {"city": "Drachten", "lat": 53.11254, "long": 6.0989},
        {"city": "Paris", "lat": 48.864716, "long": 2.349014},
    ],
)

table = glue_catalog.load_table(f"{database_name}.{table_name}")
table.append(df)
```

Tabelas particionadas

Aqui está um exemplo de criação de uma tabela Iceberg particionada com particionamento oculto usando a função e: `create_table` `PartitionSpec`

```
from pyiceberg.schema import Schema
from pyiceberg.types import (
    NestedField,
    StringType,
    FloatType,
    DoubleType,
```

```
    TimestampType,
)

# Define the schema
schema = Schema(
    NestedField(field_id=1, name="datetime", field_type=TimestampType(),
 required=True),
    NestedField(field_id=2, name="drone_id", field_type=StringType(), required=True),
    NestedField(field_id=3, name="lat", field_type=DoubleType(), required=False),
    NestedField(field_id=4, name="lon", field_type=DoubleType(), required=False),
    NestedField(field_id=5, name="height", field_type=FloatType(), required=False),
)

from pyiceberg.partitioning import PartitionSpec, PartitionField
from pyiceberg.transforms import DayTransform

partition_spec = PartitionSpec(
    PartitionField(
        source_id=1,  # Refers to "datetime"
        field_id=1000,
        transform=DayTransform(),
        name="datetime_day"
    )
)

database_name="mydb"
partitioned_table_name="pyiceberg_table_partitioned"
s3_table_path=f"s3://amzn-s3-demo-bucket/{database_name}/{partitioned_table_name}"

glue_catalog.create_table(
    identifier=f"{database_name}.{partitioned_table_name}",
    schema=schema,
    location=s3_table_path,
    partition_spec=partition_spec
)
```

Você pode inserir dados em uma tabela particionada da mesma forma que em uma tabela não particionada. O particionamento é feito automaticamente.

```
from datetime import datetime
arrow_schema = pa.schema([
    pa.field("datetime", pa.timestamp("us"), nullable=False),
    pa.field("drone_id", pa.string(), nullable=False),
```

```
    pa.field("lat", pa.float64()),
    pa.field("lon", pa.float64()),
    pa.field("height", pa.float32()),
])

data = [
    {
        "datetime": datetime(2024, 6, 1, 12, 0, 0),
        "drone_id": "drone_001",
        "lat": 52.371807,
        "lon": 4.896029,
        "height": 120.5,
    },
    {
        "datetime": datetime(2024, 6, 1, 12, 5, 0),
        "drone_id": "drone_002",
        "lat": 37.773972,
        "lon": -122.431297,
        "height": 150.0,
    },
    {
        "datetime": datetime(2024, 6, 2, 9, 0, 0),
        "drone_id": "drone_001",
        "lat": 53.11254,
        "lon": 6.0989,
        "height": 110.2,
    },
    {
        "datetime": datetime(2024, 6, 2, 9, 30, 0),
        "drone_id": "drone_003",
        "lat": 48.864716,
        "lon": 2.349014,
        "height": 145.7,
    },
]

df = pa.Table.from_pylist(data, schema=arrow_schema)

table = glue_catalog.load_table(f"{database_name}.{partitioned_table_name}")
table.append(df)
```

Leitura de dados

Você pode usar a PyIceberg `scan` função para ler dados de suas tabelas do Iceberg. Você pode filtrar linhas, selecionar colunas específicas e limitar o número de registros retornados.

```
table= glue_catalog.load_table(f"{database_name}.{table_name}")
scan_df = table.scan(
    row_filter=(
        f"city = 'Amsterdam'"
    ),
    selected_fields=("city", "lat"),
    limit=100,
).to_pandas()

print(scan_df)
```

Excluir dados

A PyIceberg `delete` função permite que você remova registros da sua tabela usando`delete_filter`:

```
table = glue_catalog.load_table(f"{database_name}.{table_name}")
table.delete(delete_filter="city == 'Paris'")
```

Acessando metadados

PyIceberg fornece várias funções para acessar os metadados da tabela. Veja como você pode visualizar informações sobre instantâneos de tabelas:

```
#List of snapshots
table.snapshots()

#Current snapshot
table.current_snapshot()

#Take a previous snapshot
second_last_snapshot_id=table.snapshots()[-2].snapshot_id
print(f"Second last SnapshotID: {second_last_snapshot_id}")
```

Para obter uma lista detalhada dos metadados disponíveis, consulte a seção de referência do código de metadados da PyIceberg documentação.

Usando a viagem no tempo

Você pode usar instantâneos da tabela para viajar no tempo para acessar os estados anteriores da sua tabela. Veja como visualizar o estado da tabela antes da última operação:

```
second_last_snapshot_id=table.snapshots()[-2].snapshot_id

time_travel_df = table.scan(
    limit=100,
    snapshot_id=second_last_snapshot_id
).to_pandas()

print(time_travel_df)
```

Para ver uma lista completa das funções disponíveis, consulte a documentação da API PyIceberg Python.

Trabalhando com a versão 3 da especificação de formato de tabela Iceberg

A versão mais recente da especificação do formato de tabela Apache Iceberg é a versão 3. Esta versão apresenta recursos avançados para criar lagos de dados em escala de petabytes com desempenho aprimorado e redução da sobrecarga operacional. Ele aborda gargalos de desempenho comuns encontrados na versão 2, principalmente em relação a atualizações em lote e operações de exclusão de conformidade.

AWS fornece suporte para vetores de exclusão e linhagem de linha, conforme definido na especificação Iceberg versão 3. Esses recursos estão disponíveis com o Apache Spark nas seguintes opções. Serviços da AWS

AWS service (Serviço da AWS)	Suporte para a versão 3
Amazon EMR para Apache Spark	Amazon EMR versão 7.12 e versões posteriores
AWS Glue	Sim
AWS Glue: API REST Iceberg, manutenção de tabelas	Sim
Notebooks Amazon SageMaker Unified Studio	Sim
Tabelas do Amazon S3: API REST Iceberg, manutenção de tabelas	Sim
Amazon Athena (Trino)	Não

Principais recursos da versão 3

Os vetores de exclusão substituem os arquivos de exclusão posicional usados na versão 2 por um formato binário eficiente armazenado como arquivos Puffin. Isso elimina a amplificação de gravação de atualizações aleatórias em lote e exclusões de conformidade com o Regulamento Geral de Proteção de Dados (GDPR) e reduz significativamente a sobrecarga de manter dados atualizados.

As organizações que processam atualizações de alta frequência verão melhorias imediatas no desempenho de gravação e custos reduzidos de armazenamento com menos arquivos pequenos.

A linhagem da linha permite o rastreamento preciso das alterações no nível da linha. Os sistemas subsequentes podem processar alterações de forma incremental, acelerando os pipelines de dados e reduzindo os custos de computação para fluxos de trabalho de captura de dados de alteração (CDC). Esse recurso integrado elimina a necessidade de implementações personalizadas de rastreamento de alterações.

Compatibilidade da versão

A versão 3 mantém a compatibilidade com versões anteriores das tabelas da versão 2. Os serviços da AWS oferecem suporte às tabelas da versão 2 e da versão 3 simultaneamente, para que você possa:

- Execute consultas nas tabelas da versão 2 e da versão 3.
- Atualize as tabelas existentes da versão 2 para a versão 3 sem reescrever dados.
- Execute consultas de viagem no tempo que abrangem instantâneos da versão 2 e da versão 3.
- Use a evolução do esquema e o particionamento oculto nas versões da tabela.

Começando com a versão 3

Pré-requisitos

Antes de trabalhar com tabelas da versão 3, verifique se você tem:

- E Conta da AWS com as permissões apropriadas AWS Identity and Access Management (IAM).
- Acesso a um ou mais serviços de AWS análise (Amazon EMR, notebooks AWS Glue Amazon SageMaker Unified Studio ou tabelas Amazon S3).
- Um bucket S3 para armazenar dados e metadados de tabelas.
- Um bucket de mesa para começar a usar o Amazon S3 Tables ou um bucket S3 de uso geral se você estiver criando sua própria infraestrutura Iceberg.
- Um AWS Glue catálogo configurado.

Criação de tabelas da versão 3

Criar novas tabelas

Para criar uma nova tabela do Iceberg versão 3, defina a propriedade da `format-version` tabela como 3.

Usar o Spark SQL:

```
CREATE TABLE IF NOT EXISTS myns.orders_v3 (
    order_id bigint,
    customer_id string,
    order_date date,
    total_amount decimal(10,2),
    status string,
    created_at timestamp
)
USING iceberg
TBLPROPERTIES (
    'format-version' = '3'
)
```

Atualizando as tabelas da versão 2 para a versão 3

Você pode atualizar as tabelas existentes da versão 2 para a versão 3 atomicamente sem reescrever os dados.

Usar o Spark SQL:

```
ALTER TABLE myns.existing_table
SET TBLPROPERTIES ('format-version' = '3')
```

> **Important**
>
> A versão 3 é uma atualização unidirecional. Depois que uma tabela é atualizada da versão 2 para a versão 3, ela não pode ser rebaixada para a versão 2 por meio de operações padrão.

O que ocorre durante a atualização:

- Um novo instantâneo de metadados é criado atomicamente.

- Os arquivos de dados existentes do Parquet são reutilizados.
- Os campos de linhagem de linha são adicionados aos metadados da tabela.

Após a atualização:

- A próxima compactação removerá os arquivos de exclusão da versão 2 antiga.
- Novas modificações usarão os arquivos vetoriais de exclusão da versão 3.

A atualização não realiza um preenchimento histórico dos registros de rastreamento de alterações de linhagem de linha.

Ativando vetores de exclusão

Para aproveitar os vetores de exclusão para atualizações, exclusões e mesclagens, configure seu modo de gravação.

Usar o Spark SQL:

```
ALTER TABLE myns.orders_v3
SET TBLPROPERTIES ('format-version' = '3',
                   'write.delete.mode' = 'merge-on-read',
                   'write.update.mode' = 'merge-on-read',
                   'write.merge.mode' = 'merge-on-read'
                  )
```

Essas configurações garantem que as operações de atualização, exclusão e mesclagem criem arquivos vetoriais de exclusão em vez de reescrever arquivos de dados inteiros.

Usando linhagem de linha para rastreamento de alterações

A versão 3 adiciona automaticamente campos de metadados de linhagem de linha para rastrear as alterações.

Usar o Spark SQL:

```
# Query with parameter value provided
last_processed_sequence = 47
```

```
SELECT
    id,
    data,
    _row_id,
    _last_updated_sequence_number
FROM myns.orders_v3
WHERE _last_updated_sequence_number > :last_processed_sequence
```

O `_row_id` campo identifica de forma exclusiva cada linha e `_last_updated_sequence_number` rastreia quando a linha foi modificada pela última vez. Use esses campos para:

- Identifique as linhas alteradas para processamento incremental.
- Rastreie a linhagem de dados para fins de conformidade.
- Otimize os pipelines do CDC.
- Reduza os custos de computação processando somente as alterações.

Práticas recomendadas para a versão 3

Quando usar a versão 3

Considere atualizar para a versão 3 ou começar com ela quando:

- Você realiza atualizações ou exclusões frequentes em lote.
- Você precisa atender aos requisitos de exclusão do GDPR ou de conformidade.
- Suas cargas de trabalho envolvem interrupções de alta frequência.
- Você precisa de fluxos de trabalho eficientes do CDC.
- Você quer reduzir os custos de armazenamento de arquivos pequenos.
- Você precisa de melhores recursos de rastreamento de alterações.

Otimizando o desempenho de gravação

- Ative vetores de exclusão para cargas de trabalho com muitas atualizações:

```
SET TBLPROPERTIES (
'write.delete.mode' = 'merge-on-read',
'write.update.mode' = 'merge-on-read',
```

```
'write.merge.mode' = 'merge-on-read'
)
```

- Configure tamanhos de arquivo apropriados:

```
SET TBLPROPERTIES (
'write.target-file-size-bytes' = '536870912'  — 512 MB
)
```

Otimizando o desempenho de leitura

- Use linhagem de linha para processamento incremental.
- Use a viagem no tempo para acessar dados históricos sem copiar.
- Habilite a coleta de estatísticas para um melhor planejamento de consultas.

Estratégia de migração

Ao migrar da versão 2 para a versão 3, siga estas melhores práticas:

- Teste primeiro em um ambiente que não seja de produção para validar o processo de upgrade e o desempenho.
- Atualize durante períodos de baixa atividade para minimizar o impacto nas operações simultâneas.
- Monitore o desempenho inicial e acompanhe as métricas após a atualização.
- Execute a compactação para consolidar os arquivos excluídos após a atualização.
- Atualize a documentação da sua equipe para refletir os recursos da versão 3.

Considerações sobre compatibilidade

- Versões do mecanismo — Certifique-se de que todos os mecanismos que acessam a tabela suportem a versão 3.
- Ferramentas de terceiros — verifique a compatibilidade da sua ferramenta com a versão 3 antes de fazer o upgrade.
- Estratégia de backup — teste os procedimentos de recuperação baseados em snapshots.
- Monitoramento — atualize os painéis de monitoramento para métricas específicas da versão 3.

Solução de problemas

Problemas comuns

Erro: “a versão de formato 3 não é suportada”

- Verifique se a versão do seu mecanismo é compatível com a versão 3. Para obter detalhes, consulte a tabela no início desta seção.
- Verifique a compatibilidade do catálogo.
- Verifique se você está usando as versões mais recentes do Serviços da AWS.

Degradação do desempenho após a atualização

- Verifique se não há falhas na compactação. Para obter mais informações, consulte Registro e monitoramento de tabelas do S3 na documentação do Amazon S3.
- Confirme se os vetores de exclusão estão habilitados. As seguintes propriedades devem ser definidas:

```
SET TBLPROPERTIES (
'write.delete.mode' = 'merge-on-read',
'write.update.mode' = 'merge-on-read',
'write.merge.mode' = 'merge-on-read'
)
```

  É possível verificar as propriedades da tabela com o seguinte código:

```
DESCRIBE FORMATTED myns.orders_v3
```

- Revise sua estratégia de partição. O particionamento excessivo pode levar a arquivos pequenos. Execute a consulta a seguir para obter o tamanho médio do arquivo para sua tabela:

```
SELECT avg(file_size_in_bytes) as avg_file_size_bytes
FROM myns.orders_v3.files
```

Incompatibilidade com ferramentas de terceiros

- Verifique se a ferramenta é compatível com a especificação da versão 3.

- Considere manter as tabelas da versão 2 para ferramentas sem suporte.
- Entre em contato com o fornecedor da ferramenta para obter o cronograma de suporte da versão 3.

Como obter ajuda

- Para questões AWS service (Serviço da AWS) específicas, entre em contato com AWS Support.
- Para obter ajuda da comunidade Iceberg, use o canal Iceberg Slack.
- Para obter informações sobre como usar Serviços da AWS para gerenciar suas cargas de trabalho de análise, consulte Analytics on AWS.

Preços

- Preços de computação e armazenamento do Amazon EMR
- SageMakerPreços da Amazon
- AWS Glue execução do trabalho e preços do catálogo de dados
- Preços de armazenamento e solicitações de tabelas do S3

Disponibilidade

O suporte à especificação de formato de tabela Iceberg versão 3 está disponível em todos os Regiões da AWS locais onde as tabelas Amazon EMR AWS Glue, AWS Glue Data Catalog, e S3 operam. Para conferir a disponibilidade de uma região, consulte Serviços da AWS by Region.

Recursos adicionais do

- Documentação do Apache Iceberg
- Especificação da tabela Apache Iceberg
- Orientação para migrar dados tabulares do Amazon S3 para tabelas do S3
- Tutorial: Introdução às tabelas do S3

Migração de tabelas existentes para o Iceberg

Esta seção se concentra na migração de suas tabelas existentes no estilo Hive para o formato Iceberg. Ela se aplica a tabelas que usam formatos tradicionais compatíveis com o Hive, como Apache Parquet ou Apache ORC. Essas informações não se aplicam a tabelas que já usam formatos de tabela modernos, como Linux Foundation Delta Lake ou Apache Hudi.

Para migrar suas tabelas atuais no estilo Hive para o formato Iceberg, você pode usar a migração de dados local ou completa:

- A migração local é o processo de gerar os arquivos de metadados do Iceberg sobre os arquivos de dados existentes.
- A migração completa de dados cria a camada de metadados do Iceberg e também reescreve os arquivos de dados existentes da tabela original para a nova tabela do Iceberg.

As seções a seguir fornecem uma visão geral detalhada de cada método de migração, incluindo step-by-step instruções e considerações para implementação. Para obter mais informações sobre essas estratégias de migração, consulte a seção Migração de tabelas da documentação do Iceberg.

Depois de analisar os detalhes dos métodos de migração de dados no local e completos, consulte as duas seções principais a seguir para auxiliar seu processo de tomada de decisão:

- A escolha de uma estratégia de migração fornece orientação por meio de uma série de perguntas e cenários, para ajudá-lo a determinar a abordagem de migração mais adequada com base em seus requisitos e casos de uso específicos.
- O resumo das opções de migração fornece uma tabela abrangente que compara as principais características e considerações em diferentes opções de migração. Essa tabela serve como um guia de referência rápida e oferece uma comparação de recursos para ajudá-lo a entender as compensações técnicas entre os métodos.

Migração local

A migração local elimina a necessidade de reescrever todos os seus arquivos de dados. Em vez disso, os arquivos de metadados do Iceberg são gerados e vinculados aos seus arquivos de dados existentes. Esse método geralmente é mais rápido e econômico, especialmente para grandes conjuntos de dados ou tabelas com formatos de arquivo compatíveis, como Parquet, Avro e ORC.

> **Note**
>
> A migração local não pode ser usada ao migrar para as tabelas do Amazon S3.

O Iceberg oferece duas opções principais para implementar a migração local:

- Usando o procedimento de captura instantânea para criar uma nova tabela Iceberg, mantendo a tabela de origem inalterada. Para obter mais informações, consulte Tabela de instantâneos na documentação do Iceberg.
- Usando o procedimento de migração para criar uma nova tabela Iceberg em substituição à tabela de origem. Para obter mais informações, consulte Migrate Table na documentação do Iceberg. Embora esse procedimento funcione com o Hive Metastore (HMS), atualmente não é compatível com o. AWS Glue Data Catalog O procedimento de replicação da migração da tabela na AWS Glue Data Catalog seção posterior deste guia fornece uma solução alternativa para obter um resultado semelhante com o Catálogo de Dados.

Depois de realizar a migração local usando um `snapshot` ou`migrate`, alguns arquivos de dados podem permanecer sem migração. Isso geralmente acontece quando os gravadores continuam gravando na tabela de origem durante ou após a migração. Para incorporar esses arquivos restantes em sua tabela Iceberg, você pode usar o procedimento add_files. Para obter mais informações, consulte Adicionar arquivos na documentação do Iceberg.

Digamos que você tenha uma `products` tabela baseada em Parquet que foi criada e preenchida no Athena da seguinte forma:

```
CREATE EXTERNAL TABLE mydb.products (
    product_id INT,
    product_name STRING
)
PARTITIONED BY (category STRING)
STORED AS PARQUET
LOCATION 's3://amzn-s3-demo-bucket/products/';

INSERT INTO mydb.products
VALUES
    (1001, 'Smartphone', 'electronics'),
    (1002, 'Laptop', 'electronics'),
    (2001, 'T-Shirt', 'clothing'),
```

```
    (2002, 'Jeans', 'clothing');
```

As seções a seguir explicam como você pode usar os `migrate` procedimentos `snapshot` e com essa tabela.

Opção 1: procedimento de captura instantânea

O `snapshot` procedimento cria uma nova tabela Iceberg que tem um nome diferente, mas replica o esquema e o particionamento da tabela de origem. Essa operação deixa a tabela de origem completamente inalterada durante e após a ação. Ele cria efetivamente uma cópia leve da tabela, o que é particularmente útil para testar cenários ou explorar dados sem correr o risco de modificações na fonte de dados original. Essa abordagem permite um período de transição em que a tabela original e a tabela Iceberg permanecem disponíveis (consulte as notas no final desta seção). Quando o teste estiver concluído, você poderá mover sua nova tabela Iceberg para produção fazendo a transição de todos os escritores e leitores para a nova tabela.

Você pode executar o `snapshot` procedimento usando o Spark em qualquer modelo de implantação do Amazon EMR (por exemplo, Amazon EMR no EC2, Amazon EMR no EKS, EMR Serverless) e. AWS Glue

Para testar a migração local com o procedimento do `snapshot` Spark, siga estas etapas:

1. Inicie um aplicativo Spark e configure a sessão do Spark com as seguintes configurações:
   - `"spark.sql.extensions":"org.apache.iceberg.spark.extensions.IcebergSparkSessi`
   - `"spark.sql.catalog.spark_catalog":"org.apache.iceberg.spark.SparkSessionCatal`
   - `"spark.sql.catalog.spark_catalog.type":"glue"`
   - `"spark.hadoop.hive.metastore.client.factory.class":"com.amazonaws.glue.catalo`
2. Execute o `snapshot` procedimento para criar uma nova tabela Iceberg que aponte para os arquivos de dados da tabela original:

```
spark.sql(f"""
CALL system.snapshot(
source_table => 'mydb.products',
table => 'mydb.products_iceberg',
location => 's3://amzn-s3-demo-bucket/products_iceberg/'
)
"""
).show(truncate=False)
```

O quadro de dados de saída contém o `imported_files_count` (o número de arquivos que foram adicionados).

3. Valide a nova tabela consultando-a:

```
spark.sql(f"""
SELECT * FROM mydb.products_iceberg LIMIT 10
"""
).show(truncate=False)
```

Observações

- Depois de executar o procedimento, qualquer modificação no arquivo de dados na tabela de origem dessincronizará a tabela gerada. Os novos arquivos que você adicionar não estarão visíveis na tabela Iceberg, e os arquivos que você removeu afetarão os recursos de consulta na tabela Iceberg. Para evitar os problemas de sincronização:
  - Se a nova tabela Iceberg for destinada ao uso em produção, interrompa todos os processos gravados na tabela original e redirecione-os para a nova tabela.
  - Se você precisar de um período de transição ou se a nova tabela do Iceberg for para fins de teste, consulte Mantendo as tabelas do Iceberg sincronizadas após a migração local, mais adiante nesta seção, para obter orientação sobre como manter a sincronização das tabelas.
- Quando você usa o `snapshot` procedimento, a `gc.enabled` propriedade é definida `false` nas propriedades da tabela Iceberg criada. Essa configuração proíbe ações como `expire_snapshotsremove_orphan_files`, ou `DROP TABLE` com a `PURGE` opção, que excluiriam fisicamente os arquivos de dados. As operações de exclusão ou mesclagem do Iceberg, que não afetam diretamente os arquivos de origem, ainda são permitidas.
- Para tornar sua nova tabela Iceberg totalmente funcional, sem limites nas ações que excluem fisicamente os arquivos de dados, você pode alterar a propriedade da `gc.enabled` tabela para`true`. No entanto, essa configuração permitirá ações que afetem os arquivos de dados de origem, o que pode corromper o acesso à tabela original. Portanto, altere a `gc.enabled` propriedade somente se você não precisar mais manter a funcionalidade da tabela original. Por exemplo:

