aws

Guía para desarrolladores

Amazon Timestream

Amazon Timestream: Guía para desarrolladores

Table of Contents

¿Para qué sirve Amazon Timestream? LiveAnalytics

Amazon Timestream LiveAnalytics for es una base de datos de series temporales rápida, escalable, totalmente administrada y diseñada específicamente que facilita el almacenamiento y el análisis de billones de puntos de datos de series temporales por día. Timestream for le LiveAnalytics permite ahorrar tiempo y costos en la administración del ciclo de vida de los datos de series temporales, ya que mantiene los datos recientes en la memoria y transfiere los datos históricos a un nivel de almacenamiento con costos optimizados en función de las políticas definidas por el usuario. El motor de consultas especialmente diseñado LiveAnalytics de Timestream for le permite acceder a los datos recientes e históricos y analizarlos juntos, sin tener que especificar su ubicación. Amazon Timestream LiveAnalytics for cuenta con funciones de análisis de series temporales integradas que le ayudan a identificar tendencias y patrones en sus datos prácticamente en tiempo real. Timestream for LiveAnalytics no tiene servidores y se escala automáticamente hacia arriba o hacia abajo para ajustar la capacidad y el rendimiento. Como no necesita administrar la infraestructura subyacente, puede concentrarse en optimizar y crear sus aplicaciones.

Timestream for LiveAnalytics también se integra con los servicios más utilizados para la recopilación de datos, la visualización y el aprendizaje automático. Puede enviar datos a Amazon Timestream LiveAnalytics para utilizarlos con Amazon Kinesis, AWS IoT Core MSK Amazon y Telegraf de código abierto. Puede visualizar los datos con Amazon QuickSight, Grafana y las herramientas de inteligencia empresarial mediante. JDBC También puedes usar Amazon SageMaker con Timestream LiveAnalytics para el aprendizaje automático.

Timestream para obtener beneficios clave LiveAnalytics

Las principales ventajas de Amazon Timestream LiveAnalytics for son:

- Sin servidores con autoscalamiento: con Amazon Timestream LiveAnalytics for, no hay servidores que administrar ni capacidad de aprovisionamiento. A medida que cambian las necesidades de su aplicación, Timestream for escala automáticamente para LiveAnalytics ajustar la capacidad.
- Administración del ciclo de vida de los datos: Amazon Timestream LiveAnalytics for simplifica el complejo proceso de administración del ciclo de vida de los datos. Ofrece almacenamiento por niveles, con un almacén de memoria para los datos recientes y un almacén magnético para los datos históricos. Amazon Timestream automatiza la transferencia de datos del almacén de memoria al almacén magnético en función de las políticas configurables por el usuario.

- Acceso simplificado a los datos: con Amazon Timestream LiveAnalytics for, ya no necesitará usar herramientas dispares para acceder a datos recientes e históricos. El motor de consultas diseñado específicamente para Amazon Timestream LiveAnalytics for accede a los datos y los combina de forma transparente en todos los niveles de almacenamiento sin tener que especificar la ubicación de los datos.
- Diseñado específicamente para series temporales: puede analizar rápidamente los datos de series temporales mediante funciones de series temporales integradas para SQL suavizar, aproximar e interpolar. Timestream for LiveAnalytics también admite agregados avanzados, funciones de ventana y tipos de datos complejos, como matrices y filas.
- Siempre cifrados: Amazon Timestream garantiza que los datos LiveAnalytics de sus series temporales estén siempre cifrados, ya sea en reposo o en tránsito. Amazon Timestream LiveAnalytics for también le permite especificar AWS KMS una clave gestionada por el cliente CMK () para cifrar los datos del almacén magnético.
- Alta disponibilidad: Amazon Timestream garantiza la alta disponibilidad de sus solicitudes de escritura y lectura mediante la replicación automática de los datos y la asignación de recursos en al menos 3 zonas de disponibilidad diferentes dentro de una misma región. AWS Para obtener más información, consulte el acuerdo de nivel de servicio de Timestream.
- Durabilidad: Amazon Timestream garantiza la durabilidad de sus datos al replicar automáticamente los datos de la memoria y del almacén magnético en diferentes zonas de disponibilidad de una sola región. AWS Todos los datos se graban en el disco antes de confirmar que la solicitud de escritura está completa.

Cronograma para casos de uso LiveAnalytics

Algunos ejemplos de una lista cada vez mayor de casos de uso de Timestream incluyen: LiveAnalytics

- Monitoree las métricas para mejorar el rendimiento y la disponibilidad de sus aplicaciones.
- Almacenamiento y análisis de la telemetría industrial para agilizar la gestión y el mantenimiento de los equipos.
- Seguimiento de la interacción del usuario con una aplicación a lo largo del tiempo.
- Almacenamiento y análisis de datos de sensores de IoT.

Cómo empezar a utilizar Timestream para LiveAnalytics

Le recomendamos que comience leyendo las secciones siguientes:

- Tutorial- Para crear una base de datos con conjuntos de datos de ejemplo y ejecutar consultas de ejemplo.
- Amazon LiveAnalytics Timestream para conceptos- Aprender los conceptos básicos de Timestream. LiveAnalytics
- Acceder a Timestream para LiveAnalytics- Para aprender a acceder a Timestream para LiveAnalytics usar la consola, o. AWS CLI API
- Cuotas- Para obtener información sobre las cuotas en la cantidad de Timestream para los LiveAnalytics componentes que puede aprovisionar.

Para obtener información sobre cómo empezar rápidamente a desarrollar aplicaciones para Timestream LiveAnalytics, consulte lo siguiente:

- Uso del AWS SDKs
- Consulta el idioma de referencia

Funcionamiento

En las siguientes secciones se proporciona una descripción general de los componentes del servicio Amazon Timestream for Live Analytics y de cómo interactúan entre ellos.

