Mejores prácticas para ajustar el rendimiento de AWS Glue los trabajos de Apache Spark

: Mejores prácticas para ajustar el rendimiento de AWS Glue los trabajos de Apache Spark

Table of Contents

Mejores prácticas para ajustar el rendimiento de AWS Glue los trabajos de Apache Spark

Roman Myers, Takashi Onikura y Noritaka Sekiyama, Amazon Web Services (AWS)

Diciembre de 2023 (historial de documentos)

AWS Glue ofrece diferentes opciones para ajustar el rendimiento. Esta guía define los temas clave para el ajuste AWS Glue de Apache Spark. Luego, proporciona una estrategia básica que puede seguir al ajustarlos AWS Glue para los trabajos de Apache Spark. Utilice esta guía para aprender a identificar los problemas de rendimiento mediante la interpretación de las métricas disponibles en AWS Glue. Luego, incorpore estrategias para abordar estos problemas, maximizando el rendimiento y minimizando los costos.

En esta guía se cubren las siguientes prácticas de ajuste:

- Escalado de la capacidad del clúster
- Utilice la última AWS Glue versión
- Reducción de la cantidad de análisis de datos
- Paralelización de las tareas
- Minimización de la sobrecarga de planificación
- Optimización de las mezclas
- Optimización de las funciones definidas por el usuario

Temas clave de Apache Spark

En esta sección se explican los conceptos básicos y los temas clave de Apache Spark para ajustar el rendimiento de AWS Glue para Apache Spark. Es importante entender estos conceptos y temas antes de analizar las estrategias de ajuste del mundo real.

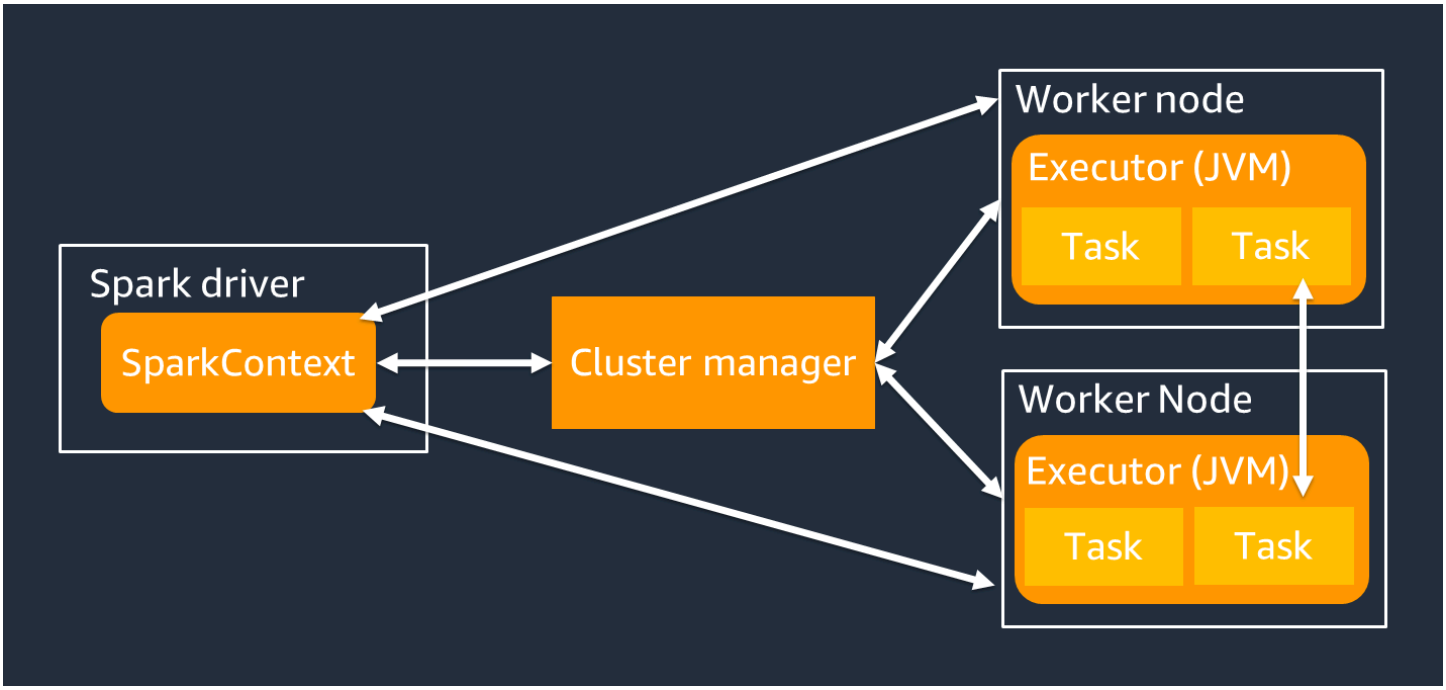
Arquitectura

El controlador de Spark es el principal responsable de dividir su aplicación de Spark en tareas que puedan llevar a cabo nodos de trabajo individuales. El controlador de Spark tiene las siguientes responsabilidades:

- Ejecutar `main()` en su código.
- Generar planes de ejecución.
- Aprovisionar los ejecutores de Spark junto con el administrador de clústeres, que administra los recursos del clúster.
- Programar tareas y solicitar tareas para los ejecutores de Spark.
- Administrar el progreso y la recuperación de las tareas.

Utiliza un objeto `SparkContext` para interactuar con el controlador de Spark durante la ejecución del trabajo.

Un ejecutor de Spark es un nodo de trabajo que se encarga de almacenar datos y ejecutar tareas que se transfieren desde el controlador de Spark. El número de ejecutores de Spark aumentará y disminuirá con el tamaño del clúster.

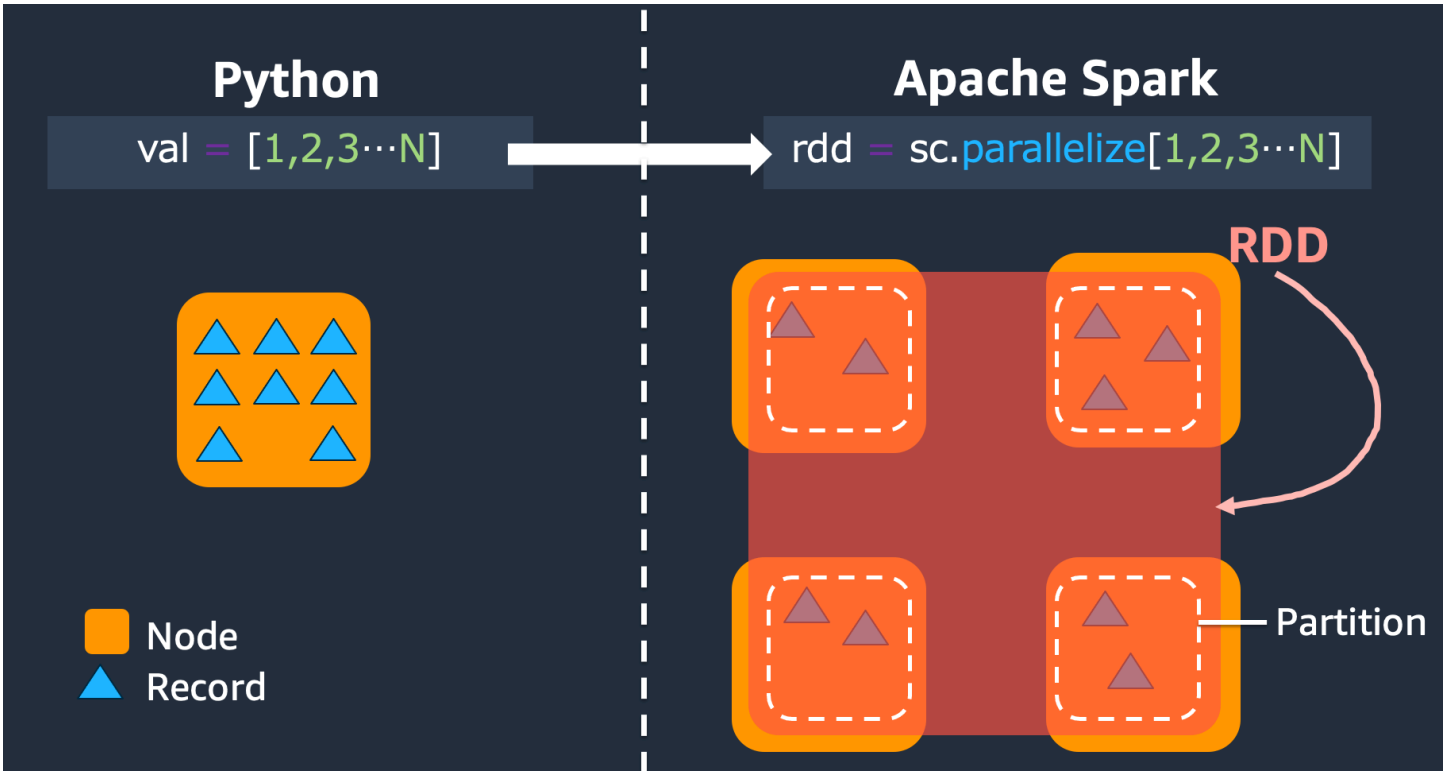


> **Note**
>
> Un ejecutor de Spark tiene múltiples ranuras para procesar múltiples tareas en paralelo. De forma predeterminada, Spark admite una tarea para cada núcleo de CPU virtual (vCPU). Por ejemplo, si un ejecutor tiene cuatro núcleos de CPU, puede ejecutar cuatro tareas simultáneas.

Conjunto de datos distribuido resiliente

Spark lleva a cabo el complejo trabajo de almacenar y rastrear grandes conjuntos de datos entre los ejecutores de Spark. Cuando escribe código para los trabajos de Spark, no tiene que pensar en los detalles del almacenamiento. Spark proporciona la abstracción de conjuntos de datos distribuidos resilientes (RDD), que es una colección de elementos que se pueden operar en paralelo y que se pueden dividir entre los ejecutores de Spark del clúster.

En la siguiente figura se muestra la diferencia en la forma de almacenar datos en memoria cuando se ejecuta un script de Python en su entorno típico y cuando se ejecuta en el marco de Spark (PySpark).



- Python: al escribir `val = [1,2,3...N]` en un script de Python se mantienen los datos en memoria en la única máquina en la que se ejecuta el código.
- PySpark: Spark proporciona la estructura de datos de RDD para cargar y procesar datos distribuidos en la memoria de varios ejecutores de Spark. Puede generar un RDD con código como `rdd = sc.parallelize[1,2,3...N]` y Spark puede distribuir y almacenar automáticamente los datos en memoria entre varios ejecutores de Spark.

  En muchos trabajos de AWS Glue, se utilizan RDD a través DynamicFrames de AWS Glue y DataFrames de Spark. Se trata de abstracciones que permiten definir el esquema de datos en un RDD y llevar a cabo tareas de nivel superior con esa información adicional. Como utilizan los RDD internamente, los datos se distribuyen y cargan de forma transparente en varios nodos en el siguiente código:

  - DynamicFrame

```
dyf= glueContext.create_dynamic_frame.from_options(
    's3', {"paths": [ "s3://<YourBucket>/<Prefix>/"]},
    format="parquet",
    transformation_ctx="dyf"
)
```

- DataFrame

```
df = spark.read.format("parquet")
    .load("s3://<YourBucket>/<Prefix>")
```

Un RDD tiene las siguientes características:

- Los RDD constan de datos divididos en varias partes denominadas particiones. Cada ejecutor de Spark almacena una o más particiones en memoria y los datos se distribuyen entre varios ejecutores.
- Los RDD son inmutables, lo que significa que no se pueden cambiar una vez creados. Para cambiar un DataFrame, puede usar transformaciones, que se definen en la siguiente sección.
- Los RDD replican los datos en los nodos disponibles, por lo que pueden recuperarse automáticamente de los errores de los nodos.

Evaluación lenta

Los RDD admiten dos tipos de operaciones: las transformaciones, que crean un nuevo conjunto de datos a partir de uno existente, y las acciones, que devuelven un valor al programa del controlador tras ejecutar un cálculo en el conjunto de datos.

- Transformaciones: como los RDD son inmutables, solo puede cambiarlos mediante una transformación.

  Por ejemplo, `map` es una transformación que pasa cada elemento del conjunto de datos a través de una función y devuelve un nuevo RDD que representa los resultados. Observe que el método `map` no devuelve ninguna salida. Spark almacena la transformación abstracta para el futuro, en lugar de permitirte interactuar con el resultado. Spark no actuará en las transformaciones hasta que llame a una acción.
- Acciones: al usar las transformaciones, crea su plan de transformaciones lógicas. Para iniciar el cálculo, ejecute una acción como `write`, `count`, `show` o `collect`.

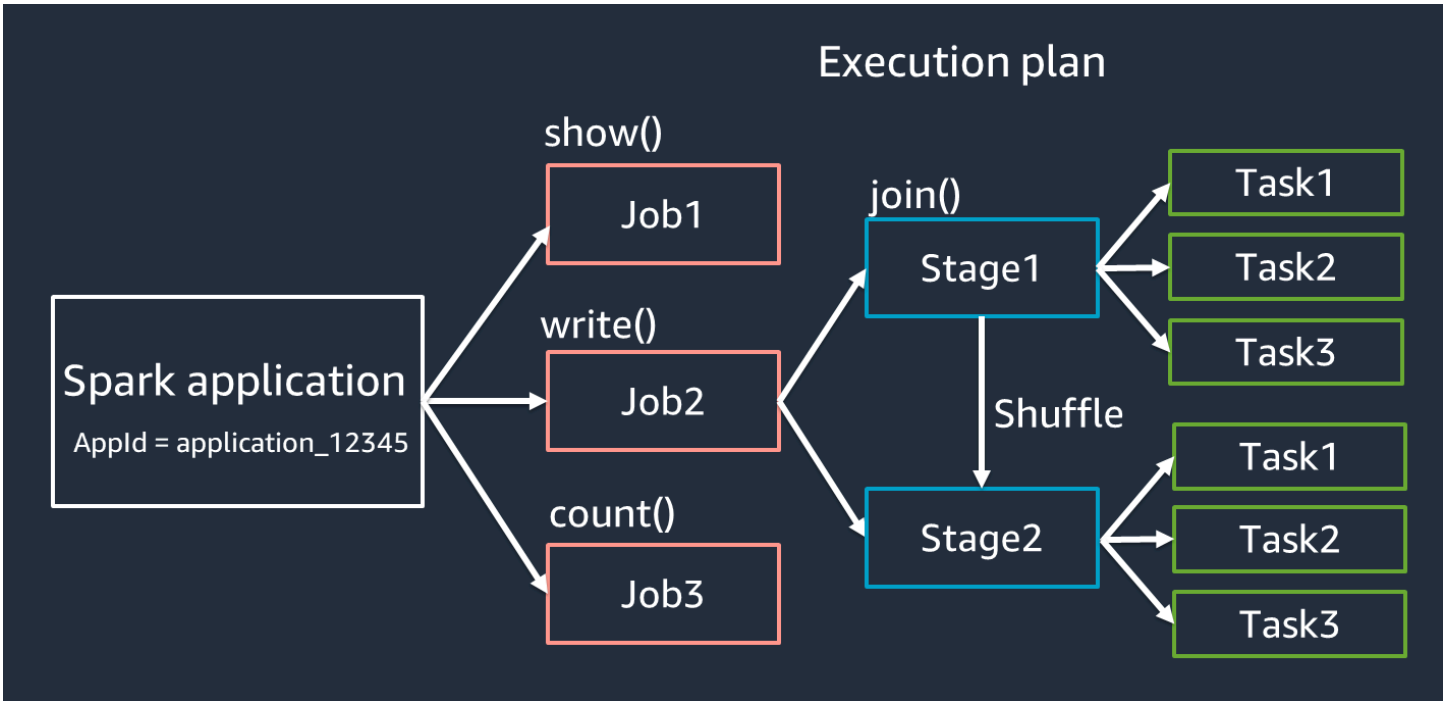
  Todas las transformaciones en Spark son lentas, en el sentido de que no calculan sus resultados de forma inmediata. En su lugar, Spark recuerda una serie de transformaciones aplicadas a algún conjunto de datos base, como objetos de Amazon Simple Storage Service (Amazon S3). Las transformaciones se calculan solo cuando una acción requiere que se devuelva un resultado

al controlador. Este diseño permite que Spark se ejecute de forma más eficiente. Por ejemplo, consideremos la situación en la que un conjunto de datos creado mediante la transformación `map` solo lo consume una transformación que reduce sustancialmente el número de filas, como `reduce`. A continuación, puede pasar el conjunto de datos más pequeño que se ha sometido a ambas transformaciones al controlador, en lugar de pasar el conjunto de datos asignado más grande.

Terminología de las aplicaciones de Spark

En esta sección se trata la terminología de las aplicaciones de Spark. El controlador de Spark crea un plan de ejecución y controla el comportamiento de las aplicaciones en varias abstracciones. Los siguientes términos son importantes para el desarrollo, la depuración y el ajuste del rendimiento con la IU de Spark.

- Aplicación: se basa en una sesión de Spark (contexto de Spark). Se identifica mediante un ID único, como `<application_XXX>`.
- Trabajos: se basan en las acciones creadas para un RDD. Un trabajo consta de una o más etapas.
- Etapas: se basan en las mezclas creadas para un RDD. Una etapa consta de una o más tareas. La mezcla es el mecanismo de Spark para redistribuir los datos de forma que se agrupen de forma diferente en las particiones de RDD. Algunas transformaciones, como `join()`, requieren una mezcla. Las mezclas se analizan con más detalle en la práctica de ajuste Optimización de las mezclas.
- Tareas: una tarea es la unidad mínima de procesamiento programada por Spark. Las tareas se crean para cada partición de RDD y el número de tareas es el número máximo de ejecuciones simultáneas de la etapa.

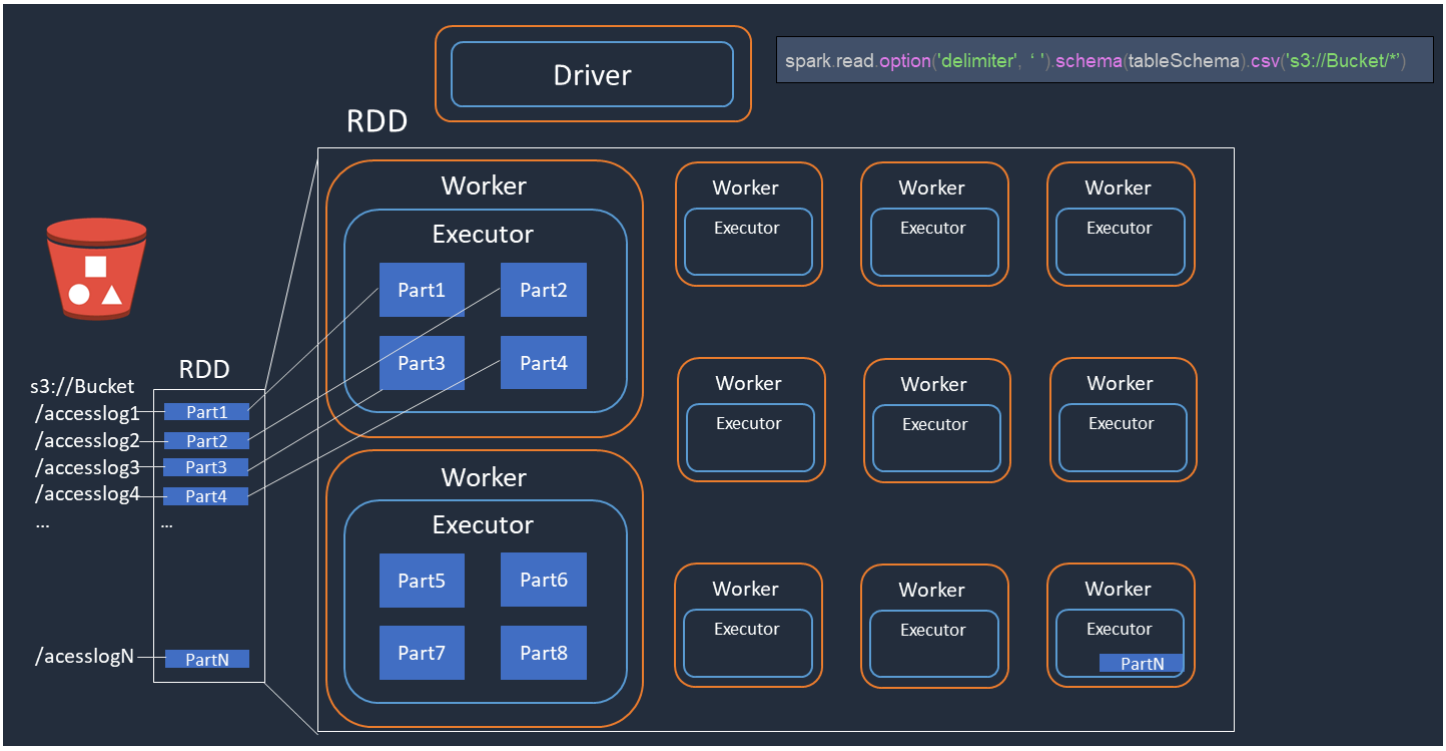


> **Note**
>
> Las tareas son lo más importante que se debe tener en cuenta a la hora de optimizar el paralelismo. El número de tareas escala con el número de RDD

Paralelismo

Spark paraleliza las tareas de carga y transformación de datos.

Considere un ejemplo en el que lleva a cabo un procesamiento distribuido de los archivos de registros de acceso (denominados `accesslog1 ... accesslogN`) en Amazon S3. En el siguiente diagrama se muestra el flujo de procesamiento distribuido.

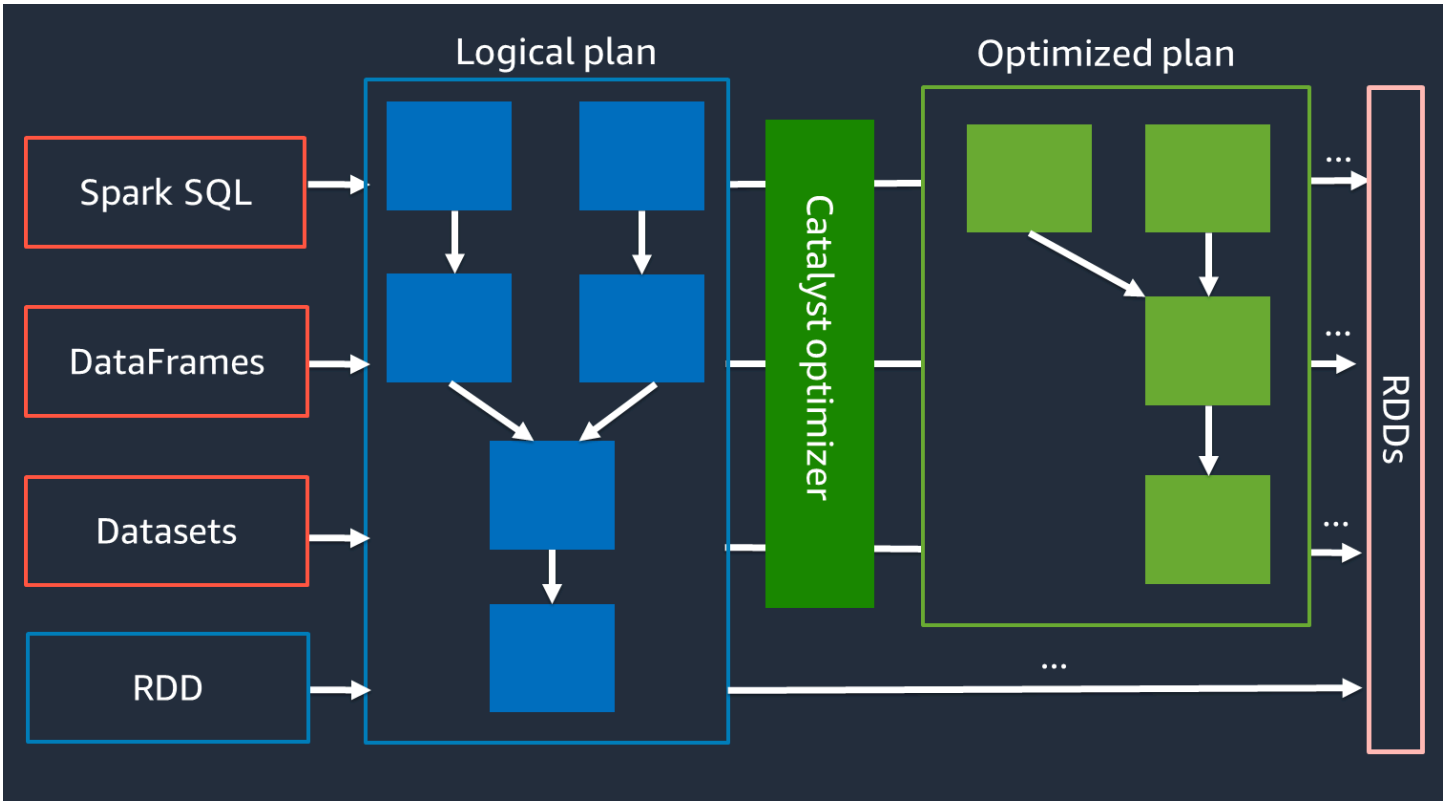


1. El controlador de Spark crea un plan de ejecución para el procesamiento distribuido entre muchos ejecutores de Spark.
2. El controlador de Spark asigna tareas a cada ejecutor en función del plan de ejecución. De forma predeterminada, el controlador de Spark crea particiones de RDD (cada una de las cuales corresponde a una tarea de Spark) para cada objeto de S3 (`Part1 ... N`). A continuación, el controlador de Spark asigna tareas a cada ejecutor.
3. Cada tarea de Spark descarga su objeto de S3 asignado y lo almacena en memoria en la partición de RDD. De esta forma, varios ejecutores de Spark descargan y procesan la tarea asignada en paralelo.

Para obtener más información sobre el número inicial de particiones y la optimización, consulte la sección Paralelización de las tareas.

Optimizador Catalyst

Internamente, Spark usa un motor llamado optimizador Catalyst para optimizar los planes de ejecución. Catalyst tiene un optimizador de consultas que puede usar al ejecutar API de Spark de alto nivel, como Spark SQL, DataFrame y Datasets, como se describe en el siguiente diagrama.



Como el optimizador Catalyst no funciona directamente con la API de RDD, las API de alto nivel suelen ser más rápidas que la API de RDD de bajo nivel. En el caso de uniones complejas, el optimizador Catalyst puede mejorar considerablemente el rendimiento al optimizar el plan de ejecución del trabajo. Puede ver el plan optimizado de su trabajo de Spark en la pestaña SQL de la IU de Spark.

Ejecución adaptativa de consultas

El optimizador Catalyst optimiza el tiempo de ejecución mediante un proceso denominado ejecución adaptativa de consultas. La ejecución adaptativa de consultas utiliza estadísticas de tiempo de ejecución para volver a optimizar el plan de ejecución de las consultas mientras el trabajo está en ejecución. La ejecución adaptativa de consultas ofrece varias soluciones a los desafíos de rendimiento, como la fusión de particiones posteriores a la mezcla, la conversión de uniones de ordenación/combinación en uniones de transmisión y la optimización de uniones sesgadas, tal y como se describe en las siguientes secciones.

La ejecución adaptativa de consultas está disponible en la versión 3.0 y posteriores de AWS Glue y está habilitada de forma predeterminada en la versión 4.0 (Spark 3.3.0) y posteriores de AWS Glue. La ejecución adaptativa de consultas se puede activar y desactivar utilizando `spark.conf.set("spark.sql.adaptive.enabled", "true")` en el código.