```
spark.sql(f"""
ALTER TABLE mydb.products_iceberg
```

```
SET TBLPROPERTIES ('gc.enabled' = 'true');
""")
```

Opção 2: procedimento de migração

O `migrate` procedimento cria uma nova tabela Iceberg que tem o mesmo nome, esquema e particionamento da tabela de origem. Quando esse procedimento é executado, ele bloqueia a tabela de origem e a renomeia para `<table_name>_BACKUP_` (ou um nome personalizado especificado pelo parâmetro do `backup_table_name` procedimento).

> **Note**
>
> Se você definir o parâmetro do `drop_backup` procedimento como`true`, a tabela original não será mantida como backup.

Consequentemente, o procedimento da `migrate` tabela exige que todas as modificações que afetam a tabela de origem sejam interrompidas antes que a ação seja executada. Antes de executar o `migrate` procedimento:

- Pare todos os escritores que interagem com a tabela de origem.
- Modifique leitores e escritores que não oferecem suporte nativo ao Iceberg para ativar o suporte ao Iceberg.

Por exemplo:

- Athena continua trabalhando sem modificações.
- O Spark requer:
  - Arquivos Iceberg Java Archive (JAR) a serem incluídos no classpath (consulte Trabalhando com o Iceberg no Amazon EMR e Trabalhando com o Iceberg nas seções anteriores deste guia). AWS Glue
  - As seguintes configurações do catálogo de sessões do Spark (usadas `SparkSessionCatalog` para adicionar suporte ao Iceberg e, ao mesmo tempo, manter as funcionalidades integradas do catálogo para tabelas que não são do Iceberg):
    - `"spark.sql.extensions":"org.apache.iceberg.spark.extensions.IcebergSparkSes`

- `"spark.sql.catalog.spark_catalog":"org.apache.iceberg.spark.SparkSessionCat`
- `"spark.sql.catalog.spark_catalog.type":"glue"`
- `"spark.hadoop.hive.metastore.client.factory.class":"com.amazonaws.glue.cata`

Depois de executar o procedimento, você pode reiniciar seus gravadores com a nova configuração do Iceberg.

Atualmente, o `migrate` procedimento não é compatível com o AWS Glue Data Catalog, porque o Catálogo de Dados não suporta a `RENAME` operação. Portanto, recomendamos que você use esse procedimento somente quando estiver trabalhando com o Hive Metastore. Se você estiver usando o Catálogo de Dados, consulte a próxima seção para obter uma abordagem alternativa.

Você pode executar o `migrate` procedimento em todos os modelos de implantação do Amazon EMR (Amazon EMR no EC2, Amazon EMR no EKS, EMR Serverless), mas isso requer uma conexão configurada com o Hive Metastore. AWS Glue O Amazon EMR no EC2 é a opção recomendada porque fornece uma configuração integrada do Hive Metastore, o que minimiza a complexidade da configuração.

Para testar a migração local com o procedimento `migrate` Spark de um cluster do Amazon EMR no EC2 configurado com o Hive Metastore, siga estas etapas:

1. Inicie um aplicativo Spark e configure a sessão do Spark para usar a implementação do catálogo Iceberg Hive. Por exemplo, se você estiver usando a `pyspark` CLI:

```
pyspark --conf
 spark.sql.extensions=org.apache.iceberg.spark.extensions.IcebergSparkSessionExtensions
 --conf spark.sql.catalog.spark_catalog=org.apache.iceberg.spark.SparkSessionCatalog
 --conf spark.sql.catalog.spark_catalog.type=hive
```

2. Crie uma `products` tabela no Hive Metastore. Essa é a tabela de origem, que já existe em uma migração típica.
   a. Crie a tabela `products` externa do Hive no Hive Metastore para apontar para os dados existentes no Amazon S3:

```
spark.sql(f"""
CREATE EXTERNAL TABLE products (
    product_id INT,
    product_name STRING
)
```

```
PARTITIONED BY (category STRING)
STORED AS PARQUET
LOCATION 's3://amzn-s3-demo-bucket/products/';
"""
)
```

b. Adicione as partições existentes usando o `MSCK REPAIR TABLE` comando:

```
spark.sql(f"""
MSCK REPAIR TABLE products
"""
)
```

c. Confirme se a tabela contém dados executando uma `SELECT` consulta:

```
spark.sql(f"""
SELECT * FROM products
"""
).show(truncate=False)
```

Exemplo de saída:

```
>>> spark.sql(f"""
... SELECT * FROM products
... """
... ).show(truncate=False)
+----------+------------+-----------+
|product_id|product_name|category   |
+----------+------------+-----------+
|1001      |Smartphone  |electronics|
|1002      |Laptop      |electronics|
|2001      |T-Shirt     |clothing   |
|2002      |Jeans       |clothing   |
+----------+------------+-----------+
```

3. Use o `migrate` procedimento Iceberg:

```
df_res=spark.sql(f"""
CALL system.migrate(
table => 'default.products'
)
"""
)
```

```
df_res.show()
```

O quadro de dados de saída contém `migrated_files_count` (os números de arquivos que foram adicionados à tabela Iceberg):

```
>>> df_res.show()
+-------------------+
|migrated_files_count|
+-------------------+
|                  2|
+-------------------+
```

4. Confirme se a tabela de backup foi criada:

```
spark.sql("show tables").show()
```

Exemplo de saída:

```
>>> spark.sql("show tables").show()
+---------+---------------+-----------+
|namespace|      tableName|isTemporary|
+---------+---------------+-----------+
|  default|       products|      false|
|  default|products_backup_|      false|
+---------+---------------+-----------+
```

5. Valide a operação consultando a tabela Iceberg:

```
spark.sql(f"""
SELECT * FROM products
"""
).show(truncate=False)
```

Observações

- Depois de executar o procedimento, todos os processos atuais que consultam ou gravam na tabela de origem serão afetados se não estiverem configurados corretamente com o suporte do Iceberg. Portanto, recomendamos que você siga estas etapas:
  1. Pare todos os processos usando a tabela de origem antes da migração.
  2. Execute a migração.

3. Reative os processos usando as configurações adequadas do Iceberg.

- Se ocorrerem modificações no arquivo de dados durante o processo de migração (novos arquivos forem adicionados ou removidos), a tabela gerada ficará fora de sincronia. Para opções de sincronização, consulte Mantendo as tabelas do Iceberg sincronizadas após a migração local, mais adiante nesta seção.

Replicando o procedimento de migração de tabelas no AWS Glue Data Catalog

Você pode replicar o resultado do procedimento de migração em AWS Glue Data Catalog (fazendo backup da tabela original e substituindo-a por uma tabela Iceberg) seguindo estas etapas:

1. Use o procedimento de captura instantânea para criar uma nova tabela Iceberg que aponta para os arquivos de dados da tabela original.
2. Faça backup dos metadados da tabela original no Catálogo de Dados:
   a. Use a GetTableAPI para recuperar a definição da tabela de origem.
   b. Use a GetPartitionsAPI para recuperar a definição da partição da tabela de origem.
   c. Use a CreateTableAPI para criar uma tabela de backup no catálogo de dados.
   d. Use a BatchCreatePartitionAPI CreatePartitionor para registrar partições na tabela de backup no Catálogo de Dados.
3. Altere a propriedade da tabela `gc.enabled` Iceberg `false` para ativar as operações completas da tabela.
4. Descarte a tabela original.
5. Localize o arquivo JSON de metadados da tabela Iceberg na pasta de metadados da localização raiz da tabela.
6. Registre a nova tabela no Catálogo de Dados usando o procedimento register_table com o nome da tabela original e a localização do `metadata.json` arquivo que foi criado pelo procedimento: `snapshot`

```
spark.sql(f"""
CALL system.register_table(
    table => 'mydb.products',
    metadata_file => '{iceberg_metadata_file}'
)
```

```
"""
).show(truncate=False)
```

Mantendo as tabelas do Iceberg sincronizadas após a migração local

O `add_files` procedimento fornece uma maneira flexível de incorporar dados existentes às tabelas do Iceberg. Especificamente, ele registra arquivos de dados existentes (como arquivos Parquet) referenciando seus caminhos absolutos na camada de metadados do Iceberg. Por padrão, o procedimento adiciona arquivos de todas as partições da tabela a uma tabela Iceberg, mas você pode adicionar seletivamente arquivos de partições específicas. Essa abordagem seletiva é particularmente útil em vários cenários:

- Quando novas partições são adicionadas à tabela de origem após a migração inicial.
- Quando os arquivos de dados são adicionados ou removidos das partições existentes após a migração inicial. No entanto, adicionar novamente partições modificadas exige primeiro a exclusão da partição. Mais informações sobre isso são fornecidas posteriormente nesta seção.

Aqui estão algumas considerações sobre o uso do `add_file` procedimento após a realização da migração local (`snapshotoumigrate`), para manter a nova tabela Iceberg sincronizada com os arquivos de dados de origem:

- Quando novos dados forem adicionados a novas partições na tabela de origem, use o `add_files` procedimento com a `partition_filter` opção de incorporar seletivamente essas adições na tabela Iceberg:

```
spark.sql(f"""
CALL system.add_files(
source_table => 'mydb.products',
table => 'mydb.products_iceberg',
partition_filter => map('category', 'electronics')
).show(truncate=False)
```

ou:

```
spark.sql(f"""
CALL system.add_files(
source_table => ``parquet`.`s3://amzn-s3-demo-bucket/products/``,
table => 'mydb.products_iceberg',
```

```
partition_filter => map('category', 'electronics')
).show(truncate=False)
```

- O `add_files` procedimento verifica arquivos em toda a tabela de origem ou em partições específicas quando você especifica a `partition_filter` opção e tenta adicionar todos os arquivos encontrados à tabela Iceberg. Por padrão, a propriedade do `check_duplicate_files` procedimento é definida como`true`, o que impede a execução do procedimento se os arquivos já existirem na tabela Iceberg. Isso é importante porque não há uma opção integrada para ignorar arquivos adicionados anteriormente, e a desativação `check_duplicate_files` fará com que os arquivos sejam adicionados duas vezes, criando duplicatas. Quando novos arquivos forem adicionados à tabela de origem, siga estas etapas:

  1. Para novas partições, use `add_files` com a `partition_filter` para importar somente arquivos da nova partição.
  2. Para partições existentes, primeiro exclua a partição da tabela Iceberg e, em seguida, execute novamente `add_files` essa partição, especificando o. `partition_filter` Por exemplo:

```
# We initially perform in-place migration with snapshot
spark.sql(f"""
CALL system.snapshot(
source_table => 'mydb.products',
table => 'mydb.products_iceberg',
location => 's3://amzn-s3-demo-bucket/products_iceberg/'
)
"""
).show(truncate=False)

# Then on the source table, some new files were generated under the
 category='electronics' partition. Example:
spark.sql("""
INSERT INTO mydb.products
VALUES (1003, 'Tablet', 'electronics')
""")

# We delete the modified partition from the Iceberg table. Note this is a metadata
 operation only
spark.sql("""
DELETE FROM mydb.products_iceberg WHERE category = 'electronics'
""")

# We add_files from the modified partition
spark.sql("""
```

```
CALL system.add_files(
  source_table => 'mydb.products',
  table => 'mydb.products_iceberg',
  partition_filter => map('category', 'electronics')
)
""").show(truncate=False)
```

> **Note**
>
> Cada `add_files` operação gera um novo instantâneo da tabela Iceberg com dados anexados.

Escolhendo a estratégia certa de migração local

Para escolher a melhor estratégia de migração local, considere as perguntas na tabela a seguir.

Pergunta	Recomendação	Explicação
Você quer migrar rapidamente sem reescrever dados e, ao mesmo tempo, manter as tabelas Hive e Iceberg acessíveis para testes ou transições graduais?	`snapshot`procedimento seguido pelo `add_files` procedimento	Use o `snapshot` procedimento para criar uma nova tabela Iceberg clonando o esquema e referenciando arquivos de dados, sem modificar a tabela de origem. Use o `add_files` procedimento para incorporar partições que foram adicionadas ou modificadas após a migração, observando que a readição de partições modificadas exige primeiro a exclusão da partição.
Você está usando o Hive Metastore e deseja substituir sua tabela do Hive por uma	`migrate`procedimento seguido pelo `add_files` procedimento	Use o `migrate` procedimento para criar uma tabela Iceberg, fazer backup da tabela de

Pergunta	Recomendação	Explicação
tabela Iceberg imediatamente, sem reescrever os dados?		origem e substituir a tabela original pela versão Iceberg. Observação: essa opção é compatível com o Hive Metastore, mas não com o. AWS Glue Data Catalog Use o `add_files` procedimento para incorporar partições que foram adicionadas ou modificadas após a migração, observando que a readição de partições modificadas exige primeiro a exclusão da partição.

Pergunta	Recomendação	Explicação
Você está usando AWS Glue Data Catalog e deseja substituir sua tabela Hive por uma tabela Iceberg imediatamente, sem reescrever os dados?	Adaptação do `migrate` procedimento, seguida pelo `add_files` procedimento	Replique o comportamento `migrate` do procedimento: 1. Use `snapshot` para criar uma tabela Iceberg. 2. Faça backup dos metadados da tabela original usando AWS Glue APIs. 3. Ative `gc.enabled` as propriedades da tabela Iceberg. 4. Descarte a tabela original. 5. Use `register_table` para criar uma nova entrada de tabela com o nome original. Observação: essa opção exige o tratamento manual de chamadas de AWS Glue API para backup de metadados. Use o `add_files` procedimento para incorporar partições que foram adicionadas ou modificadas após a migração, observando que a readição de partições modificadas exige primeiro a exclusão da partição.

Migração completa de dados

A migração completa de dados recria os arquivos de dados e os metadados. Essa abordagem leva mais tempo e requer recursos computacionais adicionais em comparação com a migração local. No entanto, a migração total de dados oferece oportunidades significativas para melhorar a qualidade da tabela e otimizar o armazenamento de dados e os padrões de acesso.

Durante a migração completa dos dados, você pode realizar várias operações benéficas, como validação de dados para garantir integridade e exatidão, modificações no esquema para melhor atender aos requisitos atuais e ajustes na estratégia de partição para melhorar o desempenho das consultas. Você também pode reordenar os dados para otimizar padrões de acesso comuns, implementar o particionamento oculto do Iceberg para melhorar a eficiência da consulta e realizar a conversão do formato de arquivo (por exemplo, de CSV para Parquet), se desejar.

Esses recursos tornam a migração completa de dados ideal para a transição para o formato Iceberg e para refinar e otimizar de forma abrangente sua estratégia de armazenamento de dados. Embora a migração total dos dados exija mais tempo e recursos iniciais, as melhorias resultantes na qualidade dos dados, na organização e no desempenho das consultas podem proporcionar benefícios a longo prazo. Para implementar a migração completa de dados, use uma das seguintes opções:

- Use a declaração `CREATE TABLE ... AS SELECT` (CTAS) no Spark (no Amazon EMR ou) AWS Glue ou no Athena. Você pode definir a especificação da partição e as propriedades da tabela para a nova tabela Iceberg usando as `TBLPROPERTIES` cláusulas `PARTITIONED BY` e. Você pode alterar o esquema e o particionamento da nova tabela de acordo com suas necessidades, em vez de herdá-los da tabela de origem.
- Leia a tabela de origem e grave os dados como uma nova tabela Iceberg usando o Spark no Amazon EMR ou. AWS Glue Para obter mais informações, consulte Criação de uma tabela na documentação do Iceberg.

Selecionar uma estratégia de migração

Ao fazer a transição para o formato Iceberg, a escolha entre a migração local e a migração completa é crucial. Para determinar a abordagem mais adequada às suas necessidades específicas, considere as seguintes perguntas e recomendações:

Pergunta	Recomendação
Qual é o formato do arquivo de dados (por exemplo, CSV ou Apache Parquet)?	• Considere a migração local se o formato do arquivo de tabela for Parquet, ORC ou Avro. • Para outros formatos, como CSV, JSON e assim por diante, use a migração de dados completa.
Você quer atualizar ou consolidar o esquema da tabela?	• Se você quiser desenvolver o esquema da tabela usando os recursos nativos do Iceberg, considere a migração local. Por exemplo, você pode renomear colunas após a migração. (O esquema pode ser alterado na camada de metadados do Iceberg.) • Se você quiser remover colunas inteiras porque elas não são mais necessárias, recomendamos que você use a migração de dados completa.
A tabela se beneficiaria com a mudança da estratégia de partição?	• Se a abordagem de particionamento do Iceberg atender aos seus requisitos (por exemplo, novos dados são armazenad os usando o novo layout de partição enquanto as partições existentes permanecem como estão), considere a migração local. • Se você quiser usar partições ocultas em sua tabela, considere a migração completa dos dados. Para obter mais informações sobre partições ocultas, consulte a seção Práticas recomendadas.
A tabela se beneficiaria com a adição ou alteração da estratégia de ordem de classificação?	• Adicionar ou alterar a ordem de classificação dos seus dados exige a reescrita do conjunto de dados. Nesse caso, considere usar a migração de dados completa. • Para tabelas grandes em que é extremamente caro reescreve r todas as partições da tabela, considere usar a migração local e executar a compactação (com a classificação ativada) para as partições acessadas com mais frequência.
A tabela tem muitos arquivos pequenos?	• A mesclagem de arquivos pequenos em arquivos maiores exige a regravação do conjunto de dados. Nesse caso, considere usar a migração de dados completa.

Pergunta	Recomendação
	• Para tabelas grandes em que é extremamente caro reescrever todas as partições da tabela, considere usar a migração local e executar a compactação (com a classificação ativada) para as partições acessadas com mais frequência.

Resumo das opções de migração

Esta tabela resume as principais características e considerações de cada opção de migração.

Recurso	Migração local snapshot	Migração local migrate	Migração completa de dados CTAS ou (CRIAR TABELA + INSERIR)
Melhoria no layout de dados como parte do processo de migração			
• Reorde dados	 N	 N	 Sim
• Alterar o particio amento (por	 N	 N	 Sim

Recurso	Migração local snapshot		Migração local migrate		Migração completa de dados CTAS ou (CRIAR TABELA + INSERIR)	
exemp para usar o particio amento oculto do Iceber						
• Alterar esquer da tabela	✗	N	✗	N	✓	Sim
• Otimiz o tamanl do arquiv	✗	N	✗	N	✓	Sim
• Valide o esquer dos dados exister s antes de adicior os dados	✗	N	✗	N	✓	Sim

Recurso	Migração local snapshot	Migração local migrate	Migração completa de dados CTAS ou (CRIAR TABELA + INSERIR)
Formatos de arquivo compatíveis	Parquet, Avro, ORC	Parquet, Avro, ORC	Parquete, Avro, ORC, JSON, CSV
Substituição da tabela de origem por uma tabela Iceberg	 N (cria uma nova tabela, mas com etapas adicionais você pode substituir a tabela de origem)	 S (cria uma tabela de backup e substitui a tabela de origem por uma tabela Iceberg)	 Não (cria uma nova tabela)
Impacto da tabela de origem			

Recurso	Migração local snapshot	Migração local migrate	Migração completa de dados CTAS ou (CRIAR TABELA + INSERIR)
• Opera de exclus de arquiv na tabela Iceber (`expi` `apsho` operaç , elimina o de uma tabela com limpez	Corrompe a tabela de origem	Corrompe a tabela de backup	Seguro, fonte não afetada
Impacto na mesa de iceberg			

Recurso	Migração local snapshot	Migração local migrate	Migração completa de dados CTAS ou (CRIAR TABELA + INSERIR)
• Impact se os arquiv da tabela de origem forem remov	Corrompe a tabela Iceberg	Corrompe a tabela Iceberg	Sem impacto na mesa Iceberg
• Impact se novos arquiv forem adicion os no local da tabela de origem	Não visível na nova tabela (é necessári o incorpora r a partição com`add_files`)	Não visível na nova tabela (é necessário incorporar a partição com`add_files`)	Não visível na nova tabela (necessidade `INSERT INTO` da nova tabela)
Custos	Baixo	Baixo	Superior (regravação completa dos dados)
Velocida e de migraçã	Fast	Fast	Mais lento

Recurso	Migração local snapshot	Migração local migrate	Migração completa de dados CTAS ou (CRIAR TABELA + INSERIR)
Pode ser usado para migrar para tabelas do Amazon S3	 N	 N	 Sim
Requer DDL manual	 N (o esquema e as partições são copiados da tabela de origem)	 N (o esquema e as partições são copiados da tabela de origem)	Se estiver usando CTAS, requer apenas a especificação do particionamento
Melhor uso	Migração rápida sem reescrever dados, permitindo o side-by-side uso do Hive e do Iceberg para testes ou transições graduais.	Substituir uma tabela do Hive no local sem reescrever dados, quando uma transição imediata é aceitável.	Otimização completa do Iceberg com reescrita de dados. Ideal para redesenhar partições ou esquemas ou melhorar o layout e o desempenho. Sempre recomendado, se possível.

Melhores práticas para otimizar as cargas de trabalho do Apache Iceberg

O Iceberg é um formato de tabela projetado para simplificar o gerenciamento de data lake e aprimorar o desempenho da carga de trabalho. Casos de uso diferentes podem priorizar aspectos diferentes, como custo, desempenho de leitura, desempenho de gravação ou retenção de dados, então o Iceberg oferece opções de configuração para gerenciar essas compensações. Esta seção fornece informações para otimizar e ajustar suas cargas de trabalho do Iceberg para atender às suas necessidades.

Tópicos

Práticas recomendadas gerais

Independentemente do seu caso de uso, ao usar o Apache Iceberg on AWS, recomendamos que você siga essas melhores práticas gerais.

- Use a versão 2 do formato Iceberg.

  Athena usa o formato Iceberg versão 2 por padrão.

  Ao usar o Spark no Amazon EMR AWS Glue ou para criar tabelas do Iceberg, especifique a versão do formato conforme descrito na documentação do Iceberg.
- Use o AWS Glue Data Catalog como seu catálogo de dados.

  Athena usa o. AWS Glue Data Catalog por padrão.

  Ao usar o Spark no Amazon EMR AWS Glue ou para trabalhar com o Iceberg, adicione a seguinte configuração à sua sessão do Spark para usar o. AWS Glue Data Catalog Para obter mais

informações, consulte a seção Configurações do Spark para o Iceberg no AWS Glue início deste guia.