Tras leer esta introducción, consulte las Acceder a Timestream para LiveAnalytics secciones para obtener información sobre cómo acceder a Timestream for Live Analytics mediante la consola, o AWS CLI SDKs

Temas

- Amazon LiveAnalytics Timestream para conceptos
- Arquitectura
- Escribe
- Almacenamiento
- Consultas
- Consultas programadas

- Unidad de cómputo Timestream () TCU

Amazon LiveAnalytics Timestream para conceptos

Los datos de series temporales son una secuencia de puntos de datos registrados durante un intervalo de tiempo. Este tipo de datos se utiliza para medir eventos que cambian con el tiempo. Algunos ejemplos son los siguientes:

- Precios de las acciones a lo largo del tiempo
- Mediciones de temperatura a lo largo del tiempo
- CPUutilización de una EC2 instancia a lo largo del tiempo

En el caso de los datos de series temporales, cada punto de datos consta de una marca temporal, uno o más atributos y el evento que cambia con el tiempo. Estos datos se pueden utilizar para obtener información sobre el rendimiento y el estado de una aplicación, detectar anomalías e identificar oportunidades de optimización. Por ejemplo, es posible que DevOps los ingenieros deseen ver los datos que miden los cambios en las métricas de rendimiento de la infraestructura. Es posible que los fabricantes deseen realizar un seguimiento de los datos de los sensores de IoT que miden los cambios en los equipos de una instalación. Es posible que los especialistas en marketing online deseen analizar los datos del flujo de clics que captan la forma en que un usuario navega por un sitio web a lo largo del tiempo. Como los datos de series temporales se generan a partir de múltiples fuentes en volúmenes extremadamente altos, es necesario recopilarlos prácticamente en tiempo real de manera rentable y, por lo tanto, requieren un almacenamiento eficiente que ayude a organizar y analizar los datos.

Los siguientes son los conceptos clave de Timestream for. LiveAnalytics

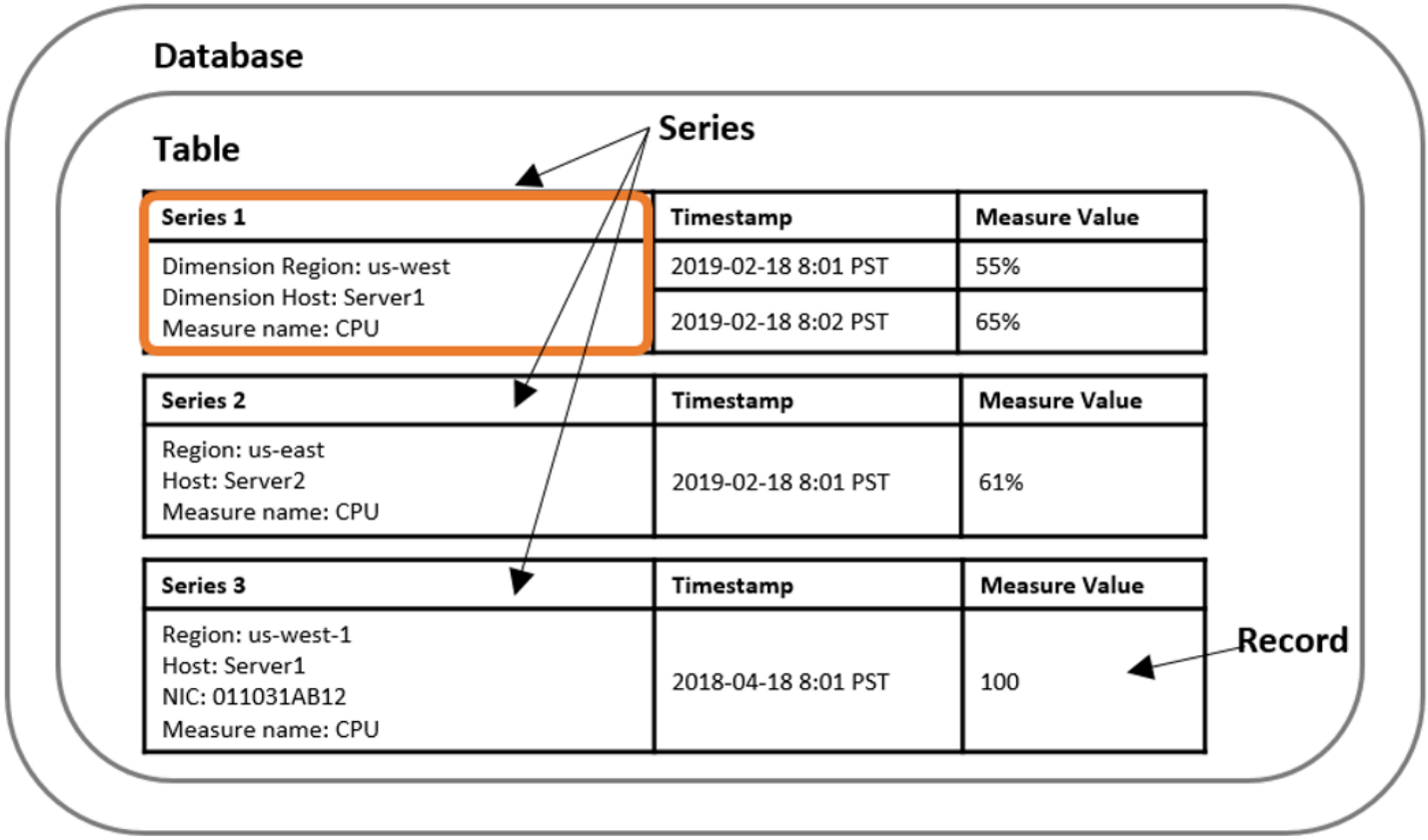
- Serie temporal: secuencia de uno o más puntos de datos (o registros) registrados durante un intervalo de tiempo. Algunos ejemplos son el precio de una acción a lo largo del tiempo, la utilización de memoria CPU o de una EC2 instancia a lo largo del tiempo y la lectura de temperatura/presión de un sensor de IoT a lo largo del tiempo.
- Registro: un único punto de datos de una serie temporal.
- Dimensión: atributo que describe los metadatos de una serie temporal. Una dimensión consta de un nombre de dimensión y un valor de dimensión. Considere los siguientes ejemplos:
  - Al considerar una bolsa de valores como una dimensión, el nombre de la dimensión es «bolsa de valores» y el valor de la dimensión es «» NYSE