Fusión de particiones posteriores a la mezcla

Esta característica reduce las particiones de RDD (fusión) después de cada mezcla en función de las estadísticas de salida de `map`. Simplifica el ajuste del número de particiones de mezcla al ejecutar consultas. No tiene que establecer un número de particiones de mezcla para que se ajuste a su conjunto de datos. Spark puede elegir el número de particiones de mezcla adecuado en tiempo de ejecución una vez que el número inicial de particiones de mezcla sea lo suficientemente grande.

La fusión de particiones posteriores a la mezcla se habilita cuando `spark.sql.adaptive.enabled` y `spark.sql.adaptive.coalescePartitions.enabled` se establecen en verdadero. Para obtener más información, consulte la documentación de Apache Spark.

Conversión de uniones de ordenación/combinación en uniones de transmisión

Esta característica reconoce cuando se unen dos conjuntos de datos de un tamaño sustancialmente diferente y adopta un algoritmo de unión más eficiente según esa información. Para obtener más información, consulte la documentación de Apache Spark. Las estrategias de unión se analizan en la sección Optimización de las mezclas.

Optimización de uniones sesgadas

El sesgo de datos es uno de los cuellos de botella más habituales para los trabajos de Spark. Describe una situación en la que los datos están sesgados hacia particiones de RDD específicas (y, en consecuencia, hacia tareas específicas), lo que retrasa el tiempo total de procesamiento de la aplicación. A menudo, esto puede reducir el rendimiento de las operaciones de unión. La característica de optimización de uniones sesgadas gestiona de forma dinámica el sesgo en las uniones de ordenación/combinación al dividir (y replicar si es necesario) las tareas sesgadas en tareas con un tamaño aproximadamente uniforme.

Esta característica se habilita cuando `spark.sql.adaptive.skewJoin.enabled` se establece en verdadero. Para obtener más información, consulte la documentación de Apache Spark. El sesgo de datos se analiza con más detalle en la sección Optimización de las mezclas.

Investigación de problemas de rendimiento mediante la IU de Spark

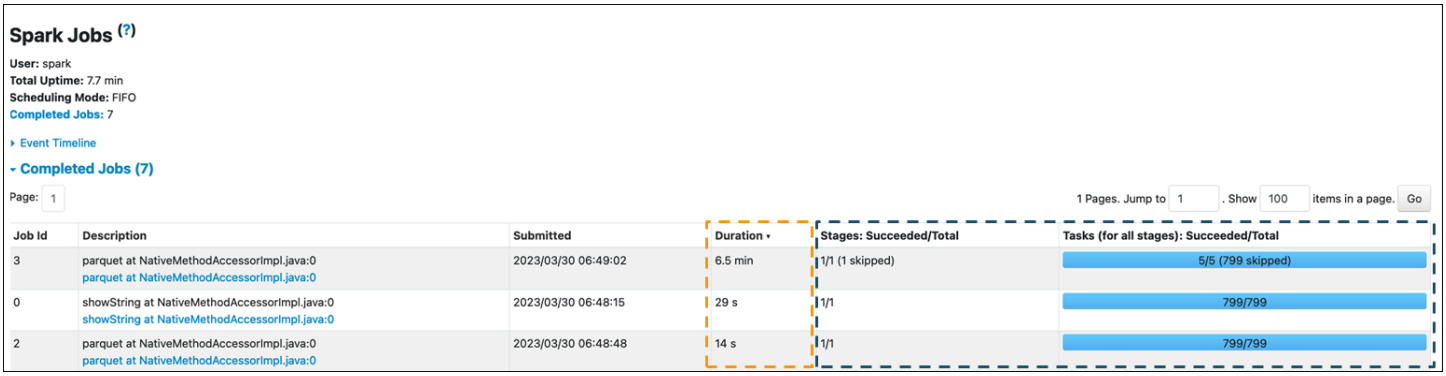
Antes de aplicar las prácticas recomendadas para ajustar el rendimiento de sus trabajos de AWS Glue, le recomendamos encarecidamente que perfile el rendimiento e identifique los cuellos de botella. Esto lo ayudará a centrarse en los aspectos correctos.

Para un análisis rápido, las métricas de Amazon CloudWatch proporcionan una vista básica de las métricas de sus trabajos. La IU de Spark proporciona una visión más detallada para ajustar el rendimiento. Para usar la IU de Spark con AWS Glue, debe habilitarla para sus trabajos de AWS Glue. Una vez que se familiarice con la IU de Spark, siga las estrategias para ajustar el rendimiento de los trabajos de Spark a fin de identificar y reducir el impacto de los cuellos de botella en función de sus resultados.

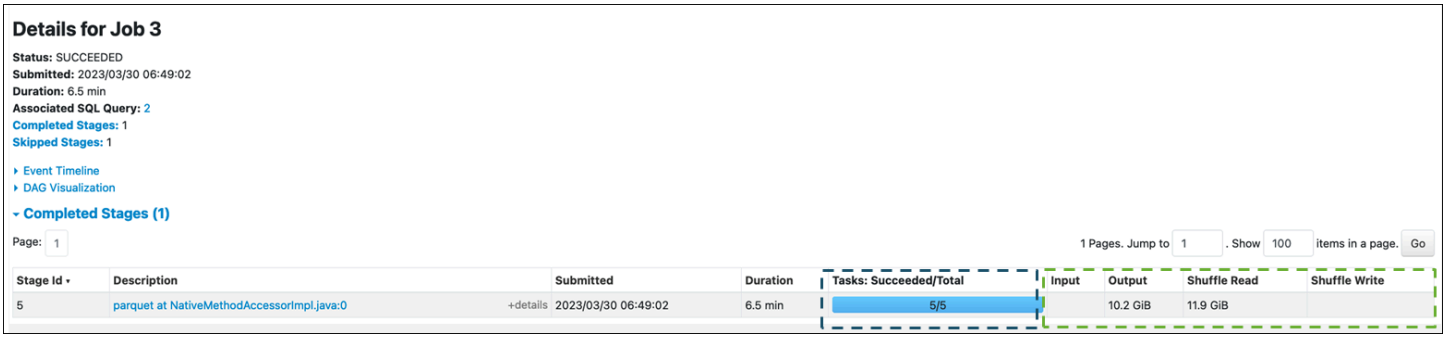
Identificación de cuellos de botella mediante la IU de Spark

Al abrir la IU de Spark, las aplicaciones de Spark aparecen en una tabla. De forma predeterminada, el nombre de la aplicación de un trabajo de AWS Glue es `nativespark-<Job Name>-<Job Run ID>`. Elija la aplicación de Spark de destino en función del ID de ejecución del trabajo para abrir la pestaña Trabajos. Las ejecuciones de trabajos incompletas, como las ejecuciones de trabajos de transmisión, aparecen en Mostrar solicitudes incompletas.

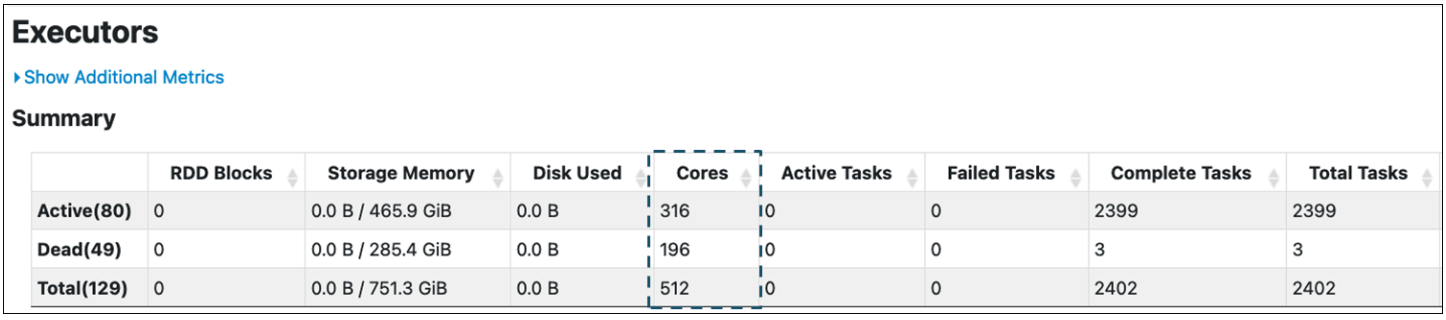
En la pestaña Trabajos se muestra un resumen de todos los trabajos de la aplicación de Spark. Para determinar los errores de alguna etapa o tarea, compruebe el número total de tareas. Para encontrar los cuellos de botella, elija Duración para ordenarlos. Consulte los detalles de los trabajos de larga duración; para ello, seleccione el enlace que se muestra en la columna Descripción.



En la página Detalles del trabajo se muestran las etapas. En esta página, puede ver información general, como la duración, el número de tareas completadas correctamente y el total, el número de entradas y salidas, y la cantidad de lecturas y escrituras de mezclas.



En la pestaña Ejecutor se muestra en detalle la capacidad de los clústeres de Spark. Puede consultar el número total de núcleos. El clúster que se muestra en la siguiente captura de pantalla contiene 316 núcleos activos y 512 núcleos en total. De forma predeterminada, cada núcleo puede procesar una tarea de Spark al mismo tiempo.



	RDD Blocks	Storage Memory	Disk Used	Cores	Active Tasks	Failed Tasks	Complete Tasks	Total Tasks
Active(80)	0	0.0 B / 465.9 GiB	0.0 B	316	0	0	2399	2399
Dead(49)	0	0.0 B / 285.4 GiB	0.0 B	196	0	0	3	3
Total(129)	0	0.0 B / 751.3 GiB	0.0 B	512	0	0	2402	2402

Según el valor 5/5 que se muestra en la página Detalles del trabajo, la etapa 5 es la más larga, pero solo utiliza 5 núcleos de un total de 512. Como el paralelismo de esta etapa es muy bajo, pero lleva mucho tiempo, puede identificarlo como un cuello de botella. Para mejorar el rendimiento, debe entender por qué. Para obtener más información sobre cómo reconocer y reducir el impacto de los cuellos de botella de rendimiento habituales, consulte Estrategias para ajustar el rendimiento de los trabajos de Spark.

Estrategias para ajustar el rendimiento de los trabajos de Spark

Cuando se prepare para ajustar los parámetros, siga las siguientes prácticas recomendadas:

- Determine sus objetivos de rendimiento antes de empezar a identificar los problemas.
- Utilice las métricas para identificar los problemas antes de intentar cambiar los parámetros de ajuste.

Para obtener resultados más consistentes al afinar un trabajo, desarrolle una estrategia básica para su trabajo de ajuste.

Estrategia básica para ajustar el rendimiento

Por lo general, el ajuste del rendimiento se realiza en el siguiente flujo de trabajo:

1. Determine los objetivos de rendimiento.
2. Mida las métricas.
3. Identifique los cuellos de botella.
4. Reduzca el impacto de los cuellos de botella.
5. Repita los pasos 2 a 4 hasta alcanzar el objetivo deseado.

En primer lugar, determine sus objetivos de rendimiento. Por ejemplo, uno de tus objetivos podría ser completar la ejecución de un AWS Glue trabajo en un plazo de 3 horas. Después de definir los objetivos, mida las métricas de rendimiento de los trabajos. Identifique las tendencias en las métricas y los cuellos de botella para cumplir con los objetivos. En particular, identificar los cuellos de botella es lo más importante para la solución de problemas, la depuración y el ajuste del rendimiento. Durante la ejecución de una aplicación de Spark, Spark registra el estado y las estadísticas de cada tarea en el registro de eventos de Spark.

En AWS Glue, puedes ver las métricas de Spark a través de la interfaz web de Spark que proporciona el servidor de historial de Spark. AWS Glue para Spark, los trabajos pueden enviar los registros de eventos de Spark a la ubicación que especifique en Amazon S3. AWS Glue también

proporciona una AWS CloudFormation plantilla de ejemplo y un Dockerfile para iniciar el servidor de historial de Spark en una instancia de Amazon EC2 o en tu ordenador local, de forma que puedas usar la interfaz de usuario de Spark con los registros de eventos.

Una vez que determine sus objetivos de rendimiento e identifique las métricas para evaluarlos, puede empezar a identificar y corregir los cuellos de botella mediante las estrategias que se describen en las siguientes secciones.

Prácticas para ajustar el rendimiento de los trabajos de Spark

Puedes usar las siguientes estrategias para ajustar AWS Glue el rendimiento de los trabajos de Spark:

- AWS Glue recursos:
  - Escalado de la capacidad del clúster
  - Utilice la versión más reciente AWS Glue
- Aplicaciones de Spark:
  - Reducción de la cantidad de análisis de datos
  - Paralelización de las tareas
  - Optimización de las mezclas
  - Minimización de la sobrecarga de planificación
  - Optimización de las funciones definidas por el usuario

Antes de usar estas estrategias, debe tener acceso a las métricas y la configuración de su trabajo de Spark. Puede encontrar esta información en la documentación de AWS Glue.

Desde la perspectiva de los AWS Glue recursos, puede lograr mejoras en el rendimiento añadiendo AWS Glue trabajadores y utilizando la última AWS Glue versión.

Desde la perspectiva de las aplicaciones de Apache Spark, tiene acceso a varias estrategias que pueden mejorar el rendimiento. Si se cargan datos innecesarios en el clúster de Spark, puede eliminarlos para reducir la cantidad de datos cargados. Si ha infrautilizado los recursos del clúster de Spark y tiene pocos datos de E/S, puede identificar tareas que paralelizar. También es posible que desee optimizar las operaciones de transferencia de datos pesados, como las uniones, si requieren mucho tiempo. También puede optimizar el plan de consultas del trabajo o reducir la complejidad computacional de las tareas individuales de Spark.

Para aplicar estas estrategias de manera eficiente, debe consultar las métricas para identificar cuándo son aplicables. Para obtener más detalles, consulte las siguientes secciones. Estas técnicas funcionan no solo para ajustar el rendimiento, sino también para resolver problemas típicos, como los errores out-of-memory (OOM).

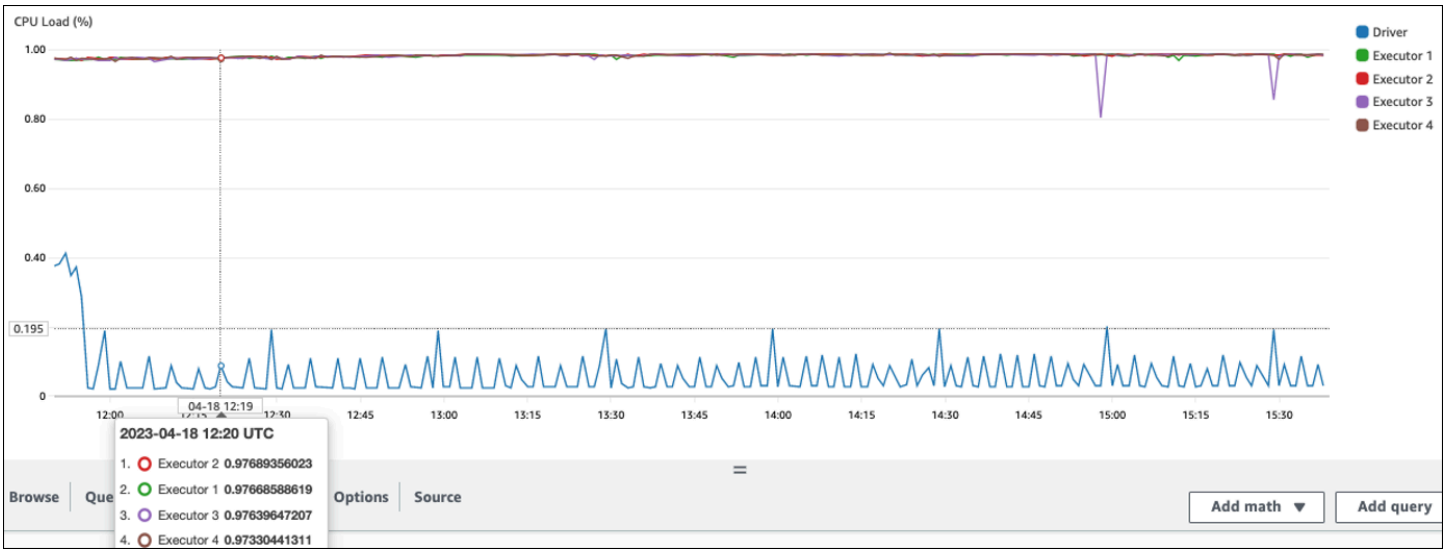
Escalado de la capacidad del clúster

Si el trabajo tarda demasiado tiempo, pero los ejecutores consumen suficientes recursos y Spark crea un gran volumen de tareas en relación con los núcleos disponibles, considere la posibilidad de escalar la capacidad del clúster. Para evaluar si esto es apropiado, use las siguientes métricas.

CloudWatch métricas

- Consulte Carga de la CPU y Utilización de la memoria para determinar si los ejecutores consumen recursos suficientes.
- Compruebe cuánto tiempo se ha ejecutado el trabajo para evaluar si el tiempo de procesamiento es demasiado largo para cumplir con sus objetivos de rendimiento.

En el siguiente ejemplo, cuatro ejecutores se ejecutan con una carga de CPU superior al 97 %, pero el procesamiento no se ha completado después de unas tres horas.



> **Note**
>
> Si la carga de la CPU es baja, probablemente no se beneficie del escalado de la capacidad del clúster.

UI de Spark

En las pestañas Trabajo o Etapa, puede ver el número de tareas de cada trabajo o etapa. En el siguiente ejemplo, Spark ha creado `58100` tareas.

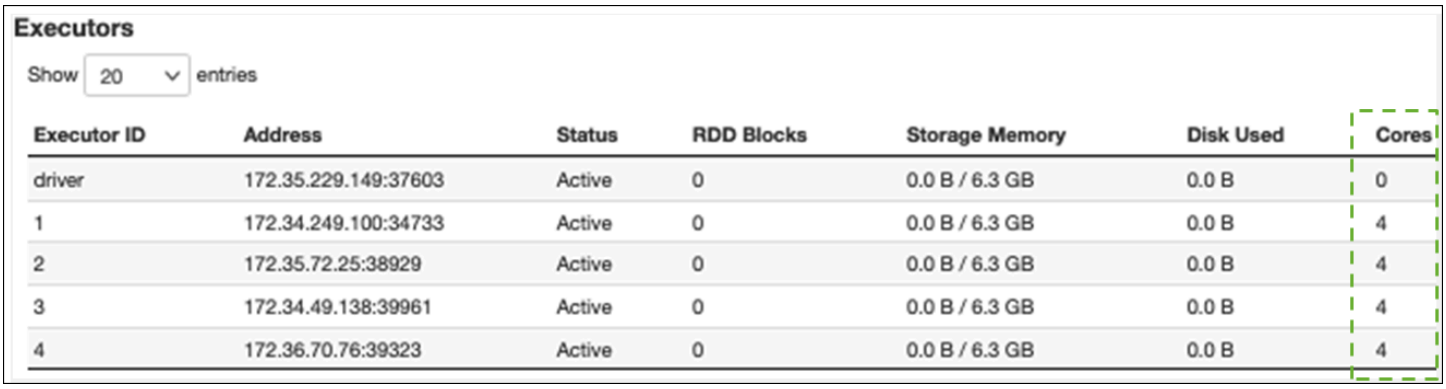
Stages for All Jobs

Completed Stages: 1

Completed Stages (1)

Stage Id	Description	Submitted	Duration	Tasks: Succeeded/Total	Input
0	count at DynamicFrame.scala:1414 +details	2023/04/18 10:59:10	4.8 h	58100/58100	28.4 GB

En la pestaña Ejecutor, puede ver el número total de ejecutores y tareas. En la siguiente captura de pantalla, cada ejecutor de Spark tiene cuatro núcleos y puede llevar a cabo cuatro tareas simultáneamente.



Executors

Show 20 entries

Executor ID	Address	Status	RDD Blocks	Storage Memory	Disk Used	Cores
driver	172.35.229.149:37603	Active	0	0.0 B / 6.3 GB	0.0 B	0
1	172.34.249.100:34733	Active	0	0.0 B / 6.3 GB	0.0 B	4
2	172.35.72.25:38929	Active	0	0.0 B / 6.3 GB	0.0 B	4
3	172.34.49.138:39961	Active	0	0.0 B / 6.3 GB	0.0 B	4
4	172.36.70.76:39323	Active	0	0.0 B / 6.3 GB	0.0 B	4

En este ejemplo, el número de tareas de Spark (`58100`) es mucho mayor que las 16 tareas que los ejecutores pueden procesar simultáneamente (4 ejecutores × 4 núcleos).

Si observa estos síntomas, considere la posibilidad de escalar el clúster. Puede escalar la capacidad del clúster mediante las siguientes opciones:

- Activar AWS Glue Auto Scaling: Auto Scaling está disponible para sus trabajos de AWS Glue extracción, transformación y carga (ETL) y de streaming en la AWS Glue versión 3.0 o posterior. AWS Glue agrega y elimina automáticamente trabajadores del clúster en función del número de

particiones en cada etapa o de la velocidad a la que se generan los microlotes durante la ejecución del trabajo.

Si observa una situación en la que el número de nodos de trabajo no aumenta aunque el escalado automático esté habilitado, considere la posibilidad de agregar nodos de trabajo manualmente. Sin embargo, tenga en cuenta que escalar manualmente para una etapa puede provocar que muchos nodos de trabajo permanezcan inactivos durante las etapas posteriores, lo que podría costar más y no aumentar el rendimiento.

Después de activar Auto Scaling, puede ver el número de ejecutores en las métricas del CloudWatch ejecutor. Utilice las siguientes métricas para supervisar la demanda de ejecutores en las aplicaciones de Spark:

- `glue.driver.ExecutorAllocationManager.executors.numberAllExecutors`
- `glue.driver.ExecutorAllocationManager.executors.numberMaxNeededExecutors`

Para obtener más información sobre las métricas, consulta Cómo monitorizar AWS Glue con CloudWatch las métricas de Amazon.

- Escalado horizontal: aumento del número de nodos de trabajo de AWS Glue : puede aumentar el número de nodos de trabajo de AWS Glue de forma manual. Agregue nodos de trabajo solo hasta que observe nodos de trabajo inactivos. En ese momento, agregar más nodos de trabajo aumentará los costos sin mejorar los resultados. Para obtener más información, consulte Paralelización de las tareas.
- Amplíe: utilice un tipo de trabajador más grande: puede cambiar manualmente el tipo de instancia de sus AWS Glue trabajadores para utilizar trabajadores con más núcleos, memoria y almacenamiento. Los tipos de nodos de trabajo más grandes le permiten escalar verticalmente y ejecutar tareas de integración de datos intensivas, como transformaciones de datos que consumen mucha memoria, agregaciones sesgadas y comprobaciones de detección de entidades con petabytes de datos.

  El escalado vertical también ayuda en los casos en que el controlador de Spark necesita una mayor capacidad, por ejemplo, porque el plan de consultas del trabajo es bastante grande. Para obtener más información sobre los tipos de trabajadores y su rendimiento, consulte la entrada del blog sobre AWS macrodatos: amplíe sus trabajos AWS Glue para Apache Spark con nuevos tipos de trabajadores más grandes, G.4X y G.8X.

El uso de nodos de trabajo más grandes también puede reducir la cantidad total de nodos de trabajo necesarios, lo que aumenta el rendimiento al reducir la mezcla en operaciones intensivas, como las uniones.

Utilice la versión más reciente AWS Glue

Recomendamos utilizar la AWS Glue versión más reciente. Hay varias optimizaciones y actualizaciones integradas en cada versión que pueden mejorar automáticamente el rendimiento de los trabajos. Por ejemplo, la AWS Glue versión 4.0 ofrece las siguientes funciones nuevas:

- Nuevo entorno de ejecución optimizado de Apache Spark 3.3.0: la AWS Glue versión 4.0 se basa en el entorno de ejecución de Apache Spark 3.3.0 y ofrece mejoras de rendimiento comparables a las de Spark de código abierto. El tiempo de ejecución de Spark 3.3.0 se basa en muchas de las innovaciones de Spark 2.x.
- Conector de Amazon Redshift mejorado: la versión 4.0 y posteriores de AWS Glue proporcionan una integración de Amazon Redshift para Apache Spark. La integración se basa en un conector de código abierto existente y lo mejora en términos de rendimiento y seguridad. La integración ayuda a que las aplicaciones funcionen hasta 10 veces más rápido. Para obtener más información, consulte la entrada del blog Amazon Redshift integration with Apache Spark.
- Ejecución basada en SIMD para lecturas vectorizadas con datos CSV y JSON: la AWS Glue versión 3.0 y las versiones posteriores incorporan lectores optimizados que pueden acelerar considerablemente el rendimiento general del trabajo en comparación con los lectores basados en filas. Para obtener más información sobre los datos CSV, consulte Optimice el rendimiento de lectura con el lector CSV SIMD vectorizado. Para obtener más información sobre los datos JSON, consulte Uso del lector SIMD JSON vectorizado con formato en columnas de Apache Arrow.

Cada AWS Glue versión incluirá actualizaciones de este tipo, entre otras muchas, como actualizaciones de conectores, controladores y bibliotecas. Para obtener más información, consulte AWS Glue Versiones y Migración de AWS Glue trabajos a la AWS Glue versión 4.0.

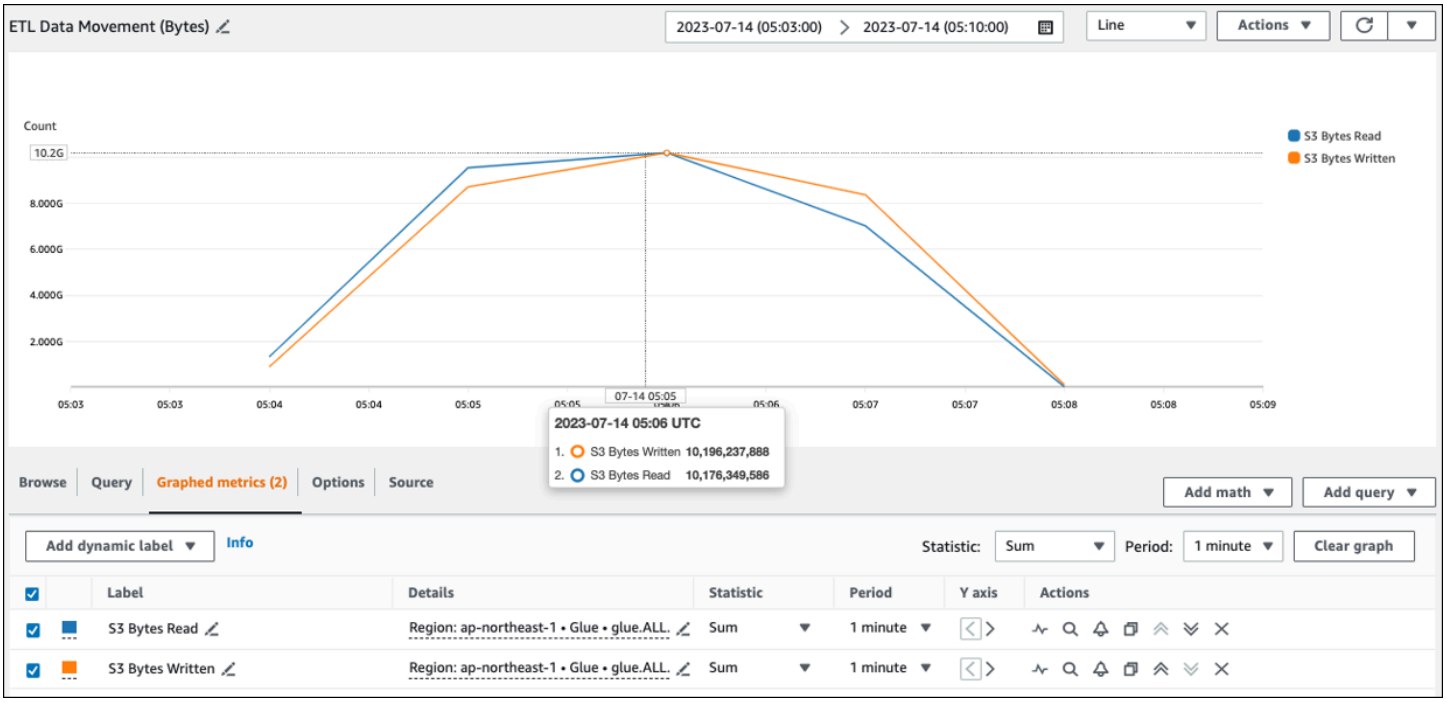
Reducción de la cantidad de análisis de datos

Para empezar, considere la posibilidad de cargar solo los datos que necesite. Para mejorar el rendimiento, puede simplemente reducir la cantidad de datos que se cargan en su clúster de Spark para cada origen de datos. Para evaluar si este enfoque es apropiado, use las siguientes métricas.