```
"spark.sql.catalog.<your_catalog_name>.type": "glue"
```

- Use o AWS Glue Data Catalog como gerenciador de bloqueio.

  O Athena usa o AWS Glue Data Catalog como gerenciador de bloqueio por padrão para tabelas Iceberg.

  Ao usar o Spark no Amazon EMR AWS Glue ou para trabalhar com o Iceberg, certifique-se de configurar sua sessão do Spark para AWS Glue Data Catalog usá-la como gerenciador de bloqueio. Para obter mais informações, consulte Optimistic Locking na documentação do Iceberg.
- Use a compressão Zstandard (ZSTD).

  O codec de compactação padrão do Iceberg é o gzip, que pode ser modificado usando a propriedade table. `write.<file_type>.compression-codec` O Athena já usa o ZSTD como codec de compactação padrão para tabelas Iceberg.

  Em geral, recomendamos o uso do codec de compactação ZSTD porque ele equilibra o GZIP e o Snappy e oferece bom read/write desempenho sem comprometer a taxa de compactação. Além disso, os níveis de compressão podem ser ajustados para atender às suas necessidades. Para obter mais informações, consulte Níveis de compressão ZSTD no Athena na documentação do Athena.

  O Snappy pode fornecer o melhor desempenho geral de leitura e gravação, mas tem uma taxa de compactação menor do que GZIP e ZSTD. Se você priorizar o desempenho, mesmo que isso signifique armazenar grandes volumes de dados no Amazon S3, o Snappy pode ser a melhor escolha.

Otimizando o desempenho de leitura

Esta seção discute as propriedades da tabela que você pode ajustar para otimizar o desempenho de leitura, independentemente do mecanismo.

Particionamento

Assim como nas tabelas do Hive, o Iceberg usa partições como camada primária de indexação para evitar a leitura de arquivos de metadados e arquivos de dados desnecessários. As estatísticas das

colunas também são consideradas como uma camada secundária de indexação para melhorar ainda mais o planejamento de consultas, o que leva a um melhor tempo geral de execução.

Particionar dados

Para reduzir a quantidade de dados que são digitalizados ao consultar tabelas do Iceberg, escolha uma estratégia de partição balanceada que se alinhe aos padrões de leitura esperados:

- Identifique as colunas que são usadas com frequência em consultas. Esses são candidatos ideais para particionamento. Por exemplo, se você normalmente consulta dados de um determinado dia, um exemplo natural de uma coluna de partição seria uma coluna de data.
- Escolha uma coluna de partição de baixa cardinalidade para evitar criar um número excessivo de partições. Muitas partições podem aumentar o número de arquivos na tabela, o que pode afetar negativamente o desempenho da consulta. Como regra geral, “muitas partições” podem ser definidas como um cenário em que o tamanho dos dados na maioria das partições é menor que 2 a 5 vezes o valor definido por. `target-file-size-bytes`

> **Note**
>
> Se você costuma consultar usando filtros em uma coluna de alta cardinalidade (por exemplo, uma `id` coluna que pode ter milhares de valores), use o recurso de particionamento oculto do Iceberg com transformações de bucket, conforme explicado na próxima seção.

Use particionamento oculto

Se suas consultas geralmente são filtradas por um derivado de uma coluna de tabela, use partições ocultas em vez de criar explicitamente novas colunas para funcionarem como partições. Para obter mais informações sobre esse recurso, consulte a documentação do Iceberg.

Por exemplo, em um conjunto de dados que tem uma coluna de carimbo de data/hora (por exemplo,`2023-01-01 09:00:00`), em vez de criar uma nova coluna com a data analisada (por exemplo,`2023-01-01`), use transformações de partição para extrair a parte da data do carimbo de data/hora e criar essas partições rapidamente.

Os casos de uso mais comuns para particionamento oculto são:

- Particionamento em data ou hora, quando os dados têm uma coluna de carimbo de data/hora. O Iceberg oferece várias transformações para extrair as partes de data ou hora de um carimbo de data/hora.
- Particionamento em uma função hash de uma coluna, quando a coluna de particionamento tem alta cardinalidade e resultaria em muitas partições. O bucket do Iceberg transforma grupos de vários valores de partição em menos partições ocultas (bucket) usando funções de hash na coluna de particionamento.

Consulte as transformações de partição na documentação do Iceberg para obter uma visão geral de todas as transformações de partição disponíveis.

As colunas usadas para particionamento oculto podem se tornar parte dos predicados de consulta por meio do uso de funções SQL regulares, como e. `year()` `month()` Os predicados também podem ser combinados com operadores como `BETWEEN` e. `AND`

> **Note**
>
> O Iceberg não pode realizar a remoção de partições para funções que geram um tipo de dados diferente; por exemplo,. `substring(event_time, 1, 10) = '2022-01-01'`

Use a evolução da partição

Use a evolução de partições do Iceberg quando a estratégia de partição existente não for a ideal. Por exemplo, se você escolher partições horárias que acabem sendo muito pequenas (apenas alguns megabytes cada), considere mudar para partições diárias ou mensais.

Você pode usar essa abordagem quando a melhor estratégia de particionamento para uma tabela inicialmente não está clara e você deseja refinar sua estratégia de particionamento à medida que obtém mais informações. Outro uso efetivo da evolução de partições é quando os volumes de dados mudam e a estratégia atual de particionamento se torna menos eficaz com o tempo.

Para obter instruções sobre como desenvolver partições, consulte as extensões ALTER TABLE SQL na documentação do Iceberg.

Ajustando tamanhos de arquivos

A otimização do desempenho da consulta envolve minimizar o número de arquivos pequenos em suas tabelas. Para um bom desempenho de consulta, geralmente recomendamos manter os arquivos Parquet e ORC maiores que 100 MB.

O tamanho do arquivo também afeta o planejamento de consultas para tabelas Iceberg. À medida que o número de arquivos em uma tabela aumenta, também aumenta o tamanho dos arquivos de metadados. Arquivos de metadados maiores podem resultar em um planejamento de consultas mais lento. Portanto, quando o tamanho da tabela aumentar, aumente o tamanho do arquivo para aliviar a expansão exponencial dos metadados.

Use as práticas recomendadas a seguir para criar arquivos de tamanho adequado nas tabelas do Iceberg.

Defina o tamanho do arquivo de destino e do grupo de linhas

O Iceberg oferece os seguintes parâmetros-chave de configuração para ajustar o layout do arquivo de dados. Recomendamos que você use esses parâmetros para definir o tamanho do arquivo de destino e o grupo de linhas ou tamanho do traçado.

Parâmetro	Valor padrão	Comentário
`write.target-file-size-bytes`	512 MB	Esse parâmetro especific a o tamanho máximo do arquivo que o Iceberg criará. No entanto, alguns arquivos podem ser gravados com um tamanho menor que esse limite.
`write.parquet.row-group-size-bytes`	128 MB	Tanto o Parquet quanto o ORC armazenam dados em partes para que os mecanismo s possam evitar a leitura do arquivo inteiro em algumas operações.

Parâmetro	Valor padrão	Comentário
`write.orc.stripe-size-bytes`	64 MB	
`write.distribution-mode`	Nenhum, para a versão 1.1 e inferior do Iceberg Hash, começando com a versão 1.2 do Iceberg	O Iceberg solicita que o Spark classifique os dados entre suas tarefas antes de gravar no armazenamento.

- Com base no tamanho esperado da tabela, siga estas diretrizes gerais:
  - Tabelas pequenas (até alguns gigabytes) — Reduza o tamanho do arquivo de destino para 128 MB. Reduza também o tamanho do grupo de linhas ou da faixa (por exemplo, para 8 ou 16 MB).
  - Tabelas de médio a grande porte (de alguns gigabytes a centenas de gigabytes) — Os valores padrão são um bom ponto de partida para essas tabelas. Se suas consultas forem muito seletivas, ajuste o grupo de linhas ou o tamanho da faixa (por exemplo, para 16 MB).
  - Tabelas muito grandes (centenas de gigabytes ou terabytes) — Aumente o tamanho do arquivo de destino para 1024 MB ou mais e considere aumentar o tamanho do grupo de linhas ou da faixa se suas consultas geralmente extraírem grandes conjuntos de dados.

- Para garantir que os aplicativos Spark que gravam nas tabelas do Iceberg criem arquivos de tamanho adequado, defina a `write.distribution-mode` propriedade como ou. `hash` `range` Para obter uma explicação detalhada da diferença entre esses modos, consulte Escrevendo modos de distribuição na documentação do Iceberg.

Essas são diretrizes gerais. Recomendamos que você execute testes para identificar os valores mais adequados para suas tabelas e cargas de trabalho específicas.

Execute a compactação regular

As configurações na tabela anterior definem um tamanho máximo de arquivo que as tarefas de gravação podem criar, mas não garantem que os arquivos tenham esse tamanho. Para garantir tamanhos de arquivo adequados, execute a compactação regularmente para combinar arquivos pequenos em arquivos maiores. Para obter orientação detalhada sobre como executar a compactação, consulte a compactação Iceberg mais adiante neste guia.

Otimize as estatísticas da coluna

O Iceberg usa estatísticas de colunas para realizar a remoção de arquivos, o que melhora o desempenho das consultas ao reduzir a quantidade de dados que são digitalizados pelas consultas. Para se beneficiar das estatísticas de colunas, certifique-se de que o Iceberg colete estatísticas de todas as colunas que são usadas com frequência em filtros de consulta.

Por padrão, o Iceberg coleta estatísticas somente para as primeiras 100 colunas em cada tabela, conforme definido pela propriedade da tabela. `write.metadata.metrics.max-inferred-column-defaults` Se sua tabela tiver mais de 100 colunas e suas consultas frequentemente referenciarem colunas fora das primeiras 100 colunas (por exemplo, você pode ter consultas que filtram na coluna 132), certifique-se de que o Iceberg colete estatísticas sobre essas colunas. Há duas opções para conseguir isso:

- Ao criar a tabela Iceberg, reordene as colunas para que as colunas necessárias para consultas estejam dentro do intervalo de colunas definido por `write.metadata.metrics.max-inferred-column-defaults` (o padrão é 100).

  Observação: se você não precisar de estatísticas em 100 colunas, poderá ajustar a `write.metadata.metrics.max-inferred-column-defaults` configuração para um valor desejado (por exemplo, 20) e reordenar as colunas para que as colunas necessárias para ler e escrever consultas caiam nas primeiras 20 colunas no lado esquerdo do conjunto de dados.
- Se você usar apenas algumas colunas nos filtros de consulta, poderá desativar a propriedade geral da coleta de métricas e escolher seletivamente colunas individuais para as quais coletar estatísticas, conforme mostrado neste exemplo:

```
.tableProperty("write.metadata.metrics.default", "none")
.tableProperty("write.metadata.metrics.column.my_col_a", "full")
.tableProperty("write.metadata.metrics.column.my_col_b", "full")
```

Observação: as estatísticas das colunas são mais eficazes quando os dados são classificados nessas colunas. Para obter mais informações, consulte a seção Definir a ordem de classificação mais adiante neste guia.

Escolha a estratégia de atualização correta

Use uma copy-on-write estratégia para otimizar o desempenho de leitura, quando operações de gravação mais lentas forem aceitáveis para seu caso de uso. Essa é a estratégia padrão usada pelo Iceberg.

Copy-on-write resulta em melhor desempenho de leitura, porque os arquivos são gravados diretamente no armazenamento de forma otimizada para leitura. No entanto, em comparação com merge-on-read, cada operação de gravação leva mais tempo e consome mais recursos computacionais. Isso apresenta uma compensação clássica entre latência de leitura e gravação. Normalmente, copy-on-write é ideal para casos de uso em que a maioria das atualizações é colocada nas mesmas partições de tabela (por exemplo, para cargas diárias em lote).

Copy-on-write as configurações (`write.update.mode`,`write.delete.mode`, `ewrite.merge.mode`) podem ser definidas no nível da tabela ou de forma independente no lado do aplicativo.

Use compressão ZSTD

Você pode modificar o codec de compactação usado pelo Iceberg usando a propriedade table. `write.<file_type>.compression-codec` Recomendamos que você use o codec de compressão ZSTD para melhorar o desempenho geral nas tabelas.

Por padrão, as versões 1.3 e anteriores do Iceberg usam a compressão GZIP, que fornece um read/write desempenho mais lento em comparação com o ZSTD.

Observação: alguns mecanismos podem usar valores padrão diferentes. Esse é o caso das tabelas Iceberg criadas com o Athena ou o Amazon EMR versão 7.x.

Definir a ordem de classificação

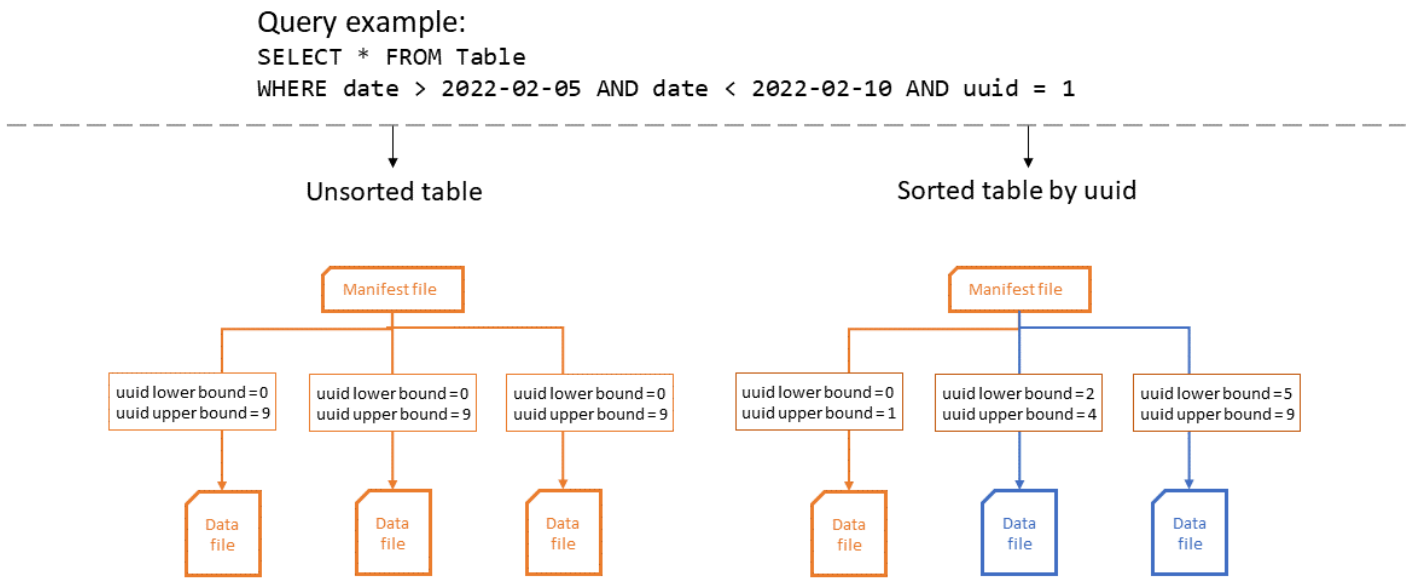
Para melhorar o desempenho de leitura nas tabelas do Iceberg, recomendamos que você classifique sua tabela com base em uma ou mais colunas usadas com frequência em filtros de consulta. A classificação, combinada com as estatísticas das colunas do Iceberg, pode tornar a remoção de arquivos significativamente mais eficiente, o que resulta em operações de leitura mais rápidas. A classificação também reduz o número de solicitações do Amazon S3 para consultas que usam as colunas de classificação nos filtros de consulta.

Você pode definir uma ordem de classificação hierárquica no nível da tabela executando uma instrução de linguagem de definição de dados (DDL) com o Spark. Para ver as opções disponíveis,

consulte a documentação do Iceberg. Depois de definir a ordem de classificação, os gravadores aplicarão essa classificação às operações subsequentes de gravação de dados na tabela Iceberg.

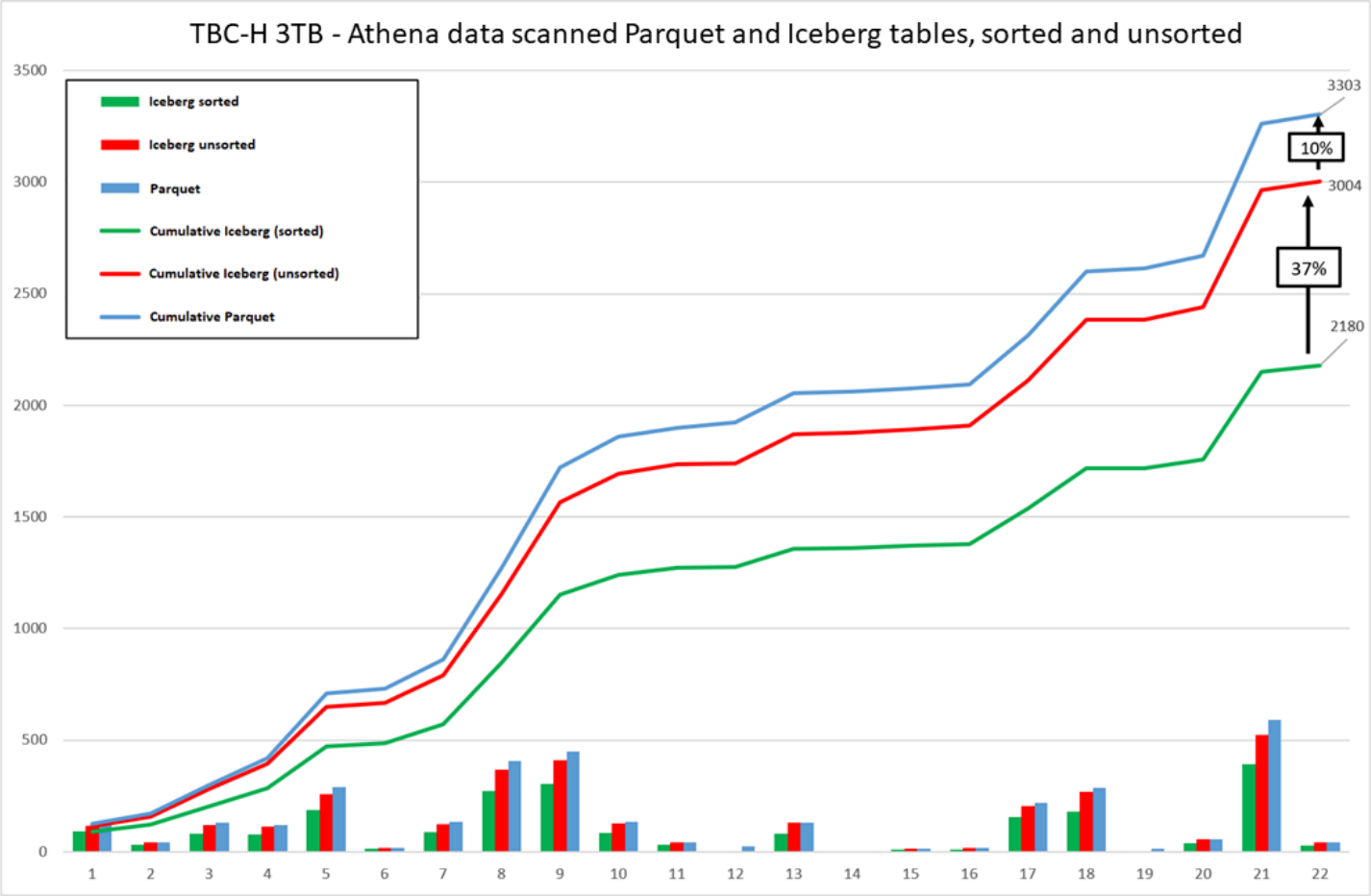
Por exemplo, em tabelas particionadas por date (`yyyy-mm-dd`) em que a maioria das consultas é filtrada`uuid`, você pode usar a opção DDL `Write Distributed By Partition Locally Ordered` para garantir que o Spark grave arquivos com intervalos não sobrepostos.

O diagrama a seguir ilustra como a eficiência das estatísticas de coluna melhora quando as tabelas são classificadas. No exemplo, a tabela classificada precisa abrir apenas um único arquivo e se beneficia ao máximo da partição e do arquivo do Iceberg. Na tabela não classificada, qualquer um `uuid` pode existir potencialmente em qualquer arquivo de dados, portanto, a consulta precisa abrir todos os arquivos de dados.

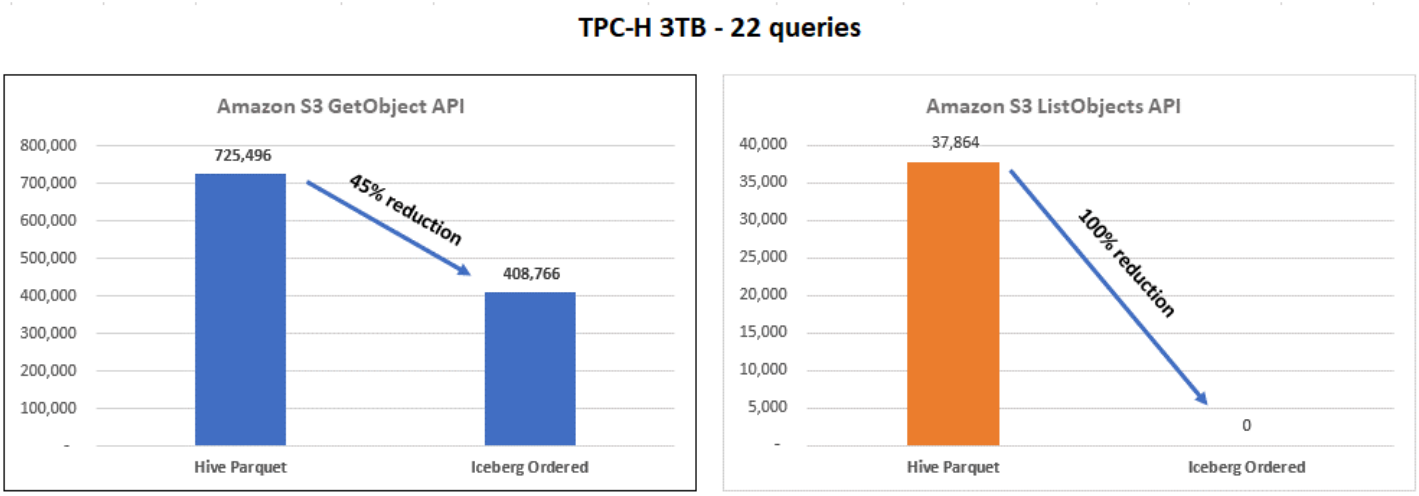


A alteração da ordem de classificação não afeta os arquivos de dados existentes. Você pode usar a compactação Iceberg para aplicar a ordem de classificação a eles.

Usar as tabelas classificadas do Iceberg pode diminuir os custos de sua carga de trabalho, conforme ilustrado no gráfico a seguir.



Esses gráficos resumem os resultados da execução do benchmark TPC-H para tabelas Hive (Parquet) em comparação com as tabelas classificadas do Iceberg. No entanto, os resultados podem ser diferentes para outros conjuntos de dados ou cargas de trabalho.



Otimizando o desempenho de gravação

Esta seção discute as propriedades da tabela que você pode ajustar para otimizar o desempenho de gravação nas tabelas do Iceberg, independentemente do mecanismo.

Definir o modo de distribuição da tabela

O Iceberg oferece vários modos de distribuição de gravação que definem como os dados de gravação são distribuídos nas tarefas do Spark. Para uma visão geral dos modos disponíveis, consulte Escrevendo modos de distribuição na documentação do Iceberg.

Para casos de uso que priorizam a velocidade de gravação, especialmente em cargas de trabalho de streaming, defina como. `write.distribution-mode` `none` Isso garante que o Iceberg não solicite embaralhamento adicional do Spark e que os dados sejam gravados assim que forem disponibilizados nas tarefas do Spark. Esse modo é particularmente adequado para aplicativos Spark Structured Streaming.

> **Note**
>
> Definir o modo de distribuição de gravação como `none` tende a produzir vários arquivos pequenos, o que degrada o desempenho de leitura. Recomendamos a compactação regular para consolidar esses arquivos pequenos em arquivos de tamanho adequado para o desempenho da consulta.

Escolha a estratégia de atualização correta

Use uma merge-on-read estratégia para otimizar o desempenho de gravação, quando operações de leitura mais lentas nos dados mais recentes forem aceitáveis para seu caso de uso.

Quando você usa merge-on-read, o Iceberg grava atualizações e exclui no armazenamento como pequenos arquivos separados. Quando a tabela é lida, o leitor precisa mesclar essas alterações com os arquivos base para retornar a visão mais recente dos dados. Isso resulta em uma penalidade de desempenho nas operações de leitura, mas acelera a gravação de atualizações e exclusões. Normalmente, merge-on-read é ideal para streaming de cargas de trabalho com atualizações ou trabalhos com poucas atualizações espalhadas por várias partições de tabela.

Você pode definir merge-on-read as configurações (`write.update.mode`write.delete.mode, ewrite.merge.mode) no nível da tabela ou de forma independente no lado do aplicativo.

O uso merge-on-read requer a execução de compactação regular para evitar que o desempenho de leitura se degrade com o tempo. A compactação reconcilia atualizações e exclusões com arquivos de dados existentes para criar um novo conjunto de arquivos de dados, eliminando assim a penalidade de desempenho incorrida no lado da leitura. Por padrão, a compactação do Iceberg não mescla arquivos excluídos, a menos que você altere o padrão da `delete-file-threshold` propriedade para um valor menor (consulte a documentação do Iceberg). Para saber mais sobre compactação, consulte a seção Compactação do Iceberg mais adiante neste guia.

Escolha o formato de arquivo correto

O Iceberg suporta a gravação de dados nos formatos Parquet, ORC e Avro. O parquet é o formato padrão. Parquet e ORC são formatos colunares que oferecem desempenho de leitura superior, mas geralmente são mais lentos para gravar. Isso representa a compensação típica entre desempenho de leitura e gravação.

Se a velocidade de gravação for importante para seu caso de uso, como em cargas de trabalho de streaming, considere escrever no formato Avro `write-format` configurando como `Avro` nas opções do gravador. Como o Avro é um formato baseado em linhas, ele fornece tempos de gravação mais rápidos ao custo de um desempenho de leitura mais lento.

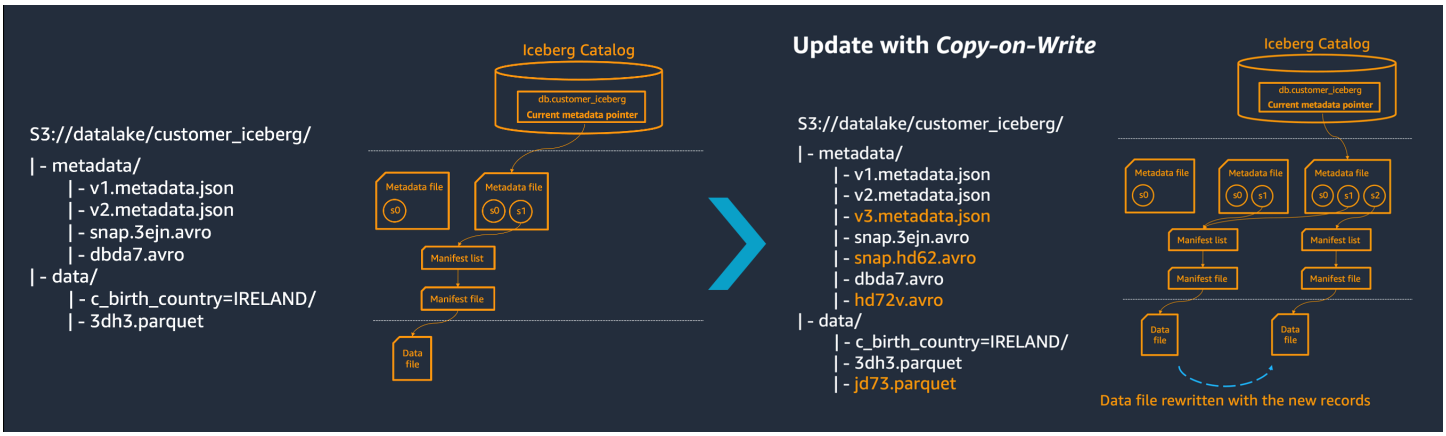
Para melhorar o desempenho de leitura, execute a compactação regular para mesclar e transformar pequenos arquivos Avro em arquivos Parquet maiores. O resultado do processo de compactação é determinado pela configuração da `write.format.default` tabela. O formato padrão do Iceberg é o Parquet, então se você escrever em Avro e depois executar a compactação, o Iceberg transformará os arquivos Avro em arquivos Parquet. Veja um exemplo abaixo:

```
spark.sql(f"""
    CREATE TABLE IF NOT EXISTS glue_catalog.{DB_NAME}.{TABLE_NAME} (
        Col_1 float,
        <<<…other columns…>>
        ts timestamp)
    USING iceberg
    PARTITIONED BY (days(ts))
    OPTIONS (
      'format-version'='2',
      write.format.default'=parquet)
""")

query = df \
    .writeStream \
```

```
    .format("iceberg") \
    .option("write-format", "avro") \
    .outputMode("append") \
    .trigger(processingTime='60 seconds') \
    .option("path", f"glue_catalog.{DB_NAME}.{TABLE_NAME}") \
    .option("checkpointLocation", f"s3://{BUCKET_NAME}/checkpoints/iceberg/")

    .start()
```

Otimizando o armazenamento

Atualizar ou excluir dados em uma tabela Iceberg aumenta o número de cópias de seus dados, conforme ilustrado no diagrama a seguir. O mesmo vale para a execução da compactação: ela aumenta o número de cópias de dados no Amazon S3. Isso porque o Iceberg trata os arquivos subjacentes a todas as tabelas como imutáveis.



Siga as práticas recomendadas nesta seção para gerenciar os custos de armazenamento.

Habilite o S3 Intelligent Tiering

Use a classe de armazenamento Amazon S3 Intelligent-Tiering para mover automaticamente os dados para o nível de acesso mais econômico quando os padrões de acesso mudarem. Essa opção não tem sobrecarga operacional nem impacto no desempenho.

Observação: não use os níveis opcionais (como Archive Access e Deep Archive Access) no S3 Intelligent-Tiering com tabelas Iceberg. Para arquivar dados, consulte as diretrizes na próxima seção.

Você também pode usar as regras de ciclo de vida do Amazon S3 para definir suas próprias regras para mover objetos para outra classe de armazenamento do Amazon S3, como S3 Standard-IA ou

S3 One Zone-IA (consulte Transições suportadas e restrições relacionadas na documentação do Amazon S3).

Arquivar ou excluir instantâneos históricos

Para cada transação confirmada (inserção, atualização, mesclagem, compactação) em uma tabela Iceberg, uma nova versão ou instantâneo da tabela é criada. Com o tempo, o número de versões e o número de arquivos de metadados no Amazon S3 se acumulam.

É necessário manter instantâneos de uma tabela para recursos como isolamento de instantâneos, reversão de tabelas e consultas de viagem no tempo. No entanto, os custos de armazenamento aumentam com o número de versões que você retém.

A tabela a seguir descreve os padrões de design que você pode implementar para gerenciar custos com base em seus requisitos de retenção de dados.

Padrão de design	Solução	Casos de uso
Excluir instantâneos antigos	• Use a instrução VACUUM no Athena para remover instantâneos antigos. Essa operação não incorre em nenhum custo computaci onal. • Como alternativa, você pode usar o Spark no Amazon EMR AWS Glue ou remover instantâneos. Para obter mais informaçõ es, consulte expire_sn apshots na documentação do Iceberg.	Essa abordagem exclui instantâneos que não são mais necessários para reduzir os custos de armazenamento. Você pode configurar quantos instantâneos devem ser retidos ou por quanto tempo, com base em seus requisitos de retenção de dados. Essa opção executa uma exclusão definitiva dos instantâneos. Você não pode reverter ou viajar no tempo para instantâneos expirados.
Defina políticas de retenção para instantâneos específicos	1. Use tags para marcar instantâneos específicos e definir uma política de retenção no Iceberg. Para	Esse padrão é útil para a conformidade com requisito s comerciais ou legais que exigem que você mostre o

Padrão de design	Solução	Casos de uso
	obter mais informações, consulte Tags históricas na documentação do Iceberg. Por exemplo, você pode reter um snapshot por mês durante um ano usando a seguinte instrução SQL no Spark no Amazon EMR: `ALTER TABLE glue_catalog.db.table CREATE TAG 'EOM-01' AS OF VERSION 30 RETAIN 365 DAYS` 2. Use o Spark no Amazon EMR AWS Glue ou remova os demais snapshots intermediários não marcados.	estado de uma tabela em um determinado momento no passado. Ao colocar políticas de retenção em instantâneos marcados específicos, você pode remover outros instantâneos (não marcados) que foram criados. Dessa forma, você pode atender aos requisitos de retenção de dados sem reter cada snapshot criado.

Padrão de design	Solução	Casos de uso
Arquive instantâneos antigos	1. Use as tags do Amazon S3 para marcar objetos com o Spark. (As tags do Amazon S3 são diferentes das tags Iceberg; para obter mais informações, consulte a documentação do Iceberg.) Por exemplo: `spark.sql.catalog.my_catalog.s3.delete-enabled=false and \ spark.sql.catalog.my_catalog.s3.delete.tags.my_key=to_archive` 2. Use o Spark no Amazon EMR AWS Glue ou para remover instantâneos. Quando você usa as configurações no exemplo, esse procedimento marca objetos e os separa dos metadados da tabela Iceberg em vez de excluí-los do Amazon S3. 3. Use as regras de ciclo de vida do S3 para fazer a transição de objetos marcados como `to_archive` uma das classes de armazenamento do S3 Glacier.	Esse padrão permite que você mantenha todas as versões e instantâneos da tabela a um custo menor. Você não pode viajar no tempo ou reverter para instantâneos arquivados sem primeiro restaurar essas versões como novas tabelas. Isso geralmente é aceitável para fins de auditoria. Você pode combinar essa abordagem com o padrão de design anterior, definindo políticas de retenção para instantâneos específicos.

Padrão de design	Solução	Casos de uso
	4. Para consultar dados arquivados: • Restaure os objetos arquivados (essa etapa não é necessária se os objetos tiverem sido transferidos para a classe de armazenamento Amazon Glacier Instant Retrieval). • Use o procedimento register_table no Iceberg para registrar o instantâneo como uma tabela no catálogo. Para obter instruções detalhadas, consulte a postagem do AWS blog Melhore a eficiência operacional das tabelas Apache Iceberg criadas nos data lakes do Amazon S3.	

Excluir arquivos órfãos

Em determinadas situações, os aplicativos do Iceberg podem falhar antes que você confirme suas transações. Isso deixa os arquivos de dados no Amazon S3. Como não houve confirmação, esses arquivos não serão associados a nenhuma tabela, então talvez seja necessário limpá-los de forma assíncrona.

Para lidar com essas exclusões, você pode usar a instrução VACUUM no Amazon Athena. Essa instrução remove instantâneos e também exclui arquivos órfãos. Isso é muito econômico, porque o Athena não cobra pelo custo computacional dessa operação. Além disso, você não precisa agendar nenhuma operação adicional ao usar a `VACUUM` instrução.

Como alternativa, você pode usar o Spark no Amazon EMR AWS Glue ou executar `remove_orphan_files` o procedimento. Essa operação tem um custo computacional e precisa ser programada de forma independente. Para obter mais informações, consulte a documentação do Iceberg.

Manutenção de tabelas usando compactação

O Iceberg inclui recursos que permitem realizar operações de manutenção da tabela depois de gravar dados na tabela. Algumas operações de manutenção se concentram na simplificação dos arquivos de metadados, enquanto outras aprimoram a forma como os dados são agrupados nos arquivos para que os mecanismos de consulta possam localizar com eficiência as informações necessárias para responder às solicitações do usuário. Esta seção se concentra nas otimizações relacionadas à compactação.

Compactação de iceberg

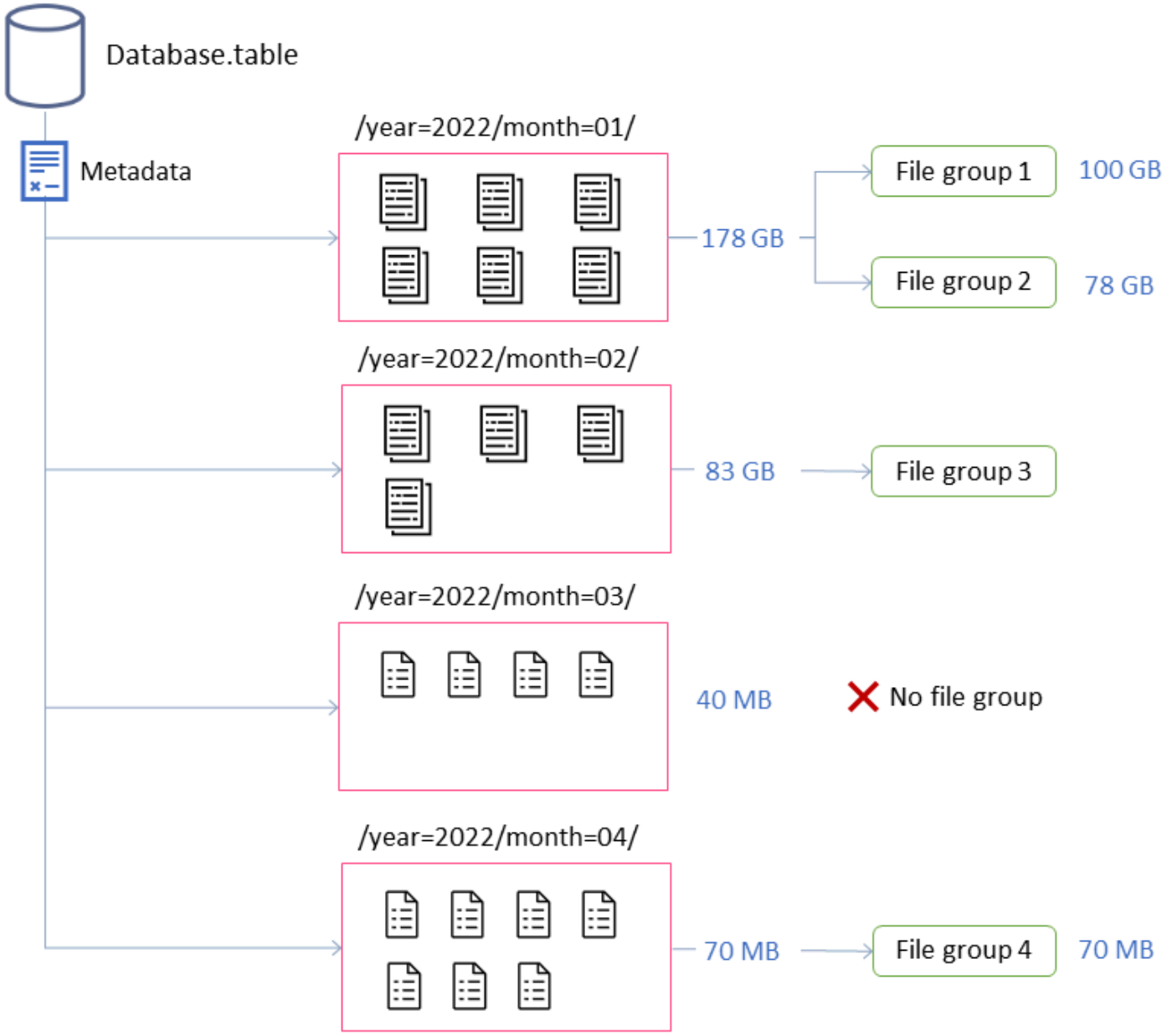
No Iceberg, você pode usar a compactação para realizar quatro tarefas:

- Combinação de arquivos pequenos em arquivos maiores que geralmente têm mais de 100 MB. Essa técnica é conhecida como embalagem de lixo.
- Mesclando arquivos excluídos com arquivos de dados. Os arquivos excluídos são gerados por atualizações ou exclusões que usam a merge-on-read abordagem.
- (Re) classificando os dados de acordo com os padrões de consulta. Os dados podem ser gravados sem qualquer ordem de classificação ou com uma ordem de classificação adequada para gravações e atualizações.
- Agrupar os dados usando curvas de preenchimento de espaço para otimizar padrões de consulta distintos, particularmente a classificação de ordem z.

Ativado AWS, você pode executar operações de compactação e manutenção de tabelas para o Iceberg por meio do Amazon Athena ou usando o Spark no Amazon EMR ou. AWS Glue

Ao executar a compactação usando o procedimento rewrite_data_files, você pode ajustar vários botões para controlar o comportamento da compactação. O diagrama a seguir mostra o

comportamento padrão do empacotamento de lixo. Compreender a compactação de compartimentos é fundamental para entender as implementações de classificação hierárquica e classificação de ordem Z, porque elas são extensões da interface de embalagem de compartimentos e operam de maneira semelhante. A principal distinção é a etapa adicional necessária para classificar ou agrupar os dados.



Neste exemplo, a tabela Iceberg consiste em quatro partições. Cada partição tem um tamanho diferente e um número diferente de arquivos. Se você iniciar um aplicativo Spark para executar a compactação, o aplicativo criará um total de quatro grupos de arquivos para processar. Um grupo de arquivos é uma abstração do Iceberg que representa uma coleção de arquivos que serão

processados por uma única tarefa do Spark. Ou seja, o aplicativo Spark que executa a compactação criará quatro trabalhos do Spark para processar os dados.

Ajustando o comportamento de compactação

As propriedades principais a seguir controlam como os arquivos de dados são selecionados para compactação:

- MAX_FILE_GROUP_SIZE_BYTES define o limite de dados para um único grupo de arquivos (tarefa do Spark) em 100 GB por padrão. Essa propriedade é especialmente importante para tabelas sem partições ou tabelas com partições que abrangem centenas de gigabytes. Ao definir esse limite, você pode dividir as operações para planejar o trabalho e progredir, evitando o esgotamento dos recursos no cluster.

  Nota: Cada grupo de arquivos é classificado separadamente. Portanto, se você quiser realizar uma classificação em nível de partição, deverá ajustar esse limite para corresponder ao tamanho da partição.
- MIN_FILE_SIZE_BYTES ou MIN_FILE_SIZE_DEFAULT_RATIO assumem como padrão 75 por cento do tamanho do arquivo de destino definido no nível da tabela. Por exemplo, se uma tabela tiver um tamanho alvo de 512 MB, qualquer arquivo menor que 384 MB será incluído no conjunto de arquivos que serão compactados.
- MAX_FILE_SIZE_BYTES ou MAX_FILE_SIZE_DEFAULT_RATIO assumem como padrão 180 por cento do tamanho do arquivo de destino. Assim como as duas propriedades que definem tamanhos mínimos de arquivo, essas propriedades são usadas para identificar arquivos candidatos para o trabalho de compactação.
- MIN_INPUT_FILES especifica o número mínimo de arquivos a serem compactados se o tamanho da partição da tabela for menor que o tamanho do arquivo de destino. O valor dessa propriedade é usado para determinar se vale a pena compactar os arquivos com base no número de arquivos (o padrão é 5).
- DELETE_FILE_THRESHOLD especifica o número mínimo de operações de exclusão de um arquivo antes de ser incluído na compactação. A menos que você especifique o contrário, a compactação não combina arquivos excluídos com arquivos de dados. Para habilitar essa funcionalidade, você deve definir um valor limite usando essa propriedade. Esse limite é específico para arquivos de dados individuais, portanto, se você defini-lo como 3, um arquivo de dados será regravado somente se houver três ou mais arquivos de exclusão que façam referência a ele.

Essas propriedades fornecem informações sobre a formação dos grupos de arquivos no diagrama anterior.

Por exemplo, a partição rotulada `month=01` inclui dois grupos de arquivos porque excede a restrição de tamanho máximo de 100 GB. Por outro lado, a `month=02` partição contém um único grupo de arquivos porque tem menos de 100 GB. A `month=03` partição não atende ao requisito mínimo padrão de arquivo de entrada de cinco arquivos. Como resultado, ele não será compactado. Por fim, embora a `month=04` partição não contenha dados suficientes para formar um único arquivo do tamanho desejado, os arquivos serão compactados porque a partição inclui mais de cinco arquivos pequenos.

Você pode definir esses parâmetros para o Spark executado no Amazon AWS Glue EMR ou. Para o Amazon Athena, você pode gerenciar propriedades semelhantes usando as propriedades da tabela (que começam com o prefixo`optimize_`).

Executando a compactação com o Spark no Amazon EMR ou AWS Glue

Esta seção descreve como dimensionar adequadamente um cluster Spark para executar o utilitário de compactação do Iceberg. O exemplo a seguir usa o Amazon EMR Serverless, mas você pode usar a mesma metodologia no Amazon EMR no EC2 ou EKS, ou no. AWS Glue

Você pode aproveitar a correlação entre grupos de arquivos e trabalhos do Spark para planejar os recursos do cluster. Para processar os grupos de arquivos sequencialmente, considerando o tamanho máximo de 100 GB por grupo de arquivos, você pode definir as seguintes propriedades do Spark:

- `spark.dynamicAllocation.enabled` = `FALSE`
- `spark.executor.memory` = `20 GB`
- `spark.executor.instances` = `5`

Se quiser acelerar a compactação, você pode escalar horizontalmente aumentando o número de grupos de arquivos que são compactados em paralelo. Você também pode escalar o Amazon EMR usando escalabilidade manual ou dinâmica.

- Dimensionamento manual (por exemplo, por um fator de 4)
  - `MAX_CONCURRENT_FILE_GROUP_REWRITES`= `4` (nosso fator)
  - `spark.executor.instances`= `5` (valor usado no exemplo) x `4` (nosso fator) = `20`
  - `spark.dynamicAllocation.enabled` = `FALSE`

- Dimensionamento dinâmico
  - `spark.dynamicAllocation.enabled= TRUE` (padrão, nenhuma ação é necessária)
  - MAX_CONCURRENT_FILE_GROUP_REWRITES = `N` (alinhe esse valor com`spark.dynamicAllocation.maxExecutors`, que é 100 por padrão; com base nas configurações do executor no exemplo, você pode definir como 20) `N`

  Essas são diretrizes para ajudar a dimensionar o cluster. No entanto, você também deve monitorar o desempenho de suas tarefas do Spark para encontrar as melhores configurações para suas cargas de trabalho.

Executando a compactação com o Amazon Athena

O Athena oferece uma implementação do utilitário de compactação do Iceberg como um recurso gerenciado por meio da declaração OPTIMIZE. Você pode usar essa instrução para executar a compactação sem precisar avaliar a infraestrutura.

Essa instrução agrupa arquivos pequenos em arquivos maiores usando o algoritmo de empacotamento de compartimentos e mescla arquivos excluídos com arquivos de dados existentes. Para agrupar os dados usando classificação hierárquica ou classificação de ordem z, use o Spark no Amazon EMR ou. AWS Glue

Você pode alterar o comportamento padrão da `OPTIMIZE` instrução na criação da tabela passando as propriedades da tabela na `CREATE TABLE` instrução ou após a criação da tabela usando a `ALTER TABLE` instrução. Para valores padrão, consulte a documentação do Athena.

Recomendações para executar a compactação

Caso de uso	Recomendação
Executando a compactação da embalagem de lixeiras com base em um cronograma	• Use a `OPTIMIZE` declaração no Athena se você não souber quantos arquivos pequenos sua tabela contém. O modelo de preços do Athena é baseado nos dados digitalizados, portanto, se não houver arquivos a serem compactados, não haverá custo associado a essas operações. Para evitar o tempo limite

Caso de uso	Recomendação
	nas tabelas do Athena, execute `OPTIMIZE` com base. per-table-partition • Use o Amazon EMR ou AWS Glue com escalabilidade dinâmica quando você espera que grandes volumes de arquivos pequenos sejam compactados.
Executando a compactação de embalagens de lixo com base em eventos	• Use o Amazon EMR ou AWS Glue com escalabilidade dinâmica quando você espera que grandes volumes de arquivos pequenos sejam compactados.
Executando a compactação para classificar dados	• Use o Amazon EMR ou AWS Glue, porque a classificação é uma operação cara e pode precisar transferir dados para o disco.
Executando a compactação para agrupar os dados usando a classificação de ordem z	• Use o Amazon EMR ou AWS Glue, porque a classificação por ordem z é uma operação muito cara e pode precisar transferir dados para o disco.
Executando a compactação em partições que podem ser atualizadas por outros aplicativos devido à chegada tardia de dados	• Use o Amazon EMR ou. AWS Glue Ative a propriedade Iceberg PARTIAL_PROGRESS_ENABLED. Quando você usa essa opção, o Iceberg divide a saída de compactação em vários commits. Se houver uma colisão (ou seja, se o arquivo de dados for atualizado durante a execução da compactação), essa configuração reduz o custo da nova tentativa, limitando-a à confirmação que inclui o arquivo afetado. Caso contrário, talvez seja necessário recompactar todos os arquivos.

Caso de uso	Recomendação
Executando a compactação em partições frias (partições de dados que não recebem mais gravações ativas)	• Use o Amazon EMR ou. AWS Glue No `rewrite_data_files` procedimento, especifique um `where` predicado que exclua partições gravadas ativamente. Essa estratégia evita conflitos de dados entre gravadores e trabalhos de compactação e deixa apenas os conflitos de metadados que o Iceberg pode resolver automaticamente.

Usando cargas de trabalho do Iceberg no Amazon S3

Esta seção discute as propriedades do Iceberg que você pode usar para otimizar a interação do Iceberg com o Amazon S3.

Evite o particionamento a quente (erros HTTP 503)

Alguns aplicativos de data lake executados no Amazon S3 manipulam milhões ou bilhões de objetos e processam petabytes de dados. Isso pode levar a prefixos que recebem um alto volume de tráfego, que normalmente são detectados por meio de erros HTTP 503 (serviço indisponível). Para evitar esse problema, use as seguintes propriedades do Iceberg:

- `write.distribution-mode`Defina `hash` para `range` que o Iceberg grave arquivos grandes, o que resulta em menos solicitações do Amazon S3. Essa é a configuração preferida e deve abordar a maioria dos casos.
- Se você continuar enfrentando erros 503 devido a um grande volume de dados em suas cargas de trabalho, você pode `write.object-storage.enabled` configurá-los no Iceberg. `true` Isso instrui o Iceberg a fazer o hash dos nomes dos objetos e distribuir a carga em vários prefixos aleatórios do Amazon S3.

Para obter mais informações sobre essas propriedades, consulte Write properties na documentação do Iceberg.

Use as operações de manutenção do Iceberg para liberar dados não utilizados

Para gerenciar tabelas do Iceberg, você pode usar a API principal do Iceberg, os clientes do Iceberg (como o Spark) ou serviços gerenciados, como o Amazon Athena. Para excluir arquivos antigos ou não utilizados do Amazon S3, recomendamos que você use somente o Iceberg APIs native para remover instantâneos, removerarquivos de metadados antigos e excluir arquivos órfãos.

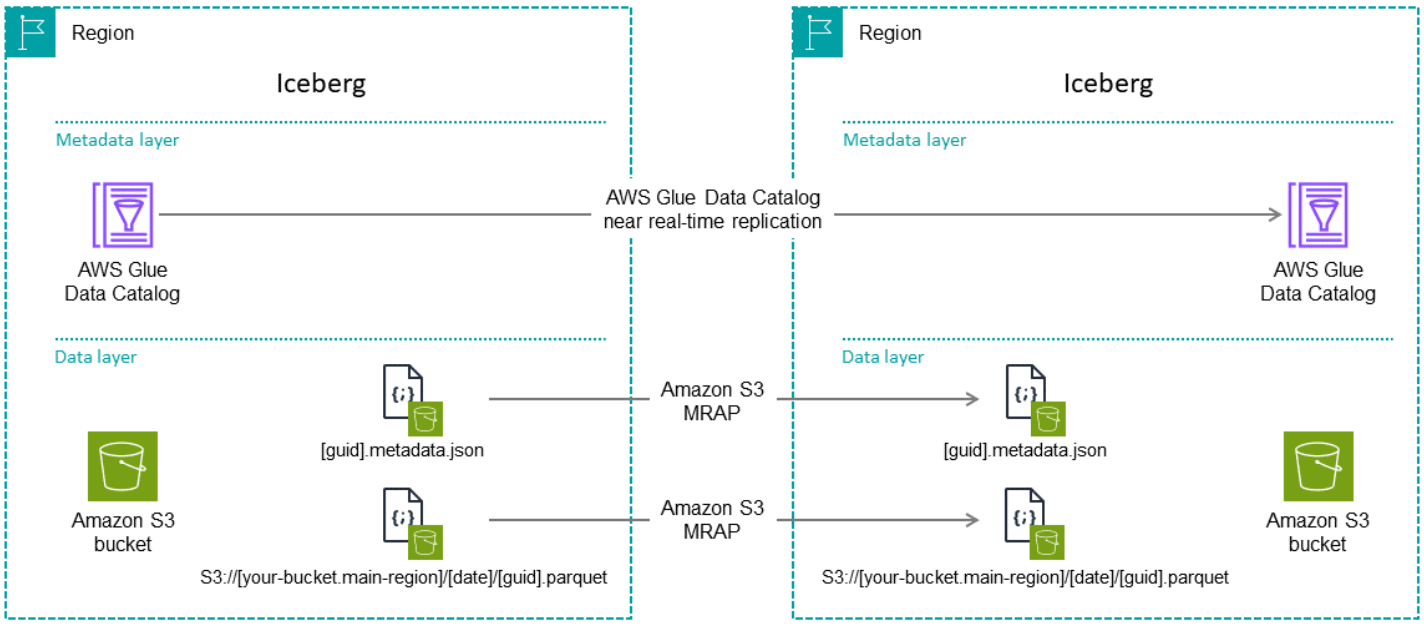
Usar o Amazon S3 APIs por meio do Boto3, o Amazon S3 SDK ou o AWS Command Line Interface (AWS CLI), ou usar qualquer outro método que não seja do Iceberg para sobrescrever ou remover arquivos do Amazon S3 de uma tabela Iceberg leva à corrupção da tabela e a falhas de consulta.

Replique dados em Regiões da AWS

Ao armazenar tabelas do Iceberg no Amazon S3, você pode usar os recursos integrados no Amazon S3, como replicação entre regiões (CRR) e pontos de acesso multirregionais (MRAP), para replicar dados em várias regiões. Regiões da AWS O MRAP fornece um endpoint global para que os aplicativos acessem buckets S3 localizados em vários. Regiões da AWS O Iceberg não oferece suporte a caminhos relativos, mas você pode usar o MRAP para realizar operações do Amazon S3 mapeando buckets para pontos de acesso. O MRAP também se integra perfeitamente ao processo de replicação entre regiões do Amazon S3, o que introduz um atraso de até 15 minutos. Você precisa replicar os arquivos de dados e metadados.

> **Important**
>
> Atualmente, a integração do Iceberg com o MRAP funciona somente com o Apache Spark. Se você precisar fazer o failover para o secundário Região da AWS, planeje redirecionar as consultas do usuário para um ambiente Spark SQL (como o Amazon EMR) na região de failover.

Os recursos CRR e MRAP ajudam você a criar uma solução de replicação entre regiões para tabelas Iceberg, conforme ilustrado no diagrama a seguir.



Para configurar essa arquitetura de replicação entre regiões:

1. Crie tabelas usando a localização do MRAP. Isso garante que os arquivos de metadados do Iceberg apontem para o local do MRAP em vez do local físico do bucket.
2. Replique arquivos Iceberg usando o Amazon S3 MRAP. O MRAP oferece suporte à replicação de dados com um contrato de nível de serviço (SLA) de 15 minutos. O Iceberg impede que as operações de leitura introduzam inconsistências durante a replicação.
3. Disponibilize as tabelas AWS Glue Data Catalog na região secundária. Você pode escolher entre duas opções:
   - Configure um pipeline para replicar os metadados da tabela Iceberg usando a replicação. AWS Glue Data Catalog Esse utilitário está disponível no repositório de replicação do GitHub Glue Catalog e do Lake Formation Permissions. Esse mecanismo controlado por eventos replica tabelas na região de destino com base nos registros de eventos.
   - Registre as tabelas na região secundária quando precisar fazer o failover. Para essa opção, você pode usar o utilitário anterior ou o procedimento register_table do Iceberg e apontá-lo para o arquivo mais recente. `metadata.json`

Monitorando cargas de trabalho do Apache Iceberg

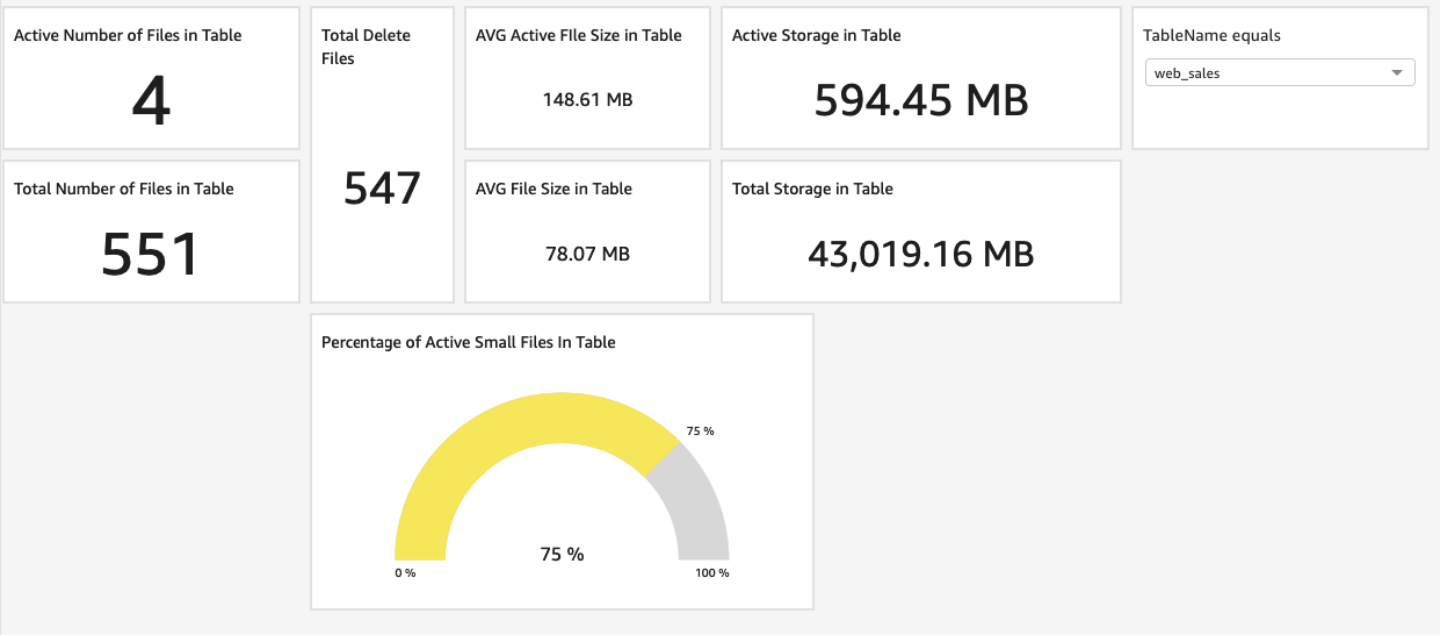
Para monitorar as cargas de trabalho do Iceberg, você tem duas opções: analisar tabelas de metadados ou usar relatórios de métricas. Os repórteres de métricas foram introduzidos na versão 1.2 do Iceberg e estão disponíveis somente para catálogos REST e JDBC.

Se você estiver usando AWS Glue Data Catalog, poderá obter informações sobre a integridade de suas tabelas do Iceberg configurando o monitoramento sobre as tabelas de metadados que o Iceberg expõe.

O monitoramento é crucial para o gerenciamento de desempenho e a solução de problemas. Por exemplo, quando uma partição em uma tabela do Iceberg atinge uma certa porcentagem de arquivos pequenos, sua carga de trabalho pode iniciar um trabalho de compactação para consolidar os arquivos em arquivos maiores. Isso evita que as consultas fiquem mais lentas além de um nível aceitável.