  - Al considerar una AWS región como una dimensión, el nombre de la dimensión es «region» y el valor de la dimensión es «us-east-1"
  - En el caso de un sensor de IoT, el nombre de la dimensión es «ID del dispositivo» y el valor de la dimensión es «12345».
- Medida: el valor real que mide el registro. Algunos ejemplos son el precio de las acciones, el CPU uso de memoria y la lectura de temperatura o humedad. Las medidas se componen de los nombres y valores de las medidas. Considere los siguientes ejemplos:
  - En el caso del precio de una acción, el nombre de la medida es «precio de la acción» y el valor de la medida es el precio real de la acción en un momento dado.
  - Para la CPU utilización, el nombre de la medida es «CPUutilización» y el valor de la medida es la CPU utilización real.

  Las medidas se pueden modelar en Timestream como registros de medidas múltiples o de una sola LiveAnalytics medida. Para obtener más información, consulte Registros de medidas múltiples frente a registros de medida única.
- Marca temporal: indica cuándo se recopiló una medida para un registro determinado. Timestream for LiveAnalytics admite marcas de tiempo con granularidad de nanosegundos.
- Tabla: un contenedor para un conjunto de series temporales relacionadas.
- Base de datos: un contenedor de nivel superior para tablas.

Un resumen de Timestream para conceptos LiveAnalytics

Una base de datos contiene 0 o más tablas. Cada tabla contiene 0 o más series temporales. Cada serie temporal consta de una secuencia de registros durante un intervalo de tiempo determinado con una granularidad especificada. Cada serie temporal se puede describir utilizando sus metadatos o dimensiones, sus datos o medidas y sus marcas temporales.