Puedes comprobar los bytes leídos de Amazon S3 en CloudWatchlas métricas y obtener más detalles en la interfaz de usuario de Spark, tal y como se describe en la sección de interfaz de usuario de Spark.

CloudWatch métricas

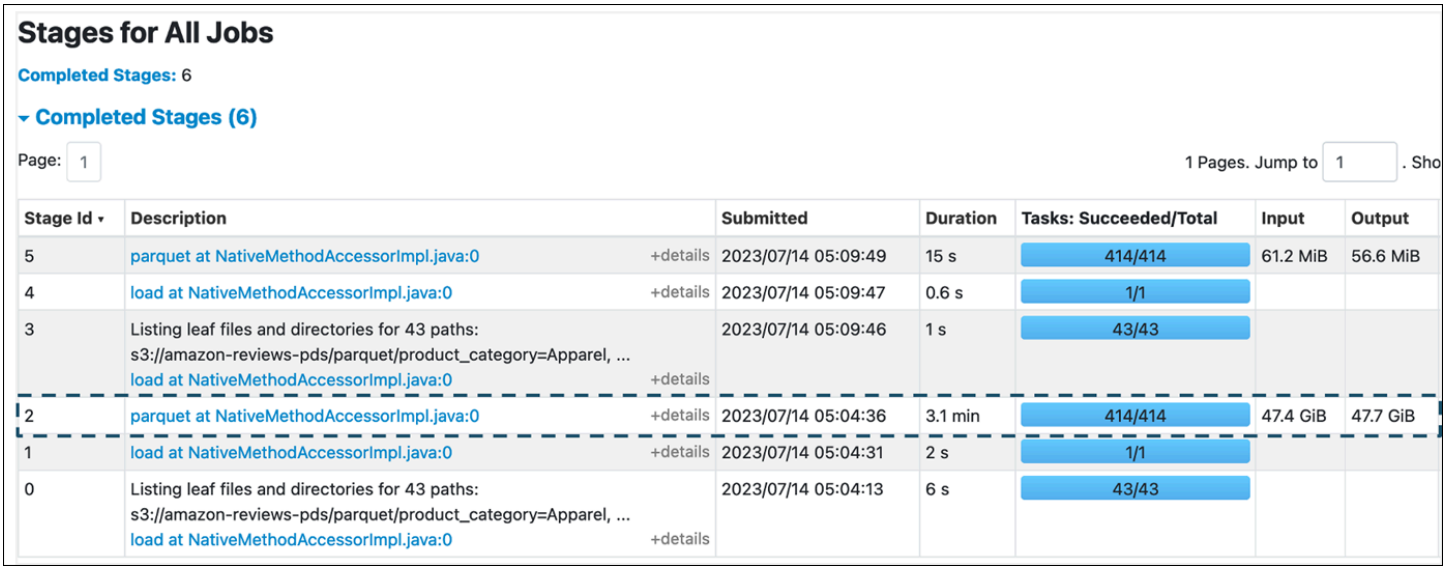
Puede ver el tamaño de lectura aproximado de Amazon S3 en Movimiento de datos de ETL (bytes). Esta métrica muestra el número de bytes que leen de Amazon S3 todos los ejecutores desde el informe anterior. Puede utilizarla para supervisar el movimiento de datos de ETL desde Amazon S3 y comparar las tasas de lectura con las tasas de ingesta de orígenes de datos externos.



Si observa un punto de datos de Lectura de bytes de S3 mayor de lo esperado, considere las siguientes soluciones.

UI de Spark

En la pestaña Stage de la interfaz AWS Glue de usuario de Spark, puedes ver el tamaño de entrada y salida. En el siguiente ejemplo, la etapa 2 lee 47,4 GiB de entrada y 47,7 GiB de salida, mientras que la etapa 5 lee 61,2 MiB de entrada y 56,6 MiB de salida.



Stages for All Jobs

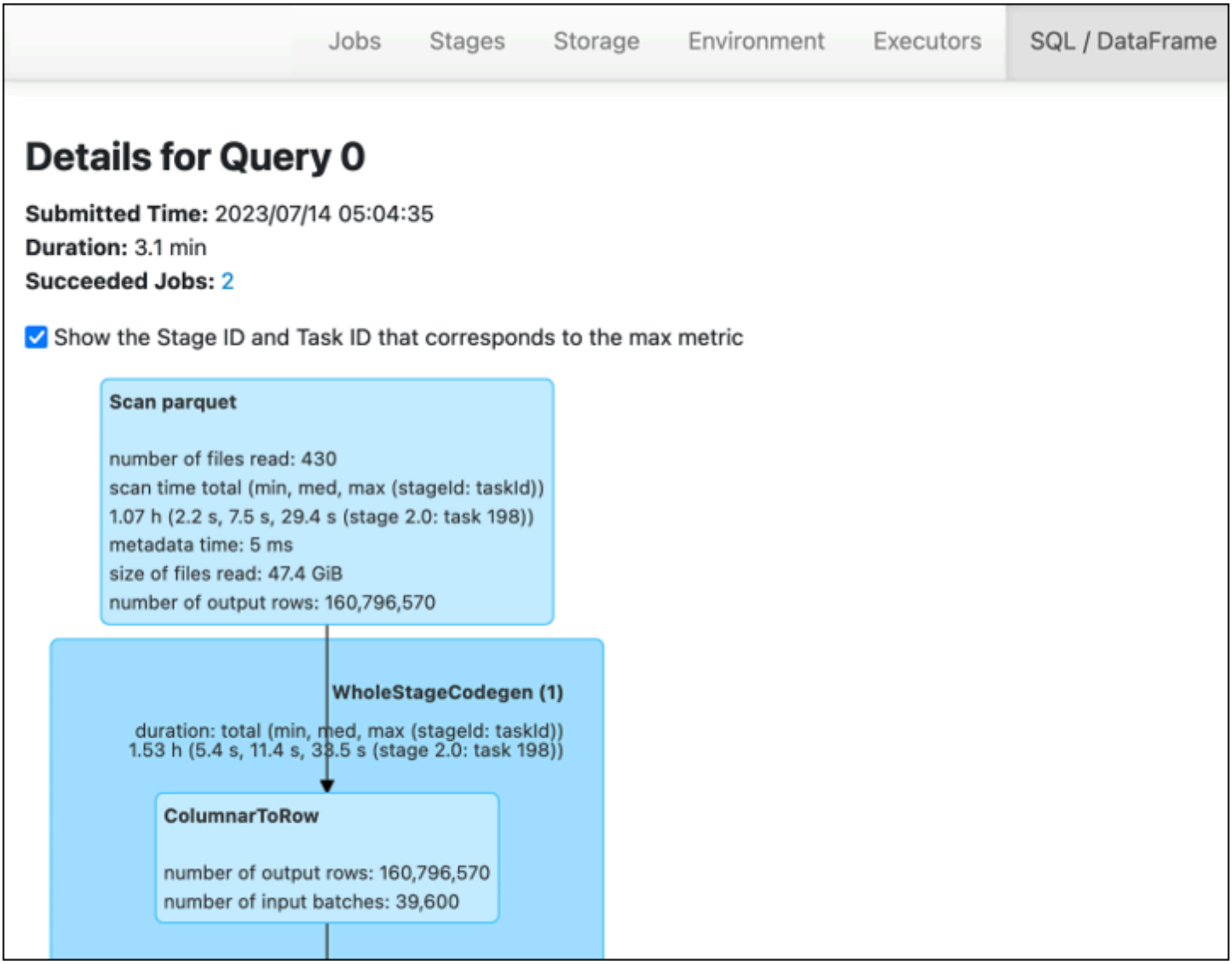
Completed Stages: 6

Completed Stages (6)

Page: 1 — 1 Pages. Jump to 1 . Sho

Stage Id	Description	Submitted	Duration	Tasks: Succeeded/Total	Input	Output
5	parquet at NativeMethodAccessorImpl.java:0 +details	2023/07/14 05:09:49	15 s	414/414	61.2 MiB	56.6 MiB
4	load at NativeMethodAccessorImpl.java:0 +details	2023/07/14 05:09:47	0.6 s	1/1		
3	Listing leaf files and directories for 43 paths: s3://amazon-reviews-pds/parquet/product_category=Apparel, ... load at NativeMethodAccessorImpl.java:0 +details	2023/07/14 05:09:46	1 s	43/43		
2	parquet at NativeMethodAccessorImpl.java:0 +details	2023/07/14 05:04:36	3.1 min	414/414	47.4 GiB	47.7 GiB
1	load at NativeMethodAccessorImpl.java:0 +details	2023/07/14 05:04:31	2 s	1/1		
0	Listing leaf files and directories for 43 paths: s3://amazon-reviews-pds/parquet/product_category=Apparel, ... load at NativeMethodAccessorImpl.java:0 +details	2023/07/14 05:04:13	6 s	43/43		

Cuando utilizas el SQL o los DataFrame enfoques de Spark en tu AWS Glue trabajo, la pestaña DataFrame de SQL /D muestra más estadísticas sobre estas etapas. En este caso, la etapa 2 muestra número de archivos leídos: 430, tamaño de los archivos leídos: 47,4 GiB y número de filas de salida: 160 796 570.



Si observa que hay una diferencia sustancial de tamaño entre los datos que lee y los datos que utiliza, pruebe las siguientes soluciones.

Amazon S3

Para reducir la cantidad de datos que se cargan en su trabajo al leer desde Amazon S3, tenga en cuenta el tamaño, la compresión, el formato y el diseño del archivo (particiones) del conjunto de datos. AWS Glue en el caso de Spark, los trabajos se suelen utilizar para la ETL de datos sin procesar, pero para un procesamiento distribuido eficiente, es necesario inspeccionar las características del formato de la fuente de datos.

- Tamaño del archivo: recomendamos mantener el tamaño de los archivos de entrada y salida dentro de un rango moderado (por ejemplo, 128 MB). Los archivos demasiado pequeños y demasiado grandes pueden causar problemas.

  Un gran número de archivos pequeños provoca los siguientes problemas:

  - Gran I/O carga de red en Amazon S3 debido a la sobrecarga necesaria para realizar solicitudes (como `ListGet`, o`Head`) para muchos objetos (en comparación con unos pocos objetos que almacenan la misma cantidad de datos).
  - El controlador Spark soporta una gran I/O carga de procesamiento, lo que generará muchas particiones y tareas y generará un paralelismo excesivo.

  Por otro lado, si el tipo de archivo no se puede dividir (por ejemplo, gzip) y los archivos son demasiado grandes, la aplicación de Spark debe esperar a que una sola tarea termine de leer todo el archivo.

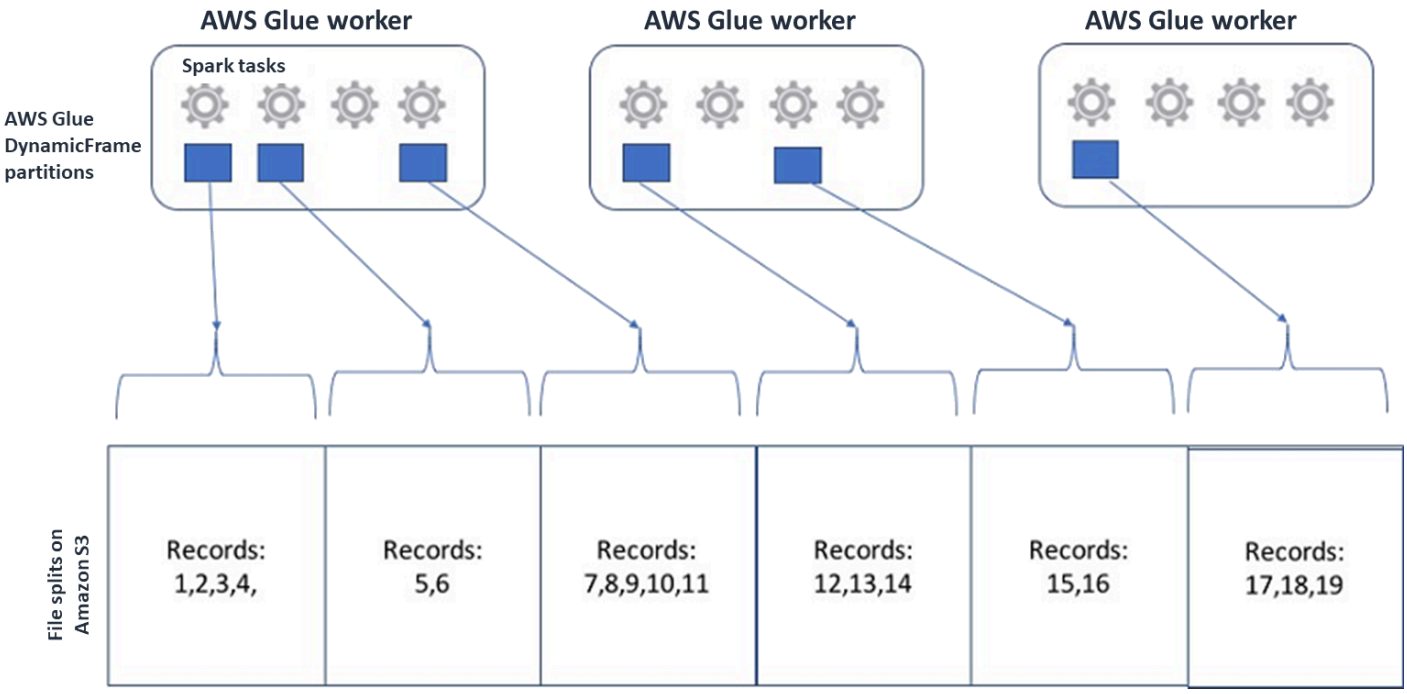
  Para reducir el paralelismo excesivo que se produce al crear una tarea de Apache Spark para cada archivo pequeño, utilice la agrupación de archivos para. DynamicFrames Este enfoque reduce las probabilidades de que se produzca una excepción de OOM por parte del controlador de Spark. Para configurar la agrupación de archivos, defina los parámetros `groupFiles` y `groupSize`. El siguiente ejemplo de código utiliza la AWS Glue DynamicFrame API de un script ETL con estos parámetros.

  ```
  dyf = glueContext.create_dynamic_frame_from_options("s3",
      {'paths': ["s3://input-s3-path/"],
      'recurse':True,
      'groupFiles': 'inPartition',
      'groupSize': '1048576'},
      format="json")
  ```

- Compresión: si los objetos de S3 ocupan cientos de megabytes, considere la posibilidad de comprimirlos. Hay varios formatos de compresión, que se pueden clasificar a grandes rasgos en dos tipos:

  - Los formatos de compresión que no se pueden dividir, como gzip, requieren que un nodo de trabajo descomprima todo el archivo.
  - Los formatos de compresión que se pueden dividir, como bzip2 o LZO (indexados), permiten la descompresión parcial de un archivo, que se puede paralelizar.

En el caso de Spark (y otros motores de procesamiento distribuido habituales), dividirá el archivo de datos de origen en fragmentos que su motor pueda procesar en paralelo. A estas unidades se las suele denominar divisiones. Una vez que los datos estén en un formato separable, los AWS Glue lectores optimizados pueden recuperar las divisiones de un objeto de S3 al ofrecer a la `GetObject` API la `Range` opción de recuperar solo bloques específicos. Tenga en cuenta el siguiente diagrama para ver cómo funcionaría esto en la práctica.



Los datos comprimidos pueden acelerar considerablemente la aplicación, siempre que los archivos tengan un tamaño óptimo o se puedan dividir. Los tamaños de datos más pequeños reducen los datos analizados de Amazon S3 y el tráfico de red de Amazon S3 a su clúster de Spark. Por otro lado, se necesita más CPU para comprimir y descomprimir los datos. La cantidad de computación necesaria escala con la relación de compresión del algoritmo de compresión. Tenga en cuenta esta concesión a la hora de elegir el formato de compresión que se pueda dividir.

> **Note**
>
> Si bien los archivos gzip generalmente no se pueden dividir, puede comprimir bloques de Parquet individuales con gzip y esos bloques se pueden paralelizar.

- Formato de archivo: utilice un formato de columnas. Apache Parquet y Apache ORC son formatos de datos de columnas habituales. Parquet y ORC almacenan los datos de manera eficiente mediante la compresión basada en columnas, codificando y comprimiendo cada columna en función de su tipo de datos. Para obtener más información sobre las codificaciones de Parquet, consulte Parquet encoding definitions. Los archivos de Parquet también se pueden dividir.

  Los formatos de columnas agrupan los valores por columnas y los almacenan juntos en bloques. Cuando utilice formatos de columnas, puede omitir bloques de datos que correspondan a columnas que no tiene previsto utilizar. Las aplicaciones de Spark pueden recuperar solo las columnas que necesita. En general, unos índices de compresión mejores o la omisión de bloques de datos implican leer menos bytes de Amazon S3, lo que se traduce en un mejor rendimiento. Ambos formatos también admiten los siguientes enfoques de inserción para reducir la E/S:

  - Inserción por proyección: la inserción por proyección es una técnica que permite recuperar únicamente las columnas especificadas en la aplicación. Puede especificar columnas en la aplicación de Spark, como se muestra en los siguientes ejemplos:
    - DataFrame ejemplo: `df.select("star_rating")`
    - Ejemplo de Spark SQL: `spark.sql("select start_rating from <table>")`
  - Inserción de predicados: la inserción de predicados es una técnica para procesar cláusulas `WHERE` y `GROUP BY` de forma eficiente. Ambos formatos tienen bloques de datos que representan valores de columnas. Cada bloque contiene las estadísticas del bloque, como los valores máximo y mínimo. Spark puede usar estas estadísticas para determinar si el bloque debe leerse u omitirse en función del valor del filtro utilizado en la aplicación. Para utilizar esta característica, agregue más filtros en las condiciones, como se muestra en los siguientes ejemplos:
    - DataFrame ejemplo: `df.select("star_rating").filter("star_rating < 2")`
    - Ejemplo de Spark SQL: `spark.sql("select * from <table> where star_rating < 2")`
- Diseño de archivo: al almacenar los datos de S3 en objetos situados en diferentes rutas en función de cómo se vayan a utilizar los datos, podrá recuperar los datos pertinentes de forma eficiente. Para obtener más información, consulte Organizar objetos con prefijos en la documentación de Amazon S3. AWS Glue admite almacenar claves y valores en prefijos de Amazon S3 en formato `key=value` y los datos se particionan según la ruta de Amazon S3. La partición de los datos le permite restringir el volumen de datos que analiza cada aplicación de análisis posterior, lo que mejora el rendimiento y reduce los costos. Para obtener más información, consulte Administrar particiones para la salida de ETL en AWS Glue.

La partición divide la tabla en diferentes partes y mantiene los datos relacionados en archivos agrupados en función de los valores de las columnas, como el año, el mes y el día, como se muestra en el siguiente ejemplo.

```
# Partitioning by /YYYY/MM/DD
s3://<YourBucket>/year=2023/month=03/day=31/0000.gz
s3://<YourBucket>/year=2023/month=03/day=01/0000.gz
s3://<YourBucket>/year=2023/month=03/day=02/0000.gz
s3://<YourBucket>/year=2023/month=03/day=03/0000.gz
...
```

Si quiere definir particiones para su conjunto de datos, modélelo con una tabla en el AWS Glue Data Catalog. A continuación, puede restringir la cantidad de datos analizados mediante la eliminación de particiones, de la siguiente manera:

- Para AWS Glue DynamicFrame, defina `push_down_predicate` (`ocatalogPartitionPredicate`).

  ```
  dyf = Glue_context.create_dynamic_frame.from_catalog(
      database=src_database_name,
      table_name=src_table_name,
      push_down_predicate = "year='2023' and month ='03'",
  )
  ```

- En el caso de Spark DataFrame, establece una ruta fija para podar las particiones.

  ```
  df = spark.read.format("json").load("s3://<YourBucket>/year=2023/month=03/*/*.gz")
  ```

- En el caso de Spark SQL, puede establecer la cláusula where para eliminar las particiones del Catálogo de datos.

  ```
  df = spark.sql("SELECT * FROM <Table> WHERE year= '2023' and month = '03'")
  ```

- Para particionar por fecha al escribir tus datos AWS Glue, configuras PartitionKeys DynamicFrame en o PartitionBy () con la información de fecha DataFrame en tus columnas de la siguiente manera.

  - DynamicFrame

    ```
    glue_context.write_dynamic_frame_from_options(
    ```

```
    frame= dyf, connection_type='s3',format='parquet'
    connection_options= {
        'partitionKeys': ["year", "month", "day"],
        'path': 's3://<YourBucket>/<Prefix>/'
    }
)
```

- DataFrame

```
df.write.mode('append')\
    .partitionBy('year','month','day')\
    .parquet('s3://<YourBucket>/<Prefix>/')
```

Esto puede aumentar el rendimiento de los consumidores de sus datos de salida.

Si no tiene acceso para modificar la canalización que crea el conjunto de datos de entrada, la creación de particiones no es una opción. En su lugar, puede excluir las rutas de S3 innecesarias mediante patrones glob. Establece exclusiones al leer. DynamicFrame Por ejemplo, el siguiente código excluye los días de los meses del 01 al 09 del año 2023.

```
dyf = glueContext.create_dynamic_frame.from_catalog(
    database=db,
    table_name=table,
    additional_options = { "exclusions":"[\"**year=2023/month=0[1-9]/**\"]" },
    transformation_ctx='dyf'
)
```

También puede establecer exclusiones en las propiedades de la tabla del Catálogo de datos:

- Clave: `exclusions`
- Valor: `["**year=2023/month=0[1-9]/**"]`

- Demasiadas particiones de Amazon S3: evite particionar los datos de Amazon S3 en columnas que contengan una amplia gama de valores, como una columna de ID con miles de valores. Esto puede aumentar considerablemente el número de particiones del bucket, ya que el número de particiones posibles es el producto de todos los campos por los que ha creado particiones. Demasiadas particiones pueden provocar lo siguiente:
  - Aumento de la latencia para recuperar los metadatos de las particiones del Catálogo de datos.
  - Mayor número de archivos pequeños, lo que requiere más solicitudes a la API de Amazon S3 (`List`, `Get` y `Head`).

Por ejemplo, si establece un tipo de fecha en `partitionBy` o `partitionKeys`, la creación de particiones de nivel de fecha, como `yyyy/mm/dd`, es adecuada para muchos casos de uso. Sin embargo, `yyyy/mm/dd/<ID>` podría generar tantas particiones que afectaría negativamente al rendimiento en su conjunto.

Por otro lado, algunos casos de uso, como las aplicaciones de procesamiento en tiempo real, requieren muchas particiones, como `yyyy/mm/dd/hh`. Si su caso de uso requiere muchas particiones, considere la posibilidad de utilizar índices de particiones de AWS Glue para reducir la latencia a la hora de recuperar los metadatos de las particiones del Catálogo de datos.

Bases de datos y JDBC

Para reducir el análisis de datos al recuperar información de una base de datos, puede especificar un predicado (o cláusula) `where` en una consulta SQL. Las bases de datos que no proporcionen una interfaz de SQL ofrecerán su propio mecanismo de consulta o filtrado.

Al utilizar conexiones de Java Database Connectivity (JDBC), proporcione una consulta de selección con la cláusula `where` para los siguientes parámetros:

- Para DynamicFrame, utilice la opción SampleQuery. Al utilizar `create_dynamic_frame.from_catalog`, configure el argumento `additional_options` de la siguiente manera.

```
query = "SELECT * FROM <TableName> where id = 'XX' AND"
datasource0 = glueContext.create_dynamic_frame.from_catalog(
  database = db,
  table_name = table,
  additional_options={
      "sampleQuery": query,
      "hashexpression": key,
      "hashpartitions": 10,
      "enablePartitioningForSampleQuery": True
    },
  transformation_ctx = "datasource0"
)
```

Cuando `using create_dynamic_frame.from_options`, configure el argumento `connection_options` de la siguiente manera.

```
query = "SELECT * FROM <TableName> where id = 'XX' AND"
datasource0 = glueContext.create_dynamic_frame.from_options(
    connection_type = connection,
    connection_options={
        "url": url,
        "user": user,
        "password": password,
        "dbtable": table,
        "sampleQuery": query,
        "hashexpression": key,
        "hashpartitions": 10,
        "enablePartitioningForSampleQuery": True
    }
)
```

- Para DataFrame, utilice la opción de consulta.

```
query = "SELECT * FROM <TableName> where id = 'XX'"
jdbcDF = spark.read \
.format('jdbc') \
.option('url', url) \
.option('user', user) \
.option('password', pwd) \
.option('query', query) \
.load()
```

- En el caso de Amazon Redshift, utilice la AWS Glue versión 4.0 o una versión posterior para aprovechar la compatibilidad con pulsaciones descendentes del conector Amazon Redshift Spark.

```
dyf = glueContext.create_dynamic_frame.from_catalog(
    database = "redshift-dc-database-name",
    table_name = "redshift-table-name",
    redshift_tmp_dir = args["temp-s3-dir"],
    additional_options = {"aws_iam_role": "arn:aws:iam::role-account-id:role/rs-role-name"}
)
```

- Para otras bases de datos, consulte la documentación correspondiente.

AWS Glue opciones

- Para evitar un análisis completo de todas las ejecuciones continuas de trabajos y procesar solo los datos que no estuvieron presentes durante la última ejecución del trabajo, habilite los marcadores de trabajos.
- Para limitar la cantidad de datos de entrada que se van a procesar, habilite la ejecución delimitada con marcadores de trabajos. Esto ayuda a reducir la cantidad de datos analizados para cada ejecución de trabajo.