Monitoramento em nível de tabela

A tela a seguir mostra um painel de monitoramento de tabelas criado no Amazon Quick Sight. Esse painel consulta as tabelas de metadados do Iceberg usando o Spark SQL e captura métricas detalhadas, como o número de arquivos ativos e o armazenamento total. Essas informações são então armazenadas em AWS Glue tabelas para fins operacionais. Por fim, um painel do Quick Sight, conforme mostrado na ilustração a seguir, é criado usando o Amazon Athena. Essas informações ajudam você a identificar e resolver problemas específicos em seus sistemas.



O painel de exemplo do Quick Sight coleta os seguintes indicadores-chave de desempenho (KPIs) para uma tabela Iceberg:

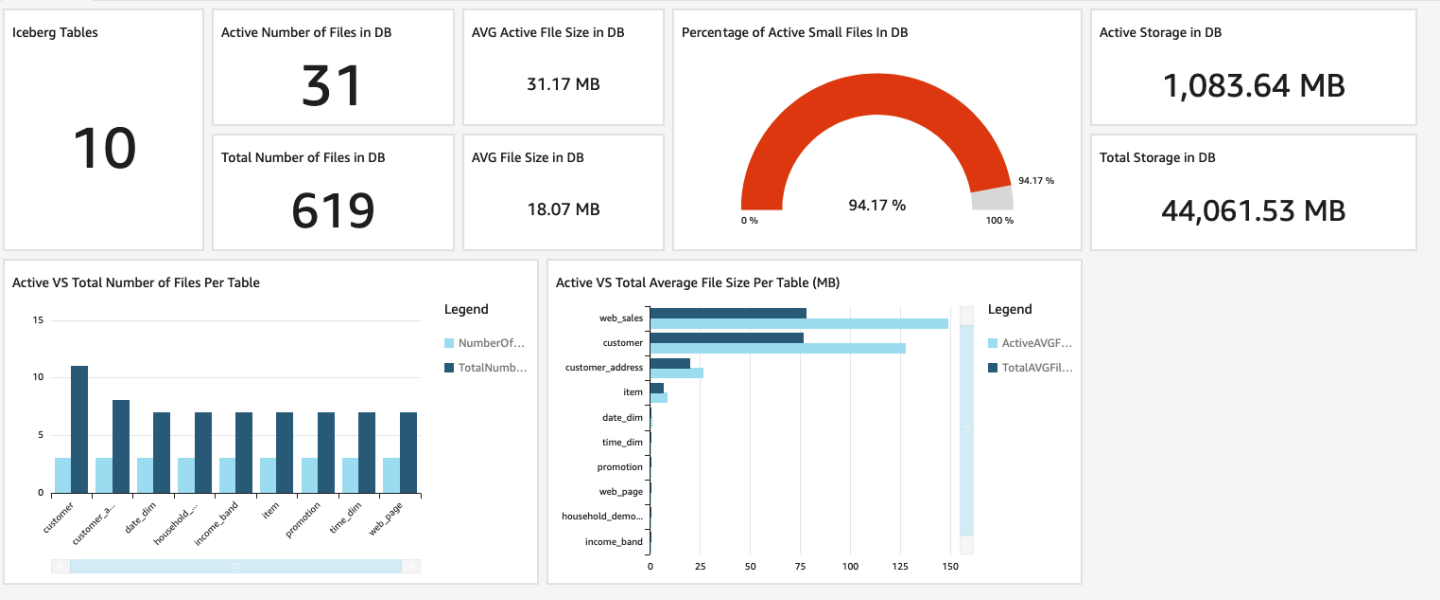
KPI	Descrição	Consulta
Número de arquivos	O número de arquivos na tabela Iceberg (para todos os instantâneos)	`select count(*) from <catalog.database.table_name>.all_files`
Número de arquivos ativos	O número de arquivos ativos no último instantâneo da tabela Iceberg	`select count(*) from <catalog.database.table_name>.files`
Tamanho médio do arquivo	O tamanho médio do arquivo, em megabytes, para todos os arquivos na tabela Iceberg	`select avg(file_size_in_bytes)/1000000 from <catalog.database.table_name>.all_files`
Tamanho médio do arquivo ativo	O tamanho médio do arquivo, em megabytes, dos arquivos ativos na tabela Iceberg	`select avg(file_size_in_bytes)/1000000`

KPI	Descrição	Consulta
		`from <catalog.database.table_name>.files`
Porcentagem de arquivos pequenos	A porcentagem de arquivos ativos menores que 100 MB	`select cast(sum( case when file_size_in_bytes < 100000000 then 1 else 0 end)*100/ count(*) as decimal(10,2)) from <catalog.database.table_name>.files`
Tamanho total do armazenamento	O tamanho total de todos os arquivos na tabela, excluindo arquivos órfãos e versões de objetos do Amazon S3 (se habilitado)	`select sum(file_size_in_bytes)/1000000 from <catalog.database.table_name>.all_files`
Tamanho total do armazenamento ativo	O tamanho total de todos os arquivos nos instantâneos atuais de uma determinada tabela	`select sum(file_size_in_bytes)/1000000 from <catalog.database.table_name>.files`

Para obter mais informações sobre a criação de painéis, consulte a documentação do Quick Sight.

Monitoramento em nível de banco de dados

O exemplo a seguir mostra um painel de monitoramento que foi criado no Quick Sight para fornecer uma visão geral do nível do banco de dados KPIs para uma coleção de tabelas Iceberg.



Esse painel coleta o seguinte: KPIs

KPI	Descrição	Consulta
Número de arquivos	O número de arquivos no banco de dados do Iceberg (para todos os instantâneos)	Esse painel usa as consultas em nível de tabela fornecidas na seção anterior e consolida os resultados.
Número de arquivos ativos	O número de arquivos ativos no banco de dados do Iceberg (com base nos últimos instantâneos das tabelas do Iceberg)	
Tamanho médio do arquivo	O tamanho médio do arquivo, em megabytes, de todos os arquivos no banco de dados Iceberg	
Tamanho médio do arquivo ativo	O tamanho médio do arquivo, em megabytes, de todos os arquivos ativos no banco de dados Iceberg	

KPI	Descrição	Consulta
Porcentagem de arquivos pequenos	A porcentagem de arquivos ativos menores que 100 MB no banco de dados do Iceberg	
Tamanho total do armazenamento	O tamanho total de todos os arquivos no banco de dados, excluindo arquivos órfãos e versões de objetos do Amazon S3 (se habilitado)	
Tamanho total do armazenamento ativo	O tamanho total de todos os arquivos nos instantâneos atuais de todas as tabelas no banco de dados	

Manutenção preventiva

Ao configurar os recursos de monitoramento discutidos nas seções anteriores, você pode abordar a manutenção da mesa de um ângulo preventivo em vez de reativo. Por exemplo, você pode usar as métricas em nível de tabela e em nível de banco de dados para programar ações como as seguintes:

- Use a compactação de compartimentos para agrupar arquivos pequenos quando uma tabela atingir N arquivos pequenos.
- Use a compactação de compartimentos para mesclar arquivos excluídos quando uma tabela atingir N arquivos excluídos em uma determinada partição.
- Remova arquivos pequenos que já foram compactados removendo instantâneos quando o armazenamento total for X vezes maior que o armazenamento ativo.

Governança e controle de acesso para o Apache Iceberg em AWS

O Apache Iceberg se integra AWS Lake Formation para simplificar a governança de dados. Essa integração permite que os administradores do data lake atribuam permissões de acesso em nível de célula às tabelas do Iceberg. Para ver um exemplo de consulta de tabelas Iceberg usando o Amazon Athena AWS Lake Formation e, consulte a postagem AWS do blog Interaja com tabelas Iceberg do Apache usando o Amazon Athena e permissões refinadas entre contas usando. AWS Lake Formation

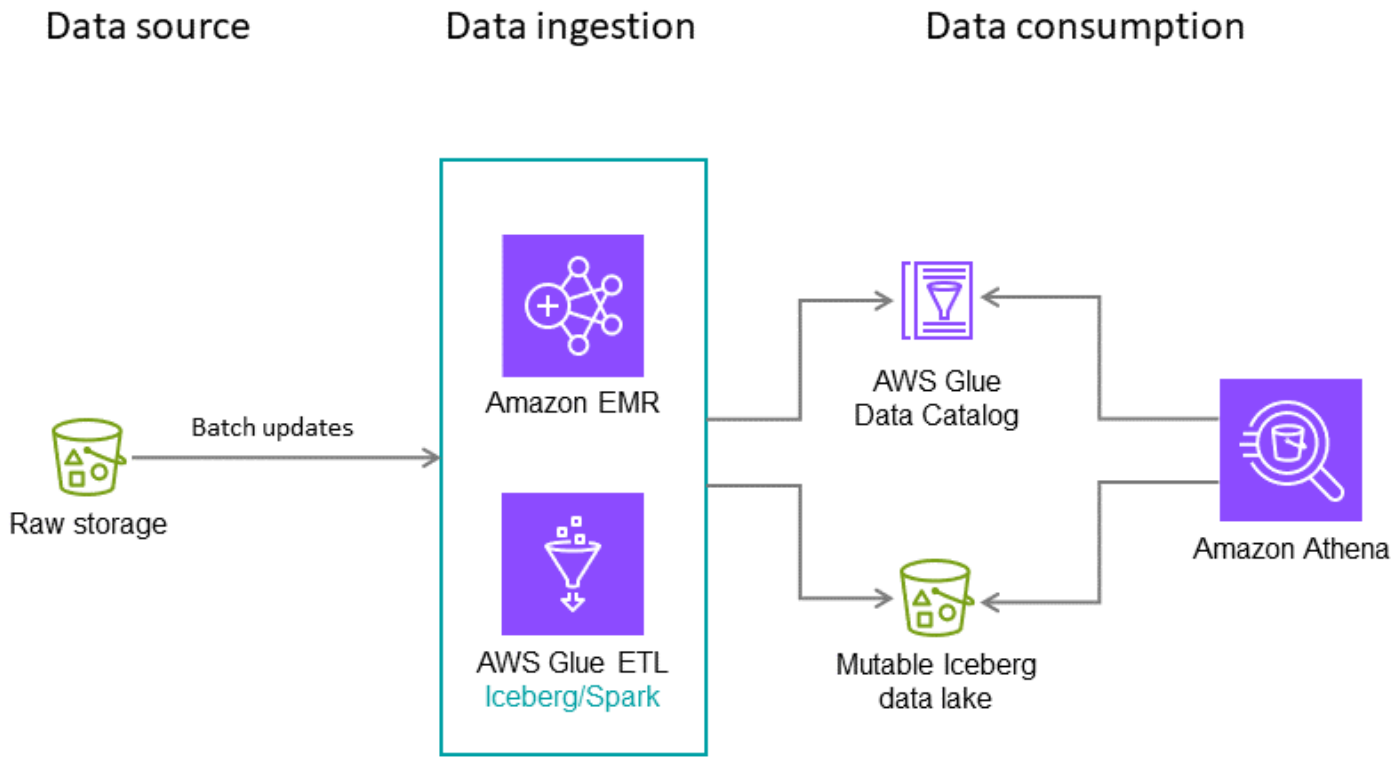
Arquiteturas de referência para o Apache Iceberg em AWS

Esta seção fornece exemplos de como aplicar as melhores práticas em diferentes casos de uso, como ingestão em lote e um data lake que combina ingestão de dados em lote e streaming.

Ingestão noturna em lote

Para esse caso de uso hipotético, digamos que sua mesa Iceberg consuma transações com cartão de crédito todas as noites. Cada lote contém somente atualizações incrementais, que devem ser mescladas na tabela de destino. Várias vezes por ano, dados históricos completos são recebidos. Para esse cenário, recomendamos a arquitetura e as configurações a seguir.

Nota: Este é apenas um exemplo. A configuração ideal depende de seus dados e requisitos.



Recomendações:

- Tamanho do arquivo: 128 MB, porque as tarefas do Apache Spark processam dados em blocos de 128 MB.
- Tipo de escrita: copy-on-write. Conforme detalhado anteriormente neste guia, essa abordagem ajuda a garantir que os dados sejam gravados de forma otimizada para leitura.

- Variáveis de partição:year/month/day. Em nosso caso de uso hipotético, consultamos dados recentes com mais frequência, embora ocasionalmente executemos varreduras completas de dados nos últimos dois anos. O objetivo do particionamento é conduzir operações de leitura rápida com base nos requisitos do caso de uso.
- Ordem de classificação: carimbo de data/hora
- Catálogo de dados: AWS Glue Data Catalog

Lago de dados que combina ingestão em lote e quase em tempo real

Você pode provisionar um data lake no Amazon S3 que compartilha dados em lote e de streaming entre contas e regiões. Para obter um diagrama de arquitetura e detalhes, consulte a postagem do AWS blog Crie um data lake transacional usando o Apache Iceberg e compartilhamentos de dados entre contas usando o Amazon Athena. AWS Glue AWS Lake Formation

Recursos

- Usando a estrutura Iceberg em AWS Glue (AWS Glue documentação)
- Iceberg (documentação do Amazon EMR)
- Usando tabelas do Apache Iceberg (documentação do Amazon Athena)
- Documentação do Amazon S3
- Documentação rápida
- Replicação do Glue Catalog e do Lake Formation Permissions (GitHub repositório)
- Documentação do Apache Iceberg
- Documentação do Apache Spark

Colaboradores

As seguintes pessoas escreveram, foram AWS coautoras e revisaram este guia.

Colaboradores

- Stefano Sandona, arquiteto de soluções, Big Data
- Imtiaz (Taz) Sayed, arquiteto de soluções e líder técnico de análise
- Shana Schipers, arquiteta de soluções, Big Data
- Prashant Singh, engenheiro de desenvolvimento de software, Amazon EMR
- Arun A K, arquiteto de soluções, Big Data e ETL
- Francisco Morillo, arquiteto de soluções, streaming
- Suthan Phillips, arquiteto de análise, Amazon EMR
- Sercan Karaoglu, arquiteto de soluções
- Yonatan Dolan, especialista em análise
- Guy Bachar, arquiteto de soluções
- Sofia Zilberman, arquiteta de soluções de streaming
- Dan Stair, arquiteto especializado em soluções
- Sakti Mishra, arquiteta de soluções
- Ron Ortloff, gerente de produto principal, Amazon S3
- David Zhang, arquiteto de soluções, análise

Revisores

- Rick Sears, gerente geral, Amazon EMR
- Linda OConnor, especialista em Amazon EMR
- Ian Meyers, diretor do Amazon EMR
- Vinita Ananth, diretora de gerenciamento de produtos do Amazon EMR
- Jason Berkowitz, gerente de produto, AWS Lake Formation
- Mahesh Mishra, gerente de produto, Amazon Redshift
- Vladimir Zlatkin, gerente de arquitetura de soluções, Big Data
- Karthik Prabhakar, arquiteto de análise, Amazon EMR

- Vijay Jain, gerente de produto
- Anupriti Warade, gerente de produto, Amazon S3
- Ajit Tandale, arquiteto de soluções, dados
- Gwen Chen, gerente de marketing de produto
- Noritaka Sekiyama, arquiteta principal, Big Data

Histórico do documento

A tabela a seguir descreve alterações significativas feitas neste guia. Se desejar receber notificações sobre futuras atualizações, inscreva-se em um feed RSS.

Alteração	Descrição	Data
Nova seção	Foram adicionadas informaçõ es sobre como trabalhar com a versão 3 da especificação de formato de tabela Iceberg.	26 de novembro de 2025
Correção	Informações corrigidas sobre `gc.enabled` as configura ções na seção de procedime nto de captura instantânea.	24 de outubro de 2025
Adições	Foram adicionadas novas seções sobre como trabalhar com tabelas do Iceberg usando o Trino e PyIceberg, e expandiu a seção sobre como migrar tabelas para o Iceberg.	12 de agosto de 2025
Atualizações	Informações aprimoradas e esclarecidas em todo o guia para refletir as versões mais recentes do AWS Glue Amazon EMR e do Apache Iceberg.	14 de julho de 2025
Adições	Foi adicionada uma nova seção sobre como trabalhar com tabelas Iceberg usando o Amazon Data Firehose.	20 de fevereiro de 2025
Publicação inicial	—	30 de abril de 2024

AWS Glossário de orientação prescritiva

A seguir estão os termos comumente usados em estratégias, guias e padrões fornecidos pela Orientação AWS Prescritiva. Para sugerir entradas, use o link Fornecer feedback no final do glossário.

Números

7 Rs

Sete estratégias comuns de migração para mover aplicações para a nuvem. Essas estratégias baseiam-se nos 5 Rs identificados pela Gartner em 2011 e consistem em:

- Refatorar/rearquitetar: mova uma aplicação e modifique sua arquitetura aproveitando ao máximo os recursos nativos de nuvem para melhorar a agilidade, a performance e a escalabilidade. Isso normalmente envolve a portabilidade do sistema operacional e do banco de dados. Exemplo: migrar seu banco de dados Oracle on-premises para o Amazon Aurora Edição Compatível com PostgreSQL.
- Redefinir a plataforma (mover e redefinir [mover e redefinir (lift-and-reshape]): mova uma aplicação para a nuvem e introduza algum nível de otimização a fim de aproveitar os recursos da nuvem. Exemplo: migrar seu banco de dados Oracle on-premises para o Amazon Relational Database Service (Amazon RDS) para Oracle na Nuvem AWS.
- Recomprar (drop and shop): mude para um produto diferente, normalmente migrando de uma licença tradicional para um modelo SaaS. Exemplo: migrar seu sistema de gerenciamento de relacionamento com o cliente (CRM) para o Salesforce.com.
- Redefinir a hospedagem (mover sem alterações [lift-and-shift])mover uma aplicação para a nuvem sem fazer nenhuma alteração a fim de aproveitar os recursos da nuvem. Exemplo: migrar seu banco de dados Oracle on-premises para o Oracle em uma instância do EC2 na Nuvem AWS.
- Realocar (mover o hipervisor sem alterações [hypervisor-level lift-and-shift]): mover a infraestrutura para a nuvem sem comprar novo hardware, reescrever aplicações ou modificar suas operações existentes. Você migra servidores de uma plataforma on-premises para um serviço de nuvem para a mesma plataforma. Exemplo: migrar um Microsoft Hyper-V aplicativo para o. AWS
- Reter (revisitar): mantenha as aplicações em seu ambiente de origem. Isso pode incluir aplicações que exigem grande refatoração, e você deseja adiar esse trabalho para um

momento posterior, e aplicações antigas que você deseja manter porque não há justificativa comercial para migrá-las.

- Retirar: desative ou remova aplicações que não são mais necessárias em seu ambiente de origem.

A

ABAC

Consulte controle de acesso baseado em atributo.

serviços abstraídos

Veja serviços gerenciados.

ACID

Veja atomicidade, consistência, isolamento, durabilidade.

migração ativa-ativa

Um método de migração de banco de dados no qual os bancos de dados de origem e de destino são mantidos em sincronia (por meio de uma ferramenta de replicação bidirecional ou operações de gravação dupla), e ambos os bancos de dados lidam com transações de aplicações conectadas durante a migração. Esse método oferece suporte à migração em lotes pequenos e controlados, em vez de exigir uma substituição única. É mais flexível, mas exige mais trabalho do que a migração ativa-passiva.

migração ativa-passiva

Um método de migração de banco de dados em que os bancos de dados de origem e de destino são mantidos em sincronia, mas somente o banco de dados de origem manipula as transações das aplicações conectadas, enquanto os dados são replicados no banco de dados de destino. O banco de dados de destino não aceita nenhuma transação durante a migração.

AGGREGATE FUNCTION

Uma função SQL que opera em um grupo de linhas e calcula um único valor de retorno para o grupo. Exemplos de funções agregadas incluem `SUM` e `MAX`.

AI

Veja inteligência artificial.

AIOps

Veja operações de inteligência artificial.

anonimização

O processo de excluir permanentemente informações pessoais em um conjunto de dados. A anonimização pode ajudar a proteger a privacidade pessoal. Dados anônimos não são mais considerados dados pessoais.

antipadrões

Uma solução frequentemente usada para um problema recorrente em que a solução é contraproducente, ineficaz ou menos eficaz do que uma alternativa.

controle de aplicações

Uma abordagem de segurança que permite o uso somente de aplicações aprovadas para ajudar a proteger um sistema contra malware.

portfólio de aplicações

Uma coleção de informações detalhadas sobre cada aplicação usada por uma organização, incluindo o custo para criar e manter a aplicação e seu valor comercial. Essas informações são fundamentais para o processo de descoberta e análise de portfólio e ajudam a identificar e priorizar as aplicações a serem migradas, modernizadas e otimizadas.

inteligência artificial (IA)

O campo da ciência da computação que se dedica ao uso de tecnologias de computação para desempenhar funções cognitivas normalmente associadas aos humanos, como aprender, resolver problemas e reconhecer padrões. Para obter mais informações, consulte O que é inteligência artificial?

operações de inteligência artificial (AIOps)

O processo de usar técnicas de machine learning para resolver problemas operacionais, reduzir incidentes operacionais e intervenção humana e aumentar a qualidade do serviço. Para obter mais informações sobre como AIOps é usado na estratégia de AWS migração, consulte o guia de integração de operações.

criptografia assimétrica

Um algoritmo de criptografia que usa um par de chaves, uma chave pública para criptografia e uma chave privada para descriptografia. É possível compartilhar a chave pública porque ela não é usada na descriptografia, mas o acesso à chave privada deve ser altamente restrito.

atomicidade, consistência, isolamento, durabilidade (ACID)

Um conjunto de propriedades de software que garantem a validade dos dados e a confiabilidade operacional de um banco de dados, mesmo no caso de erros, falhas de energia ou outros problemas.

controle de acesso por atributo (ABAC)

A prática de criar permissões minuciosas com base nos atributos do usuário, como departamento, cargo e nome da equipe. Para obter mais informações, consulte ABAC AWS na documentação AWS Identity and Access Management (IAM).

fonte de dados autorizada

Um local onde você armazena a versão principal dos dados, que é considerada a fonte de informações mais confiável. Você pode copiar dados da fonte de dados autorizada para outros locais com o objetivo de processar ou modificar os dados, como anonimizá-los, redigi-los ou pseudonimizá-los.

Zona de disponibilidade

Um local distinto dentro de um Região da AWS que está isolado de falhas em outras zonas de disponibilidade e fornece conectividade de rede barata e de baixa latência a outras zonas de disponibilidade na mesma região.

AWS Estrutura de adoção da nuvem (AWS CAF)

Uma estrutura de diretrizes e melhores práticas AWS para ajudar as organizações a desenvolver um plano eficiente e eficaz para migrar com sucesso para a nuvem. AWS O CAF organiza a orientação em seis áreas de foco chamadas perspectivas: negócios, pessoas, governança, plataforma, segurança e operações. As perspectivas de negócios, pessoas e governança têm como foco habilidades e processos de negócios; as perspectivas de plataforma, segurança e operações concentram-se em habilidades e processos técnicos. Por exemplo, a perspectiva das pessoas tem como alvo as partes interessadas que lidam com recursos humanos (RH), funções de pessoal e gerenciamento de pessoal. Nessa perspectiva, o AWS CAF fornece orientação para desenvolvimento, treinamento e comunicação de pessoas para ajudar a preparar a organização para a adoção bem-sucedida da nuvem. Para obter mais informações, consulte o site da AWS CAF e o whitepaper da AWS CAF.

AWS Estrutura de qualificação da carga de trabalho (AWS WQF)

Uma ferramenta que avalia as cargas de trabalho de migração do banco de dados, recomenda estratégias de migração e fornece estimativas de trabalho. AWS O WQF está incluído com AWS

Schema Conversion Tool ()AWS SCT. Ela analisa esquemas de banco de dados e objetos de código, código de aplicações, dependências e características de performance, além de fornecer relatórios de avaliação.

B

bot malicioso

Um bot destinado a causar disrupção ou danos a indivíduos ou organizações.

BCP

Veja planejamento de continuidade de negócios

gráfico de comportamento

Uma visualização unificada e interativa do comportamento e das interações de recursos ao longo do tempo. É possível usar um gráfico de comportamento com o Amazon Detective para examinar tentativas de login malsucedidas, chamadas de API suspeitas e ações similares. Para obter mais informações, consulte Dados em um gráfico de comportamento na documentação do Detective.

sistema big-endian

Um sistema que armazena o byte mais significativo antes. Veja também endianness.

classificação binária

Um processo que prevê um resultado binário (uma de duas classes possíveis). Por exemplo, seu modelo de ML pode precisar prever problemas como “Este e-mail é ou não é spam?” ou “Este produto é um livro ou um carro?”

filtro de bloom

Uma estrutura de dados probabilística e eficiente em termos de memória que é usada para testar se um elemento é membro de um conjunto.

blue/green deployment (implantação azul/verde)

Uma estratégia de implantação em que você cria dois ambientes separados, mas idênticos. Você executa a versão atual da aplicação em um ambiente (azul) e a nova versão da aplicação no outro ambiente (verde). Essa estratégia ajuda você a reverter rapidamente com o mínimo de impacto.

bot

Uma aplicação de software que executa tarefas automatizadas na internet e simula a atividade ou interação humana. Alguns bots são úteis ou benéficos, como crawlers da web que indexam informações na internet. Outros bots, conhecidos como bots maliciosos, têm como objetivo causar disrupção ou danos a indivíduos ou organizações.

botnet

Redes de bots infectadas por malware e sob o controle de uma única parte, conhecidas como bot herder ou operador de bots. Os botnets são o mecanismo mais conhecido para escalar bots e seu impacto.

ramo

Uma área contida de um repositório de código. A primeira ramificação criada em um repositório é a ramificação principal. Você pode criar uma nova ramificação a partir de uma ramificação existente e, em seguida, desenvolver recursos ou corrigir bugs na nova ramificação. Uma ramificação que você cria para gerar um recurso é comumente chamada de ramificação de recurso. Quando o recurso estiver pronto para lançamento, você mesclará a ramificação do recurso de volta com a ramificação principal. Para obter mais informações, consulte Sobre filiais (GitHub documentação).

Acesso de emergência

Em circunstâncias excepcionais e por meio de um processo aprovado, um meio rápido para um usuário obter acesso a um Conta da AWS que ele normalmente não tem permissão para acessar. Para obter mais informações, consulte o indicador Implement break-glass procedures nas orientações do AWS Well-Architected.

estratégia brownfield

A infraestrutura existente em seu ambiente. Ao adotar uma estratégia brownfield para uma arquitetura de sistema, você desenvolve a arquitetura de acordo com as restrições dos sistemas e da infraestrutura atuais. Se estiver expandindo a infraestrutura existente, poderá combinar as estratégias brownfield e greenfield.

cache do buffer

A área da memória em que os dados acessados com mais frequência são armazenados.

capacidade de negócios

O que uma empresa faz para gerar valor (por exemplo, vendas, atendimento ao cliente ou marketing). As arquiteturas de microsserviços e as decisões de desenvolvimento podem

ser orientadas por recursos de negócios. Para obter mais informações, consulte a seção Organizados de acordo com as capacidades de negócios do whitepaper Executar microsserviços conteinerizados na AWS.

planejamento de continuidade de negócios (BCP)

Um plano que aborda o impacto potencial de um evento disruptivo, como uma migração em grande escala, nas operações e permite que uma empresa retome as operações rapidamente.

C

CAF

Veja AWS Cloud Adoption Framework.

implantação canário

O lançamento lento e incremental de uma versão para usuários finais. Quando estiver confiante, você implanta a nova versão e substitui a versão atual por completo.

CCoE

Veja Centro de Excelência da Nuvem.

CDC

Veja captura de dados de alteração.

captura de dados de alterações (CDC)

O processo de rastrear alterações em uma fonte de dados, como uma tabela de banco de dados, e registrar metadados sobre a alteração. É possível usar o CDC para várias finalidades, como auditar ou replicar alterações em um sistema de destino para manter a sincronização.

engenharia do caos

Introduzir intencionalmente falhas ou eventos disruptivos para testar a resiliência de um sistema. Você pode usar AWS Fault Injection Service (AWS FIS) para realizar experimentos que estressam suas AWS cargas de trabalho e avaliar sua resposta.