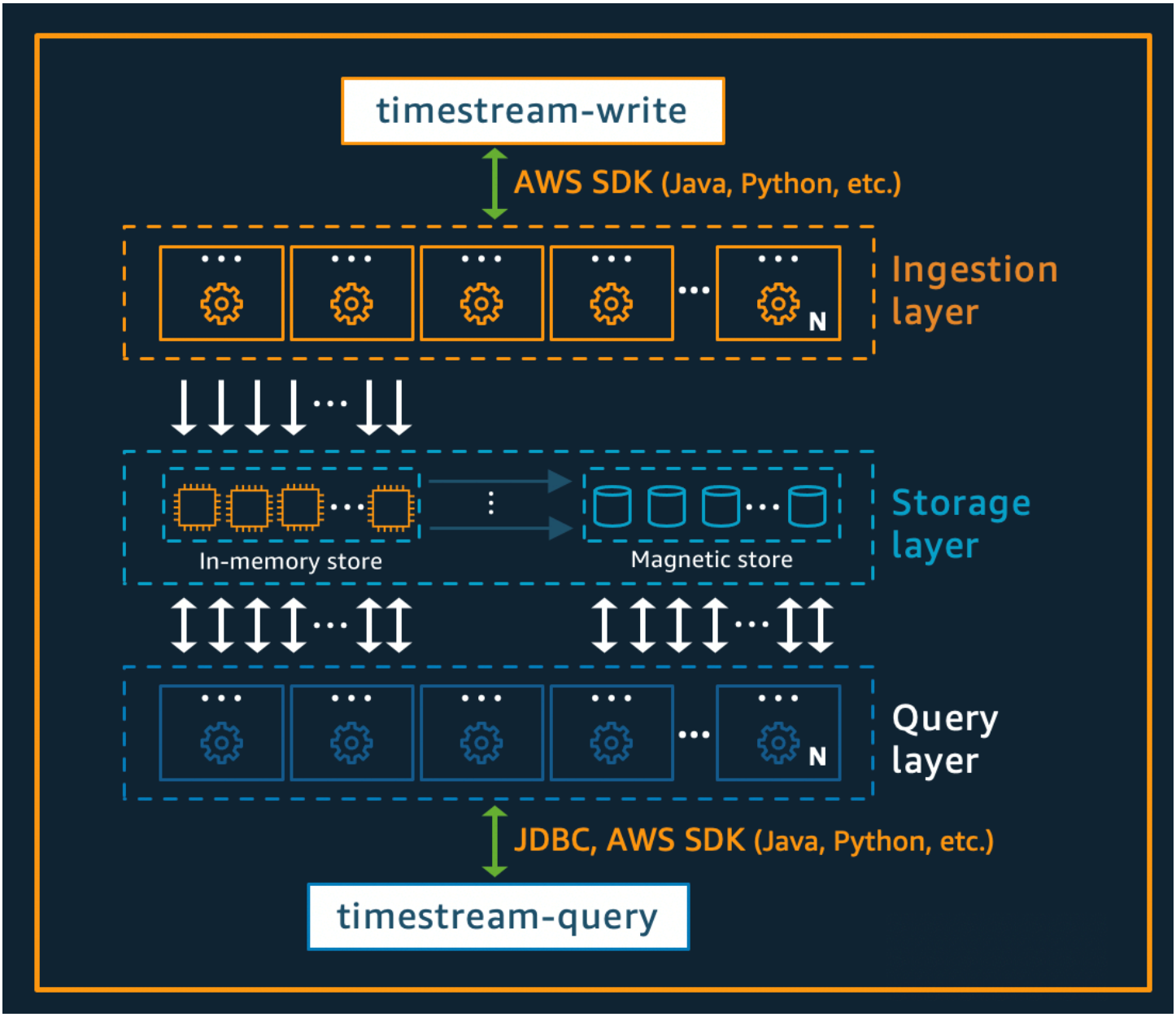


Arquitectura

Amazon Timestream for Live Analytics se ha diseñado desde cero para recopilar, almacenar y procesar datos de series temporales a escala. Su arquitectura sin servidor admite sistemas de procesamiento de consultas, almacenamiento y ingesta de datos totalmente disociados que pueden escalarse de forma independiente. Este diseño simplifica cada subsistema, lo que facilita el logro de una fiabilidad inquebrantable, elimina los cuellos de botella de escalado y reduce las probabilidades de que se produzcan fallos correlacionados con el sistema. Cada uno de estos factores adquiere más importancia a medida que el sistema se amplía.

Temas

- Escribe la arquitectura
- Arquitectura de almacenamiento
- Arquitectura de consultas
- Arquitectura celular



Escribe la arquitectura

Al escribir datos de series temporales, Amazon Timestream para Live Analytics enruta las escrituras de una tabla o partición a una instancia de almacenamiento de memoria tolerante a errores que procesa escrituras de datos de alto rendimiento. El almacén de memoria, a su vez, logra una mayor durabilidad en un sistema de almacenamiento independiente que replica los datos en tres zonas de disponibilidad (). AZs La replicación se basa en el quórum, de modo que la pérdida de nodos, o de una zona de disponibilidad completa, no interrumpirá la disponibilidad de escritura. Casi en tiempo real, otros nodos de almacenamiento en memoria se sincronizan con los datos para atender

las consultas. Los nodos de réplica del lector AZs también se extienden para garantizar una alta disponibilidad de lectura.

Timestream for Live Analytics permite escribir datos directamente en el almacén magnético, para aplicaciones que generan datos de menor rendimiento y llegan tarde. Los datos que llegan tarde son datos con una marca de tiempo anterior a la hora actual. Al igual que ocurre con las escrituras de alto rendimiento en el almacén de memoria, los datos escritos en el almacén magnético se replican en tres AZs y la replicación se basa en el quórum.

Tanto si los datos se escriben en la memoria como en un almacén magnético, Timestream for Live Analytics indexa y divide automáticamente los datos antes de guardarlos en el almacenamiento. Una sola tabla de Timestream for Live Analytics puede tener cientos, miles o incluso millones de particiones. Las particiones individuales no se comunican directamente entre sí y no comparten ningún dato (arquitectura que no comparte nada). En su lugar, el seguimiento de la partición de una tabla se realiza mediante un servicio de indexación y seguimiento de particiones de alta disponibilidad. Esto proporciona otra separación de problemas diseñada específicamente para minimizar el efecto de las fallas en el sistema y reducir la probabilidad de que se produzcan fallas correlacionadas.

Arquitectura de almacenamiento

Cuando los datos se almacenan en Timestream for Live Analytics, los datos se organizan en orden temporal y temporal en función de los atributos de contexto escritos con los datos. Disponer de un esquema de particiones que divida el «espacio» del tiempo es importante para escalar masivamente un sistema de series temporales. Esto se debe a que la mayoría de los datos de series temporales se escriben en la hora actual o en torno a ella. En consecuencia, la partición basada únicamente en el tiempo no distribuye bien el tráfico de escritura ni permite reducir eficazmente los datos en el momento de la consulta. Esto es importante para el procesamiento de series temporales a escala extrema, y ha permitido a Timestream for Live Analytics escalar órdenes de magnitud más que los demás sistemas líderes que existen hoy en día sin servidores. Las particiones resultantes se denominan «teselas» porque representan divisiones de un espacio bidimensional (que están diseñadas para tener un tamaño similar). Las tablas Timestream for Live Analytics comienzan como una partición única (mosaico) y, después, se dividen en la dimensión espacial según lo requiera el rendimiento. Cuando las teselas alcanzan un tamaño determinado, se dividen en la dimensión temporal para lograr un mejor paralelismo de lectura a medida que aumenta el tamaño de los datos.

Timestream for Live Analytics está diseñado para gestionar automáticamente el ciclo de vida de los datos de series temporales. Timestream for Live Analytics ofrece dos almacenes de datos: un almacén en memoria y un almacén magnético rentable. También permite configurar políticas a

nivel de tabla para transferir automáticamente los datos entre las tiendas. Las escrituras de datos entrantes de alto rendimiento aterrizan en el almacén de memoria, donde los datos se optimizan para la escritura, así como las lecturas que se realizan en torno a la hora actual, lo que facilita las consultas de tipo panel y alerta. Una vez transcurrido el plazo principal para escribir, enviar alertas y crear cuadros de mando, se permite que los datos fluyan automáticamente del almacén de memoria al almacén magnético para optimizar los costes. Timestream for Live Analytics permite establecer una política de retención de datos en el almacén de memoria con este fin. Las escrituras de datos para los datos que llegan tarde se escriben directamente en el almacén magnético.

Una vez que los datos están disponibles en el almacén magnético (debido a la expiración del período de retención del almacenamiento de memoria o debido a la escritura directa en el almacén magnético), se reorganizan en un formato altamente optimizado para lecturas de grandes volúmenes de datos. El almacén magnético también tiene una política de retención de datos que se puede configurar si hay un límite de tiempo en el que los datos dejan de ser útiles. Cuando los datos superan el intervalo de tiempo definido para la política de retención del almacén magnético, se eliminan automáticamente. Por lo tanto, con Timestream for Live Analytics, salvo alguna configuración, la gestión del ciclo de vida de los datos se lleva a cabo sin problemas entre bastidores.