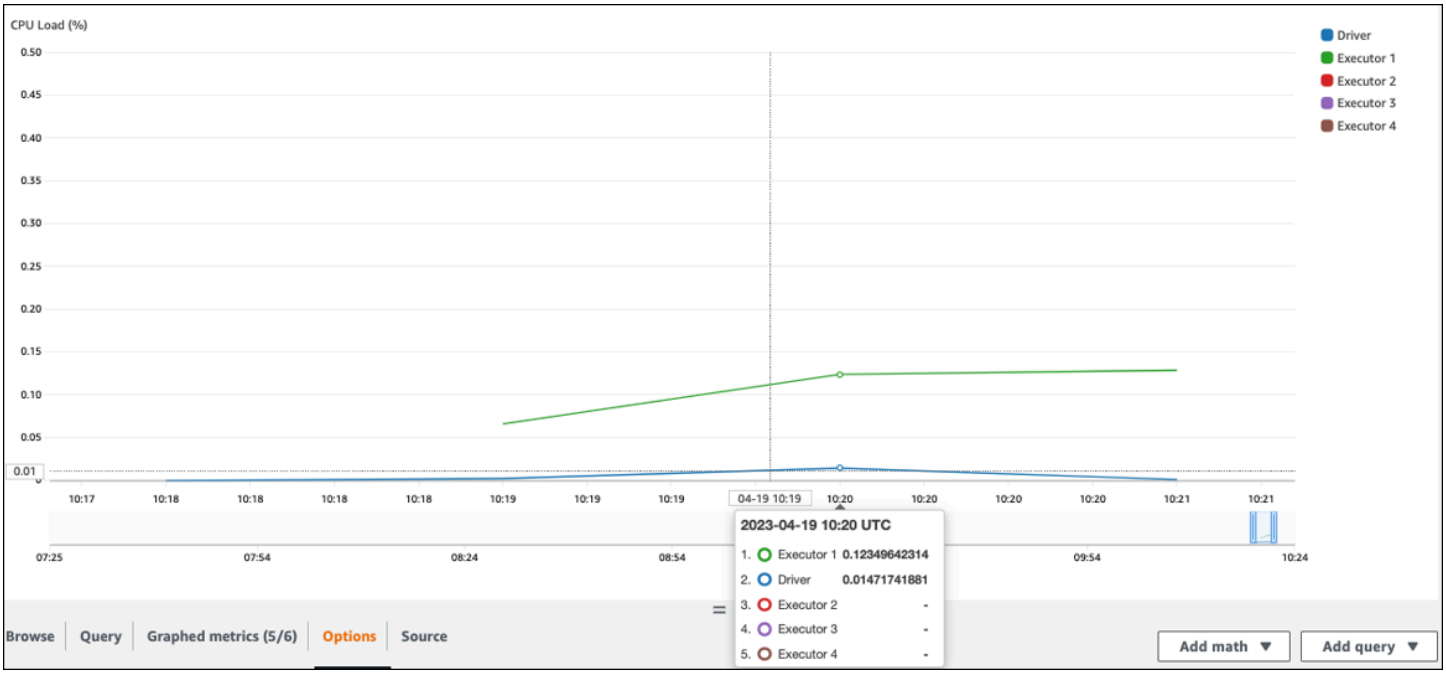
Paralelización de las tareas

Para optimizar el rendimiento, es importante paralelizar las tareas de carga y transformación de datos. Como explicamos en Temas clave de Apache Spark, el número de particiones de conjuntos de datos distribuidos resilientes (RDD) es importante porque determina el grado de paralelismo. Cada tarea que crea Spark corresponde a una partición de RDD de forma 1:1. Para lograr el mejor rendimiento, debe entender cómo se determina el número de particiones de RDD y cómo se optimiza ese número.

Si no tienes suficiente paralelismo, los siguientes síntomas se registrarán en las CloudWatchmétricas y en la interfaz de usuario de Spark.

CloudWatch métricas

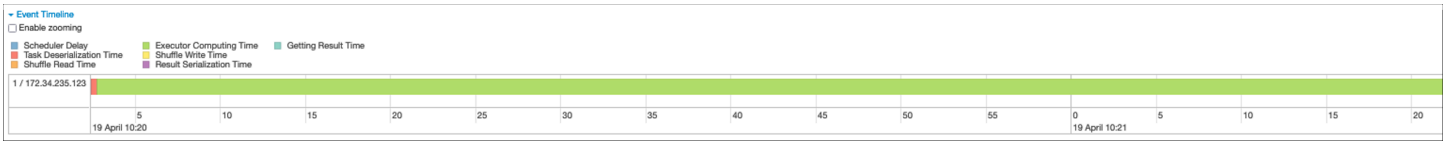
Consulte Carga de la CPU y Utilización de la memoria. Si algunos ejecutores no se procesan durante una fase de su trabajo, es conveniente mejorar el paralelismo. En este caso, durante el periodo de tiempo visualizado, el ejecutor 1 estaba llevando a cabo una tarea, pero los ejecutores restantes (2, 3 y 4) no. Se puede deducir que el controlador de Spark no asignó tareas a esos ejecutores.



UI de Spark

En la pestaña Etapa de la IU de Spark, puede ver el número de tareas de una etapa. En este caso, Spark solo ha llevado a cabo una tarea.

Además, en el cronograma del evento se muestra al ejecutor 1 procesando una tarea. Esto significa que el trabajo de esta etapa se ejecutó íntegramente con un ejecutor, mientras que los demás estaban inactivos.



Si observa estos síntomas, pruebe las siguientes soluciones para cada origen de datos.

Paralelización de la carga de datos desde Amazon S3

Para paralelizar las cargas de datos de Amazon S3, compruebe primero el número predeterminado de particiones. A continuación, puede determinar manualmente el número objetivo de particiones, pero asegúrese de evitar tener demasiadas particiones.

Determinación del número predeterminado de particiones

En Amazon S3, el número inicial de particiones de RDD de Spark (cada una de las cuales corresponde a una tarea de Spark) se determina según las características del conjunto de datos de Amazon S3 (por ejemplo, el formato, la compresión y el tamaño). Al crear un Spark AWS Glue DynamicFrame o un Spark DataFrame a partir de objetos CSV almacenados en Amazon S3, el número inicial de particiones RDD (`NumPartitions`) se puede calcular aproximadamente de la siguiente manera:

- Tamaño del objeto <= 64 MB: `NumPartitions = Number of Objects`
- Tamaño del objeto > 64 MB: `NumPartitions = Total Object Size / 64 MB`
- No se puede dividir (gzip): `NumPartitions = Number of Objects`

Como se explica en la sección Reducción de la cantidad de análisis de datos, Spark divide los objetos de S3 grandes en divisiones que se pueden procesar en paralelo. Cuando el objeto es más grande que el tamaño de la división, Spark divide el objeto y crea una partición de RDD (y una tarea) para cada división. El tamaño de división de Spark se basa en el formato de datos y el entorno del tiempo de ejecución, pero esta es una aproximación inicial razonable. Algunos objetos se comprimen con formatos de compresión que no se pueden dividir, como gzip, por lo que Spark no puede dividirlos.

El `NumPartitions` valor puede variar en función del formato de datos, la compresión, la AWS Glue versión, el número de AWS Glue trabajadores y la configuración de Spark.

Por ejemplo, cuando cargas un único `csv.gz` objeto de 10 GB con un Spark DataFrame, el controlador de Spark solo creará una partición RDD (`NumPartitions=1`) porque gzip no se puede dividir. Esto supone una gran carga para un ejecutor de Spark en concreto y no se asigna ninguna tarea a los ejecutores restantes, como se describe en la siguiente figura.

Consulte el número real de tareas (`NumPartitions`) de la etapa en la pestaña Etapa de la IU web de Spark o ejecute `df.rdd.getNumPartitions()` en su código para comprobar el paralelismo.

Cuando encuentre un archivo gzip de 10 GB, compruebe si el sistema que lo genera puede generarlo en un formato divisible. Si no es una opción, es posible que tenga que escalar la capacidad del clúster para procesar el archivo. Para ejecutar las transformaciones de forma eficaz en los datos que ha cargado, tendrá que reequilibrar el RDD entre los nodos de trabajo del clúster mediante la repartición.

Determinación manual del número objetivo de particiones

En función de las propiedades de sus datos y de la implementación de determinadas funcionalidades por parte de Spark, es posible que acabe con un valor de `NumPartitions` bajo, aunque el trabajo subyacente aún se pueda paralelizar. Si `NumPartitions` es demasiado pequeño, ejecute `df.repartition(N)` para aumentar el número de particiones y poder distribuir el procesamiento entre varios ejecutores de Spark.

En este caso, la ejecución de `df.repartition(100)` aumentará el valor de `NumPartitions` de 1 a 100, lo que generará 100 particiones de sus datos, cada una con una tarea que podrá asignarse a los demás ejecutores.

La operación `repartition(N)` divide todos los datos en partes iguales (10 GB / 100 particiones = 100 MB/partición), lo que evita que los datos se sesguen hacia determinadas particiones.

> **Note**
>
> Cuando se ejecuta una operación de mezcla como `join`, el número de particiones aumenta o disminuye de forma dinámica en función del valor de `spark.sql.shuffle.partitions` o `spark.default.parallelism`. Esto facilita un intercambio de datos más eficiente entre los ejecutores de Spark. Para obtener más información, consulte la documentación de Spark.

Al determinar el número objetivo de particiones, tu objetivo es maximizar el uso de los trabajadores aprovisionados. AWS Glue El número de AWS Glue trabajadores y el número de tareas de Spark se relacionan mediante el número de vCPUs. Spark admite una tarea para cada núcleo de vCPU. En AWS Glue la versión 3.0 o posterior, puedes calcular el número objetivo de particiones mediante la siguiente fórmula.

```
# Calculate NumPartitions by WorkerType
numExecutors = (NumberOfWorkers - 1)
numSlotsPerExecutor =
  4 if WorkerType is G.1X
  8 if WorkerType is G.2X
  16 if WorkerType is G.4X
  32 if WorkerType is G.8X
NumPartitions = numSlotsPerExecutor * numExecutors

# Example: Glue 4.0 / G.1X / 10 Workers
numExecutors = ( 10 - 1 ) = 9  # 1 Worker reserved on Spark Driver
```

```
numSlotsPerExecutor       = 4  # G.1X has 4 vCpu core ( Glue 3.0 or later )
NumPartitions = 9  * 4    = 36
```

En este ejemplo, cada nodo de trabajo G.1X proporciona cuatro núcleos de vCPU a un ejecutor de Spark (`spark.executor.cores = 4`). Spark admite una tarea para cada núcleo de vCPU, por lo que los ejecutores G.1X de Spark pueden ejecutar cuatro tareas simultáneamente (`numSlotPerExecutor`). Este número de particiones aprovecha al máximo el clúster si las tareas tardan el mismo tiempo. Sin embargo, algunas tareas tardarán más que otras y se crearán núcleos inactivos. Si esto ocurre, considere la posibilidad de multiplicar `numPartitions` por 2 o 3 para dividir y programar de manera eficiente las tareas que producen los cuellos de botella.

Demasiadas particiones

Un número excesivo de particiones crea un número excesivo de tareas. Esto provoca una gran carga en el controlador de Spark debido a la sobrecarga relacionada con el procesamiento distribuido, como las tareas de administración y el intercambio de datos entre los ejecutores de Spark.

Si el número de particiones de su trabajo es considerablemente mayor que el número objetivo de particiones, considere la posibilidad de reducir el número de particiones. Puede reducir las particiones mediante las siguientes opciones:

- Si el tamaño de los archivos es muy pequeño, utilice AWS Glue GroupFiles. Puede reducir el paralelismo excesivo que resulta del lanzamiento de una tarea de Apache Spark para procesar cada archivo.
- Use `coalesce(N)` para fusionar las particiones. Se trata de un proceso de bajo costo. A la hora de reducir el número de particiones, `coalesce(N)` se prefiere en lugar de `repartition(N)`, ya que `repartition(N)` lleva a cabo una mezcla para distribuir equitativamente la cantidad de registros de cada partición. Esto aumenta los costos y la sobrecarga administrativa.
- Utilice la ejecución adaptativa de consultas de Spark 3.x. Como se explica en la sección Temas clave de Apache Spark, la ejecución adaptativa de consultas proporciona una función para fusionar automáticamente el número de particiones. Puede usar este enfoque cuando no sepa el número de particiones hasta llevar a cabo la ejecución.

Paralelización de la carga de datos desde JDBC

El número de particiones de RDD de Spark se determina según la configuración. Tenga en cuenta que, de forma predeterminada, solo se ejecuta una tarea para analizar un conjunto de datos de origen completo mediante una consulta `SELECT`.

AWS Glue DynamicFrames Tanto Spark como Spark DataFrames admiten la carga de datos JDBC paralelizada en múltiples tareas. Esto se hace mediante predicados `where` para dividir una consulta `SELECT` en varias consultas. Para paralelizar las lecturas de JDBC, configure las siguientes opciones:

- Para AWS Glue DynamicFrame, establece (o) y. `hashfield` `hashexpression`) `hashpartition` Para obtener más información, consulte Lectura desde tablas de JDBC en paralelo.

```
connection_mysql8_options = {
    "url": "jdbc:mysql://XXXXXXXXXX.XXXXXXX.us-east-1.rds.amazonaws.com:3306/test",
    "dbtable": "medicare_tb",
    "user": "test",
    "password": "XXXXXXXXX",
    "hashexpression":"id",
    "hashpartitions":"10"
}
datasource0 = glueContext.create_dynamic_frame.from_options(
    'mysql',
    connection_options=connection_mysql8_options,
    transformation_ctx= "datasource0"
)
```

- Para Spark DataFrame, establece `numPartitions`,`partitionColumn`,`lowerBound`, y`upperBound`. Para obtener más información, consulte JDBC To Other Databases.

```
df = spark.read \
    .format("jdbc") \
    .option("url", "jdbc:mysql://XXXXXXXXXX.XXXXXXX.us-east-1.rds.amazonaws.com:3306/
test") \
    .option("dbtable", "medicare_tb") \
    .option("user", "test") \
    .option("password", "XXXXXXXXXX") \
    .option("partitionColumn", "id") \
    .option("numPartitions", "10") \
    .option("lowerBound", "0") \
    .option("upperBound", "1141455") \
    .load()

df.write.format("json").save("s3://bucket_name/Tests/sparkjdbc/with_parallel/")
```

Paralelización de la carga de datos de DynamoDB al utilizar el conector de ETL

El número de particiones de RDD de Spark se determina según el parámetro `dynamodb.splits`. Para paralelizar las lecturas de Amazon DynamoDB, configure las siguientes opciones:

- Aumente el valor de `dynamodb.splits`.
- Optimice el parámetro siguiendo la fórmula explicada en Tipos y opciones de conexión para ETL en AWS Glue Spark.

Paralelización de la carga de datos de Kinesis Data Streams

El número de particiones de RDD de Spark se determina según el número de particiones del flujo de datos de Amazon Kinesis Data Streams de origen. Si solo tiene unas pocas particiones en su flujo de datos, solo habrá unas pocas tareas de Spark. Esto puede provocar un bajo paralelismo en los procesos posteriores. Para paralelizar las lecturas de Kinesis Data Streams, configure las siguientes opciones:

- Aumente el número de particiones para obtener más paralelismo al cargar datos de Kinesis Data Streams.
- Si la lógica del microlote es lo suficientemente compleja, considere la posibilidad de volver a particionar los datos al principio del lote, después de eliminar las columnas innecesarias.

Para obtener más información, consulta las prácticas recomendadas para optimizar el coste y el rendimiento de la AWS Glue transmisión de trabajos de ETL.

Paralelización de las tareas después de la carga de datos

Para paralelizar las tareas después de la carga de datos, aumente el número de particiones de RDD mediante las siguientes opciones:

- Vuelva a particionar los datos para generar un mayor número de particiones, especialmente justo después de la carga inicial si la carga en sí no se pudo paralelizar.

  Llame a `repartition()` DynamicFrame o DataFrame especifique el número de particiones. Una buena regla general es multiplicar por dos o tres el número de núcleos disponibles.

  Sin embargo, al escribir una tabla particionada, esto puede provocar una explosión de archivos (cada partición puede generar un archivo en cada partición de la tabla). Para evitarlo, puedes

reparticionar tu columna DataFrame por columnas. Esto utiliza las columnas de partición de la tabla para que los datos se organicen antes de escribirlos. Puede especificar un número mayor de particiones sin tener archivos pequeños en las particiones de la tabla. Sin embargo, tenga cuidado para evitar el sesgo de datos, ya que algunos valores de partición acaban con la mayoría de los datos y se retrasa la finalización de la tarea.

- Cuando haya mezclas, aumente el valor de `spark.sql.shuffle.partitions`. Esto también puede ayudar a solucionar cualquier problema de memoria durante la mezcla.

  Cuando tiene más de 2001 particiones de mezcla, Spark utiliza un formato de memoria comprimido. Si tiene un número cercano a ese valor, quizás quiera establecer el valor de `spark.sql.shuffle.partitions` por encima de ese límite para obtener una representación más eficiente.

Optimización de las mezclas

Algunas operaciones, como `join()` y `groupByKey()`, requieren que Spark lleve a cabo una mezcla. La mezcla es el mecanismo de Spark para redistribuir los datos de forma que se agrupen de forma diferente en las particiones de RDD. Las mezclas pueden ayudarlo a corregir los cuellos de botella de rendimiento. Sin embargo, dado que la mezcla suele implicar la copia de datos entre los ejecutores de Spark, se trata de una operación compleja y costosa. Por ejemplo, las mezclas generan los siguientes costos:

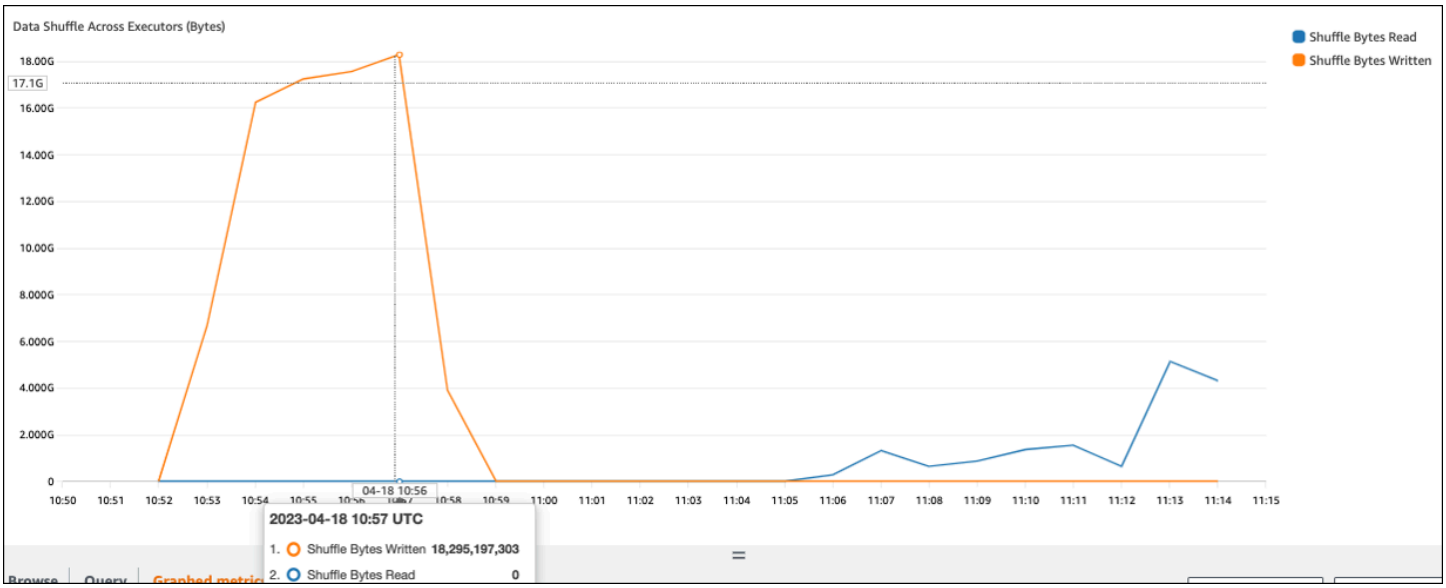
- E/S de disco:
  - Genera una gran cantidad de archivos intermedios en el disco.
- E/S de red:
  - Necesita muchas conexiones de red (número de conexiones = `Mapper × Reducer`).
  - Como los registros se agregan a nuevas particiones de RDD que podrían alojarse en un ejecutor de Spark diferente, una parte sustancial del conjunto de datos podría moverse entre los ejecutores de Spark a través de la red.
- Carga de CPU y memoria:
  - Ordena los valores y fusiona conjuntos de datos. Estas operaciones se planifican en el ejecutor, lo que supone una gran carga para este.

La mezcla es uno de los factores más importantes en la degradación del rendimiento de su aplicación de Spark. Al almacenar los datos intermedios, puede agotar el espacio en el disco local del ejecutor, lo que provoca un error en el trabajo de Spark.

Puedes evaluar tu rendimiento en el shuffle en CloudWatch las métricas y en la interfaz de usuario de Spark.

CloudWatch métricas

Si el valor de Bytes de mezcla escritos es alto en comparación con Bytes de mezcla leídos, su trabajo de Spark podría utilizar operaciones de mezcla, como `join()` o `groupByKey()`.

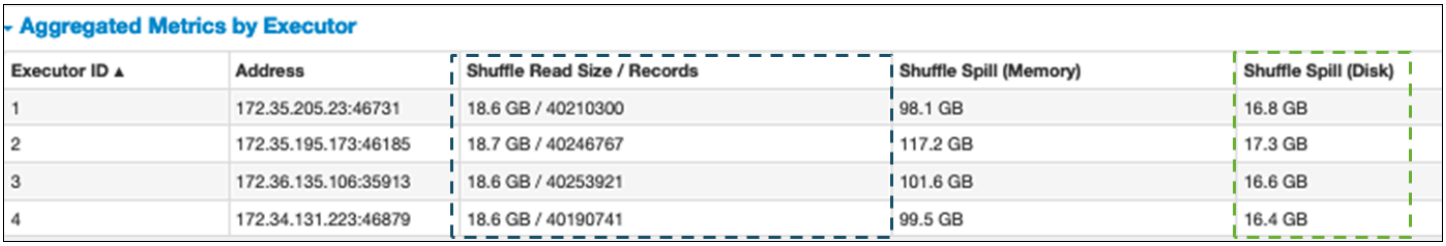


UI de Spark

En la pestaña Etapa de la IU de Spark, puede comprobar los valores de Tamaño de lectura de la mezcla / Registros. También puede verlos en la pestaña Ejecutores.

En la siguiente captura de pantalla, cada ejecutor intercambia aproximadamente 18,6 GB/4 020 000 registros con el proceso de mezcla, lo que supone un tamaño total de lectura de mezcla de unos 75 GB.

En la columna Volcado de mezcla (disco) se muestra una gran cantidad de datos que se vuelca de la memoria al disco, lo que puede provocar que el disco se llene o que se produzca un problema de rendimiento.



Aggregated Metrics by Executor

Executor ID ▲	Address	Shuffle Read Size / Records	Shuffle Spill (Memory)	Shuffle Spill (Disk)
1	172.35.205.23:46731	18.6 GB / 40210300	98.1 GB	16.8 GB
2	172.35.195.173:46185	18.7 GB / 40246767	117.2 GB	17.3 GB
3	172.36.135.106:35913	18.6 GB / 40253921	101.6 GB	16.6 GB
4	172.34.131.223:46879	18.6 GB / 40190741	99.5 GB	16.4 GB

Si observa estos síntomas y la etapa tarda demasiado en comparación con sus objetivos de rendimiento, o si se producen los errores `Out Of Memory` o `No space left on device`, considere las siguientes soluciones.

Optimización de la unión

La operación `join()`, que une tablas, es la operación de mezcla más utilizada, pero suele provocar un cuello de botella de rendimiento. Como la unión es una operación costosa, le recomendamos no utilizarla a menos que sea esencial para los requisitos de su empresa. Haga las siguientes preguntas para comprobar que está haciendo un uso eficiente de su canalización de datos:

- ¿Va a volver a calcular una unión que también se lleva a cabo en otros trabajos que puede reutilizar?
- ¿Va a llevar a cabo una unión para resolver claves externas a valores que no utilizan los consumidores de su resultado?

Tras confirmar que las operaciones de unión son esenciales para los requisitos de su empresa, consulte las siguientes opciones para optimizar la unión de forma que se ajuste a sus requisitos.

Uso de una inserción antes de una unión

Filtra las filas y columnas innecesarias DataFrame antes de realizar una unión. Esto tiene las siguientes ventajas:

- Reduce la cantidad de transferencia de datos durante la mezcla.
- Reduce la cantidad de procesamiento en el ejecutor de Spark.
- Reduce la cantidad de análisis de datos.

```
# Default
df_joined  = df1.join(df2, ["product_id"])
```

```
# Use Pushdown
df1_select =
 df1.select("product_id","product_title","star_rating").filter(col("star_rating")>=4.0)
df2_select = df2.select("product_id","category_id")
df_joined  = df1_select.join(df2_select, ["product_id"])
```

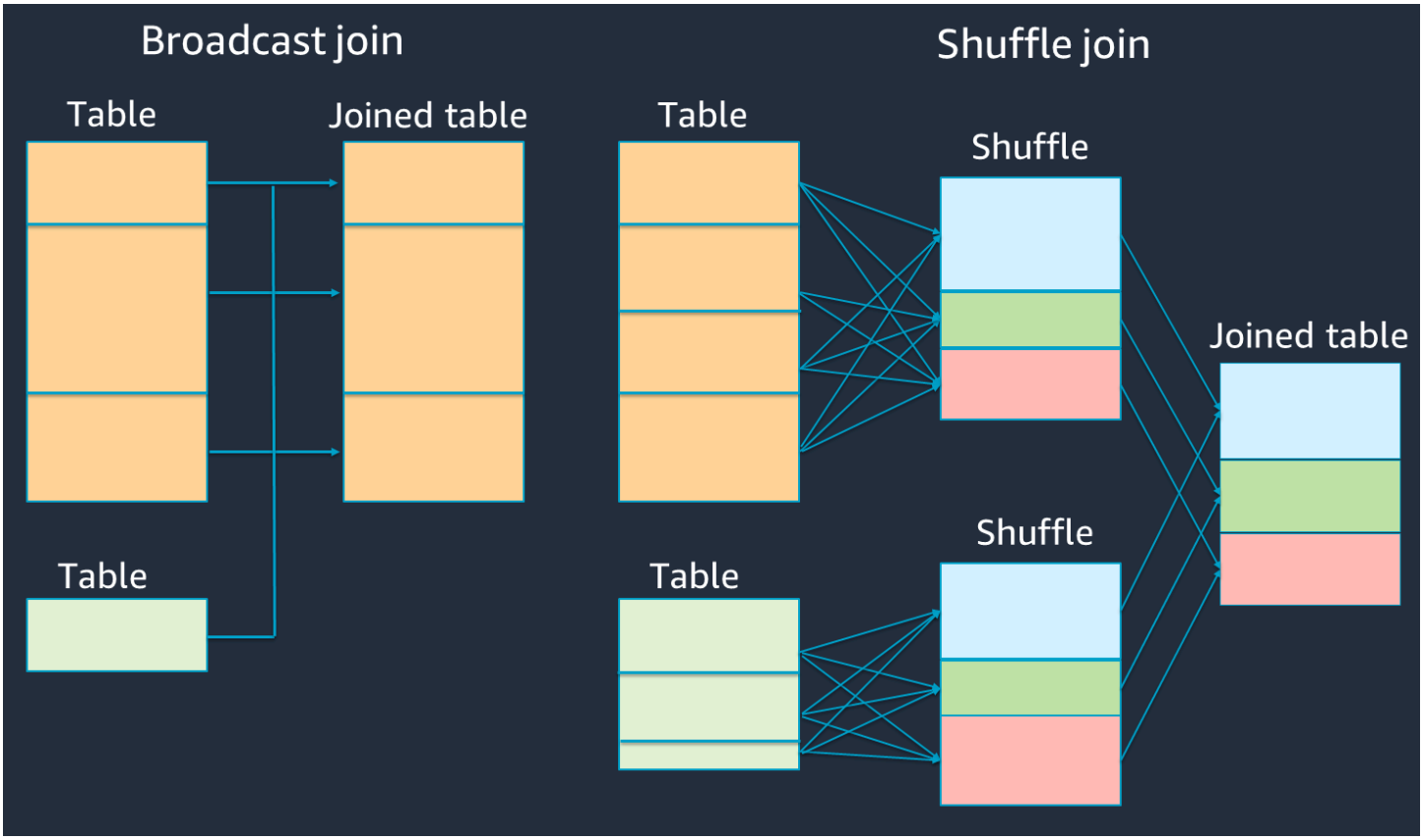
Utilice DataFrame Join

Intenta usar una API de alto nivel de Spark, como SparkSQL y Datasets DataFrame, en lugar de la API RDD o join. DynamicFrame Puedes realizar la conversión DataFrame con una llamada DynamicFrame a un método como. `dyf.toDF()` Como se explica en la sección Temas clave de Apache Spark, estas operaciones de unión aprovechan internamente la optimización de consultas del optimizador Catalyst.