CI/CD

Veja integração e entrega contínuas.

classificação

Um processo de categorização que ajuda a gerar previsões. Os modelos de ML para problemas de classificação predizem um valor discreto. Os valores discretos são sempre diferentes uns dos outros. Por exemplo, um modelo pode precisar avaliar se há ou não um carro em uma imagem.

criptografia no lado do cliente

Criptografia de dados localmente, antes que o alvo os AWS service (Serviço da AWS) receba.

Centro de excelência em nuvem (CCoE)

Uma equipe multidisciplinar que impulsiona os esforços de adoção da nuvem em toda a organização, incluindo o desenvolvimento de práticas recomendas de nuvem, a mobilização de recursos, o estabelecimento de cronogramas de migração e a liderança da organização em transformações em grande escala. Para obter mais informações, consulte as publicações CCo E no blog de estratégia Nuvem AWS corporativa.

computação em nuvem

A tecnologia de nuvem normalmente usada para armazenamento de dados remoto e gerenciamento de dispositivos de IoT. A computação em nuvem é normalmente conectada à tecnologia de computação de borda.

modelo operacional em nuvem

Em uma organização de TI, o modelo operacional usado para criar, amadurecer e otimizar um ou mais ambientes de nuvem. Para obter mais informações, consulte Criar seu modelo operacional de nuvem.

estágios de adoção da nuvem

As quatro fases pelas quais as organizações normalmente passam ao migrar para a Nuvem AWS:

- Projeto: executar alguns projetos relacionados à nuvem para fins de prova de conceito e aprendizado
- Fundação — Fazer investimentos fundamentais para escalar sua adoção da nuvem (por exemplo, criar uma landing zone, definir um CCo E, estabelecer um modelo de operações)
- Migração: migrar aplicações individuais
- Reinvenção: otimizar produtos e serviços e inovar na nuvem

Esses estágios foram definidos por Stephen Orban na postagem do blog The Journey Toward Cloud-First & the Stages of Adoption no blog de estratégia Nuvem AWS empresarial. Para obter

informações sobre como eles se relacionam com a estratégia de AWS migração, consulte o guia de preparação para migração.

CMDB

Veja banco de dados de gerenciamento de configuração.

repositório de código

Um local onde o código-fonte e outros ativos, como documentação, amostras e scripts, são armazenados e atualizados por meio de processos de controle de versão. Os repositórios de nuvem comuns incluem o GitHub ou o Bitbucket Cloud. Cada versão do código é chamada de ramificação. Em uma estrutura de microsserviços, cada repositório é dedicado a uma única peça de funcionalidade. Um único pipeline de CI/CD pode usar vários repositórios.

cache frio

Um cache de buffer que está vazio, não está bem preenchido ou contém dados obsoletos ou irrelevantes. Isso afeta a performance porque a instância do banco de dados deve ler da memória principal ou do disco, um processo que é mais lento do que a leitura do cache do buffer.

dados frios

Dados que raramente são acessados e geralmente são históricos. Ao consultar esse tipo de dados, consultas lentas geralmente são aceitáveis. Mover esses dados para níveis ou classes de armazenamento de baixo desempenho e menos caros pode reduzir os custos.

visão computacional (CV)

Um campo de IA que usa machine learning para analisar e extrair informações de formatos visuais, como vídeos e imagens digitais. Por exemplo, a Amazon SageMaker AI fornece algoritmos de processamento de imagem para CV.

desvio de configuração

Em uma workload, uma alteração de configuração em relação ao estado esperado. Isso pode fazer com que a workload se torne incompatível e, normalmente, é gradual e não intencional.

banco de dados de gerenciamento de configuração (CMDB)

Um repositório que armazena e gerencia informações sobre um banco de dados e seu ambiente de TI, incluindo componentes de hardware e software e suas configurações. Normalmente, os dados de um CMDB são usados no estágio de descoberta e análise do portfólio da migração.

pacote de conformidade

Uma coleção de AWS Config regras e ações de remediação que você pode montar para personalizar suas verificações de conformidade e segurança. Você pode implantar um pacote de conformidade como uma entidade única em uma Conta da AWS região ou em uma organização usando um modelo YAML. Para obter mais informações, consulte Pacotes de conformidade na documentação. AWS Config

integração contínua e entrega contínua (CI/CD)

O processo de automatizar os estágios de origem, criação, teste, preparação e produção do processo de lançamento do software. CI/CD é comumente descrito como um pipeline. CI/CD pode ajudá-lo a automatizar processos, melhorar a produtividade, melhorar a qualidade do código e entregar com mais rapidez. Para obter mais informações, consulte Benefícios da entrega contínua. CD também pode significar implantação contínua. Para obter mais informações, consulte Entrega contínua versus implantação contínua.

CV

Veja visão computacional.

D

dados em repouso

Dados estacionários em sua rede, por exemplo, dados que estão em um armazenamento.

classificação de dados

Um processo para identificar e categorizar os dados em sua rede com base em criticalidade e confidencialidade. É um componente crítico de qualquer estratégia de gerenciamento de riscos de segurança cibernética, pois ajuda a determinar os controles adequados de proteção e retenção para os dados. A classificação de dados é um componente do pilar de segurança no AWS Well-Architected Framework. Para obter mais informações, consulte Classificação de dados.

desvio de dados

Uma variação significativa entre os dados de produção e os dados usados para treinar um modelo de ML ou uma alteração significativa nos dados de entrada ao longo do tempo. O desvio de dados pode reduzir a qualidade geral, a precisão e a imparcialidade das previsões do modelo de ML.

dados em trânsito

Dados que estão se movendo ativamente pela sua rede, como entre os recursos da rede.

data mesh

Um framework de arquitetura que fornece propriedade de dados distribuída e descentralizada com gerenciamento e governança centralizados.

minimização de dados

O princípio de coletar e processar apenas os dados estritamente necessários. Praticar a minimização de dados no Nuvem AWS pode reduzir os riscos de privacidade, os custos e a pegada de carbono de sua análise.

perímetro de dados

Um conjunto de proteções preventivas em seu AWS ambiente que ajudam a garantir que somente identidades confiáveis acessem recursos confiáveis das redes esperadas. Para obter mais informações, consulte Construindo um perímetro de dados em. AWS

pré-processamento de dados

A transformação de dados brutos em um formato que seja facilmente analisado por seu modelo de ML. O pré-processamento de dados pode significar a remoção de determinadas colunas ou linhas e o tratamento de valores ausentes, inconsistentes ou duplicados.

proveniência dos dados

O processo de rastrear a origem e o histórico dos dados ao longo de seu ciclo de vida, por exemplo, como os dados foram gerados, transmitidos e armazenados.

titular dos dados

Um indivíduo cujos dados estão sendo coletados e processados.

data warehouse

Um sistema de gerenciamento de dados compatível com business intelligence, como analytics. Os data warehouses geralmente contêm grandes quantidades de dados históricos e geralmente são usados para consultas e análises.

linguagem de definição de dados (DDL)

Instruções ou comandos para criar ou modificar a estrutura de tabelas e objetos em um banco de dados.

linguagem de manipulação de dados (DML)

Instruções ou comandos para modificar (inserir, atualizar e excluir) informações em um banco de dados.

DDL

Veja linguagem de definição de banco de dados.

deep ensemble

A combinação de vários modelos de aprendizado profundo para gerar previsões. Os deep ensembles podem ser usados para produzir uma previsão mais precisa ou para estimar a incerteza nas previsões.

Aprendizado profundo

Um subcampo do ML que usa várias camadas de redes neurais artificiais para identificar o mapeamento entre os dados de entrada e as variáveis-alvo de interesse.

defense-in-depth

Uma abordagem de segurança da informação na qual uma série de mecanismos e controles de segurança são cuidadosamente distribuídos por toda a rede de computadores para proteger a confidencialidade, a integridade e a disponibilidade da rede e dos dados nela contidos. Ao adotar essa estratégia AWS, você adiciona vários controles em diferentes camadas da AWS Organizations estrutura para ajudar a proteger os recursos. Por exemplo, uma defense-in-depth abordagem pode combinar autenticação multifatorial, segmentação de rede e criptografia.

administrador delegado

Em AWS Organizations, um serviço compatível pode registrar uma conta de AWS membro para administrar as contas da organização e gerenciar as permissões desse serviço. Essa conta é chamada de administrador delegado para esse serviço Para obter mais informações e uma lista de serviços compatíveis, consulte Serviços que funcionam com o AWS Organizations na documentação do AWS Organizations .

implantação

O processo de criar uma aplicação, novos recursos ou correções de código disponíveis no ambiente de destino. A implantação envolve a implementação de mudanças em uma base de código e, em seguida, a criação e execução dessa base de código nos ambientes da aplicação

ambiente de desenvolvimento

Veja ambiente.

controle detectivo

Um controle de segurança projetado para detectar, registrar e alertar após a ocorrência de um evento. Esses controles são uma segunda linha de defesa, alertando você sobre eventos de segurança que contornaram os controles preventivos em vigor. Para obter mais informações, consulte Controles detectivos em Como implementar controles de segurança na AWS.

mapeamento do fluxo de valor de desenvolvimento (DVSM)

Um processo usado para identificar e priorizar restrições que afetam negativamente a velocidade e a qualidade em um ciclo de vida de desenvolvimento de software. O DVSM estende o processo de mapeamento do fluxo de valor originalmente projetado para práticas de manufatura enxuta. Ele se concentra nas etapas e equipes necessárias para criar e movimentar valor por meio do processo de desenvolvimento de software.

gêmeo digital

Uma representação virtual de um sistema real, como um prédio, fábrica, equipamento industrial ou linha de produção. Os gêmeos digitais oferecem suporte à manutenção preditiva, ao monitoramento remoto e à otimização da produção.

tabela de dimensões

Em um esquema em estrela, uma tabela menor que contém atributos de dados sobre dados quantitativos em uma tabela de fatos. Os atributos da tabela de dimensões geralmente são campos de texto ou números discretos que se comportam como texto. Esses atributos normalmente são usados para restringir consultas, filtrar e rotular conjuntos de resultados.

desastre

Um evento que impede que uma workload ou sistema cumpra seus objetivos de negócios em seu local principal de implantação. Esses eventos podem ser desastres naturais, falhas técnicas ou o resultado de ações humanas, como configuração incorreta não intencional ou ataque de malware.

Recuperação de desastres (RD)

A estratégia e o processo que você usa para minimizar o tempo de inatividade e a perda de dados causados por um disastre. Para obter mais informações, consulte Recuperação de desastres de cargas de trabalho em AWS: Recuperação na nuvem no AWS Well-Architected Framework.

DML

Veja linguagem de manipulação de banco de dados.

design orientado por domínio

Uma abordagem ao desenvolvimento de um sistema de software complexo conectando seus componentes aos domínios em evolução, ou principais metas de negócios, atendidos por cada componente. Esse conceito foi introduzido por Eric Evans em seu livro, Design orientado por domínio: lidando com a complexidade no coração do software (Boston: Addison-Wesley Professional, 2003). Para obter informações sobre como usar o design orientado por domínio com o padrão strangler fig, consulte Modernizar incrementalmente os serviços web herdados do Microsoft ASP.NET (ASMX) usando contêineres e o Amazon API Gateway.

DR

Veja recuperação de desastres.

Detecção da oscilação

Rastreamento de desvios de uma configuração de linha de base. Por exemplo, você pode usar AWS CloudFormation para detectar desvios nos recursos do sistema ou AWS Control Tower para detectar mudanças em seu landing zone que possam afetar a conformidade com os requisitos de governança.

DVSM

Veja mapeamento do fluxo de valor de desenvolvimento.

E

EDA

Veja análise exploratória de dados.

EDI

Veja intercâmbio eletrônico de dados.

computação de borda

A tecnologia que aumenta o poder computacional de dispositivos inteligentes nas bordas de uma rede de IoT. Quando comparada com a computação em nuvem, a computação de borda pode reduzir a latência da comunicação e melhorar o tempo de resposta.

intercâmbio eletrônico de dados (EDI)

A troca automatizada de documentos comerciais entre organizações. Para obter mais informações, consulte O que é EDI (Intercâmbio eletrônico de dados)?.

criptografia

Um processo de computação que transforma dados de texto simples, legíveis por humanos, em texto cifrado.

chave de criptografia

Uma sequência criptográfica de bits aleatórios que é gerada por um algoritmo de criptografia. As chaves podem variar em tamanho, e cada chave foi projetada para ser imprevisível e exclusiva.

endianismo

A ordem na qual os bytes são armazenados na memória do computador. Os sistemas big-endian armazenam o byte mais significativo antes. Os sistemas little-endian armazenam o byte menos significativo antes.

endpoint

Veja endpoint de serviço.

serviço de endpoint

Um serviço que pode ser hospedado em uma nuvem privada virtual (VPC) para ser compartilhado com outros usuários. Você pode criar um serviço de endpoint com AWS PrivateLink e conceder permissões a outros diretores Contas da AWS ou a AWS Identity and Access Management (IAM). Essas contas ou entidades principais podem se conectar ao serviço de endpoint de maneira privada criando endpoints da VPC de interface. Para obter mais informações, consulte Criar um serviço de endpoint na documentação do Amazon Virtual Private Cloud (Amazon VPC).

planejamento de recursos empresariais (ERP)

Um sistema que automatiza e gerencia os principais processos de negócios (como contabilidade, MES e gerenciamento de projetos) para uma empresa.

criptografia envelopada

O processo de criptografar uma chave de criptografia com outra chave de criptografia. Para obter mais informações, consulte Criptografia de envelope na documentação AWS Key Management Service (AWS KMS).

ambiente

Uma instância de uma aplicação em execução. Estes são tipos comuns de ambientes na computação em nuvem:

- ambiente de desenvolvimento: uma instância de uma aplicação em execução que está disponível somente para a equipe principal responsável pela manutenção da aplicação. Ambientes de desenvolvimento são usados para testar mudanças antes de promovê-las para ambientes superiores. Esse tipo de ambiente às vezes é chamado de ambiente de teste.
- ambientes inferiores: todos os ambientes de desenvolvimento para uma aplicação, como aqueles usados para compilações e testes iniciais.
- ambiente de produção: uma instância de uma aplicação em execução que os usuários finais podem acessar. Em um CI/CD pipeline, o ambiente de produção é o último ambiente de implantação.
- ambientes superiores: todos os ambientes que podem ser acessados por usuários que não sejam a equipe principal de desenvolvimento. Isso pode incluir um ambiente de produção, ambientes de pré-produção e ambientes para testes de aceitação do usuário.

epic

Em metodologias ágeis, categorias funcionais que ajudam a organizar e priorizar seu trabalho. Os epics fornecem uma descrição de alto nível dos requisitos e das tarefas de implementação. Por exemplo, os épicos de segurança AWS da CAF incluem gerenciamento de identidade e acesso, controles de detetive, segurança de infraestrutura, proteção de dados e resposta a incidentes. Para obter mais informações sobre epics na estratégia de migração da AWS , consulte o guia de implementação do programa.

ERP

Veja planejamento de recursos empresariais.

análise exploratória de dados (EDA)

O processo de analisar um conjunto de dados para entender suas principais características. Você coleta ou agrega dados e, em seguida, realiza investigações iniciais para encontrar padrões, detectar anomalias e verificar suposições. O EDA é realizado por meio do cálculo de estatísticas resumidas e da criação de visualizações de dados.

F

tabela de fatos

A tabela central em um esquema em estrela. Ela armazena dados quantitativos sobre as operações comerciais. Normalmente, uma tabela de fatos contém dois tipos de colunas: as que contêm medidas e as que contêm uma chave externa para uma tabela de dimensões.

Antecipar-se à falha

Uma filosofia que usa testes frequentes e incrementais para reduzir o ciclo de vida do desenvolvimento. É uma parte essencial de uma abordagem ágil.

delimitação de isolamento contra falhas

No Nuvem AWS, um limite, como uma zona de disponibilidade, Região da AWS um plano de controle ou um plano de dados, que limita o efeito de uma falha e ajuda a melhorar a resiliência das cargas de trabalho. Para obter mais informações, consulte AWS Fault Isolation Boundaries.

ramificação de recursos

Veja ramificação.

recursos

Os dados de entrada usados para fazer uma previsão. Por exemplo, em um contexto de manufatura, os recursos podem ser imagens capturadas periodicamente na linha de fabricação.

importância do recurso

O quanto um recurso é importante para as previsões de um modelo. Isso geralmente é expresso como uma pontuação numérica que pode ser calculada por meio de várias técnicas, como Shapley Additive Explanations (SHAP) e gradientes integrados. Para obter mais informações, consulte Interpretabilidade do modelo de aprendizado de máquina com AWS.

transformação de recursos

O processo de otimizar dados para o processo de ML, incluindo enriquecer dados com fontes adicionais, escalar valores ou extrair vários conjuntos de informações de um único campo de dados. Isso permite que o modelo de ML se beneficie dos dados. Por exemplo, se a data “2021-05-27 00:15:37” for dividida em “2021”, “maio”, “quinta” e “15”, isso poderá ajudar o algoritmo de aprendizado a aprender padrões diferenciados associados a diferentes componentes de dados.

prompt few shot

Fornecer a um LLM um pequeno número de exemplos que demonstram a tarefa e o resultado desejado antes de solicitar que ele execute uma tarefa semelhante. Essa técnica é uma aplicação do aprendizado em contexto, em que os modelos aprendem com exemplos (shots) incorporados aos prompts. Prompts few-shot podem ser eficazes para tarefas que exigem formatação, raciocínio ou conhecimento de domínio específicos. Veja também prompts zero-shot.

FGAC

Veja controle de acesso refinado.

Controle de acesso refinado (FGAC)

O uso de várias condições para permitir ou negar uma solicitação de acesso.

migração flash-cut

Um método de migração de banco de dados que usa replicação contínua de dados via captura de dados de alteração para migrar os dados no menor tempo possível, em vez de usar uma abordagem em fases. O objetivo é reduzir ao mínimo o tempo de inatividade.

FM

Veja modelo de base.

modelo de base (FM)

Uma grande rede neural de aprendizado profundo que vem treinando em grandes conjuntos de dados generalizados e não rotulados. FMs são capazes de realizar uma ampla variedade de tarefas gerais, como entender a linguagem, gerar texto e imagens e conversar em linguagem natural. Para obter mais informações, consulte O que são modelos de base?.

G

IA generativa

Um subconjunto de modelos de IA que foram treinados em grandes quantidades de dados e que podem usar um simples prompt de texto para criar novos artefatos e conteúdo, como imagens, vídeos, texto e áudio. Para obter mais informações, consulte O que é IA generativa?.

bloqueio geográfico

Veja restrições geográficas.

restrições geográficas (bloqueio geográfico)

Na Amazon CloudFront, uma opção para impedir que usuários em países específicos acessem distribuições de conteúdo. É possível usar uma lista de permissões ou uma lista de bloqueios para especificar países aprovados e banidos. Para obter mais informações, consulte Restringir a distribuição geográfica do seu conteúdo na CloudFront documentação.

Fluxo de trabalho do GitFlow

Uma abordagem na qual ambientes inferiores e superiores usam ramificações diferentes em um repositório de código-fonte. O fluxo de trabalho do Gitflow é considerado legado, e o fluxo de trabalho trunk-based é a abordagem moderna e preferencial.

golden image

Um snapshot de um sistema ou software usado como modelo para implantar novas instâncias desse sistema ou software. Por exemplo, na manufatura, uma golden image pode ser usada para provisionar software em vários dispositivos e ajudar a melhorar a velocidade, a escalabilidade e a produtividade nas operações de fabricação de dispositivos.

estratégia greenfield

A ausência de infraestrutura existente em um novo ambiente. Ao adotar uma estratégia greenfield para uma arquitetura de sistema, é possível selecionar todas as novas tecnologias sem a restrição da compatibilidade com a infraestrutura existente, também conhecida como brownfield. Se estiver expandindo a infraestrutura existente, poderá combinar as estratégias brownfield e greenfield.

barreira de proteção

Uma regra de alto nível que ajuda a governar recursos, políticas e conformidade em todas as unidades organizacionais ()OUs. Barreiras de proteção preventivas impõem políticas para garantir o alinhamento a padrões de conformidade. Elas são implementadas usando políticas de controle de serviço e limites de permissões do IAM. Barreiras de proteção detectivas detectam violações de políticas e problemas de conformidade e geram alertas para remediação. Eles são implementados usando AWS Config, AWS Security Hub CSPM, Amazon GuardDuty AWS Trusted Advisor, Amazon Inspector e verificações personalizadas AWS Lambda .

H

HA

Veja alta disponibilidade.

migração heterogênea de bancos de dados

Migrar seu banco de dados de origem para um banco de dados de destino que usa um mecanismo de banco de dados diferente (por exemplo, Oracle para Amazon Aurora). A migração heterogênea geralmente faz parte de um esforço de redefinição da arquitetura, e converter