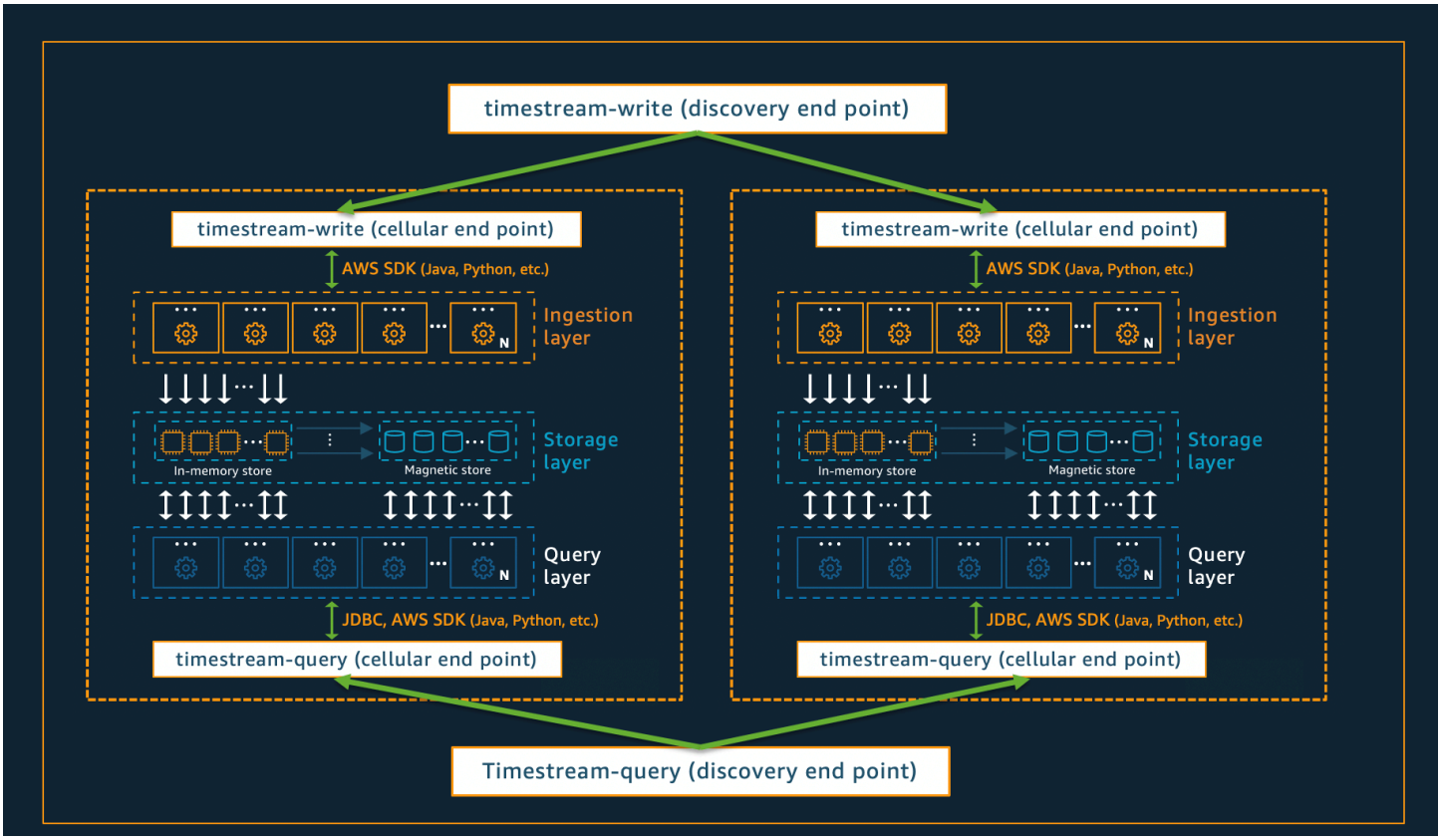
Arquitectura de consultas

Las consultas de Timestream for Live Analytics se expresan en una SQL gramática que tiene extensiones para admitir series temporales específicas (funciones y tipos de datos específicos de series temporales), por lo que los desarrolladores que ya estén familiarizados pueden aprender fácilmente la curva de aprendizaje. SQL A continuación, las consultas se procesan mediante un motor de consultas adaptable y distribuido que utiliza los metadatos del servicio de indexación y seguimiento de teselas para acceder sin problemas a los datos de los distintos almacenes de datos y combinarlos en el momento de emitir la consulta. Esto hace que la experiencia resulte atractiva para los clientes, ya que reduce muchas de las complejidades de Rube Goldberg en una simple y familiar abstracción de bases de datos.

Las consultas son gestionadas por una flota de trabajadores dedicada, en la que el número de trabajadores contratados para ejecutar una consulta determinada viene determinado por la complejidad de la consulta y el tamaño de los datos. El rendimiento de las consultas complejas en conjuntos de datos de gran tamaño se logra mediante un paralelismo masivo, tanto en la flota de tiempo de ejecución de las consultas como en las flotas de almacenamiento del sistema. La capacidad de analizar grandes cantidades de datos de forma rápida y eficiente es uno de los puntos fuertes de Timestream for Live Analytics. Una sola consulta que abarque terabytes o incluso petabytes de datos puede tener miles de máquinas trabajando en ella al mismo tiempo.