Uniones de mezcla y hash de transmisión y sugerencias

Spark admite dos tipos de unión: unión de mezcla y unión hash de transmisión. Una unión hash de transmisión no requiere mezclas y puede requerir menos procesamiento que una unión de mezcla. Sin embargo, solo se aplica al unir una tabla pequeña a una grande. Al unir una tabla que quepa en la memoria de un solo ejecutor de Spark, considere la posibilidad de usar una unión hash de transmisión.

En el siguiente diagrama se muestran la estructura de alto nivel y los pasos de una unión hash de transmisión y una unión de mezcla.



Los detalles de cada unión son los siguientes:

- Unión de mezcla:
  - La unión hash de mezcla une dos tablas sin ordenación y distribuye la unión entre las dos tablas. Es adecuada para uniones de tablas pequeñas que se pueden almacenar en la memoria del ejecutor de Spark.
  - La unión de ordenación/combinación distribuye las dos tablas que se van a unir por clave y las ordena antes de unirlas. Es adecuada para unir tablas grandes.
- Unión hash de transmisión:
  - Una unión hash de transmisión envía el RDD o tabla más pequeño a cada uno de los nodos de trabajo. Luego, hace una combinación del lado de asignación con cada partición del RDD o la tabla más grande.

    Es adecuado para las uniones cuando una de las tablas RDDs o tablas puede caber en la memoria o se puede hacer que quepa en la memoria. Siempre que sea posible, es recomendable hacer una unión hash de transmisión, ya que no se necesita una mezcla.

Puede usar una sugerencia de unión para solicitar una unión de transmisión desde Spark de la siguiente manera.

```
# DataFrame
from pySpark.sql.functions import broadcast
df_joined= df_big.join(broadcast(df_small), right_df[key] == left_df[key],
 how='inner')

-- SparkSQL
SELECT /*+ BROADCAST(t1) /  FROM t1 INNER JOIN t2 ON t1.key = t2.key;
```

Para obtener más información sobre las sugerencias de unión, consulte Join hints.

En la AWS Glue versión 3.0 y versiones posteriores, puede aprovechar las uniones hash emitidas automáticamente al habilitar la ejecución adaptativa de consultas y parámetros adicionales. La ejecución adaptativa de consultas convierte una unión de ordenación/combinación en una unión hash de transmisión cuando las estadísticas de tiempo de ejecución de cualquiera de los lados de la unión son inferiores al umbral de unión hash de transmisión adaptativo.

En la AWS Glue versión 3.0, puede habilitar la ejecución adaptativa de consultas mediante la configuración`spark.sql.adaptive.enabled=true`. La ejecución adaptativa de consultas está habilitada de forma predeterminada en AWS Glue 4.0.

Puede establecer parámetros adicionales relacionados con las mezclas y las uniones hash de transmisión:

- `spark.sql.adaptive.localShuffleReader.enabled`
- `spark.sql.adaptive.autoBroadcastJoinThreshold`

Para obtener más información sobre los parámetros relacionados, consulte Conversión de uniones de ordenación/combinación en uniones de transmisión.

En la AWS Glue versión 3.0 o versiones posteriores, puede utilizar otras sugerencias de unión para mezclar y ajustar su comportamiento.

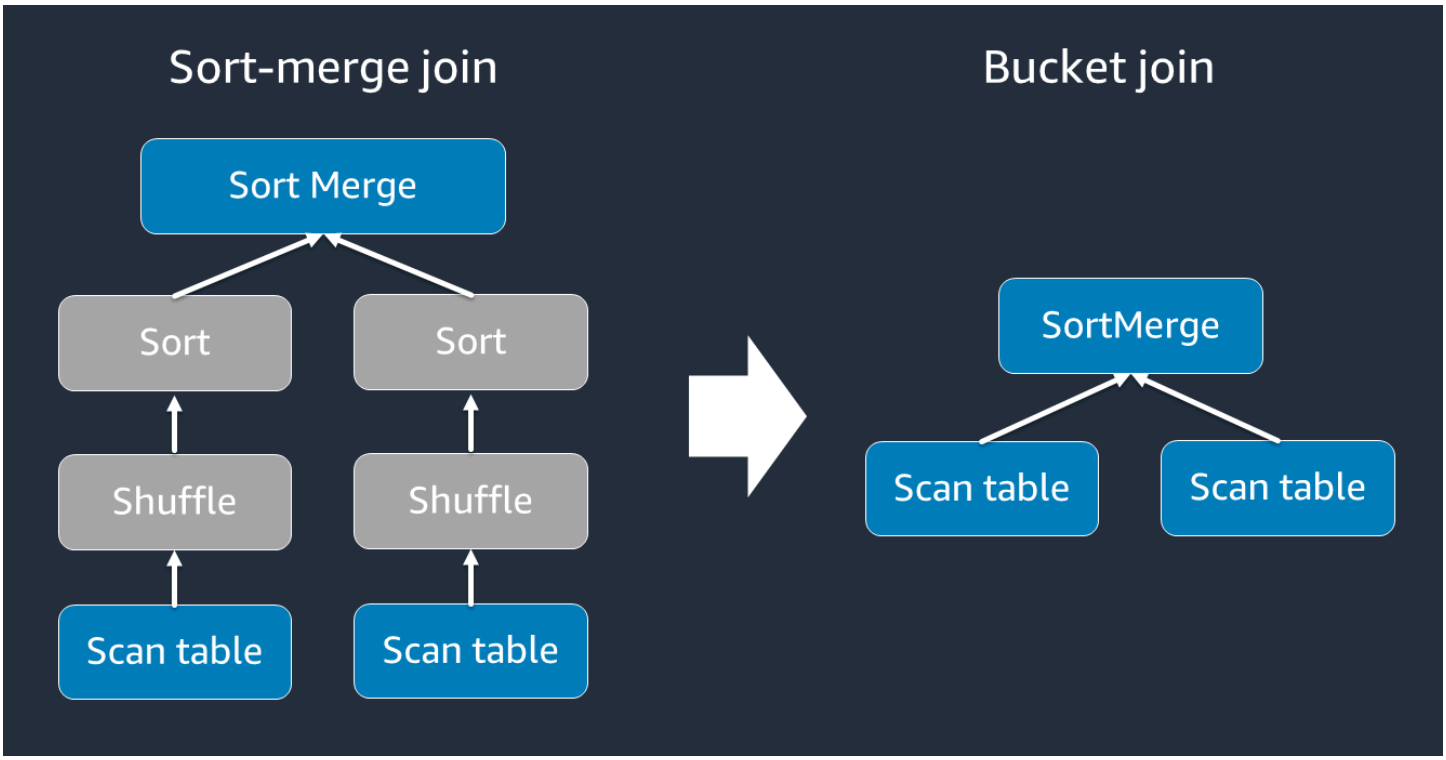
```
-- Join Hints for shuffle sort merge join
SELECT /*+ SHUFFLE_MERGE(t1) /  FROM t1 INNER JOIN t2 ON t1.key = t2.key;
SELECT /*+ MERGEJOIN(t2) /  FROM t1 INNER JOIN t2 ON t1.key = t2.key;
SELECT /*+ MERGE(t1) /  FROM t1 INNER JOIN t2 ON t1.key = t2.key;
```

```
-- Join Hints for shuffle hash join
SELECT /*+ SHUFFLE_HASH(t1) /  FROM t1 INNER JOIN t2 ON t1.key = t2.key;

-- Join Hints for shuffle-and-replicate nested loop join
SELECT /*+ SHUFFLE_REPLICATE_NL(t1) /  FROM t1 INNER JOIN t2 ON t1.key = t2.key;
```

Uso de la asignación de buckets

La unión de ordenación/combinación requiere dos fases: mezcla y ordenación y, a continuación, fusión. Estas dos fases pueden sobrecargar el ejecutor de Spark y provocar problemas de OOM y rendimiento cuando algunos ejecutores se fusionan y otros se ordenan simultáneamente. En esos casos, podría ser posible llevar a cabo una unión eficiente mediante la asignación de buckets. La asignación de buckets mezclará y ordenará previamente las entradas en las claves de unión y, a continuación, escribirá esos datos ordenados en una tabla intermedia. El costo de los pasos de mezcla y ordenación se puede reducir al unir tablas grandes definiendo las tablas intermedias ordenadas con antelación.



Las tablas en buckets son útiles para lo siguiente:

- Datos que se unen con frecuencia a través de la misma clave, como `account_id`.

- Carga de tablas acumulativas diarias, como las tablas de base y delta, que podrían agruparse en un bucket en una columna común.

Puede crear una tabla en buckets mediante el siguiente código.

```
df.write.bucketBy(50, "account_id").sortBy("age").saveAsTable("bucketed_table")
```

Reparticione DataFrames las claves de unión antes de la unión

Para reparticionar los dos elementos DataFrames de las claves de combinación antes de la unión, utilice las siguientes instrucciones.

```
df1_repartitioned = df1.repartition(N,"join_key")
df2_repartitioned = df2.repartition(N,"join_key")
df_joined  = df1_repartitioned.join(df2_repartitioned,"product_id")
```

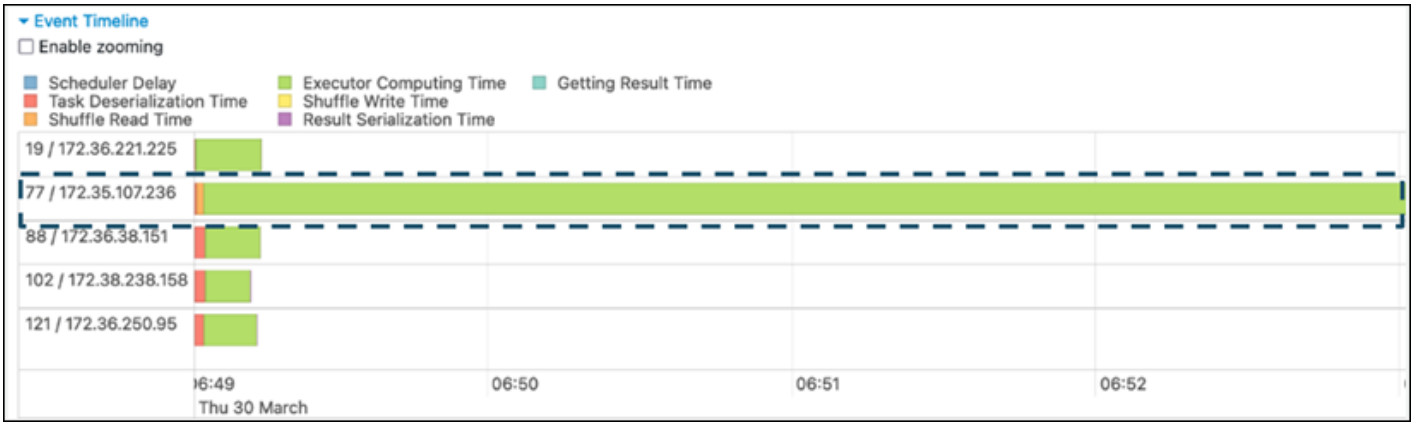
Esto dividirá dos (aún separados) de RDDs la clave de unión antes de iniciar la unión. Si las dos RDDs están particionadas en la misma clave y con el mismo código de partición, los registros RDD que planea unir tendrán una alta probabilidad de estar ubicados en el mismo elemento de trabajo antes de mezclarlos para la unión. Esto podría mejorar el rendimiento al reducir la actividad de la red y el sesgo de datos durante la unión.

Cómo superar el sesgo de datos

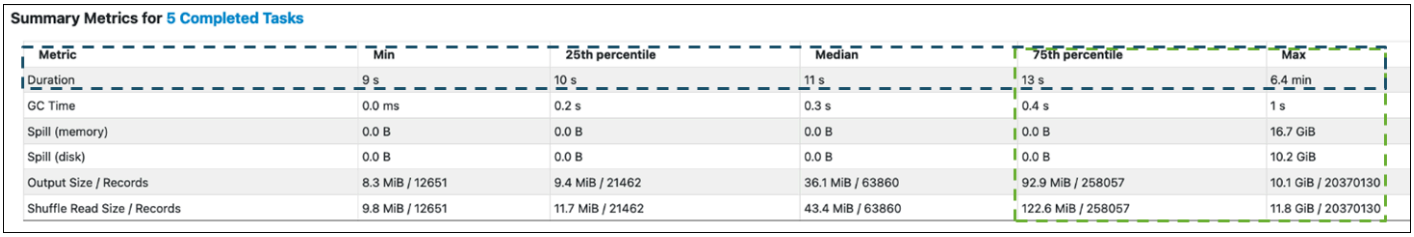
El sesgo de datos es una de las causas más habituales de los cuellos de botella para los trabajos de Spark. Se produce cuando los datos no se distribuyen uniformemente en las particiones de RDD. Esto hace que las tareas de esa partición tarden mucho más que otras, lo que retrasa el tiempo total de procesamiento de la aplicación.

Para identificar el sesgo de datos, evalúe las siguientes métricas en la IU de Spark:

- En la pestaña Etapa de la IU de Spark, examine la página Cronograma del evento. Puede ver una distribución desigual de tareas en la siguiente captura de pantalla. Las tareas que se distribuyen de forma desigual o que tardan demasiado en ejecutarse pueden indicar un sesgo de datos.



- Otra página importante es Métricas de resumen, en la que se muestran las estadísticas de las tareas de Spark. En la siguiente captura de pantalla se muestran las métricas con los percentiles de Duración, Tiempo de GC, Volcado (memoria), Volcado (disco), etc.



Summary Metrics for 5 Completed Tasks

Metric	Min	25th percentile	Median	75th percentile	Max
Duration	9 s	10 s	11 s	13 s	6.4 min
GC Time	0.0 ms	0.2 s	0.3 s	0.4 s	1 s
Spill (memory)	0.0 B	0.0 B	0.0 B	0.0 B	16.7 GiB
Spill (disk)	0.0 B	0.0 B	0.0 B	0.0 B	10.2 GiB
Output Size / Records	8.3 MiB / 12651	9.4 MiB / 21462	36.1 MiB / 63860	92.9 MiB / 258057	10.1 GiB / 20370130
Shuffle Read Size / Records	9.8 MiB / 12651	11.7 MiB / 21462	43.4 MiB / 63860	122.6 MiB / 258057	11.8 GiB / 20370130

  Cuando las tareas estén distribuidas uniformemente, verá números similares en todos los percentiles. Cuando los datos estén sesgados, verá valores muy sesgados en cada percentil. En el ejemplo, la duración de la tarea es inferior a 13 segundos en Mínimo, Percentil 25, Mediana y Percentil 75. Si bien la tarea máxima procesó 100 veces más datos que el percentil 75, su duración de 6,4 minutos es aproximadamente 30 veces mayor. Esto significa que al menos una tarea (o hasta un 25 % de las tareas) tardó mucho más tiempo que el resto de las tareas.

Si observa un sesgo de datos, pruebe lo siguiente:

- Si usa la AWS Glue versión 3.0, habilite la ejecución adaptativa de consultas mediante la configuración. `spark.sql.adaptive.enabled=true` La ejecución adaptativa de consultas está habilitada de forma predeterminada en la AWS Glue versión 4.0.

  También puede utilizar la ejecución adaptativa de consultas para el sesgo de datos introducido mediante las uniones; para ello, defina los siguientes parámetros relacionados:

  - `spark.sql.adaptive.skewJoin.skewedPartitionFactor`
  - `spark.sql.adaptive.skewJoin.skewedPartitionThresholdInBytes`

- `spark.sql.adaptive.advisoryPartitionSizeInBytes=128m (128 mebibytes or larger should be good)`
- `spark.sql.adaptive.coalescePartitions.enabled=true (when you want to coalesce partitions)`

Para obtener más información, consulte la documentación de Apache Spark.

- Utilice claves con un amplio rango de valores para las claves de unión. En una unión de mezcla, las particiones se determinan para cada valor de hash de una clave. Si la cardinalidad de una clave de unión es demasiado baja, es más probable que la función hash distribuya mal los datos entre las particiones. Por lo tanto, si su aplicación y su lógica empresarial lo admiten, considere la posibilidad de utilizar una clave compuesta o una clave de cardinalidad más alta.

```
# Use Single Primary Key
df_joined = df1_select.join(df2_select, ["primary_key"])

# Use Composite Key
df_joined = df1_select.join(df2_select, ["primary_key","secondary_key"])
```

Uso de la caché

Cuando utilices la función repetitiva DataFrames, evita tener que `df.persist()` hacer cálculos o barajar de forma adicional utilizando `df.cache()` o almacenando en caché los resultados de los cálculos en la memoria de cada ejecutor de Spark y en el disco. Spark también admite la persistencia RDDs en el disco o la replicación en varios nodos (nivel de almacenamiento).

Por ejemplo, puedes conservarlos DataFrames añadiendo. `df.persist()` Cuando la caché ya no sea necesaria, puede utilizar un`persist` para descartar los datos almacenados en caché.

```
df = spark.read.parquet("s3://<Bucket>/parquet/product_category=Books/")
df_high_rate = df.filter(col("star_rating")>=4.0)
df_high_rate.persist()

df_joined1 = df_high_rate.join(<Table1>, ["key"])
df_joined2 = df_high_rate.join(<Table2>, ["key"])
df_joined3 = df_high_rate.join(<Table3>, ["key"])
...
df_high_rate.unpersist()
```

Eliminación de las acciones innecesarias de Spark

Evita ejecutar acciones innecesarias como `count`, `show` o `collect`. Como se explica en la sección Temas clave de Apache Spark, Spark es lento. Cada RDD transformado puede volver a calcularse cada vez que ejecute una acción en él. Cuando utiliza muchas acciones de Spark, se llama a varios accesos a orígenes, cálculos de tareas y ejecuciones de mezcla para cada acción.

Si no necesita `collect()` ni otras acciones en su entorno comercial, considere la posibilidad de eliminarlas.

> **Note**
>
> Evite usar `collect()` de Spark en entornos comerciales en la medida de lo posible. La acción `collect()` devuelve todos los resultados de un cálculo del ejecutor de Spark al controlador de Spark, lo que podría provocar que el controlador de Spark devuelva un error de OOM. Para evitar un error de OOM, Spark define `spark.driver.maxResultSize = 1GB` de forma predeterminada, lo que limita el tamaño máximo de los datos devueltos al controlador de Spark a 1 GB.

Minimización de la sobrecarga de planificación

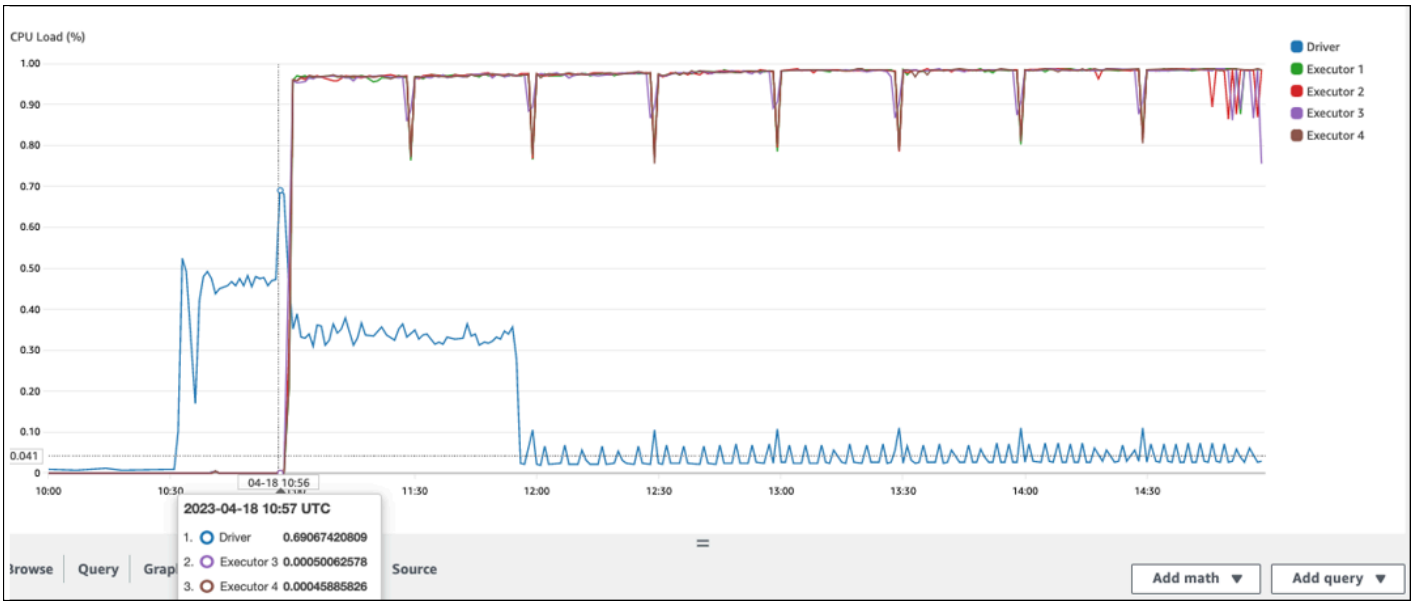
Como se analiza en Temas clave de Apache Spark, el controlador de Spark genera el plan de ejecución. Según ese plan, las tareas se asignan al ejecutor de Spark para su procesamiento distribuido. Sin embargo, el controlador de Spark puede convertirse en un cuello de botella si hay una gran cantidad de archivos pequeños o si el AWS Glue Data Catalog contiene una gran cantidad de particiones. Para identificar una sobrecarga de planificación elevada, evalúe las siguientes métricas.

CloudWatch métricas

Compruebe Carga de la CPU y Utilización de la memoria en las siguientes situaciones:

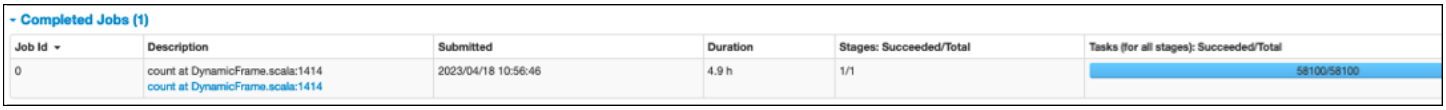
- Los valores de Carga de la CPU y Utilización de la memoria del controlador de Spark se registran como altos. Normalmente, el controlador de Spark no procesa los datos, por lo que la carga de la CPU y la utilización de la memoria no aumentan. Sin embargo, si el origen de datos de Amazon S3 tiene demasiados archivos pequeños, la enumeración de todos los objetos de S3 y la administración de un gran número de tareas puede provocar una utilización elevada de los recursos.

- Hay un largo intervalo antes de que comience el procesamiento en el ejecutor de Spark. En la siguiente captura de pantalla de ejemplo, la carga de CPU del ejecutor de Spark es demasiado baja hasta las 10:57, a pesar de que el AWS Glue trabajo comenzó a las 10:00. Esto indica que es posible que el controlador de Spark esté tardando mucho en generar un plan de ejecución. En este ejemplo, recuperar la gran cantidad de particiones del Catálogo de datos y enumerar la gran cantidad de archivos pequeños en el controlador de Spark lleva mucho tiempo.



IU de Spark

En la pestaña Trabajo de la IU de Spark, puede ver la hora de envío. En el siguiente ejemplo, el controlador de Spark inició el trabajo 0 a las 10:56:46, aunque el trabajo comenzó a las 10:00:00. AWS Glue



También puede ver la hora de Tareas (para todas las etapas): completadas correctamente/total en la pestaña Trabajo. En este caso, el número de tareas se registra como `58100`. Como se explica en la sección de Amazon S3 de la página Paralelización de las tareas, la cantidad de tareas corresponde aproximadamente a la cantidad de objetos de S3. Esto significa que hay unos 58 100 objetos en Amazon S3.

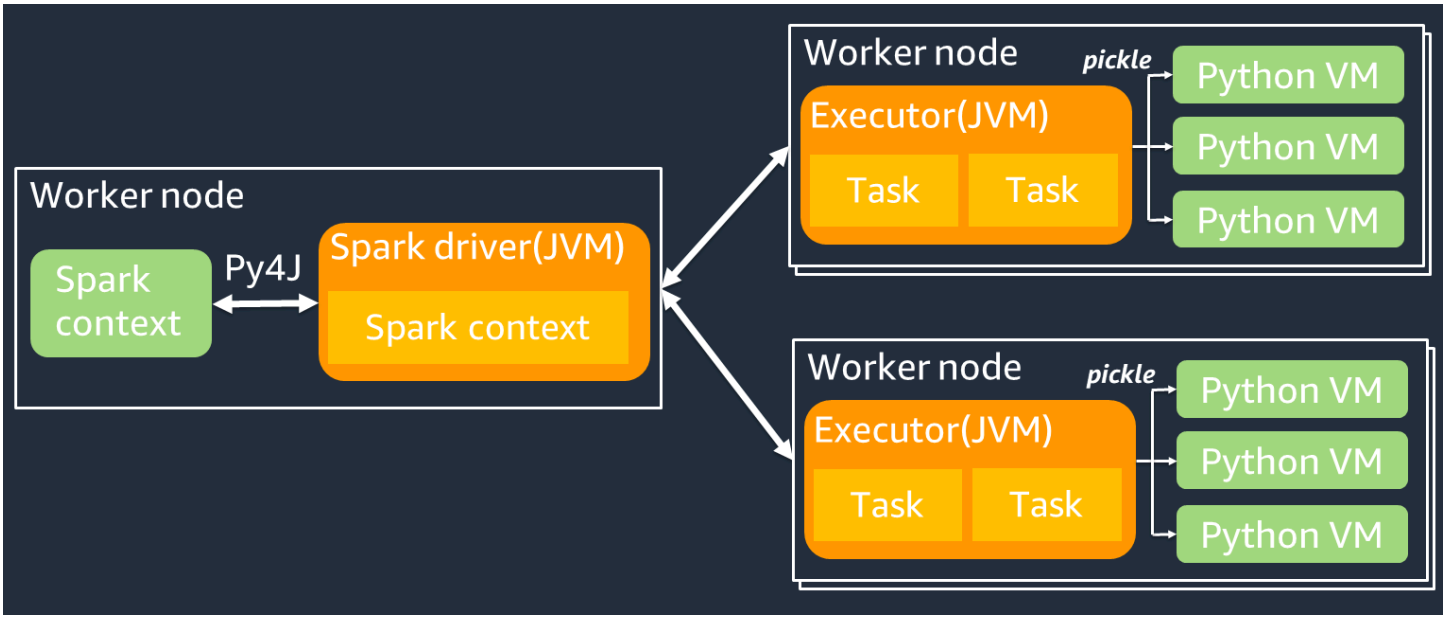
Para obtener más información sobre este trabajo y su cronograma, consulte la pestaña Etapa. Si observa un cuello de botella con el controlador de Spark, considere las siguientes soluciones:

- Si Amazon S3 tiene demasiados archivos, consulte las instrucciones sobre el paralelismo excesivo de la sección Demasiadas particiones de la página Paralelización de las tareas.
- Si Amazon S3 tiene demasiadas particiones, consulte las instrucciones sobre la partición excesiva de la sección Demasiadas particiones de Amazon S3 de la página Reducción de la cantidad de análisis de datos. Habilite los índices de particiones de AWS Glue si hay muchas particiones para reducir la latencia a la hora de recuperar los metadatos de las particiones del Catálogo de datos. Para obtener más información, consulte Mejorar el rendimiento de las consultas mediante índices de partición. AWS Glue
- Si JDBC tiene demasiadas particiones, reduzca el valor de `hashpartition`.
- Si DynamoDB tiene demasiadas particiones, reduzca el valor de `dynamodb.splits`.
- Si los trabajos de transmisión tienen demasiadas particiones, reduzca el número de particiones.