o esquema pode ser uma tarefa complexa. O AWS fornece o AWS SCT para ajudar nas conversões de esquemas.

alta disponibilidade (HA)

A capacidade de uma workload operar continuamente, sem intervenção, em caso de desafios ou desastres. Os sistemas AH são projetados para realizar o failover automático, oferecer consistentemente desempenho de alta qualidade e lidar com diferentes cargas e falhas com impacto mínimo no desempenho.

modernização de historiador

Uma abordagem usada para modernizar e atualizar os sistemas de tecnologia operacional (OT) para melhor atender às necessidades do setor de manufatura. Um historiador é um tipo de banco de dados usado para coletar e armazenar dados de várias fontes em uma fábrica.

dados de hold-out

Uma parte dos dados históricos rotulados que são retidos de um conjunto de dados usado para treinar um modelo de machine learning. Você pode usar dados de hold-out para avaliar a performance do modelo comparando as predições do modelo com os dados de retenção.

migração homogênea de bancos de dados

Migrar seu banco de dados de origem para um banco de dados de destino que compartilha o mesmo mecanismo de banco de dados (por exemplo, Microsoft SQL Server para Amazon RDS para SQL Server). A migração homogênea geralmente faz parte de um esforço de redefinição da hospedagem ou da plataforma. É possível usar utilitários de banco de dados nativos para migrar o esquema.

dados quentes

Dados acessados com frequência, como dados em tempo real ou dados translacionais recentes. Esses dados normalmente exigem uma camada ou classe de armazenamento de alto desempenho para fornecer respostas rápidas às consultas.

hotfix

Uma correção urgente para um problema crítico em um ambiente de produção. Devido à sua urgência, um hotfix geralmente é feito fora do fluxo de trabalho normal de DevOps lançamento.

período de hipercuidados

Imediatamente após a substituição, o período em que uma equipe de migração gerencia e monitora as aplicações migradas na nuvem para resolver quaisquer problemas. Normalmente,

a duração desse período é de 1 a 4 dias. No final do período de hipercuidados, a equipe de migração normalmente transfere a responsabilidade pelas aplicações para a equipe de operações de nuvem.

eu

IaC

Veja infraestrutura como código.

Política baseada em identidade

Uma política anexada a um ou mais diretores do IAM que define suas permissões no Nuvem AWS ambiente.

aplicação ociosa

Uma aplicação que tem um uso médio de CPU e memória entre 5 e 20% em um período de 90 dias. Em um projeto de migração, é comum retirar essas aplicações ou retê-las on-premises.

IIoT

Veja Internet das Coisas Industrial.

infraestrutura imutável

Um modelo que implanta uma nova infraestrutura para workloads de produção em vez de atualizar, aplicar patches ou modificar a infraestrutura existente. Infraestruturas imutáveis são inerentemente mais consistentes, confiáveis e preditivas do que infraestruturas mutáveis. Para obter mais informações, consulte a prática recomendada Implantar usando infraestrutura imutável no AWS Well-Architected Framework.

VPC de entrada (admissão)

Em uma arquitetura de AWS várias contas, uma VPC que aceita, inspeciona e roteia conexões de rede de fora de um aplicativo. A Arquitetura de Referência de AWS Segurança recomenda configurar sua conta de rede com entrada, saída e inspeção VPCs para proteger a interface bidirecional entre seu aplicativo e a Internet em geral.

migração incremental

Uma estratégia de substituição na qual você migra a aplicação em pequenas partes, em vez de realizar uma única substituição completa. Por exemplo, é possível mover inicialmente

apenas alguns microsserviços ou usuários para o novo sistema. Depois de verificar se tudo está funcionando corretamente, mova os microsserviços ou usuários adicionais de forma incremental até poder descomissionar seu sistema herdado. Essa estratégia reduz os riscos associados a migrações de grande porte.

Indústria 4.0

Um termo que foi introduzido por Klaus Schwab em 2016 para se referir à modernização dos processos de manufatura por meio de avanços em conectividade, dados em tempo real, automação, analytics e IA/ML.

infraestrutura

Todos os recursos e ativos contidos no ambiente de uma aplicação.

Infraestrutura como código (IaC)

O processo de provisionamento e gerenciamento da infraestrutura de uma aplicação por meio de um conjunto de arquivos de configuração. A IaC foi projetada para ajudar você a centralizar o gerenciamento da infraestrutura, padronizar recursos e escalar rapidamente para que novos ambientes sejam reproduzíveis, confiáveis e consistentes.

Internet industrial das coisas (IIoT)

O uso de sensores e dispositivos conectados à Internet nos setores industriais, como manufatura, energia, automotivo, saúde, ciências biológicas e agricultura. Para obter mais informações, consulte Criando uma estratégia de transformação digital industrial da Internet das Coisas (IIoT).

VPC de inspeção

Em uma arquitetura de AWS várias contas, uma VPC centralizada que gerencia as inspeções do tráfego de rede entre VPCs (na mesma ou em diferentes Regiões da AWS) a Internet e as redes locais. A Arquitetura de Referência de AWS Segurança recomenda configurar sua conta de rede com entrada, saída e inspeção VPCs para proteger a interface bidirecional entre seu aplicativo e a Internet em geral.

Internet das coisas (IoT)

A rede de objetos físicos conectados com sensores ou processadores incorporados que se comunicam com outros dispositivos e sistemas pela Internet ou por uma rede de comunicação local. Para obter mais informações, consulte O que é IoT?

interpretabilidade

Uma característica de um modelo de machine learning que descreve o grau em que um ser humano pode entender como as previsões do modelo dependem de suas entradas. Para obter mais informações, consulte Interpretabilidade do modelo de aprendizado de máquina com AWS.

IoT

Veja Internet das Coisas.

Biblioteca de informações de TI (ITIL)

Um conjunto de práticas recomendadas para fornecer serviços de TI e alinhar esses serviços a requisitos de negócios. A ITIL fornece a base para o ITSM.

Gerenciamento de serviços de TI (ITSM)

Atividades associadas a design, implementação, gerenciamento e suporte de serviços de TI para uma organização. Para obter informações sobre a integração de operações em nuvem com ferramentas de ITSM, consulte o guia de integração de operações.

ITIL

Veja biblioteca de informações de TI.

ITSM

Veja gerenciamento de serviços de TI.

L

controle de acesso baseado em etiqueta (LBAC)

Uma implementação do controle de acesso obrigatório (MAC) em que os usuários e os dados em si recebem explicitamente um valor de etiqueta de segurança. A interseção entre a etiqueta de segurança do usuário e a etiqueta de segurança dos dados determina quais linhas e colunas podem ser vistas pelo usuário.

zona de pouso

Uma landing zone é um AWS ambiente bem arquitetado, com várias contas, escalável e seguro. Um ponto a partir do qual suas organizações podem iniciar e implantar rapidamente workloads e aplicações com confiança em seu ambiente de segurança e infraestrutura. Para obter mais

informações sobre zonas de pouso, consulte Configurar um ambiente da AWS com várias contas seguro e escalável.

grande modelo de linguagem (LLM)

Um modelo de IA de aprendizado profundo pré-treinado em uma grande quantidade de dados. Um LLM pode realizar várias tarefas, como responder a perguntas, resumir documentos, traduzir texto para outros idiomas e completar frases. Para obter mais informações, consulte O que são LLMs.

migração de grande porte

Uma migração de 300 servidores ou mais.

LBAC

Veja controle de acesso baseado em rótulo.

privilégio mínimo

A prática recomendada de segurança de conceder as permissões mínimas necessárias para executar uma tarefa. Para obter mais informações, consulte Aplicar permissões de privilégios mínimos na documentação do IAM.

mover sem alterações (lift-and-shift)

Veja 7 Rs.

sistema little-endian

Um sistema que armazena o byte menos significativo antes. Veja também endianness.

LLM

Veja grande modelo de linguagem.

ambientes inferiores

Veja ambiente.

M

machine learning (ML)

Um tipo de inteligência artificial que usa algoritmos e técnicas para reconhecimento e aprendizado de padrões. O ML analisa e aprende com dados gravados, por exemplo, dados da

Internet das Coisas (IoT), para gerar um modelo estatístico baseado em padrões. Para obter mais informações, consulte Machine learning.

ramificação principal

Veja ramificação.

Malware

Software projetado para comprometer a segurança ou a privacidade do computador. O malware pode interromper os sistemas do computador, vazar informações sensíveis ou obter acesso não autorizado. Exemplos de malware incluem vírus, worms, ransomware, cavalos de Troia, spyware e keyloggers.

Serviços gerenciados

Serviços da AWS para o qual AWS opera a camada de infraestrutura, o sistema operacional e as plataformas, e você acessa os endpoints para armazenar e recuperar dados. O Amazon Simple Storage Service (Amazon S3) e o Amazon DynamoDB são exemplos de serviços gerenciados. Eles também são conhecidos como serviços abstraídos.

sistema de execução de manufatura (MES)

Um sistema de software para rastrear, monitorar, documentar e controlar processos de produção que convertem matérias-primas em produtos acabados no chão de fábrica.

MAP

Veja Programa de Aceleração da Migração.

mecanismo

Um processo completo em que você cria uma ferramenta, impulsiona a adoção da ferramenta e, em seguida, inspeciona os resultados para fazer ajustes. Um mecanismo é um ciclo que se reforça e se aprimora à medida que opera. Para obter mais informações, consulte Construindo mecanismos no AWS Well-Architected Framework.

conta de membro

Todos, Contas da AWS exceto a conta de gerenciamento, que fazem parte de uma organização em AWS Organizations. Uma conta só pode ser membro de uma organização de cada vez.

MES

Veja sistema de execução de manufatura.

Transporte de Telemetria de Enfileiramento de Mensagens (MQTT)

: Um protocolo de comunicação leve machine-to-machine (M2M), baseado no padrão de publicação/assinatura, para dispositivos de IoT com recursos limitados.

microsserviço

: Um serviço pequeno e independente que se comunica de forma bem definida APIs e normalmente é de propriedade de equipes pequenas e independentes. Por exemplo, um sistema de seguradora pode incluir microsserviços que mapeiam as capacidades comerciais, como vendas ou marketing, ou subdomínios, como compras, reclamações ou análises. Os benefícios dos microsserviços incluem agilidade, escalabilidade flexível, fácil implantação, código reutilizável e resiliência. Para obter mais informações, consulte Integração de microsserviços usando serviços sem AWS servidor.

arquitetura de microsserviços

: Uma abordagem à criação de aplicações com componentes independentes que executam cada processo de aplicação como um microsserviço. Esses microsserviços se comunicam por meio de uma interface bem definida usando leveza. APIs Cada microsserviço nessa arquitetura pode ser atualizado, implantado e escalado para atender à demanda por funções específicas de uma aplicação. Para obter mais informações, consulte Implementação de microsserviços em. AWS

Programa de Aceleração da Migração (MAP)

: Um AWS programa que fornece suporte de consultoria, treinamento e serviços para ajudar as organizações a criar uma base operacional sólida para migrar para a nuvem e ajudar a compensar o custo inicial das migrações. O MAP inclui uma metodologia de migração para executar migrações legadas de forma metódica e um conjunto de ferramentas para automatizar e acelerar cenários comuns de migração.

migração em escala

: O processo de mover a maior parte do portfólio de aplicações para a nuvem em ondas, com mais aplicações sendo movidas em um ritmo mais rápido a cada onda. Essa fase usa as práticas recomendadas e lições aprendidas nas fases anteriores para implementar uma fábrica de migração de equipes, ferramentas e processos para agilizar a migração de workloads por meio de automação e entrega ágeis. Esta é a terceira fase da estratégia de migração para a AWS.

fábrica de migração

: Equipes multifuncionais que simplificam a migração de workloads por meio de abordagens automatizadas e ágeis. As equipes da fábrica de migração geralmente incluem operações,

analistas e proprietários de negócios, engenheiros de migração, desenvolvedores e DevOps profissionais que trabalham em sprints. Entre 20 e 50% de um portfólio de aplicações corporativas consiste em padrões repetidos que podem ser otimizados por meio de uma abordagem de fábrica. Para obter mais informações, consulte discussão sobre fábricas de migração e o guia do Cloud Migration Factory neste conjunto de conteúdo.

metadados de migração

As informações sobre a aplicação e o servidor necessárias para concluir a migração. Cada padrão de migração exige um conjunto de metadados de migração diferente. Exemplos de metadados de migração incluem a sub-rede, o grupo de segurança e AWS a conta de destino.

padrão de migração

Uma tarefa de migração repetível que detalha a estratégia de migração, o destino da migração e a aplicação ou o serviço de migração usado. Exemplo: rehospede a migração para o Amazon EC2 AWS com o Application Migration Service.

Avaliação de Portfólio para Migração (MPA)

Uma ferramenta on-line que fornece informações para validar o caso de negócios para migrar para a Nuvem AWS. O MPA fornece avaliação detalhada do portfólio (dimensionamento correto do servidor, preços, comparações de TCO, análise de custos de migração), bem como planejamento de migração (análise e coleta de dados de aplicações, agrupamento de aplicações, priorização de migração e planejamento de ondas). A ferramenta MPA (requer login) está disponível gratuitamente para todos os AWS consultores e consultores parceiros da APN.

Avaliação de Preparação para Migração (MRA)

O processo de obter insights sobre o status de prontidão de uma organização para a nuvem, identificar pontos fortes e fracos e criar um plano de ação para fechar as lacunas identificadas, usando o CAF. AWS Para mais informações, consulte o guia de preparação para migração. A MRA é a primeira fase da estratégia de migração para a AWS.

estratégia de migração

A abordagem usada para migrar uma workload para a Nuvem AWS. Para obter mais informações, veja a entrada 7 Rs neste glossário e consulte Mobilize sua organização para acelerar migrações em grande escala.

ML

Veja machine learning.

modernização

Transformar uma aplicação desatualizada (herdada ou monolítica) e sua infraestrutura em um sistema ágil, elástico e altamente disponível na nuvem para reduzir custos, ganhar eficiência e aproveitar as inovações. Para obter mais informações, consulte Strategy for modernizing applications in the Nuvem AWS.

avaliação de preparação para modernização

Uma avaliação que ajuda a determinar a preparação para modernização das aplicações de uma organização. Ela identifica benefícios, riscos e dependências e determina o quão bem a organização pode acomodar o estado futuro dessas aplicações. O resultado da avaliação é um esquema da arquitetura de destino, um roteiro que detalha as fases de desenvolvimento e os marcos do processo de modernização e um plano de ação para abordar as lacunas identificadas. Para obter mais informações, consulte Evaluating modernization readiness for applications in the Nuvem AWS.

aplicações monolíticas (monólitos)

Aplicações que são executadas como um único serviço com processos fortemente acoplados. As aplicações monolíticas apresentam várias desvantagens. Se um recurso da aplicação apresentar um aumento na demanda, toda a arquitetura deverá ser escalada. Adicionar ou melhorar os recursos de uma aplicação monolítica também se torna mais complexo quando a base de código cresce. Para resolver esses problemas, é possível criar uma arquitetura de microsserviços. Para obter mais informações, consulte Decompor monólitos em microsserviços.

MPA

Veja Avaliação do Portfólio para Migração.

MQTT

Veja Transporte de Telemetria de Enfileiramento de Mensagens.

classificação multiclasse

Um processo que ajuda a gerar previsões para várias classes (prevendo um ou mais de dois resultados). Por exemplo, um modelo de ML pode perguntar “Este produto é um livro, um carro ou um telefone?” ou “Qual categoria de produtos é mais interessante para este cliente?”

infraestrutura mutável

Um modelo que atualiza e modifica a infraestrutura existente para workloads de produção. Para melhorar a consistência, confiabilidade e previsibilidade, o AWS Well-Architected Framework recomenda o uso de infraestrutura imutável como uma prática recomendada.

O

OAC

Veja controle de acesso de origem.

OAI

Veja identidade de acesso de origem.

OCM

Veja gerenciamento de alterações organizacionais.

migração offline

Um método de migração no qual a workload de origem é desativada durante o processo de migração. Esse método envolve tempo de inatividade prolongado e geralmente é usado para workloads pequenas e não críticas.

OI

Veja integração de operações.

Ola

Veja acordo de nível operacional.

migração online

Um método de migração no qual a workload de origem é copiada para o sistema de destino sem ser colocada offline. As aplicações conectadas à workload podem continuar funcionando durante a migração. Esse método envolve um tempo de inatividade nulo ou mínimo e normalmente é usado para workloads essenciais para a produção.

OPC-UA

Veja Open Process Communications - Unified Architecture.

Open Process Communications - Unified Architecture (OPC-UA)

Um protocolo de comunicação machine-to-machine (M2M) para automação industrial. O OPC-UA fornece um padrão de interoperabilidade com esquemas de criptografia, autenticação e autorização de dados.

acordo de nível operacional (OLA)

Um acordo que esclarece o que os grupos funcionais de TI prometem oferecer uns aos outros para apoiar um acordo de serviço (SLA).

análise de prontidão operacional (ORR)

Uma lista de verificação de perguntas e práticas recomendadas associadas que ajudam você a entender, avaliar, prevenir ou reduzir o escopo de incidentes e possíveis falhas. Para obter mais informações, consulte Operational Readiness Reviews (ORR) no AWS Well-Architected Framework.

tecnologia operacional (TO)

Sistemas de hardware e software que trabalham com o ambiente físico para controlar operações, equipamentos e infraestrutura industriais. Na manufatura, a integração dos sistemas de tecnologia da informação (TI) e tecnologia operacional (TO) é o foco principal das transformações da Indústria 4.0.

integração de operações (OI)

O processo de modernização das operações na nuvem, que envolve planejamento de preparação, automação e integração. Para obter mais informações, consulte o guia de integração de operações.

trilha organizacional

Uma trilha criada por ela AWS CloudTrail registra todos os eventos de todos Contas da AWS em uma organização em AWS Organizations. Essa trilha é criada em cada Conta da AWS que faz parte da organização e monitora a atividade em cada conta. Para obter mais informações, consulte Criação de uma trilha para uma organização na CloudTrail documentação.

gerenciamento de alterações organizacionais (OCM)

Uma estrutura para gerenciar grandes transformações de negócios disruptivas de uma perspectiva de pessoas, cultura e liderança. O OCM ajuda as organizações a se prepararem e fazerem a transição para novos sistemas e estratégias, acelerando a adoção de alterações, abordando questões de transição e promovendo mudanças culturais e organizacionais. Na estratégia de AWS migração, essa estrutura é chamada de aceleração de pessoas, devido à velocidade de mudança exigida nos projetos de adoção da nuvem. Para obter mais informações, consulte o guia do OCM.

controle de acesso de origem (OAC)

Em CloudFront, uma opção aprimorada para restringir o acesso para proteger seu conteúdo do Amazon Simple Storage Service (Amazon S3). O OAC oferece suporte a todos os buckets S3 Regiões da AWS, criptografia do lado do servidor com AWS KMS (SSE-KMS) e solicitações dinâmicas ao bucket S3. `PUT` `DELETE`

Identidade do acesso de origem (OAI)

Em CloudFront, uma opção para restringir o acesso para proteger seu conteúdo do Amazon S3. Quando você usa o OAI, CloudFront cria um principal com o qual o Amazon S3 pode se autenticar. Os diretores autenticados podem acessar o conteúdo em um bucket do S3 somente por meio de uma distribuição específica. CloudFront Veja também OAC, que fornece um controle de acesso mais granular e aprimorado.

ORR

Veja análise de prontidão operacional.

OT

Veja tecnologia operacional.

VPC de saída (egresso)

Em uma arquitetura de AWS várias contas, uma VPC que gerencia conexões de rede que são iniciadas de dentro de um aplicativo. A Arquitetura de Referência de AWS Segurança recomenda configurar sua conta de rede com entrada, saída e inspeção VPCs para proteger a interface bidirecional entre seu aplicativo e a Internet em geral.

P

limite de permissões

Uma política de gerenciamento do IAM anexada a entidades principais do IAM para definir as permissões máximas que o usuário ou perfil podem ter. Para obter mais informações, consulte Limites de permissões na documentação do IAM.

Informações de identificação pessoal (PII)

Informações que, quando visualizadas diretamente ou combinadas com outros dados relacionados, podem ser usadas para inferir razoavelmente a identidade de um indivíduo. Exemplos de PII incluem nomes, endereços e informações de contato.

PII

Veja informações de identificação pessoal.

manual

Um conjunto de etapas predefinidas que capturam o trabalho associado às migrações, como a entrega das principais funções operacionais na nuvem. Um manual pode assumir a forma de scripts, runbooks automatizados ou um resumo dos processos ou etapas necessários para operar seu ambiente modernizado.

PLC

Veja controlador lógico programável.

PLM

Veja gerenciamento do ciclo de vida do produto.

política

Um objeto que pode definir permissões (veja política baseada em identidade), especificar condições de acesso (veja políta baseada em recurso) ou definir as permissões máximas para todas as contas em uma organização no AWS Organizations (veja política de controle de serviços).

persistência poliglota