Arquitectura celular

Para garantizar que Timestream for Live Analytics pueda ofrecer una escala prácticamente infinita para sus aplicaciones y, al mismo tiempo, garantizar una disponibilidad del 99,99%, el sistema también está diseñado con una arquitectura celular. En lugar de escalar el sistema como un todo, Timestream for Live Analytics se segmenta en varias copias más pequeñas de sí mismo, denominadas celdas. Esto permite analizar las células a gran escala y evita que un problema del sistema en una célula afecte a la actividad de cualquier otra célula de una región determinada. Si bien Timestream for Live Analytics está diseñado para admitir varias celdas por región, considere el siguiente escenario ficticio, en el que hay 2 celdas en una región.



En el escenario descrito anteriormente, las solicitudes de ingesta y consulta de datos son procesadas primero por el terminal de descubrimiento para la ingesta y consulta de datos, respectivamente. A continuación, el punto final de detección identifica la celda que contiene los datos del cliente y dirige la solicitud al punto final de recepción o consulta correspondiente a esa celda. Cuando se utilizaSDKs, estas tareas de administración de terminales se gestionan de forma transparente para usted.

> **Note**
>
> Cuando utilice VPC puntos de conexión con Timestream para Live Analytics o acceda directamente a REST API las operaciones de Timestream for Live Analytics, tendrá que interactuar directamente con los puntos de conexión móviles. Para obtener información sobre cómo hacerlo, consulte VPCEndpoints para obtener instrucciones sobre cómo configurar los puntos finales y VPC Endpoint Discovery Pattern para obtener instrucciones sobre la invocación directa de las operaciones. REST API

Escribe

Puede recopilar datos de series temporales de dispositivos conectados, sistemas de TI y equipos industriales y escribirlos en Timestream for Live Analytics. Timestream para Live Analytics le permite escribir puntos de datos de una sola serie temporal o puntos de datos de varias series en una sola solicitud de escritura cuando las series temporales pertenecen a la misma tabla. Para su comodidad, Timestream for Live Analytics le ofrece un esquema flexible que detecta automáticamente los nombres de las columnas y los tipos de datos de las tablas de Timestream for Live Analytics en función de los nombres de las dimensiones y los tipos de datos de los valores de medida que especifique al invocar las escrituras en la base de datos. También puede escribir lotes de datos en Timestream for Live Analytics.

> **Note**
>
> Timestream for Live Analytics admite una semántica de coherencia eventual para las lecturas. Esto significa que cuando consulta datos inmediatamente después de escribir un lote de datos en Timestream for Live Analytics, es posible que los resultados de la consulta no reflejen los resultados de una operación de escritura completada recientemente. Es posible que los resultados también incluyan algunos datos obsoletos. Del mismo modo, al escribir datos de series temporales con una o más dimensiones nuevas, una consulta puede devolver un subconjunto parcial de columnas durante un breve período de tiempo. Si repites estas solicitudes de consulta al cabo de poco tiempo, los resultados deberían devolver los datos más recientes.

Puede escribir datos con AWS SDKs, AWS CLI, o mediante AWS Lambda AWS IoT CoreAmazon Managed Service para Apache Flink, Amazon KinesisAmazon MSK, yTelegraf de código abierto.

Temas

Tipos de datos

Timestream for Live Analytics admite los siguientes tipos de datos para las escrituras.

Tipo de datos	Descripción
BIGINT	Representa un entero con signo de 64 bits.
BOOLEAN	Representa los dos valores de verdad de la lógica, a saber, verdadero y falso.
DOUBLE	Precisión variable de 64 bits que implementa el IEEE estándar 754 para aritmética binaria de punto flotante. **Note** Hay funciones de lenguaje de consulta para valores `NaN` dobles que se pueden utilizar en `Infinity` las consultas. Pero no puede escribir esos valores en Timestream.
VARCHAR	Datos de caracteres de longitud variable con una longitud máxima opcional. El límite máximo es de 2 KB.
MULTI	Tipo de datos para registros de medidas múltiples. Este tipo de datos incluye una o más medidas del tipo`BIGINT`,`BOOLEAN`, `DOUBLEVARCHAR`, y`TIMESTAMP` .

Tipo de datos	Descripción
TIMESTAMP	Representa una instancia en el tiempo utilizando un tiempo de precisión de nanosegundos enUTC, registrando el tiempo transcurrido desde la hora de Unix. Actualmente, este tipo de datos solo se admite para registros de medidas múltiples (es decir, dentro de los valores de medición del tipo). `MULTI` *YYYY*-*MM*-*DD* *hh*:*mm*:*ss*.*sssssssss* Escribe marcas de tiempo de soporte en el intervalo de 0 a. `1970-01-01 00:00:00.000000000` `2262-04-11 23:47:16.854775807`

No hay una definición de esquema inicial

Antes de enviar datos a Amazon Timestream for Live Analytics, debe crear una base de datos y una tabla mediante AWS Management Console las operaciones Timestream for Live Analytics o Timestream for Live SDKs Analytics. API Para obtener más información, consulte Creación de una base de datos de y Creación de una tabla. Al crear la tabla, no necesita definir el esquema por adelantado. Amazon Timestream para Live Analytics detecta automáticamente el esquema en función de las medidas y dimensiones de los puntos de datos que se envían, por lo que ya no es necesario modificar el esquema sin conexión para adaptarlo a los datos de series temporales que cambian rápidamente.