Optimización de las funciones definidas por el usuario

Las funciones definidas por el usuario (UDFs) y`RDD.map`, a menudo, reducen el rendimiento de manera significativa. PySpark Esto se debe a la sobrecarga necesaria para representar con precisión su código Python en la implementación de Scala subyacente de Spark.

El siguiente diagrama muestra la arquitectura de PySpark los trabajos. Cuando lo usas PySpark, el controlador Spark usa la biblioteca Py4j para llamar a los métodos de Java desde Python. Al llamar a Spark SQL o a funciones DataFrame integradas, hay poca diferencia de rendimiento entre Python y Scala porque las funciones se ejecutan en la JVM de cada ejecutor mediante un plan de ejecución optimizado.



Si usa su propia lógica de Python, como `map/ mapPartitions/ udf`, la tarea se ejecutará en un entorno de tiempo de ejecución de Python. La administración de dos entornos genera un costo general. Además, los datos en memoria deben transformarse para que las funciones integradas del entorno de tiempo de ejecución de JVM puedan utilizarlos. Pickle es un formato de serialización que se utiliza de forma predeterminada para el intercambio entre los tiempos de ejecución de JVM y Python. Sin embargo, el coste de esta serialización y deserialización es muy elevado, por lo que UDFs escribir en Java o Scala es más rápido que en Python. UDFs

Para evitar la sobrecarga de serialización y deserialización, tenga en cuenta lo siguiente: PySpark

- Usa las funciones integradas de Spark SQL: considera reemplazar tu propia función de mapa o UDF por Spark SQL o funciones integradas. DataFrame Al ejecutar Spark SQL o funciones DataFrame integradas, hay poca diferencia de rendimiento entre Python y Scala porque las tareas se gestionan en la JVM de cada ejecutor.
- UDFs Impleméntalo en Scala o Java: considera usar una UDF escrita en Java o Scala, ya que se ejecuta en la JVM.
- Use Apache basado en Arrow para cargas de trabajo vectorizadas: considere usar UDFs el basado en Arrow. UDFs Esta característica también se conoce como UDF vectorizada (UDF de pandas). Apache Arrow es un formato de datos en memoria independiente del lenguaje que se AWS Glue puede utilizar para transferir datos de manera eficiente entre los procesos de JVM y Python. Actualmente, esto es más beneficioso para los usuarios de Python que trabajan con Pandas o NumPy datos.

Arrow es un formato de columnas (vectorizado). Su uso no es automático y puede requerir algunos cambios menores en la configuración o el código para aprovechar al máximo y garantizar la compatibilidad. Para obtener más detalles y conocer las limitaciones, consulte Apache Arrow en PySpark.

En el siguiente ejemplo se compara una UDF incremental básica en Python estándar, como una UDF vectorizada, y en Spark SQL.

UDF de Python estándar

El tiempo del ejemplo son 3,20 (segundos).

Código de ejemplo

```
# DataSet
df = spark.range(10000000).selectExpr("id AS a","id AS b")

# UDF Example
def plus(a,b):
    return a+b
spark.udf.register("plus",plus)

df.selectExpr("count(plus(a,b))").collect()
```

Plan de ejecución

```
== Physical Plan ==
AdaptiveSparkPlan isFinalPlan=false
+- HashAggregate(keys=[], functions=[count(pythonUDF0#124)])
+- Exchange SinglePartition, ENSURE_REQUIREMENTS, [id=#580]
+- HashAggregate(keys=[], functions=[partial_count(pythonUDF0#124)])
+- Project [pythonUDF0#124]
+- BatchEvalPython [plus(a#116L, b#117L)], [pythonUDF0#124]
+- Project [id#114L AS a#116L, id#114L AS b#117L]
+- Range (0, 10000000, step=1, splits=16)
```

UDF vectorizada

El tiempo del ejemplo son 0,59 (segundos).

La UDF vectorizada es 5 veces más rápida que el ejemplo de la UDF anterior. Si consulta `Physical Plan`, puede ver `ArrowEvalPython`, que indica que esta aplicación está vectorizada por Apache Arrow. Para habilitar la UDF vectorizada, debe especificar `spark.sql.execution.arrow.pyspark.enabled = true` en el código.

Código de ejemplo

```
# Vectorized UDF
from pyspark.sql.types import LongType
from pyspark.sql.functions import count, pandas_udf

# Enable Apache Arrow Support
spark.conf.set("spark.sql.execution.arrow.pyspark.enabled", "true")

# DataSet
df = spark.range(10000000).selectExpr("id AS a","id AS b")

# Annotate pandas_udf to use Vectorized UDF
@pandas_udf(LongType())
def pandas_plus(a,b):
    return a+b
spark.udf.register("pandas_plus",pandas_plus)

df.selectExpr("count(pandas_plus(a,b))").collect()
```

Plan de ejecución

```
== Physical Plan ==
AdaptiveSparkPlan isFinalPlan=false
+- HashAggregate(keys=[], functions=[count(pythonUDF0#1082L)],
 output=[count(pandas_plus(a, b))#1080L])
+- Exchange SinglePartition, ENSURE_REQUIREMENTS, [id=#5985]
+- HashAggregate(keys=[], functions=[partial_count(pythonUDF0#1082L)],
 output=[count#1084L])
+- Project [pythonUDF0#1082L]
+- ArrowEvalPython [pandas_plus(a#1074L, b#1075L)], [pythonUDF0#1082L], 200
+- Project [id#1072L AS a#1074L, id#1072L AS b#1075L]
+- Range (0, 10000000, step=1, splits=16)
```

Spark SQL

El tiempo del ejemplo son 0,087 (segundos).

Spark SQL es mucho más rápido que la UDF vectorizada, ya que las tareas se ejecutan en la JVM de cada ejecutor sin un tiempo de ejecución de Python. Si puede reemplazar la UDF por una función integrada, le recomendamos que lo haga.

Código de ejemplo

```
df.createOrReplaceTempView("test")
spark.sql("select count(a+b) from test").collect()
```

Uso de pandas para macrodatos

Si ya estás familiarizado con los pandas y quieres usar Spark para macrodatos, puedes usar la API de pandas en Spark. AWS Glue La versión 4.0 y las versiones posteriores la admiten. Para empezar, puede usar el cuaderno oficial Quickstart: Pandas API on Spark. Para obtener más información, consulte la Documentación de PySpark.

Recursos

- AWS Glue
- Performance Tuning (guía de Spark SQL)
- AWS Glue Optimization Workshop

Historial de documentos

En la siguiente tabla, se describen cambios significativos de esta guía. Si quiere recibir notificaciones de futuras actualizaciones, puede suscribirse a las notificaciones RSS.

Cambio	Descripción	Fecha
Publicación inicial	—	2 de enero de 2024

AWS Glosario de orientación prescriptiva

Los siguientes son términos de uso común en las estrategias, guías y patrones proporcionados por la Guía AWS prescriptiva. Para sugerir entradas, utilice el enlace Enviar comentarios al final del glosario.

Números

Las 7 R

Siete estrategias de migración comunes para trasladar aplicaciones a la nube. Estas estrategias se basan en las 5 R que Gartner identificó en 2011 y consisten en lo siguiente:

- Refactorizar/rediseñar: traslade una aplicación y modifique su arquitectura mediante el máximo aprovechamiento de las características nativas en la nube para mejorar la agilidad, el rendimiento y la escalabilidad. Por lo general, esto implica trasladar el sistema operativo y la base de datos. Ejemplo: Migrar la base de datos de Oracle en las instalaciones a Amazon Aurora PostgreSQL-Compatible Edition.
- Redefinir la plataforma (transportar y redefinir): traslade una aplicación a la nube e introduzca algún nivel de optimización para aprovechar las capacidades de la nube. Ejemplo: Migrar la base de datos Oracle en las instalaciones a Amazon Relational Database Service (Amazon RDS) para Oracle en la nube de Nube de AWS.
- Recomprar (readquirir): cambie a un producto diferente, lo cual se suele llevar a cabo al pasar de una licencia tradicional a un modelo SaaS. Ejemplo: Migrar el sistema de administración de las relaciones con los clientes (CRM) a Salesforce.com.
- Volver a alojar (migrar mediante lift-and-shift): traslade una aplicación a la nube sin realizar cambios para aprovechar las capacidades de la nube. Ejemplo: Migrar la base de datos de Oracle en las instalaciones a Oracle en una instancia de EC2 en la Nube de AWS.
- Reubicar: (migrar el hipervisor mediante lift and shift): traslade la infraestructura a la nube sin comprar equipo nuevo, reescribir aplicaciones o modificar las operaciones actuales. Los servidores se migran de una plataforma en las instalaciones a un servicio en la nube para la misma plataforma. Ejemplo: migrar una Microsoft Hyper-V aplicación a AWS.
- Retener (revisitar): conserve las aplicaciones en el entorno de origen. Estas pueden incluir las aplicaciones que requieren una refactorización importante, que desee posponer para más adelante, y las aplicaciones heredadas que desee retener, ya que no hay ninguna justificación empresarial para migrarlas.

- Retirar: retire o elimine las aplicaciones que ya no sean necesarias en un entorno de origen.

A

ABAC

Consulte control de acceso basado en atributos.

servicios abstractos

Consulte servicios administrados.

ACID

Consulte atomicidad, consistencia, aislamiento, durabilidad.

migración activa-activa

Método de migración de bases de datos en el que las bases de datos de origen y destino se mantienen sincronizadas (mediante una herramienta de replicación bidireccional o mediante operaciones de escritura doble) y ambas bases de datos gestionan las transacciones de las aplicaciones conectadas durante la migración. Este método permite la migración en lotes pequeños y controlados, en lugar de requerir una transición única. Es más flexible, pero requiere más trabajo que una migración activa-pasiva.

migración activa-pasiva

Método de migración de bases de datos en el que las bases de datos de origen y destino se mantienen sincronizadas, pero solo la de origen gestiona las transacciones de las aplicaciones conectadas, mientras los datos se replican en la de destino. La base de datos de destino no acepta ninguna transacción durante la migración.

función de agregación

Función SQL que actúa en un grupo de filas y calcula un único valor de devolución para el grupo. Entre los ejemplos de funciones de agregación se incluyen `SUM` y `MAX`.

IA

Consulte inteligencia artificial.

AIOps

Consulte operaciones de inteligencia artificial

anonimización

El proceso de eliminar permanentemente la información personal de un conjunto de datos. La anonimización puede ayudar a proteger la privacidad personal. Los datos anonimizados ya no se consideran datos personales.

antipatrones

Una solución que se utiliza con frecuencia para un problema recurrente en el que la solución es contraproducente, ineficaz o menos eficaz que una alternativa.

control de aplicaciones

Enfoque de seguridad que permite usar de manera exclusiva aplicaciones aprobadas para ayudar a proteger un sistema contra el malware.

cartera de aplicaciones

Recopilación de información detallada sobre cada aplicación que utiliza una organización, incluido el costo de creación y mantenimiento de la aplicación y su valor empresarial. Esta información es clave para el proceso de detección y análisis de la cartera y ayuda a identificar y priorizar las aplicaciones que se van a migrar, modernizar y optimizar.

inteligencia artificial (IA)

El campo de la informática que se dedica al uso de tecnologías informáticas para realizar funciones cognitivas que suelen estar asociadas a los seres humanos, como el aprendizaje, la resolución de problemas y el reconocimiento de patrones. Para más información, consulte ¿Qué es la inteligencia artificial?

operaciones de inteligencia artificial (AIOps)

El proceso de utilizar técnicas de machine learning para resolver problemas operativos, reducir los incidentes operativos y la intervención humana, y mejorar la calidad del servicio. Para obtener más información sobre cómo AIOps se utiliza en la estrategia de AWS migración, consulte la guía de integración de operaciones.

cifrado asimétrico

Algoritmo de cifrado que utiliza un par de claves, una clave pública para el cifrado y una clave privada para el descifrado. Puede compartir la clave pública porque no se utiliza para el descifrado, pero el acceso a la clave privada debe estar sumamente restringido.

atomicidad, consistencia, aislamiento, durabilidad (ACID)

Conjunto de propiedades de software que garantizan la validez de los datos y la fiabilidad operativa de una base de datos, incluso en caso de errores, cortes de energía u otros problemas.

control de acceso basado en atributos (ABAC)

La práctica de crear permisos detallados basados en los atributos del usuario, como el departamento, el puesto de trabajo y el nombre del equipo. Para obtener más información, consulte ABAC AWS en la documentación AWS Identity and Access Management (IAM).

origen de datos fidedigno

Ubicación en la que se almacena la versión principal de los datos, que se considera la fuente de información más fiable. Puede copiar los datos del origen de datos autorizado a otras ubicaciones con el fin de procesarlos o modificarlos, por ejemplo, anonimizarlos, redactarlos o seudonimizarlos.

Zona de disponibilidad

Una ubicación distinta dentro de una Región de AWS que está aislada de los fallos en otras zonas de disponibilidad y que proporciona una conectividad de red económica y de baja latencia a otras zonas de disponibilidad de la misma región.

AWS Marco de adopción de la nube (AWS CAF)

Un marco de directrices y mejores prácticas AWS para ayudar a las organizaciones a desarrollar un plan eficiente y eficaz para migrar con éxito a la nube. AWS CAF organiza la orientación en seis áreas de enfoque denominadas perspectivas: negocios, personas, gobierno, plataforma, seguridad y operaciones. Las perspectivas empresariales, humanas y de gobernanza se centran en las habilidades y los procesos empresariales; las perspectivas de plataforma, seguridad y operaciones se centran en las habilidades y los procesos técnicos. Por ejemplo, la perspectiva humana se dirige a las partes interesadas que se ocupan de los Recursos Humanos (RR. HH.), las funciones del personal y la administración de las personas. Desde esta perspectiva, AWS CAF proporciona orientación para el desarrollo, la formación y la comunicación de las personas a fin de preparar a la organización para una adopción exitosa de la nube. Para obtener más información, consulte la Página web de AWS CAF y el Documento técnico de AWS CAF.

AWS Marco de calificación de la carga de trabajo (AWS WQF)

Herramienta que evalúa las cargas de trabajo de migración de bases de datos, recomienda estrategias de migración y proporciona estimaciones de trabajo. AWS WQF se incluye con AWS

Schema Conversion Tool ().AWS SCT Analiza los esquemas de bases de datos y los objetos de código, el código de las aplicaciones, las dependencias y las características de rendimiento y proporciona informes de evaluación.

B

bot malicioso

Bot destinado a causar interrupciones o daños a personas u organizaciones.

BCP

Consulte planificación de la continuidad del negocio.

gráfico de comportamiento

Una vista unificada e interactiva del comportamiento de los recursos y de las interacciones a lo largo del tiempo. Puede utilizar un gráfico de comportamiento con Amazon Detective para examinar los intentos de inicio de sesión fallidos, las llamadas sospechosas a la API y acciones similares. Para obtener más información, consulte Datos en un gráfico de comportamiento en la documentación de Detective.

sistema big-endian

Un sistema que almacena primero el byte más significativo. Consulte también endianidad.

clasificación binaria

Un proceso que predice un resultado binario (una de las dos clases posibles). Por ejemplo, es posible que su modelo de ML necesite predecir problemas como “¿Este correo electrónico es spam o no es spam?” o “¿Este producto es un libro o un automóvil?”.

filtro de floración

Estructura de datos probabilística y eficiente en términos de memoria que se utiliza para comprobar si un elemento es miembro de un conjunto.

implementación azul/verde

Estrategia de implementación en la que se crean dos entornos separados, pero idénticos. La versión actual de la aplicación se ejecuta en un entorno (azul) y la nueva versión de la aplicación se ejecuta en el otro entorno (verde). Esta estrategia lo ayuda a hacer reversiones rápidas con un impacto mínimo.

bot

Aplicación de software que ejecuta tareas automatizadas a través de Internet y simula la actividad o interacción humana. Algunos bots son útiles o beneficiosos, como los rastreadores web que indexan la información de Internet. Otros bots, conocidos como bots maliciosos, tienen como objetivo causar interrupciones o daños a personas u organizaciones.

botnet

Redes de bots infectadas por malware y que están bajo el control de una sola parte, conocida como pastor de bots u operador de bots. Las botnets son el mecanismo más conocido para escalar los bots y su impacto.

branch

Área contenida de un repositorio de código. La primera rama que se crea en un repositorio es la rama principal. Puede crear una rama nueva a partir de una rama existente y, a continuación, desarrollar características o corregir errores en la rama nueva. Una rama que se genera para crear una característica se denomina comúnmente rama de característica. Cuando la característica se encuentra lista para su lanzamiento, se vuelve a combinar la rama de característica con la rama principal. Para obtener más información, consulte Acerca de las sucursales (GitHub documentación).

acceso de emergencia

En circunstancias excepcionales y mediante un proceso aprobado, es una forma rápida de que un usuario pueda acceder a un Cuenta de AWS sitio al que normalmente no tiene permisos de acceso. Para más información, consulte el indicador Implement break-glass procedures en la guía de AWS Well-Architected.

estrategia de implementación sobre infraestructura existente

La infraestructura existente en su entorno. Al adoptar una estrategia de implementación sobre infraestructura existente para una arquitectura de sistemas, se diseña la arquitectura en función de las limitaciones de los sistemas y la infraestructura actuales. Si está ampliando la infraestructura existente, puede combinar las estrategias de implementación sobre infraestructuras existentes y de implementación desde cero.

caché de búfer

El área de memoria donde se almacenan los datos a los que se accede con más frecuencia.

capacidad empresarial

Lo que hace una empresa para generar valor (por ejemplo, ventas, servicio al cliente o marketing). Las arquitecturas de microservicios y las decisiones de desarrollo pueden estar impulsadas por las capacidades empresariales. Para obtener más información, consulte la sección Organizado en torno a las capacidades empresariales del documento técnico Ejecutar microservicios en contenedores en AWS.

planificación de la continuidad del negocio (BCP)

Plan que aborda el posible impacto de un evento disruptivo, como una migración a gran escala en las operaciones y permite a la empresa reanudar las operaciones rápidamente.

C

CAF

Consulte AWS Cloud Adoption Framework.

implementación canario

Lanzamiento lento e incremental de una versión para los usuarios finales. Cuando tenga mayor confianza en la nueva versión, la implementa y reemplaza la versión actual en su totalidad.

CCoE

Consulte Centro de excelencia en la nube.

CDC

Consulte captura de datos de cambios.

captura de datos de cambio (CDC)

Proceso de seguimiento de los cambios en un origen de datos, como una tabla de base de datos, y registro de los metadatos relacionados con el cambio. Puede utilizar los CDC para diversos fines, como auditar o replicar los cambios en un sistema de destino para mantener la sincronización.

ingeniería del caos

Introducción intencionada de fallos o eventos disruptivos para poner a prueba la resiliencia de un sistema. Puedes usar AWS Fault Injection Service (AWS FIS) para realizar experimentos que estresen tus AWS cargas de trabajo y evalúen su respuesta.

CI/CD

Consulte integración continua y entrega continua.

clasificación

Un proceso de categorización que permite generar predicciones. Los modelos de ML para problemas de clasificación predicen un valor discreto. Los valores discretos siempre son distintos entre sí. Por ejemplo, es posible que un modelo necesite evaluar si hay o no un automóvil en una imagen.

cifrado del cliente

Cifrado de datos localmente, antes de que el objetivo los Servicio de AWS reciba.

Centro de excelencia en la nube (CCoE)

Equipo multidisciplinario que impulsa los esfuerzos de adopción de la nube en toda la organización, incluido el desarrollo de las prácticas recomendadas en la nube, la movilización de recursos, el establecimiento de plazos de migración y la dirección de la organización durante las transformaciones a gran escala. Para obtener más información, consulte las publicaciones de CCo E en el blog de estrategia Nube de AWS empresarial.

computación en la nube

La tecnología en la nube que se utiliza normalmente para la administración de dispositivos de IoT y el almacenamiento de datos de forma remota. La computación en la nube suele estar relacionada con la tecnología de computación de periferia.

modelo operativo en la nube

En una organización de TI, el modelo operativo que se utiliza para crear, madurar y optimizar uno o más entornos de nube. Para obtener más información, consulte Creación de su modelo operativo de nube.

etapas de adopción de la nube

Las siguientes son las cuatro fases por las que suelen pasar las empresas cuando migran a la Nube de AWS:

- Proyecto: ejecución de algunos proyectos relacionados con la nube con fines de prueba de concepto y aprendizaje
- Fundamento: realizar inversiones fundamentales para escalar su adopción de la nube (p. ej., crear una landing zone, definir una CCo E, establecer un modelo de operaciones)

- Migración: migración de aplicaciones individuales
- Reinvención: optimización de productos y servicios e innovación en la nube

Stephen Orban definió estas etapas en la entrada del blog The Journey Toward Cloud-First & the Stages of Adoption en el blog Nube de AWS Enterprise Strategy. Para obtener información sobre su relación con la estrategia de AWS migración, consulte la guía de preparación para la migración.

CMDB

Consulte base de datos de administración de configuración.

repositorio de código

Una ubicación donde el código fuente y otros activos, como documentación, muestras y scripts, se almacenan y actualizan mediante procesos de control de versiones. Algunos repositorios en la nube comunes son GitHub o Bitbucket Cloud. Cada versión del código se denomina rama. En una estructura de microservicios, cada repositorio se encuentra dedicado a una única funcionalidad. Una sola canalización de CI/CD puede utilizar varios repositorios.

caché en frío

Una caché de búfer que está vacía no está bien poblada o contiene datos obsoletos o irrelevantes. Esto afecta al rendimiento, ya que la instancia de la base de datos debe leer desde la memoria principal o el disco, lo que es más lento que leer desde la memoria caché del búfer.

datos fríos

Datos a los que se accede con poca frecuencia y que suelen ser históricos. Al consultar este tipo de datos, normalmente se aceptan consultas lentas. Trasladar estos datos a niveles o clases de almacenamiento de menor rendimiento y menos costosos puede reducir los costos.

visión artificial (CV)

Campo de la IA que utiliza el machine learning para analizar y extraer información de formatos visuales, como imágenes y videos digitales. Por ejemplo, Amazon SageMaker AI proporciona algoritmos de procesamiento de imágenes para CV.

deriva de configuración

En el caso de una carga de trabajo, un cambio en la configuración con respecto al estado esperado. Podría provocar que la carga de trabajo deje de cumplir las normas y, por lo general, es gradual e involuntaria.

base de datos de administración de configuración (CMDB)

Repositorio que almacena y administra información sobre una base de datos y su entorno de TI, incluidos los componentes de hardware y software y sus configuraciones. Por lo general, los datos de una CMDB se utilizan en la etapa de detección y análisis de la cartera de productos durante la migración.

paquete de conformidad

Un conjunto de AWS Config reglas y medidas correctivas que puede reunir para personalizar sus controles de conformidad y seguridad. Puede implementar un paquete de conformidad como una entidad única en una región Cuenta de AWS y, o en una organización, mediante una plantilla YAML. Para obtener más información, consulta los paquetes de conformidad en la documentación. AWS Config

integración y entrega continuas (CI/CD)

El proceso de automatización de las etapas de origen, compilación, prueba, puesta en escena y producción del proceso de publicación del software. CI/CD se describe comúnmente como una canalización. CI/CD puede ayudarlo a automatizar los procesos, mejorar la productividad, mejorar la calidad del código y entregar más rápido. Para obtener más información, consulte Beneficios de la entrega continua. CD también puede significar implementación continua. Para obtener más información, consulte Entrega continua frente a implementación continua.

CV

Consulte visión artificial.

D

datos en reposo

Datos que están estacionarios en la red, como los datos que se encuentran almacenados.

clasificación de datos

Un proceso para identificar y clasificar los datos de su red en función de su importancia y sensibilidad. Es un componente fundamental de cualquier estrategia de administración de riesgos de ciberseguridad porque lo ayuda a determinar los controles de protección y retención adecuados para los datos. La clasificación de datos es un componente del pilar de seguridad

del AWS Well-Architected Framework. Para obtener más información, consulte Clasificación de datos.

deriva de datos

Una variación significativa entre los datos de producción y los datos que se utilizaron para entrenar un modelo de machine learning, o un cambio significativo en los datos de entrada a lo largo del tiempo. La deriva de datos puede reducir la calidad, la precisión y la imparcialidad generales de las predicciones de los modelos de machine learning.

datos en tránsito

Datos que se mueven de forma activa por la red, por ejemplo, entre los recursos de la red.

malla de datos

Marco de arquitectura que proporciona una propiedad de datos distribuida y descentralizada con una administración y una gobernanza centralizadas.

minimización de datos

El principio de recopilar y procesar solo los datos estrictamente necesarios. Practicar la minimización de los datos Nube de AWS puede reducir los riesgos de privacidad, los costos y la huella de carbono de la analítica.

perímetro de datos

Un conjunto de barreras preventivas en su AWS entorno que ayudan a garantizar que solo las identidades confiables accedan a los recursos confiables desde las redes esperadas. Para obtener más información, consulte Crear un perímetro de datos sobre. AWS

preprocesamiento de datos

Transformar los datos sin procesar en un formato que su modelo de ML pueda analizar fácilmente. El preprocesamiento de datos puede implicar eliminar determinadas columnas o filas y corregir los valores faltantes, incoherentes o duplicados.

procedencia de los datos

El proceso de rastrear el origen y el historial de los datos a lo largo de su ciclo de vida, por ejemplo, la forma en que se generaron, transmitieron y almacenaron los datos.

titular de los datos

Persona cuyos datos se recopilan y procesan.

almacenamiento de datos

Sistema de administración de datos que respalda la inteligencia empresarial, como los análisis. Los almacenes de datos suelen contener grandes cantidades de datos históricos y, por lo general, se utilizan para las consultas y los análisis.

lenguaje de definición de datos (DDL)

Instrucciones o comandos para crear o modificar la estructura de tablas y objetos de una base de datos.

lenguaje de manipulación de datos (DML)

Instrucciones o comandos para modificar (insertar, actualizar y eliminar) la información de una base de datos.

DDL

Consulte lenguaje de definición de bases de datos.

conjunto profundo

Combinar varios modelos de aprendizaje profundo para la predicción. Puede utilizar conjuntos profundos para obtener una predicción más precisa o para estimar la incertidumbre de las predicciones.

aprendizaje profundo

Un subcampo del ML que utiliza múltiples capas de redes neuronales artificiales para identificar el mapeo entre los datos de entrada y las variables objetivo de interés.

defense-in-depth

Un enfoque de seguridad de la información en el que se distribuyen cuidadosamente una serie de mecanismos y controles de seguridad en una red informática para proteger la confidencialidad, la integridad y la disponibilidad de la red y de los datos que contiene. Al adoptar esta estrategia AWS, se añaden varios controles en diferentes capas de la AWS Organizations estructura para ayudar a proteger los recursos. Por ejemplo, un defense-in-depth enfoque podría combinar la autenticación multifactorial, la segmentación de la red y el cifrado.

administrador delegado

En AWS Organizations, un servicio compatible puede registrar una cuenta de AWS miembro para administrar las cuentas de la organización y gestionar los permisos de ese servicio. Esta

cuenta se denomina administrador delegado para ese servicio. Para obtener más información y una lista de servicios compatibles, consulte Servicios que funcionan con AWS Organizations en la documentación de AWS Organizations .