Escolher de forma independente a tecnologia de armazenamento de dados de um microsserviço com base em padrões de acesso a dados e outros requisitos. Se seus microsserviços tiverem a mesma tecnologia de armazenamento de dados, eles poderão enfrentar desafios de implementação ou apresentar baixa performance. Os microsserviços serão implementados com mais facilidade e alcançarão performance e escalabilidade melhores se usarem o armazenamento de dados mais bem adaptado às suas necessidades.

avaliação do portfólio

Um processo de descobrir, analisar e priorizar o portfólio de aplicações para planejar a migração. Para obter mais informações, consulte Avaliar a preparação para a migração.

predicado

Uma condição de consulta que retorna `true` ou `false`, normalmente localizada em uma cláusula `WHERE`.

pushdown de predicados

Uma técnica de otimização de consultas de banco de dados que filtra os dados na consulta antes da transferência. Isso reduz a quantidade de dados que devem ser recuperados e processados do banco de dados relacional e melhora a performance das consultas.

controle preventivo

Um controle de segurança projetado para evitar que um evento ocorra. Esses controles são a primeira linha de defesa para ajudar a evitar acesso não autorizado ou alterações indesejadas em sua rede. Para obter mais informações, consulte Controles preventivos em Como implementar controles de segurança na AWS.

principal (entidade principal)

Uma entidade AWS que pode realizar ações e acessar recursos. Essa entidade geralmente é um usuário raiz para um Conta da AWS, uma função do IAM ou um usuário. Para obter mais informações, consulte Entidade principal em Termos e conceitos de perfis na documentação do IAM.

Privacidade por design

Uma abordagem em engenharia de sistemas que leva em consideração a privacidade em todo o processo de desenvolvimento.

zonas hospedadas privadas

Um contêiner que contém informações sobre como você deseja que o Amazon Route 53 responda às consultas de DNS para um domínio e seus subdomínios em um ou mais. VPCs Para obter mais informações, consulte Como trabalhar com zonas hospedadas privadas na documentação do Route 53.

controle proativo

Um controle de segurança desenvolvido para evitar a implantação de recursos não conformes. Esses controles verificam os recursos antes de serem provisionados. Se o recurso não estiver em conformidade com o controle, ele não será provisionado. Para obter mais informações, consulte o guia de referência de controles na AWS Control Tower documentação e consulte Controles proativos em Implementação de controles de segurança em AWS.

gerenciamento do ciclo de vida do produto (PLM)

O gerenciamento de dados e processos de um produto em todo o seu ciclo de vida, desde a concepção, o desenvolvimento e o lançamento, passando pelo crescimento e maturidade, até o declínio e a remoção.

ambiente de produção

Veja ambiente.

controlador lógico programável (PLC)

Na manufatura, um computador altamente confiável e adaptável que monitora as máquinas e automatiza os processos de fabricação.

encadeamento de prompts

Uso da saída de um prompt do LLM como entrada para o próximo prompt para gerar respostas melhores. Essa técnica é usada para dividir uma tarefa complexa em subtarefas, ou para refinar ou expandir iterativamente uma resposta preliminar. Isso ajuda a melhorar a precisão e a relevância das respostas de um modelo e permite resultados mais granulares e personalizados.

pseudonimização

O processo de substituir identificadores pessoais em um conjunto de dados por valores de espaço reservado. A pseudonimização pode ajudar a proteger a privacidade pessoal. Os dados pseudonimizados ainda são considerados dados pessoais.

publish/subscribe (pub/sub)

Um padrão que permite comunicações assíncronas entre microsserviços para melhorar a escalabilidade e a capacidade de resposta. Por exemplo, em um MES baseado em microsserviços, um microsserviço pode publicar mensagens de eventos em um canal em que outros microsserviços possam assinar. O sistema pode adicionar novos microsserviços sem alterar o serviço de publicação.

Q

plano de consulta

Uma série de etapas, como instruções, usadas para acessar os dados em um sistema de banco de dados relacional SQL.

regressão de planos de consultas

Quando um otimizador de serviço de banco de dados escolhe um plano menos adequado do que escolhia antes de uma determinada alteração no ambiente de banco de dados ocorrer. Isso pode ser causado por alterações em estatísticas, restrições, configurações do ambiente, associações de parâmetros de consulta e atualizações do mecanismo de banco de dados.

R

Matriz RACI

Veja responsável, aprovador, consultado, informado (RACI).

RAG

Veja geração aumentada via recuperação.

ransomware

Um software mal-intencionado desenvolvido para bloquear o acesso a um sistema ou dados de computador até que um pagamento seja feito.

Matriz RASCI

Veja responsável, aprovador, consultado, informado (RACI).

RCAC

Veja controle de acesso por linha e coluna.

réplica de leitura

Uma cópia de um banco de dados usada somente para leitura. É possível encaminhar consultas para a réplica de leitura e reduzir a carga no banco de dados principal.

Redefinir arquitetura

Veja 7 Rs.

objetivo de ponto de recuperação (RPO).

O máximo período de tempo aceitável desde o último ponto de recuperação de dados. Isso determina o que é considerado uma perda aceitável de dados entre o último ponto de recuperação e a interrupção do serviço.

objetivo de tempo de recuperação (RTO)

O máximo atraso aceitável entre a interrupção e a restauração do serviço.

refatorar

Veja 7 Rs.

Região

Uma coleção de AWS recursos em uma área geográfica. Cada um Região da AWS é isolado e independente dos outros para fornecer tolerância a falhas, estabilidade e resiliência. Para obter informações, consulte Specify which Regiões da AWS your account can use.

regressão

Uma técnica de ML que prevê um valor numérico. Por exemplo, para resolver o problema de “Por qual preço esta casa será vendida?” um modelo de ML pode usar um modelo de regressão linear para prever o preço de venda de uma casa com base em fatos conhecidos sobre a casa (por exemplo, a metragem quadrada).

redefinir a hospedagem

Veja 7 Rs.

versão

Em um processo de implantação, o ato de promover mudanças em um ambiente de produção.

realocar

Veja 7 Rs.

redefinir a plataforma

Veja 7 Rs.

recomprar

Veja 7 Rs.

resiliência

A capacidade de uma aplicação de resistir ou se recuperar de interrupções. Alta disponibilidade e recuperação de desastres são considerações comuns ao planejar a resiliência na Nuvem AWS. Para obter mais informações, consulte Nuvem AWS Resilience.

política baseada em recurso

Uma política associada a um recurso, como um bucket do Amazon S3, um endpoint ou uma chave de criptografia. Esse tipo de política especifica quais entidades principais têm acesso permitido, ações válidas e quaisquer outras condições que devem ser atendidas.

matriz responsável, accountable, consultada, informada (RACI)

Uma matriz que define as funções e responsabilidades de todas as partes envolvidas nas atividades de migração e nas operações de nuvem. O nome da matriz é derivado dos tipos de responsabilidade definidos na matriz: responsável (R), responsabilizável (A), consultado (C) e informado (I). O tipo de suporte (S) é opcional. Se você incluir suporte, a matriz será chamada de matriz RASCI e, se excluir, será chamada de matriz RACI.

controle responsivo

Um controle de segurança desenvolvido para conduzir a remediação de eventos adversos ou desvios em relação à linha de base de segurança. Para obter mais informações, consulte Controles responsivos em Como implementar controles de segurança na AWS.

reter

Veja 7 Rs.

Retirada

Veja 7 Rs.

Geração Aumentada de Recuperação (RAG)

Uma tecnologia de IA generativa em que um LLM faz referência a uma fonte de dados autorizada que está fora de suas fontes de dados de treinamento antes de gerar uma resposta. Por exemplo, um modelo RAG pode realizar uma pesquisa semântica na base de conhecimento ou nos dados personalizados de uma organização. Para obter mais informações, consulte O que é RAG (geração aumentada via recuperação)?.

alternância

O processo de atualizar periodicamente um segredo para dificultar o acesso de um invasor às credenciais.

controle de acesso por linha e coluna (RCAC)

O uso de expressões SQL básicas e flexíveis que tenham regras de acesso definidas. O RCAC consiste em permissões de linha e máscaras de coluna.

RPO

Veja objetivo de ponto de recuperação.

RTO

Veja objetivo de tempo de recuperação.

runbook

Um conjunto de procedimentos manuais ou automatizados necessários para realizar uma tarefa específica. Eles são normalmente criados para agilizar operações ou procedimentos repetitivos com altas taxas de erro.

S

SAML 2.0

Um padrão aberto que muitos provedores de identidade (IdPs) usam. Esse recurso permite o login único federado (SSO), para que os usuários possam fazer login Console de gerenciamento da AWS ou chamar as operações da AWS API sem que você precise criar um usuário no IAM para todos em sua organização. Para obter mais informações sobre a federação baseada em SAML 2.0, consulte Sobre a federação baseada em SAML 2.0 na documentação do IAM.

SCADA

Veja controle de supervisão e aquisição de dados.

SCP

Veja política de controle de serviço.

secret

Em AWS Secrets Manager, informações confidenciais ou restritas, como uma senha ou credenciais de usuário, que você armazena de forma criptografada. Consiste no valor secreto e em seus metadados. O valor secreto pode ser binário, uma única string ou várias strings. Para obter mais informações, consulte What's in a Secrets Manager secret? na documentação do Secrets Manager.

segurança desde a concepção

Uma abordagem em engenharia de sistemas que leva em consideração a segurança em todo o processo de desenvolvimento.

controle de segurança

Uma barreira de proteção técnica ou administrativa que impede, detecta ou reduz a capacidade de uma ameaça explorar uma vulnerabilidade de segurança. Existem quatro tipos primários de controles de segurança: preventivos, detectivos, responsivos e proativos.

hardening da segurança

O processo de reduzir a superfície de ataque para torná-la mais resistente a ataques. Isso pode incluir ações como remover recursos que não são mais necessários, implementar a prática recomendada de segurança de conceder privilégios mínimos ou desativar recursos desnecessários em arquivos de configuração.

sistema de gerenciamento de eventos e informações de segurança (SIEM)

Ferramentas e serviços que combinam sistemas de gerenciamento de informações de segurança (SIM) e gerenciamento de eventos de segurança (SEM). Um sistema SIEM coleta, monitora e analisa dados de servidores, redes, dispositivos e outras fontes para detectar ameaças e violações de segurança e gerar alertas.

automação de resposta de segurança

Uma ação predefinida e programada projetada para responder ou remediar automaticamente um evento de segurança. Essas automações servem como controles de segurança responsivos ou detectivos que ajudam você a implementar as melhores práticas AWS de segurança. Exemplos de ações de resposta automatizada incluem a modificação de um grupo de segurança da VPC, a aplicação de patches em uma instância do Amazon EC2 ou a alternância de credenciais.

Criptografia do lado do servidor

Criptografia dos dados em seu destino, por AWS service (Serviço da AWS) quem os recebe.

política de controle de serviços (SCP)

Uma política que fornece controle centralizado sobre as permissões de todas as contas em uma organização em AWS Organizations. SCPs defina barreiras ou estabeleça limites nas ações que um administrador pode delegar a usuários ou funções. Você pode usar SCPs como listas de permissão ou listas de negação para especificar quais serviços ou ações são permitidos ou proibidos. Para obter mais informações, consulte Políticas de controle de serviço na AWS Organizations documentação.

service endpoint (endpoint de serviço)

O URL do ponto de entrada para um AWS service (Serviço da AWS). Você pode usar o endpoint para se conectar programaticamente ao serviço de destino. Para obter mais informações, consulte Endpoints do AWS service (Serviço da AWS) na Referência geral da AWS.

acordo de serviço (SLA)

Um acordo que esclarece o que uma equipe de TI promete fornecer aos clientes, como tempo de atividade e performance do serviço.

indicador de nível de serviço (SLI)

Uma avaliação de um aspecto de performance de um serviço, como taxa de erro, disponibilidade ou throughput.

objetivo de nível de serviço (SLO)

Uma métrica alvo que representa a integridade de um serviço, conforme avaliado por um indicador de nível de serviço.

modelo de responsabilidade compartilhada

Um modelo que descreve a responsabilidade com a qual você compartilha AWS pela segurança e conformidade na nuvem. AWS é responsável pela segurança da nuvem, enquanto você é responsável pela segurança na nuvem. Para obter mais informações, consulte o Modelo de responsabilidade compartilhada.

SIEM

Veja sistema de gerenciamento de eventos e informações de segurança.

ponto único de falha (SPOF)

Uma falha em um único componente crítico de uma aplicação que pode interromper o sistema.

SLA

Veja cacordo de serviço.

SLI

Veja indicador de nível de serviço.

SLO

Veja objetivo de nível de serviço.

split-and-seed modelo

Um padrão para escalar e acelerar projetos de modernização. À medida que novos recursos e lançamentos de produtos são definidos, a equipe principal se divide para criar novas equipes de produtos. Isso ajuda a escalar os recursos e os serviços da sua organização, melhora a produtividade do desenvolvedor e possibilita inovações rápidas. Para obter mais informações, consulte Phased approach to modernizing applications in the Nuvem AWS.

SPOF

Veja ponto único de falha.

esquema em estrela

Uma estrutura organizacional de banco de dados que usa uma grande tabela de fatos para armazenar dados transacionais ou medidos e usa uma ou mais tabelas dimensionais menores para armazenar atributos de dados. Essa estrutura foi projetada para ser usada em um data warehouse ou para fins de inteligência comercial.

padrão strangler fig

Uma abordagem à modernização de sistemas monolíticos que consiste em reescrever e substituir incrementalmente a funcionalidade do sistema até que o sistema herdado possa ser desativado. Esse padrão usa a analogia de uma videira que cresce e se torna uma árvore estabelecida e, eventualmente, supera e substitui sua hospedeira. O padrão foi apresentado por Martin Fowler como forma de gerenciar riscos ao reescrever sistemas monolíticos. Para ver um exemplo de como aplicar esse padrão, consulte Modernizar incrementalmente os serviços Web herdados do Microsoft ASP.NET (ASMX) usando contêineres e o Amazon API Gateway.

sub-rede

Um intervalo de endereços IP na VPC. Cada sub-rede fica alocada em uma única zona de disponibilidade.

controle supervisório e aquisição de dados (SCADA)

Na manufatura, um sistema que usa hardware e software para monitorar ativos físicos e operações de produção.

symmetric encryption (criptografia simétrica)

Um algoritmo de criptografia que usa a mesma chave para criptografar e descriptografar dados.

testes sintéticos

Testar um sistema de forma que simule as interações do usuário para detectar possíveis problemas ou monitorar a performance. Você pode usar o Amazon CloudWatch Synthetics para criar esses testes.

prompt do sistema

Uma técnica para fornecer contexto, instruções ou orientações a um LLM a fim de direcionar seu comportamento. Os prompts do sistema ajudam a definir o contexto e a estabelecer regras para interações com os usuários.

T

tags

Pares de valores-chave que atuam como metadados para organizar seus recursos. AWS As tags podem ajudar você a gerenciar, identificar, organizar, pesquisar e filtrar recursos da . Para obter mais informações, consulte Marcar seus recursos do AWS.

variável-alvo

O valor que você está tentando prever no ML supervisionado. Ela também é conhecida como variável de resultado. Por exemplo, em uma configuração de fabricação, a variável-alvo pode ser um defeito do produto.

lista de tarefas

Uma ferramenta usada para monitorar o progresso por meio de um runbook. Uma lista de tarefas contém uma visão geral do runbook e uma lista de tarefas gerais a serem concluídas. Para cada tarefa geral, ela inclui o tempo estimado necessário, o proprietário e o progresso.

ambiente de teste

Veja ambiente.

treinamento

O processo de fornecer dados para que seu modelo de ML aprenda. Os dados de treinamento devem conter a resposta correta. O algoritmo de aprendizado descobre padrões nos dados de treinamento que mapeiam os atributos dos dados de entrada no destino (a resposta que você deseja prever). Ele gera um modelo de ML que captura esses padrões. Você pode usar o modelo de ML para obter previsões de novos dados cujo destino você não conhece.

gateway de trânsito

Um hub de trânsito de rede que você pode usar para interconectar sua rede com VPCs a rede local. Para obter mais informações, consulte O que é um gateway de trânsito na AWS Transit Gateway documentação.

fluxo de trabalho baseado em troncos

Uma abordagem na qual os desenvolvedores criam e testam recursos localmente em uma ramificação de recursos e, em seguida, mesclam essas alterações na ramificação principal. A ramificação principal é então criada para os ambientes de desenvolvimento, pré-produção e produção, sequencialmente.

Acesso confiável

Conceder permissões a um serviço que você especifica para realizar tarefas em sua organização AWS Organizations e em suas contas em seu nome. O serviço confiável cria um perfil vinculado ao serviço em cada conta, quando esse perfil é necessário, para realizar tarefas de gerenciamento para você. Para obter mais informações, consulte Usando AWS Organizations com outros AWS serviços na AWS Organizations documentação.

tuning (ajustar)

Alterar aspectos do processo de treinamento para melhorar a precisão do modelo de ML. Por exemplo, você pode treinar o modelo de ML gerando um conjunto de rótulos, adicionando rótulos e repetindo essas etapas várias vezes em configurações diferentes para otimizar o modelo.

equipe de duas pizzas

Uma pequena DevOps equipe que você pode alimentar com duas pizzas. Uma equipe de duas pizzas garante a melhor oportunidade possível de colaboração no desenvolvimento de software.

U

incerteza

Um conceito que se refere a informações imprecisas, incompletas ou desconhecidas que podem minar a confiabilidade dos modelos preditivos de ML. Há dois tipos de incertezas: a incerteza epistêmica é causada por dados limitados e incompletos, enquanto a incerteza aleatória é causada pelo ruído e pela aleatoriedade inerentes aos dados. Para obter mais informações, consulte o guia Como quantificar a incerteza em sistemas de aprendizado profundo.

tarefas indiferenciadas

Também conhecido como trabalho pesado, trabalho necessário para criar e operar um aplicativo, mas que não fornece valor direto ao usuário final nem oferece vantagem competitiva. Exemplos de tarefas indiferenciadas incluem aquisição, manutenção e planejamento de capacidade.

ambientes superiores

Veja ambiente.

V

aspiração

Uma operação de manutenção de banco de dados que envolve limpeza após atualizações incrementais para recuperar armazenamento e melhorar a performance.

controle de versões

Processos e ferramentas que rastreiam mudanças, como alterações no código-fonte em um repositório.

emparelhamento da VPC

Uma conexão entre duas VPCs que permite rotear o tráfego usando endereços IP privados. Para ter mais informações, consulte O que é emparelhamento de VPC? na documentação da Amazon VPC.

Vulnerabilidade

Uma falha de software ou hardware que compromete a segurança do sistema.

W

cache quente

Um cache de buffer que contém dados atuais e relevantes que são acessados com frequência. A instância do banco de dados pode ler do cache do buffer, o que é mais rápido do que ler da memória principal ou do disco.

dados mornos

Dados acessados raramente. Ao consultar esse tipo de dados, consultas moderadamente lentas geralmente são aceitáveis.

função de janela

Uma função SQL que executa um cálculo em um grupo de linhas que se relacionam de alguma forma com o registro atual. As funções de janela são úteis para processar tarefas, como calcular uma média móvel ou acessar o valor das linhas com base na posição relativa da linha atual.

workload

Uma coleção de códigos e recursos que geram valor empresarial, como uma aplicação voltada para o cliente ou um processo de backend.

workstreams

Grupos funcionais em um projeto de migração que são responsáveis por um conjunto específico de tarefas. Cada workstream é independente, mas oferece suporte aos outros workstreams do projeto. Por exemplo, o workstream de portfólio é responsável por priorizar aplicações, planejar ondas e coletar metadados de migração. O workstream de portfólio entrega esses ativos ao workstream de migração, que então migra os servidores e as aplicações.

WORM

Veja gravação única e várias leituras.

WQF

Veja AWS Workload Qualification Framework.

gravação única e várias leituras (WORM)

Um modelo de armazenamento que grava dados uma única vez e evita que os dados sejam excluídos ou modificados. Os usuários autorizados podem ler os dados quantas vezes forem necessárias, mas não podem alterá-los. Essa infraestrutura de armazenamento de dados é considerada imutável.

Z

exploração de dia zero

Um ataque, normalmente malware, que tira proveito de uma vulnerabilidade zero-day.

vulnerabilidade de dia zero

Uma falha ou vulnerabilidade não mitigada em um sistema de produção. Os agentes de ameaças podem usar esse tipo de vulnerabilidade para atacar o sistema. Os desenvolvedores frequentemente ficam cientes da vulnerabilidade como resultado do ataque.

prompt zero shot

Fornecer a um LLM instruções para realizar uma tarefa, mas sem exemplos (shots) que possam ajudar a orientá-lo. O LLM deve usar seu conhecimento pré-treinado para lidar com a tarefa. A

eficácia dos prompts zero-shot depende da complexidade da tarefa e da qualidade do prompt. Veja também prompts few-shot.

aplicação zumbi

Uma aplicação que tem um uso médio de CPU e memória inferior a 5%. Em um projeto de migração, é comum retirar essas aplicações.

As traduções são geradas por tradução automática. Em caso de conflito entre o conteúdo da tradução e da versão original em inglês, a versão em inglês prevalecerá.