Escribir datos (inserciones y ajustes)

La operación de escritura en Amazon Timestream for Live Analytics le permite insertar y modificar datos. De forma predeterminada, las escrituras en Amazon Timestream para Live Analytics siguen al primer escritor gana la semántica, en la que los datos se almacenan solo como datos adjuntos y se rechazan los registros duplicados. Si bien el primer escritor gana, la semántica satisface los requisitos de muchas aplicaciones de series temporales, hay situaciones en las que las aplicaciones necesitan actualizar los registros existentes de forma idempotente o escribir datos con la semántica que gana el último escritor, gana la semántica, en la que el registro con la versión más alta se almacena en el servicio. Para abordar estos escenarios, Amazon Timestream for Live Analytics ofrece la posibilidad de alterar los datos. Upsert es una operación que inserta un registro en el

sistema cuando el registro no existe o lo actualiza cuando existe uno. Cuando se actualiza el registro, se actualiza de forma idempotente.

No hay ninguna operación de borrado a nivel de registro. Sin embargo, las tablas y las bases de datos se pueden eliminar.

Escribir datos en el almacén de memoria y en el almacén magnético

Amazon Timestream for Live Analytics ofrece la posibilidad de escribir datos directamente en el almacén de memoria y en el almacén magnético. El almacén de memoria está optimizado para escrituras de datos de alto rendimiento y el almacenamiento magnético está optimizado para escrituras con menor rendimiento de los datos que llegan tarde.

Los datos que llegan tarde son datos con una marca de tiempo anterior a la hora actual y fuera del período de retención del almacén de memoria. Debe habilitar explícitamente la capacidad de escribir los datos que lleguen tarde en el almacén magnético habilitando la escritura en el almacén magnético para la tabla. Además, `MagneticStoreRejectedDataLocation` se define cuando se crea una tabla. Para escribir en el almacén magnético, las personas que llaman `WriteRecords` deben tener los `S3:PutObject` permisos para acceder al bucket de S3 especificado `MagneticStoreRejectedDataLocation` durante la creación de la tabla. Para obtener más información, consulte CreateTableWriteRecords, y PutObject.

Escribir datos con registros de una sola medida y registros de múltiples medidas

Amazon Timestream for Live Analytics ofrece la posibilidad de escribir datos mediante dos tipos de registros, a saber, registros de una sola medida y registros de múltiples medidas.

Registros de medida única

Los registros de una sola medida le permiten enviar una sola medida por registro. Cuando los datos se envían a Timestream for Live Analytics con este formato, Timestream for Live Analytics crea una fila de tabla por registro. Esto significa que si un dispositivo emite 4 métricas y cada métrica se envía como un registro de una sola medida, Timestream for Live Analytics creará 4 filas en la tabla para almacenar estos datos y los atributos del dispositivo se repetirán para cada fila. Se recomienda este formato en los casos en los que desee supervisar una única métrica de una aplicación o cuando la aplicación no emita varias métricas al mismo tiempo.

Registros de medidas múltiples

Con los registros de múltiples medidas, puede almacenar varias medidas en una sola fila de la tabla, en lugar de almacenar una medida por fila de la tabla. Por lo tanto, los registros de medidas múltiples

le permiten migrar los datos existentes de las bases de datos relacionales a Amazon Timestream for Live Analytics con cambios mínimos.

También puede agrupar más datos en una sola solicitud de escritura que los registros de una sola medida. Esto aumenta el rendimiento y el rendimiento de la escritura de datos y, además, reduce el coste de la escritura de datos. Esto se debe a que agrupar más datos en una solicitud de escritura permite a Amazon Timestream for Live Analytics identificar más datos repetibles en una sola solicitud de escritura (cuando proceda) y cobrar solo una vez por los datos repetidos.

Temas

Registros de medidas múltiples

Con los registros de medidas múltiples, puede almacenar sus datos de series temporales en un formato más compacto en la memoria y en el almacén magnético, lo que ayuda a reducir los costos de almacenamiento de datos. Además, el almacenamiento de datos compacto se presta para escribir consultas más sencillas para la recuperación de datos, mejora el rendimiento de las consultas y reduce el coste de las consultas.

Además, los registros de múltiples medidas también admiten el tipo de TIMESTAMP datos para almacenar más de una marca de tiempo en un registro de series temporales. TIMESTAMPlos atributos de un registro de múltiples medidas admiten marcas de tiempo en el futuro o en el pasado. Por lo tanto, los registros de medidas múltiples ayudan a mejorar el rendimiento, el coste y la simplicidad de las consultas, y ofrecen más flexibilidad para almacenar diferentes tipos de medidas correlacionadas.

Ventajas

Los siguientes son los beneficios de usar registros de múltiples medidas.

- Rendimiento y coste: los registros de múltiples medidas permiten escribir varias medidas de series temporales en una sola solicitud de escritura. Esto aumenta el rendimiento de escritura y también reduce el costo de las escrituras. Con los registros de múltiples medidas, puede almacenar los datos de una manera más compacta, lo que ayuda a reducir los costos de almacenamiento de datos. El almacenamiento compacto de datos de los registros de múltiples medidas hace que las consultas procesen menos datos. Esto está diseñado para mejorar el rendimiento general de las consultas y ayudar a reducir el costo de las consultas.

- Simplicidad de consulta: con los registros de varias medidas, no es necesario escribir expresiones de tabla comunes y complejas (CTEs) en una consulta para leer varias medidas con la misma marca de tiempo. Esto se debe a que las medidas se almacenan como columnas en una sola fila de la tabla. Por lo tanto, los registros de medidas múltiples permiten escribir consultas más sencillas.
- Flexibilidad de modelado de datos: puede escribir marcas de tiempo futuras en Timestream for Live Analytics mediante el tipo de TIMESTAMP datos y los registros de medidas múltiples. Un registro de múltiples medidas puede tener varios atributos de tipo de TIMESTAMP datos, además del campo de tiempo de un registro. TIMESTAMPLos atributos, en un registro de múltiples medidas, pueden tener marcas de tiempo en el futuro o en el pasado y comportarse como el campo de tiempo, excepto que Timestream for Live Analytics no indexa los valores de tipo TIMESTAMP en un registro de múltiples medidas.