Implementación

El proceso de hacer que una aplicación, características nuevas o correcciones de código se encuentren disponibles en el entorno de destino. La implementación abarca implementar cambios en una base de código y, a continuación, crear y ejecutar esa base en los entornos de la aplicación.

entorno de desarrollo

Consulte entorno.

control de detección

Un control de seguridad que se ha diseñado para detectar, registrar y alertar después de que se produzca un evento. Estos controles son una segunda línea de defensa, ya que lo advierten sobre los eventos de seguridad que han eludido los controles preventivos establecidos. Para obtener más información, consulte Controles de detección en Implementación de controles de seguridad en AWS.

asignación de flujos de valor para el desarrollo (DVSM)

Proceso que se utiliza para identificar y priorizar las restricciones que afectan negativamente a la velocidad y la calidad en el ciclo de vida del desarrollo de software. DVSM amplía el proceso de asignación del flujo de valor diseñado originalmente para las prácticas de fabricación ajustada. Se centra en los pasos y los equipos necesarios para crear y transferir valor a través del proceso de desarrollo de software.

gemelo digital

Representación virtual de un sistema del mundo real, como un edificio, una fábrica, un equipo industrial o una línea de producción. Los gemelos digitales son compatibles con el mantenimiento predictivo, la supervisión remota y la optimización de la producción.

tabla de dimensiones

En un esquema en estrella, tabla más pequeña que contiene los atributos de datos sobre los datos cuantitativos en una tabla de hechos. Los atributos de la tabla de dimensiones suelen ser campos de texto o números discretos que se comportan como texto. Estos atributos se suelen utilizar para restringir consultas, filtrarlas y etiquetar los conjuntos de resultados.

desastre

Un evento que impide que una carga de trabajo o un sistema cumplan sus objetivos empresariales en su ubicación principal de implementación. Estos eventos pueden ser desastres naturales, fallos técnicos o el resultado de acciones humanas, como una configuración incorrecta involuntaria o un ataque de malware.

recuperación de desastres (DR)

Estrategia y proceso que utiliza para minimizar el tiempo de inactividad y la pérdida de datos a causa de un desastre. Para obtener más información, consulte Recuperación ante desastres de cargas de trabajo en AWS: Recovery in the Cloud in the AWS Well-Architected Framework.

DML

Consulte lenguaje de manipulación de bases de datos.

diseño basado en el dominio

Un enfoque para desarrollar un sistema de software complejo mediante la conexión de sus componentes a dominios en evolución, o a los objetivos empresariales principales, a los que sirve cada componente. Este concepto lo introdujo Eric Evans en su libro, Diseño impulsado por el dominio: abordando la complejidad en el corazón del software (Boston: Addison-Wesley Professional, 2003). Para obtener información sobre cómo utilizar el diseño basado en dominios con el patrón de higos estranguladores, consulte Modernización gradual de los servicios web antiguos de Microsoft ASP.NET (ASMX) mediante contenedores y Amazon API Gateway.

DR

Consulte recuperación ante desastres.

Detección de desviaciones

Seguimiento de las desviaciones con respecto a una configuración con línea de base. Por ejemplo, puedes usarlo AWS CloudFormation para detectar desviaciones en los recursos del sistema o puedes usarlo AWS Control Tower para detectar cambios en tu landing zone que puedan afectar al cumplimiento de los requisitos de gobierno.

DVSM

Consulte asignación de flujos de valor para el desarrollo.

E

EDA

Consulte análisis de datos de tipo exploratorio.

EDI

Consulte intercambio electrónico de datos.

computación en la periferia

La tecnología que aumenta la potencia de cálculo de los dispositivos inteligentes en la periferia de una red de IoT. En comparación con la computación en la nube, la computación de periferia puede reducir la latencia de la comunicación y mejorar el tiempo de respuesta.

intercambio electrónico de datos (EDI)

Intercambio automatizado de documentos comerciales entre organizaciones. Para más información, consulte ¿Qué es el intercambio electrónico de datos?

cifrado

Proceso de computación que transforma datos de texto plano, que son legibles por humanos, en texto cifrado.

clave de cifrado

Cadena criptográfica de bits aleatorios que se genera mediante un algoritmo de cifrado. Las claves pueden variar en longitud y cada una se ha diseñado para ser impredecible y única.

endianidad

El orden en el que se almacenan los bytes en la memoria del ordenador. Los sistemas big-endianos almacenan primero el byte más significativo. Los sistemas Little-Endian almacenan primero el byte menos significativo.

punto de conexión

Consulte punto de conexión de servicio.

servicio de punto de conexión

Servicio que puede alojar en una nube privada virtual (VPC) para compartir con otros usuarios. Puede crear un servicio de punto final AWS PrivateLink y conceder permisos a otros Cuentas de AWS o a responsables AWS Identity and Access Management (de IAM). Estas cuentas o entidades principales pueden conectarse a su servicio de punto de conexión de forma privada

mediante la creación de puntos de conexión de VPC de interfaz. Para obtener más información, consulte Creación de un servicio de punto de conexión en la documentación de Amazon Virtual Private Cloud (Amazon VPC).

planificación de recursos empresariales (ERP)

Sistema que automatiza y administra los procesos empresariales clave (como la contabilidad, MES y la administración de proyectos) de una empresa.

cifrado de sobre

El proceso de cifrar una clave de cifrado con otra clave de cifrado. Para obtener más información, consulte el cifrado de sobres en la documentación de AWS Key Management Service (AWS KMS).

entorno

Una instancia de una aplicación en ejecución. Los siguientes son los tipos de entornos más comunes en la computación en la nube:

- entorno de desarrollo: instancia de una aplicación en ejecución que solo se encuentra disponible para el equipo principal responsable del mantenimiento de la aplicación. Los entornos de desarrollo se utilizan para probar los cambios antes de promocionarlos a los entornos superiores. Este tipo de entorno a veces se denomina entorno de prueba.
- entornos inferiores: todos los entornos de desarrollo de una aplicación, como los que se utilizan para las compilaciones y pruebas iniciales.
- entorno de producción: instancia de una aplicación en ejecución a la que pueden acceder los usuarios finales. En un CI/CD proceso, el entorno de producción es el último entorno de implementación.
- entornos superiores: todos los entornos a los que pueden acceder usuarios que no sean del equipo de desarrollo principal. Esto puede incluir un entorno de producción, entornos de preproducción y entornos para las pruebas de aceptación por parte de los usuarios.

epopeya

En las metodologías ágiles, son categorías funcionales que ayudan a organizar y priorizar el trabajo. Las epopeyas brindan una descripción detallada de los requisitos y las tareas de implementación. Por ejemplo, las epopeyas AWS de seguridad de CAF incluyen la gestión de identidades y accesos, los controles de detección, la seguridad de la infraestructura, la protección de datos y la respuesta a incidentes. Para obtener más información sobre las epopeyas en la estrategia de migración de AWS , consulte la Guía de implementación del programa.

ERP

Consulte planificación de recursos empresariales.

análisis de datos de tipo exploratorio (EDA)

El proceso de analizar un conjunto de datos para comprender sus características principales. Se recopilan o agregan datos y, a continuación, se realizan las investigaciones iniciales para encontrar patrones, detectar anomalías y comprobar las suposiciones. El EDA se realiza mediante el cálculo de estadísticas resumidas y la creación de visualizaciones de datos.

F

tabla de hechos

Tabla central de un esquema en estrella. Almacena datos cuantitativos sobre operaciones empresariales. Por lo general, una tabla de hechos contiene dos tipos de columnas: las que contienen medidas y las que contienen una clave externa para una tabla de dimensiones.

Fail Fast

Filosofía que utiliza pruebas frecuentes e incrementales para reducir el ciclo de vida del desarrollo. Es una parte fundamental de los enfoques ágiles.

límite de aislamiento de errores

En el Nube de AWS, un límite, como una zona de disponibilidad Región de AWS, un plano de control o un plano de datos, que limita el efecto de una falla y ayuda a mejorar la resiliencia de las cargas de trabajo. Para más información, consulte AWS Fault Isolation Boundaries.

rama de característica

Consulte rama.

características

Los datos de entrada que se utilizan para hacer una predicción. Por ejemplo, en un contexto de fabricación, las características pueden ser imágenes que se capturan periódicamente desde la línea de fabricación.

importancia de las características

La importancia que tiene una característica para las predicciones de un modelo. Por lo general, esto se expresa como una puntuación numérica que se puede calcular mediante diversas

técnicas, como las explicaciones aditivas de Shapley (SHAP) y los gradientes integrados. Para obtener más información, consulte Interpretabilidad del modelo de aprendizaje automático con. AWS

transformación de funciones

Optimizar los datos para el proceso de ML, lo que incluye enriquecer los datos con fuentes adicionales, escalar los valores o extraer varios conjuntos de información de un solo campo de datos. Esto permite que el modelo de ML se beneficie de los datos. Por ejemplo, si divide la fecha del “27 de mayo de 2021 00:15:37” en “jueves”, “mayo”, “2021” y “15”, puede ayudar al algoritmo de aprendizaje a aprender patrones matizados asociados a los diferentes componentes de los datos.

peticiones con pocos pasos

Proporcionar a un LLM una pequeña cantidad de ejemplos que demuestren la tarea y el resultado deseado antes de pedirle que lleve a cabo una tarea similar. Esta técnica es una aplicación del aprendizaje contextual, mediante el que los modelos aprenden a partir de ejemplos (pasos) incrustados en las peticiones. La técnica de peticiones con pocos pasos puede ser eficaz para las tareas que requieren un formato, un razonamiento o un conocimiento del dominio específicos. Consulte también peticiones desde cero.

FGAC

Consulte control de acceso detallado.

control de acceso preciso (FGAC)

El uso de varias condiciones que tienen por objetivo permitir o denegar una solicitud de acceso.

migración relámpago

Método de migración de bases de datos que utiliza la replicación continua de datos mediante la captura de datos de cambio para migrar los datos en el menor tiempo posible, en lugar de utilizar un enfoque gradual. El objetivo es reducir al mínimo el tiempo de inactividad.

FM

Consulte modelo fundacional.

Modelo fundacional (FM)

Una gran red neuronal de aprendizaje profundo que se ha estado entrenando con conjuntos de datos masivos de datos generalizados y sin etiquetar. FMs son capaces de realizar una

amplia variedad de tareas generales, como comprender el lenguaje, generar texto e imágenes y conversar en lenguaje natural. Para más información, consulte ¿Qué son los modelos fundacionales?

G

IA generativa

Subconjunto de modelos de IA que se entrenaron con grandes cantidades de datos y que pueden utilizar una simple petición de texto para crear contenido y artefactos nuevos, como imágenes, videos, texto y audio. Para más información, consulte ¿Qué es la IA generativa?

bloqueo geográfico

Consulte restricciones geográficas.

restricciones geográficas (bloqueo geográfico)

En Amazon CloudFront, una opción para impedir que los usuarios de países específicos accedan a las distribuciones de contenido. Puede utilizar una lista de permitidos o bloqueados para especificar los países aprobados y prohibidos. Para obtener más información, consulta la sección Restringir la distribución geográfica del contenido en la CloudFront documentación.

Flujo de trabajo de Gitflow

Un enfoque en el que los entornos inferiores y superiores utilizan diferentes ramas en un repositorio de código fuente. El flujo de trabajo de Gitflow se considera heredado, mientras que el flujo de trabajo basado en enlaces troncales es el enfoque moderno preferido.

imagen dorada

Instantánea de un sistema o software que se usa como plantilla para implementar nuevas instancias de ese sistema o software. Por ejemplo, en la fabricación, una imagen dorada se puede utilizar para aprovisionar software en varios dispositivos y ayuda a mejorar la velocidad, la escalabilidad y la productividad de las operaciones de fabricación de dispositivos.

estrategia de implementación desde cero

La ausencia de infraestructura existente en un entorno nuevo. Al adoptar una estrategia de implementación desde cero para una arquitectura de sistemas, puede seleccionar todas las tecnologías nuevas sin que estas deban ser compatibles con una infraestructura existente, lo que también se conoce como implementación sobre infraestructura existente. Si está

ampliando la infraestructura existente, puede combinar las estrategias de implementación sobre infraestructuras existentes y de implementación desde cero.

barrera de protección

Una regla de alto nivel que ayuda a regular los recursos, las políticas y el cumplimiento en todas las unidades organizativas (OUs). Las barreras de protección preventivas aplican políticas para garantizar la alineación con los estándares de conformidad. Se implementan mediante políticas de control de servicios y límites de permisos de IAM. Las barreras de protección de detección detectan las vulneraciones de las políticas y los problemas de conformidad, y generan alertas para su corrección. Se implementan mediante Amazon AWS Config AWS Security Hub CSPM GuardDuty AWS Trusted Advisor, Amazon Inspector y AWS Lambda cheques personalizados.

H

HA

Consulte alta disponibilidad.

migración heterogénea de bases de datos

Migración de la base de datos de origen a una base de datos de destino que utilice un motor de base de datos diferente (por ejemplo, de Oracle a Amazon Aurora). La migración heterogénea suele ser parte de un esfuerzo de rediseño de la arquitectura y convertir el esquema puede ser una tarea compleja. AWS ofrece AWS SCT, lo cual ayuda con las conversiones de esquemas.

alta disponibilidad (HA)

La capacidad de una carga de trabajo para funcionar de forma continua, sin intervención, en caso de desafíos o desastres. Los sistemas de alta disponibilidad están diseñados para realizar una conmutación por error automática, ofrecer un rendimiento de alta calidad de forma constante y gestionar diferentes cargas y fallos con un impacto mínimo en el rendimiento.

modernización histórica

Un enfoque utilizado para modernizar y actualizar los sistemas de tecnología operativa (TO) a fin de satisfacer mejor las necesidades de la industria manufacturera. Un histórico es un tipo de base de datos que se utiliza para recopilar y almacenar datos de diversas fuentes en una fábrica.

datos de reserva

Parte de los datos históricos etiquetados que se ocultan de un conjunto de datos que se utiliza para entrenar un modelo de machine learning. Puede utilizar los datos de reserva para evaluar el rendimiento del modelo mediante la comparación de las predicciones del modelo con los datos de reserva.

migración homogénea de bases de datos

Migración de la base de datos de origen a una base de datos de destino que comparte el mismo motor de base de datos (por ejemplo, Microsoft SQL Server a Amazon RDS para SQL Server). La migración homogénea suele formar parte de un esfuerzo para volver a alojar o redefinir la plataforma. Puede utilizar las utilidades de bases de datos nativas para migrar el esquema.

datos recientes

Datos a los que se accede con frecuencia, como datos en tiempo real o datos traslacionales recientes. Por lo general, estos datos requieren un nivel o una clase de almacenamiento de alto rendimiento para proporcionar respuestas rápidas a las consultas.

hotfix

Una solución urgente para un problema crítico en un entorno de producción. Debido a su urgencia, una revisión suele realizarse fuera del flujo de trabajo de DevOps publicación típico.

periodo de hiperatención

Periodo, inmediatamente después de la transición, durante el cual un equipo de migración administra y monitorea las aplicaciones migradas en la nube para solucionar cualquier problema. Por lo general, este periodo dura de 1 a 4 días. Al final del periodo de hiperatención, el equipo de migración suele transferir la responsabilidad de las aplicaciones al equipo de operaciones en la nube.

I

IaC

Consulte infraestructura como código.

políticas basadas en identidades

Política asociada a uno o más directores de IAM que define sus permisos en el entorno. Nube de AWS

aplicación inactiva

Aplicación que utiliza un promedio de CPU y memoria de entre 5 y 20 por ciento durante un periodo de 90 días. En un proyecto de migración, es habitual retirar estas aplicaciones o mantenerlas en las instalaciones.

IIoT

Consulte Internet de las cosas industrial.

infraestructura inmutable

Modelo que implementa una nueva infraestructura para las cargas de trabajo de producción en lugar de actualizar o modificar la infraestructura existente o aplicarle revisiones. Las infraestructuras inmutables son de manera intrínseca más coherentes, fiables y predecibles que las infraestructuras mutables. Para más información, consulte la práctica recomendada Implementación mediante una infraestructura inmutable en el Marco de AWS Well-Architected.

VPC entrante (de entrada)

En una arquitectura de AWS cuentas múltiples, una VPC que acepta, inspecciona y enruta las conexiones de red desde fuera de una aplicación. La arquitectura AWS de referencia de seguridad recomienda configurar la cuenta de red con entradas, salidas e inspección VPCs para proteger la interfaz bidireccional entre la aplicación y el resto de Internet.

migración gradual

Estrategia de transición en la que se migra la aplicación en partes pequeñas en lugar de realizar una transición única y completa. Por ejemplo, puede trasladar inicialmente solo unos pocos microservicios o usuarios al nuevo sistema. Tras comprobar que todo funciona correctamente, puede trasladar microservicios o usuarios adicionales de forma gradual hasta que pueda retirar su sistema heredado. Esta estrategia reduce los riesgos asociados a las grandes migraciones.

Industria 4.0

Término que introdujo Klaus Schwab en 2016 para referirse a la modernización de los procesos de fabricación mediante los avances en la conectividad, los datos en tiempo real, la automatización, el análisis, la IA y el ML.

infraestructura

Todos los recursos y activos que se encuentran en el entorno de una aplicación.

infraestructura como código (IaC)

Proceso de aprovisionamiento y administración de la infraestructura de una aplicación mediante un conjunto de archivos de configuración. La IaC se ha diseñado para ayudarlo a centralizar la administración de la infraestructura, estandarizar los recursos y escalar con rapidez a fin de que los entornos nuevos sean repetibles, fiables y consistentes.

Internet de las cosas industrial (T) IIo

El uso de sensores y dispositivos conectados a Internet en los sectores industriales, como el productivo, el eléctrico, el automotriz, el sanitario, el de las ciencias de la vida y el de la agricultura. Para obtener más información, consulte Creación de una estrategia de transformación digital de la Internet de las cosas (IIoT) industrial.

VPC de inspección

En una arquitectura de AWS cuentas múltiples, una VPC centralizada que gestiona las inspecciones del tráfico de red VPCs entre Internet y las redes locales (en una misma o Regiones de AWS diferente). La arquitectura AWS de referencia de seguridad recomienda configurar su cuenta de red con entrada, salida e inspección VPCs para proteger la interfaz bidireccional entre la aplicación e Internet en general.

Internet de las cosas (IoT)

Red de objetos físicos conectados con sensores o procesadores integrados que se comunican con otros dispositivos y sistemas a través de Internet o de una red de comunicación local. Para obtener más información, consulte ¿Qué es IoT?.

interpretabilidad

Característica de un modelo de machine learning que describe el grado en que un ser humano puede entender cómo las predicciones del modelo dependen de sus entradas. Para obtener más información, consulte Interpretabilidad del modelo de aprendizaje automático con. AWS

IoT

Consulte Internet de las cosas.

biblioteca de información de TI (ITIL)

Conjunto de prácticas recomendadas para ofrecer servicios de TI y alinearlos con los requisitos empresariales. La ITIL proporciona la base para la ITSM.

administración de servicios de TI (ITSM)

Actividades asociadas con el diseño, la implementación, la administración y el soporte de los servicios de TI para una organización. Para obtener información sobre la integración de las operaciones en la nube con las herramientas de ITSM, consulte la Guía de integración de operaciones.

ITIL

Consulte biblioteca de información de TI.

ITSM

Consulte administración de servicios de TI.

L

control de acceso basado en etiquetas (LBAC)

Una implementación del control de acceso obligatorio (MAC) en la que a los usuarios y a los propios datos se les asigna explícitamente un valor de etiqueta de seguridad. La intersección entre la etiqueta de seguridad del usuario y la etiqueta de seguridad de los datos determina qué filas y columnas puede ver el usuario.

zona de aterrizaje

Una landing zone es un AWS entorno multicuenta bien diseñado, escalable y seguro. Este es un punto de partida desde el cual las empresas pueden lanzar e implementar rápidamente cargas de trabajo y aplicaciones con confianza en su entorno de seguridad e infraestructura. Para obtener más información sobre las zonas de aterrizaje, consulte Configuración de un entorno de AWS seguro y escalable con varias cuentas.

modelo de lenguaje de gran tamaño (LLM)

Modelo de IA de aprendizaje profundo que se entrenó previamente con una gran cantidad de datos. Un LLM puede llevar a cabo varias tareas, como responder preguntas, resumir documentos, traducir textos a otros idiomas y completar oraciones. Para obtener más información, consulte Qué son. LLMs

migración grande

Migración de 300 servidores o más.

LBAC

Consulte control de acceso basado en etiquetas.

privilegio mínimo

La práctica recomendada de seguridad que consiste en conceder los permisos mínimos necesarios para realizar una tarea. Para obtener más información, consulte Aplicar permisos de privilegio mínimo en la documentación de IAM.

migrar mediante lift-and-shift

Consulte Las 7 R.

sistema little-endian

Un sistema que almacena primero el byte menos significativo. Consulte también endianidad.

LLM

Consulte modelo de lenguaje de gran tamaño.

entornos inferiores

Consulte entorno.

M

machine learning (ML)

Un tipo de inteligencia artificial que utiliza algoritmos y técnicas para el reconocimiento y el aprendizaje de patrones. El ML analiza y aprende de los datos registrados, como los datos del Internet de las cosas (IoT), para generar un modelo estadístico basado en patrones. Para más información, consulte Machine learning.

rama principal

Consulte rama.

malware

Software diseñado para comprometer la seguridad o la privacidad de la computadora. El malware podría interrumpir los sistemas informáticos, filtrar información confidencial u obtener acceso

no autorizado. Algunos ejemplos de malware son los virus, los gusanos, el ransomware, los troyanos, el spyware y los registradores de pulsaciones de teclas.

Servicios administrados

Servicios de AWS para lo cual AWS opera la capa de infraestructura, el sistema operativo y las plataformas, y se accede a los puntos finales para almacenar y recuperar datos. Amazon Simple Storage Service (Amazon S3) y Amazon DynamoDB son ejemplos de servicios administrados. También se conocen como servicios abstractos.

sistema de ejecución de fabricación (MES)

Sistema de software para seguir, supervisar, documentar y controlar los procesos de producción que convierten las materias primas en productos acabados en la zona de producción.

MAP

Consulte Programa de aceleración de la migración.

mecanismo

Proceso completo mediante el que se crea una herramienta, se impulsa su adopción y, a continuación, se inspeccionan los resultados para hacer ajustes. Un mecanismo es un ciclo que se refuerza y mejora por sí mismo a medida que funciona. Para obtener más información, consulte Creación de mecanismos en el AWS Well-Architected Framework.

cuenta de miembro

Todas las Cuentas de AWS demás cuentas, excepto la de administración, que forman parte de una organización. AWS Organizations Una cuenta no puede pertenecer a más de una organización a la vez.

MES

Consulte sistema de ejecución de fabricación.

Message Queuing Telemetry Transport (MQTT)

Un protocolo de comunicación ligero machine-to-machine (M2M), basado en el patrón de publicación/suscripción, para dispositivos de IoT con recursos limitados.

microservicio

Un servicio pequeño e independiente que se comunica a través de una red bien definida APIs y que, por lo general, es propiedad de equipos pequeños e independientes. Por ejemplo,

un sistema de seguros puede incluir microservicios que se adapten a las capacidades empresariales, como las de ventas o marketing, o a subdominios, como las de compras, reclamaciones o análisis. Los beneficios de los microservicios incluyen la agilidad, la escalabilidad flexible, la facilidad de implementación, el código reutilizable y la resiliencia. Para obtener más información, consulte Integrar microservicios mediante AWS servicios sin servidor.

arquitectura de microservicios

Un enfoque para crear una aplicación con componentes independientes que ejecutan cada proceso de la aplicación como un microservicio. Estos microservicios se comunican a través de una interfaz bien definida mediante un uso ligero. APIs Cada microservicio de esta arquitectura se puede actualizar, implementar y escalar para satisfacer la demanda de funciones específicas de una aplicación. Para obtener más información, consulte Implementación de microservicios en. AWS

Programa de aceleración de la migración (MAP)

Un AWS programa que proporciona soporte de consultoría, formación y servicios para ayudar a las organizaciones a crear una base operativa sólida para migrar a la nube y para ayudar a compensar el costo inicial de las migraciones. El MAP incluye una metodología de migración para ejecutar las migraciones antiguas de forma metódica y un conjunto de herramientas para automatizar y acelerar los escenarios de migración más comunes.

migración a escala

Proceso de transferencia de la mayoría de la cartera de aplicaciones a la nube en oleadas, con más aplicaciones desplazadas a un ritmo más rápido en cada oleada. En esta fase, se utilizan las prácticas recomendadas y las lecciones aprendidas en las fases anteriores para implementar una fábrica de migración de equipos, herramientas y procesos con el fin de agilizar la migración de las cargas de trabajo mediante la automatización y la entrega ágil. Esta es la tercera fase de la estrategia de migración de AWS.

fábrica de migración

Equipos multifuncionales que agilizan la migración de las cargas de trabajo mediante enfoques automatizados y ágiles. Los equipos de las fábricas de migración suelen incluir a analistas y propietarios de operaciones, empresas, ingenieros de migración, desarrolladores y DevOps profesionales que trabajan a pasos agigantados. Entre el 20 y el 50 por ciento de la cartera de aplicaciones empresariales se compone de patrones repetidos que pueden optimizarse mediante un enfoque de fábrica. Para obtener más información, consulte la discusión sobre las fábricas de migración y la Guía de fábricas de migración a la nube en este contenido.

metadatos de migración

Información sobre la aplicación y el servidor que se necesita para completar la migración. Cada patrón de migración requiere un conjunto diferente de metadatos de migración. Algunos ejemplos de metadatos de migración son la subred de destino, el grupo de seguridad y AWS la cuenta.

patrón de migración

Tarea de migración repetible que detalla la estrategia de migración, el destino de la migración y la aplicación o el servicio de migración utilizados. Ejemplo: rehospede la migración a Amazon EC2 AWS con Application Migration Service.

Migration Portfolio Assessment (MPA)

Herramienta en línea que proporciona información a fin de validar los argumentos comerciales necesarios para migrar a la Nube de AWS. La MPA ofrece una evaluación detallada de la cartera (adecuación del tamaño de los servidores, precios, comparaciones del costo total de propiedad, análisis de los costos de migración), así como una planificación de la migración (análisis y recopilación de datos de aplicaciones, agrupación de aplicaciones, priorización de la migración y planificación de oleadas). La herramienta MPA (requiere iniciar sesión) está disponible de forma gratuita para todos los AWS consultores y consultores de los socios de APN.

Evaluación de la preparación para la migración (MRA)

Proceso que consiste en obtener información sobre el estado de preparación de una organización para la nube, identificar sus puntos fuertes y débiles y elaborar un plan de acción para cerrar las brechas identificadas mediante el AWS CAF. Para obtener más información, consulte la Guía de preparación para la migración. La MRA es la primera fase de la estrategia de migración de AWS.

estrategia de migración

Enfoque utilizado para migrar una carga de trabajo a la Nube de AWS. Para más información, consulte la entrada Las 7 R de este glosario y también Mobilize your organization to accelerate large-scale migrations.

ML

Consulte machine learning.

modernización

Transformar una aplicación obsoleta (antigua o monolítica) y su infraestructura en un sistema ágil, elástico y de alta disponibilidad en la nube para reducir los gastos, aumentar la eficiencia

y aprovechar las innovaciones. Para más información, consulte Strategy for modernizing applications in the Nube de AWS.

evaluación de la preparación para la modernización

Evaluación que ayuda a determinar la preparación para la modernización de las aplicaciones de una organización; identifica los beneficios, los riesgos y las dependencias; y determina qué tan bien la organización puede soportar el estado futuro de esas aplicaciones. El resultado de la evaluación es un esquema de la arquitectura objetivo, una hoja de ruta que detalla las fases de desarrollo y los hitos del proceso de modernización y un plan de acción para abordar las brechas identificadas. Para más información, consulte Evaluating modernization readiness for applications in the Nube de AWS.

aplicaciones monolíticas (monolitos)

Aplicaciones que se ejecutan como un único servicio con procesos estrechamente acoplados. Las aplicaciones monolíticas presentan varios inconvenientes. Si una característica de la aplicación experimenta un aumento en la demanda, se debe escalar toda la arquitectura. Agregar o mejorar las características de una aplicación monolítica también se vuelve más complejo a medida que crece la base de código. Para solucionar problemas con la aplicación, puede utilizar una arquitectura de microservicios. Para obtener más información, consulte Descomposición de monolitos en microservicios.

MPA

Consulte Migration Portfolio Assessment.

MQTT

Consulte Message Queuing Telemetry Transport.

clasificación multiclase

Un proceso que ayuda a generar predicciones para varias clases (predice uno de más de dos resultados). Por ejemplo, un modelo de ML podría preguntar “¿Este producto es un libro, un automóvil o un teléfono?” o “¿Qué categoría de productos es más interesante para este cliente?”.

infraestructura mutable

Modelo que actualiza y modifica la infraestructura actual para las cargas de trabajo de producción. Para mejorar la coherencia, la fiabilidad y la previsibilidad, el AWS Well-Architected Framework recomienda el uso de una infraestructura inmutable como práctica recomendada.

O

OAC

Consulte control de acceso de origen.

OAI

Consulte identidad de acceso de origen.

OCM

Consulte administración del cambio organizacional.

migración fuera de línea

Método de migración en el que la carga de trabajo de origen se elimina durante el proceso de migración. Este método implica un tiempo de inactividad prolongado y, por lo general, se utiliza para cargas de trabajo pequeñas y no críticas.

OI

Consulte integración de operaciones.

OLA

Consulte acuerdo de nivel operativo.

migración en línea

Método de migración en el que la carga de trabajo de origen se copia al sistema de destino sin que se desconecte. Las aplicaciones que están conectadas a la carga de trabajo pueden seguir funcionando durante la migración. Este método implica un tiempo de inactividad nulo o mínimo y, por lo general, se utiliza para cargas de trabajo de producción críticas.

OPC-UA

Consulte Open Process Communications: arquitectura unificada.

Open Process Communications: arquitectura unificada (OPC-UA)

Un protocolo de machine-to-machine comunicación (M2M) para la automatización industrial. OPC-UA establece un estándar de interoperabilidad con esquemas de autenticación, autorización y cifrado de datos.

acuerdo de nivel operativo (OLA)

Acuerdo que aclara lo que los grupos de TI operativos se comprometen a ofrecerse entre sí, para respaldar un acuerdo de nivel de servicio (SLA).

revisión de la preparación operativa (ORR)

Lista de comprobación de preguntas y prácticas recomendadas asociadas que son útiles para comprender, evaluar, prevenir o reducir el alcance de los incidentes y posibles errores. Para más información, consulte Operational Readiness Reviews (ORR) en el Marco de AWS Well-Architected.

tecnología operativa (TO)

Sistemas de hardware y software que funcionan con el entorno físico para controlar las operaciones, los equipos y la infraestructura industriales. En el sector de la fabricación, la integración de los sistemas de TO y tecnología de la información (TI) es un enfoque clave para las transformaciones de la industria 4.0.

integración de operaciones (OI)

Proceso de modernización de las operaciones en la nube, que implica la planificación de la preparación, la automatización y la integración. Para obtener más información, consulte la Guía de integración de las operaciones.

registro de seguimiento organizativo

Un registro creado por y AWS CloudTrail que registra todos los eventos para todos los miembros Cuentas de AWS de una organización. AWS Organizations Este registro de seguimiento se crea en cada Cuenta de AWS que forma parte de la organización y realiza un seguimiento de la actividad en cada cuenta. Para obtener más información, consulte Crear un registro para una organización en la CloudTrail documentación.

administración del cambio organizacional (OCM)

Marco para administrar las transformaciones empresariales importantes y disruptivas desde la perspectiva de las personas, la cultura y el liderazgo. La OCM ayuda a las empresas a prepararse para nuevos sistemas y estrategias y a realizar la transición a ellos, al acelerar la adopción de cambios, abordar los problemas de transición e impulsar cambios culturales y organizacionales. En la estrategia de AWS migración, este marco se denomina aceleración de personal, debido a la velocidad de cambio que requieren los proyectos de adopción de la nube. Para obtener más información, consulte la Guía de OCM.

control de acceso de origen (OAC)

En CloudFront, una opción mejorada para restringir el acceso y proteger el contenido del Amazon Simple Storage Service (Amazon S3). El OAC admite todos los buckets de S3 Regiones de AWS, el cifrado del lado del servidor AWS KMS (SSE-KMS) y las solicitudes dinámicas `PUT` y `DELETE` dirigidas al bucket de S3.

identidad de acceso de origen (OAI)

En CloudFront, una opción para restringir el acceso y proteger el contenido de Amazon S3. Cuando utiliza OAI, CloudFront crea un principal con el que Amazon S3 puede autenticarse. Los directores autenticados solo pueden acceder al contenido de un bucket de S3 a través de una distribución específica. CloudFront Consulte también el OAC, que proporciona un control de acceso más detallado y mejorado.

ORR

Consulte revisión de la preparación operativa.

OT

Consulte tecnología operativa.

VPC saliente (de salida)

En una arquitectura de AWS cuentas múltiples, una VPC que gestiona las conexiones de red que se inician desde una aplicación. La arquitectura AWS de referencia de seguridad recomienda configurar la cuenta de red con entradas, salidas e inspección VPCs para proteger la interfaz bidireccional entre la aplicación e Internet en general.

P

límite de permisos

Una política de administración de IAM que se adjunta a las entidades principales de IAM para establecer los permisos máximos que puede tener el usuario o el rol. Para obtener más información, consulte Límites de permisos en la documentación de IAM.

información de identificación personal (PII)

Información que, vista directamente o combinada con otros datos relacionados, puede utilizarse para deducir de manera razonable la identidad de una persona. Algunos ejemplos de información de identificación personal son los nombres, las direcciones y la información de contacto.

PII

Consulte información de identificación personal.

manual de estrategias

Conjunto de pasos predefinidos que capturan el trabajo asociado a las migraciones, como la entrega de las funciones de operaciones principales en la nube. Un manual puede adoptar la forma de scripts, manuales de procedimientos automatizados o resúmenes de los procesos o pasos necesarios para operar un entorno modernizado.

PLC

Consulte controlador lógico programable.

PLM

Consulte administración del ciclo de vida del producto.

policy

Objeto que puede definir permisos (consulte política basada en identidad), especificar las condiciones de acceso (consulte política basada en recursos) o definir los permisos máximos para todas las cuentas de una organización de AWS Organizations (consulte política de control de servicio).

persistencia políglota

Elegir de forma independiente la tecnología de almacenamiento de datos de un microservicio en función de los patrones de acceso a los datos y otros requisitos. Si sus microservicios tienen la misma tecnología de almacenamiento de datos, pueden enfrentarse a desafíos de implementación o experimentar un rendimiento deficiente. Los microservicios se implementan más fácilmente y logran un mejor rendimiento y escalabilidad si utilizan el almacén de datos que mejor se adapte a sus necesidades.

evaluación de cartera

Proceso de detección, análisis y priorización de la cartera de aplicaciones para planificar la migración. Para obtener más información, consulte la Evaluación de la preparación para la migración.

predicate

Condición de consulta que devuelve `true` o `false`. En general, se encuentra en una cláusula `WHERE`.

inserción de predicados

Técnica de optimización de consultas en bases de datos que filtra los datos de la consulta antes de transferirlos. Esta técnica reduce la cantidad de datos de la base de datos relacional que se tienen que recuperar y procesar. Además, mejora el rendimiento de las consultas.

control preventivo

Un control de seguridad diseñado para evitar que ocurra un evento. Estos controles son la primera línea de defensa para evitar el acceso no autorizado o los cambios no deseados en la red. Para obtener más información, consulte Controles preventivos en Implementación de controles de seguridad en AWS.

entidad principal

Una entidad AWS que puede realizar acciones y acceder a los recursos. Esta entidad suele ser un usuario raíz para un Cuenta de AWS rol de IAM o un usuario. Para obtener más información, consulte Entidad principal en Términos y conceptos de roles en la documentación de IAM.

Privacidad desde el diseño

Enfoque de ingeniería de sistemas que tiene en cuenta la privacidad durante todo el proceso de desarrollo.

zonas alojadas privadas

Un contenedor que contiene información sobre cómo desea que Amazon Route 53 responda a las consultas de DNS de un dominio y sus subdominios dentro de uno o más VPCs. Para obtener más información, consulte Uso de zonas alojadas privadas en la documentación de Route 53.

control proactivo

Control de seguridad que se diseñó para evitar la implementación de recursos que no cumplan con la normativa. Estos controles analizan los recursos antes de aprovisionarlos. Si el recurso no cumple con los requisitos del control, no se aprovisiona. Para obtener más información, consulte la guía de referencia de controles en la AWS Control Tower documentación y consulte Controles proactivos en la sección Implementación de controles de seguridad en AWS.

administración del ciclo de vida del producto (PLM)

Administración de los datos y los procesos de un producto a lo largo de todo su ciclo de vida, desde el diseño, el desarrollo y el lanzamiento, pasando por el crecimiento y la madurez, hasta la reducción de su uso y su retirada.

entorno de producción

Consulte entorno.

controlador lógico programable (PLC)

En el sector de la fabricación, computadora adaptable y altamente fiable que supervisa las máquinas y automatiza los procesos de fabricación.

encadenamiento de peticiones

Uso de la salida de una petición de LLM como entrada para la siguiente petición a fin de generar mejores respuestas. Esta técnica se utiliza para dividir una tarea compleja en tareas secundarias o para refinar o ampliar de forma iterativa una respuesta preliminar. Ayuda a mejorar la precisión y la relevancia de las respuestas de un modelo y permite obtener resultados más detallados y personalizados.

seudonimización

El proceso de reemplazar los identificadores personales de un conjunto de datos por valores de marcadores de posición. La seudonimización puede ayudar a proteger la privacidad personal. Los datos seudonimizados siguen considerándose datos personales.

publish/subscribe (pub/sub)

Patrón que permite establecer comunicaciones asíncronas entre microservicios para mejorar la escalabilidad y la capacidad de respuesta. Por ejemplo, en un MES basado en microservicios, un microservicio puede publicar mensajes de eventos en un canal al que se pueden suscribir otros microservicios. El sistema puede agregar nuevos microservicios sin cambiar el servicio de publicación.

Q

plan de consulta

Serie de pasos, como instrucciones, que se utilizan para acceder a los datos de un sistema de base de datos relacional SQL.

regresión del plan de consulta

El optimizador de servicios de la base de datos elige un plan menos óptimo que antes de un cambio determinado en el entorno de la base de datos. Los cambios en estadísticas,

restricciones, configuración del entorno, enlaces de parámetros de consultas y actualizaciones del motor de base de datos PostgreSQL pueden provocar una regresión del plan.

R

Matriz RACI

Consulte responsable, fiable, consultada e informada (RACI).

RAG

Consulte generación aumentada por recuperación.

ransomware

Software malicioso que se ha diseñado para bloquear el acceso a un sistema informático o a los datos hasta que se efectúe un pago.

Matriz RASCI

Consulte responsable, fiable, consultada e informada (RACI).

RCAC

Consulte control de acceso por filas y columnas.

réplica de lectura

Una copia de una base de datos que se utiliza con fines de solo lectura. Puede enrutar las consultas a la réplica de lectura para reducir la carga en la base de datos principal.

rediseñar

Consulte Las 7 R.

objetivo de punto de recuperación (RPO)

La cantidad de tiempo máximo aceptable desde el último punto de recuperación de datos. Esto determina qué se considera una pérdida de datos aceptable entre el último punto de recuperación y la interrupción del servicio.

objetivo de tiempo de recuperación (RTO)

La demora máxima aceptable entre la interrupción del servicio y el restablecimiento del servicio.

refactorizar

Consulte Las 7 R.

Region

Conjunto de AWS recursos en un área geográfica. Cada uno Región de AWS está aislado e independiente de los demás para proporcionar tolerancia a las fallas, estabilidad y resiliencia. Para más información, consulte Specify which Regiones de AWS your account can use.

regresión

Una técnica de ML que predice un valor numérico. Por ejemplo, para resolver el problema de "¿A qué precio se venderá esta casa?", un modelo de ML podría utilizar un modelo de regresión lineal para predecir el precio de venta de una vivienda en función de datos conocidos sobre ella (por ejemplo, los metros cuadrados).

volver a alojar

Consulte Las 7 R.

versión

En un proceso de implementación, el acto de promover cambios en un entorno de producción.

reubicar

Consulte Las 7 R.

redefinir la plataforma

Consulte Las 7 R.

recomprar

Consulte Las 7 R.

resiliencia

Capacidad de una aplicación para resistir interrupciones o recuperarse de ellas. Al planificar la resiliencia en la Nube de AWS, la alta disponibilidad y la recuperación ante desastres son consideraciones comunes. Para más información, consulte Resiliencia en la Nube de AWS.

política basada en recursos

Una política asociada a un recurso, como un bucket de Amazon S3, un punto de conexión o una clave de cifrado. Este tipo de política especifica a qué entidades principales se les permite el acceso, las acciones compatibles y cualquier otra condición que deba cumplirse.

matriz responsable, confiable, consultada e informada (RACI)

Una matriz que define las funciones y responsabilidades de todas las partes involucradas en las actividades de migración y las operaciones de la nube. El nombre de la matriz se deriva de los tipos de responsabilidad definidos en la matriz: responsable (R), contable (A), consultado (C) e informado (I). El tipo de soporte (S) es opcional. Si incluye el soporte, la matriz se denomina matriz RASCI y, si la excluye, se denomina matriz RACI.

control receptivo

Un control de seguridad que se ha diseñado para corregir los eventos adversos o las desviaciones con respecto a su base de seguridad. Para obtener más información, consulte Controles receptivos en Implementación de controles de seguridad en AWS.

retain

Consulte Las 7 R.

retirar

Consulte Las 7 R.

Generación aumentada de recuperación (RAG)

Tecnología de IA generativa mediante la que un LLM hace referencia a un origen de datos autorizado que se encuentra fuera de sus orígenes de datos de entrenamiento antes de generar una respuesta. Por ejemplo, un modelo de RAG podría hacer una búsqueda semántica en la base de conocimientos o en los datos personalizados de una organización. Para más información, consulte ¿Qué es RAG (generación aumentada por recuperación)?

rotación

Proceso mediante el que periódicamente se actualiza un secreto para que resulte más difícil que un atacante pueda acceder a las credenciales.

control de acceso por filas y columnas (RCAC)

El uso de expresiones SQL básicas y flexibles que tienen reglas de acceso definidas. El RCAC consta de permisos de fila y máscaras de columnas.

RPO

Consulte objetivo de punto de recuperación.

RTO

Consulte objetivo de tiempo de recuperación.

manual de procedimientos

Conjunto de procedimientos manuales o automatizados necesarios para realizar una tarea específica. Por lo general, se diseñan para agilizar las operaciones o los procedimientos repetitivos con altas tasas de error.

S

SAML 2.0

Un estándar abierto que utilizan muchos proveedores de identidad (IdPs). Esta función permite el inicio de sesión único (SSO) federado, de modo que los usuarios pueden iniciar sesión Consola de administración de AWS o llamar a las operaciones de la AWS API sin tener que crear un usuario en IAM para todos los miembros de la organización. Para obtener más información sobre la federación basada en SAML 2.0, consulte Acerca de la federación basada en SAML 2.0 en la documentación de IAM.

SCADA

Consulte control de supervisión y adquisición de datos.

SCP

Consulte política de control de servicio.

secreta

En AWS Secrets Manager, información confidencial o restringida, como una contraseña o credenciales de usuario, que se almacena de forma cifrada. Se compone del valor del secreto y de sus metadatos. El valor del secreto puede ser binario, una sola cadena o varias cadenas. Para más información, consulte What's in a Secrets Manager secret? en la documentación de Secrets Manager.

seguridad desde el diseño

Enfoque de ingeniería de sistemas que tiene en cuenta la seguridad durante todo el proceso de desarrollo.

control de seguridad

Barrera de protección técnica o administrativa que impide, detecta o reduce la capacidad de un agente de amenazas para aprovechar una vulnerabilidad de seguridad. Existen cuatro tipos de controles de seguridad principales: preventivos, de detección, de respuesta y proactivos.

refuerzo de la seguridad

Proceso de reducir la superficie expuesta a ataques para hacerla más resistente a los ataques. Esto puede incluir acciones, como la eliminación de los recursos que ya no se necesitan, la implementación de prácticas recomendadas de seguridad consistente en conceder privilegios mínimos o la desactivación de características innecesarias en los archivos de configuración.

sistema de información sobre seguridad y administración de eventos (SIEM)

Herramientas y servicios que combinan sistemas de administración de información sobre seguridad (SIM) y de administración de eventos de seguridad (SEM). Un sistema de SIEM recopila, monitorea y analiza los datos de servidores, redes, dispositivos y otras fuentes para detectar amenazas y brechas de seguridad y generar alertas.

automatización de la respuesta de seguridad

Acción predefinida y programada que está diseñada para responder automáticamente a un evento de seguridad o corregirlo. Estas automatizaciones sirven como controles de seguridad preventivos o adaptables que le ayudan a implementar las mejores prácticas AWS de seguridad. La modificación de un grupo de seguridad de VPC, la aplicación de revisiones a una instancia de Amazon EC2 o la rotación de credenciales son algunos ejemplos de acciones de respuesta automatizadas.

cifrado del servidor

Cifrado de los datos en su destino, por parte de Servicio de AWS quien los recibe.

política de control de servicio (SCP)

Política que proporciona un control centralizado de los permisos de todas las cuentas de una organización en AWS Organizations. SCPs defina barreras o establezca límites a las acciones que un administrador puede delegar en usuarios o roles. Puede utilizarlas SCPs como listas de permitidos o rechazados para especificar qué servicios o acciones están permitidos o prohibidos. Para obtener más información, consulte las políticas de control de servicios en la AWS Organizations documentación.

punto de enlace de servicio

La URL del punto de entrada de un Servicio de AWS. Para conectarse mediante programación a un servicio de destino, puede utilizar un punto de conexión. Para obtener más información, consulte Puntos de conexión de Servicio de AWS en Referencia general de AWS.

acuerdo de nivel de servicio (SLA)

Acuerdo que aclara lo que un equipo de TI se compromete a ofrecer a los clientes, como el tiempo de actividad y el rendimiento del servicio.

indicador de nivel de servicio (SLI)

Medición de un aspecto del rendimiento de un servicio, como la tasa de errores, la disponibilidad o el rendimiento.

objetivo de nivel de servicio (SLO)

Métrica objetivo que representa el estado de un servicio medido mediante un indicador de nivel de servicio.

modelo de responsabilidad compartida

Un modelo que describe la responsabilidad con AWS la que compartes la seguridad y el cumplimiento de la nube. AWS es responsable de la seguridad de la nube, mientras que usted es responsable de la seguridad en la nube. Para obtener más información, consulte el Modelo de responsabilidad compartida.

SIEM

Consulte sistema de administración de eventos e información de seguridad.

único punto de error (SPOF)

Error en un único componente crítico de una aplicación que puede interrumpir el sistema.

SLA

Consulte acuerdo de nivel de servicio.

SLI

Consulte indicador de nivel de servicio.

SLO

Consulte objetivo de nivel de servicio.

split-and-seed modelo

Un patrón para escalar y acelerar los proyectos de modernización. A medida que se definen las nuevas funciones y los lanzamientos de los productos, el equipo principal se divide para

crear nuevos equipos de productos. Esto ayuda a ampliar las capacidades y los servicios de su organización, mejora la productividad de los desarrolladores y apoya la innovación rápida. Para más información, consulte Phased approach to modernizing applications in the Nube de AWS.

SPOF

Consulte único punto de error.

esquema en estrella

Estructura organizativa de una base de datos que utiliza una tabla de hechos de gran tamaño para almacenar datos transaccionales o medidos y una o varias tablas dimensionales más pequeñas para almacenar los atributos de los datos. Esta estructura está diseñada para utilizarse en un almacén de datos o con fines de inteligencia empresarial.

patrón de higo estrangulador

Un enfoque para modernizar los sistemas monolíticos mediante la reescritura y el reemplazo gradual de las funciones del sistema hasta que se pueda desmantelar el sistema heredado. Este patrón utiliza la analogía de una higuera que crece hasta convertirse en un árbol estable y, finalmente, se apodera y reemplaza a su host. El patrón fue presentado por Martin Fowler como una forma de gestionar el riesgo al reescribir sistemas monolíticos. Para ver un ejemplo con la aplicación de este patrón, consulte Modernización gradual de los servicios web antiguos de Microsoft ASP.NET (ASMX) mediante contenedores y Amazon API Gateway.

subred

Un intervalo de direcciones IP en la VPC. Una subred debe residir en una sola zona de disponibilidad.

control de supervisión y adquisición de datos (SCADA)

En el sector de la fabricación, sistema que utiliza hardware y software para supervisar los activos físicos y las operaciones de producción.

cifrado simétrico

Un algoritmo de cifrado que utiliza la misma clave para cifrar y descifrar los datos.

pruebas sintéticas

Prueba de un sistema de manera que simule las interacciones de los usuarios para detectar posibles problemas o supervisar el rendimiento. Puede usar Amazon CloudWatch Synthetics para crear estas pruebas.

petición del sistema

Técnica para proporcionar contexto, instrucciones o pautas a un LLM para dirigir su comportamiento. Las peticiones del sistema ayudan a establecer el contexto y las reglas para las interacciones con los usuarios.

T

etiquetas

Pares clave-valor que actúan como metadatos para organizar los recursos. AWS Las etiquetas pueden ayudar a administrar, identificar, organizar, buscar y filtrar recursos de . Para obtener más información, consulte Etiquetado de los recursos de AWS.

variable de destino

El valor que intenta predecir en el ML supervisado. Esto también se conoce como variable de resultado. Por ejemplo, en un entorno de fabricación, la variable objetivo podría ser un defecto del producto.

lista de tareas

Herramienta que se utiliza para hacer un seguimiento del progreso mediante un manual de procedimientos. La lista de tareas contiene una descripción general del manual de procedimientos y una lista de las tareas generales que deben completarse. Para cada tarea general, se incluye la cantidad estimada de tiempo necesario, el propietario y el progreso.

entorno de prueba

Consulte entorno.

entrenamiento

Proporcionar datos de los que pueda aprender su modelo de ML. Los datos de entrenamiento deben contener la respuesta correcta. El algoritmo de aprendizaje encuentra patrones en los datos de entrenamiento que asignan los atributos de los datos de entrada al destino (la respuesta que desea predecir). Genera un modelo de ML que captura estos patrones. Luego, el modelo de ML se puede utilizar para obtener predicciones sobre datos nuevos para los que no se conoce el destino.

puerta de enlace de tránsito

Un centro de tránsito de red que puede usar para interconectar sus redes con VPCs las locales. Para obtener más información, consulte Qué es una pasarela de tránsito en la AWS Transit Gateway documentación.

flujo de trabajo basado en enlaces troncales

Un enfoque en el que los desarrolladores crean y prueban características de forma local en una rama de característica y, a continuación, combinan esos cambios en la rama principal. Luego, la rama principal se adapta a los entornos de desarrollo, preproducción y producción, de forma secuencial.

acceso de confianza

Otorgar permisos a un servicio que especifique para realizar tareas en su organización AWS Organizations y en sus cuentas en su nombre. El servicio de confianza crea un rol vinculado al servicio en cada cuenta, cuando ese rol es necesario, para realizar las tareas de administración por usted. Para obtener más información, consulte AWS Organizations Utilización con otros AWS servicios en la AWS Organizations documentación.

ajuste

Cambiar aspectos de su proceso de formación a fin de mejorar la precisión del modelo de ML. Por ejemplo, puede entrenar el modelo de ML al generar un conjunto de etiquetas, incorporar etiquetas y, luego, repetir estos pasos varias veces con diferentes ajustes para optimizar el modelo.

equipo de dos pizzas

Un DevOps equipo pequeño al que puedes alimentar con dos pizzas. Un equipo formado por dos integrantes garantiza la mejor oportunidad posible de colaboración en el desarrollo de software.

U

incertidumbre

Un concepto que hace referencia a información imprecisa, incompleta o desconocida que puede socavar la fiabilidad de los modelos predictivos de ML. Hay dos tipos de incertidumbre: la incertidumbre epistémica se debe a datos limitados e incompletos, mientras que la incertidumbre aleatoria se debe al ruido y la aleatoriedad inherentes a los datos. Para más información, consulte la guía Cuantificación de la incertidumbre en los sistemas de aprendizaje profundo.

tareas indiferenciadas

También conocido como tareas arduas, es el trabajo que es necesario para crear y operar una aplicación, pero que no proporciona un valor directo al usuario final ni proporciona una ventaja competitiva. Algunos ejemplos de tareas indiferenciadas son la adquisición, el mantenimiento y la planificación de la capacidad.

entornos superiores

Consulte entorno.

V

succión

Una operación de mantenimiento de bases de datos que implica limpiar después de las actualizaciones incrementales para recuperar espacio de almacenamiento y mejorar el rendimiento.

control de versión

Procesos y herramientas que realizan un seguimiento de los cambios, como los cambios en el código fuente de un repositorio.

Emparejamiento de VPC

Una conexión entre dos VPCs que le permite enrutar el tráfico mediante direcciones IP privadas. Para obtener más información, consulte ¿Qué es una interconexión de VPC? en la documentación de Amazon VPC.

vulnerabilidad

Defecto de software o hardware que pone en peligro la seguridad del sistema.

W

caché caliente

Un búfer caché que contiene datos actuales y relevantes a los que se accede con frecuencia. La instancia de base de datos puede leer desde la caché del búfer, lo que es más rápido que leer desde la memoria principal o el disco.

datos templados

Datos a los que el acceso es infrecuente. Al consultar este tipo de datos, normalmente se aceptan consultas moderadamente lentas.

función de ventana

Función SQL que hace un cálculo en un grupo de filas que se relacionan de alguna manera con el registro actual. Las funciones de ventana son útiles para las tareas de procesamiento, como calcular una media móvil o acceder al valor de las filas en función de la posición relativa de la fila actual.

carga de trabajo

Conjunto de recursos y código que ofrece valor comercial, como una aplicación orientada al cliente o un proceso de backend.

flujo de trabajo

Grupos funcionales de un proyecto de migración que son responsables de un conjunto específico de tareas. Cada flujo de trabajo es independiente, pero respalda a los demás flujos de trabajo del proyecto. Por ejemplo, el flujo de trabajo de la cartera es responsable de priorizar las aplicaciones, planificar las oleadas y recopilar los metadatos de migración. El flujo de trabajo de la cartera entrega estos recursos al flujo de trabajo de migración, que luego migra los servidores y las aplicaciones.

WORM

Consulte escritura única y lectura múltiple.

WQF

Consulte AWS Workload Qualification Framework.

escritura única y lectura múltiple (WORM)

Modelo de almacenamiento que escribe los datos una sola vez y evita que se eliminen o modifiquen. Los usuarios autorizados pueden leer los datos tantas veces como sea necesario, pero no los pueden cambiar. Esta infraestructura de almacenamiento de datos se considera inmutable.

Z

ataque de día cero

Ataque, normalmente de malware, que se aprovecha de una vulnerabilidad de día cero.

vulnerabilidad de día cero

Un defecto o una vulnerabilidad sin mitigación en un sistema de producción. Los agentes de amenazas pueden usar este tipo de vulnerabilidad para atacar el sistema. Los desarrolladores suelen darse cuenta de la vulnerabilidad a raíz del ataque.

peticiones desde cero

Proporcionar a un LLM instrucciones para llevar a cabo una tarea, pero sin ejemplos (pasos) que puedan ayudar a guiarlo. El LLM debe usar los conocimientos del entrenamiento previo para llevar a cabo la tarea. La eficacia de la petición desde cero depende de la complejidad de la tarea y de la calidad de la petición. Consulte también peticiones con pocos pasos.

aplicación zombi

Aplicación que utiliza un promedio de CPU y memoria menor al 5 por ciento. En un proyecto de migración, es habitual retirar estas aplicaciones.

Las traducciones son generadas a través de traducción automática. En caso de conflicto entre la traducción y la version original de inglés, prevalecerá la version en inglés.