Casos de uso

Puede usar registros de múltiples medidas para cualquier aplicación de series temporales que genere más de una medición desde el mismo dispositivo en un momento dado. A continuación se muestran algunos ejemplos de aplicaciones.

- Una plataforma de transmisión de vídeo que genera cientos de métricas en un momento dado.
- Dispositivos médicos que generan mediciones como los niveles de oxígeno en sangre, la frecuencia cardíaca y el pulso.
- Equipos industriales, como plataformas petrolíferas, que generan sensores métricos, de temperatura y meteorológicos.
- Otras aplicaciones que están diseñadas con uno o más microservicios.

Ejemplo: supervisar el rendimiento y el estado de una aplicación de streaming de vídeo

Considere una aplicación de streaming de vídeo que se ejecute en 200 EC2 instancias. Desea utilizar Amazon Timestream for Live Analytics para almacenar y analizar las métricas emitidas por la aplicación, de modo que pueda comprender el rendimiento y el estado de la aplicación, identificar rápidamente las anomalías, resolver problemas y descubrir oportunidades de optimización.

Modelaremos este escenario con registros de una sola medida y registros de múltiples medidas y, a continuación, compararemos o contrastaremos ambos enfoques. Para cada enfoque, hacemos las siguientes suposiciones.

- Cada EC2 instancia emite cuatro medidas (video_startup_time, rebuffering_ratio, video_playback_failures y average_frame_rate) y cuatro dimensiones (device_id, device_type, os_version y region) por segundo.
- Desea almacenar 6 horas de datos en el almacén de memoria y 6 meses de datos en el almacén magnético.
- Para identificar las anomalías, ha configurado 10 consultas que se ejecutan cada minuto para identificar cualquier actividad inusual que se haya producido en los últimos minutos. También ha creado un panel de control con ocho widgets que muestran los datos de las últimas 6 horas, para que pueda supervisar su aplicación de forma eficaz. Cinco usuarios acceden a este panel en un momento dado y se actualiza automáticamente cada hora.

Uso de registros de una sola medida

Modelado de datos: con los registros de una sola medida, crearemos un registro para cada una de las cuatro medidas (tiempo de inicio del vídeo, relación de realmacenamiento en búfer, errores de reproducción del vídeo y velocidad media de fotogramas). Cada registro tendrá las cuatro dimensiones (device_id, device_type, os_version y region) y una marca de tiempo.

Escritura: al escribir datos en Amazon Timestream for Live Analytics, los registros se crean de la siguiente manera.

```
public void writeRecords() {
    System.out.println("Writing records");
    // Specify repeated values for all records
    List<Record> records = new ArrayList<>();
    final long time = System.currentTimeMillis();

    List<Dimension> dimensions = new ArrayList<>();

    final Dimension device_id = new
 Dimension().withName("device_id").withValue("12345678");
    final Dimension device_type = new
 Dimension().withName("device_type").withValue("iPhone 11");
    final Dimension os_version = new
 Dimension().withName("os_version").withValue("14.8");
    final Dimension region = new Dimension().withName("region").withValue("us-east-1");

    dimensions.add(device_id);
    dimensions.add(device_type);
    dimensions.add(os_version);
```

```
    dimensions.add(region);

    Record videoStartupTime = new Record()
        .withDimensions(dimensions)
        .withMeasureName("video_startup_time")
        .withMeasureValue("200")
        .withMeasureValueType(MeasureValueType.BIGINT)
        .withTime(String.valueOf(time));
    Record rebufferingRatio = new Record()
        .withDimensions(dimensions)
        .withMeasureName("rebuffering_ratio")
        .withMeasureValue("0.5")
        .withMeasureValueType(MeasureValueType.DOUBLE)
        .withTime(String.valueOf(time));
    Record videoPlaybackFailures = new Record()
        .withDimensions(dimensions)
        .withMeasureName("video_playback_failures")
        .withMeasureValue("0")
        .withMeasureValueType(MeasureValueType.BIGINT)
        .withTime(String.valueOf(time));
    Record averageFrameRate = new Record()
        .withDimensions(dimensions)
        .withMeasureName("average_frame_rate")
        .withMeasureValue("0.5")
        .withMeasureValueType(MeasureValueType.DOUBLE)
        .withTime(String.valueOf(time));

    records.add(videoStartupTime);
    records.add(rebufferingRatio);
    records.add(videoPlaybackFailures);
    records.add(averageFrameRate);

    WriteRecordsRequest writeRecordsRequest = new WriteRecordsRequest()
        .withDatabaseName(DATABASE_NAME)
        .withTableName(TABLE_NAME)
        .withRecords(records);

    try {
      WriteRecordsResult writeRecordsResult =
amazonTimestreamWrite.writeRecords(writeRecordsRequest);
      System.out.println("WriteRecords Status: " +
writeRecordsResult.getSdkHttpMetadata().getHttpStatusCode());
    } catch (RejectedRecordsException e) {
      System.out.println("RejectedRecords: " + e);
```

```
      for (RejectedRecord rejectedRecord : e.getRejectedRecords()) {
        System.out.println("Rejected Index " + rejectedRecord.getRecordIndex() + ": "
            + rejectedRecord.getReason());
      }
      System.out.println("Other records were written successfully. ");
    } catch (Exception e) {
      System.out.println("Error: " + e);
    }
  }
```

Al almacenar registros de una sola medida, los datos se representan lógicamente de la siguiente manera.

Tiempo	device_id	tipo_dispositivo	os_version	región	measure_name	measure_value::bigint	measure_value::double
2021-09-07 21:48:44.000000000	12345678	iPhone 11	14.8	us-east-1	video_startup_time	200	
2021-09-07 21:48:44.000000000	12345678	iPhone 11	14.8	us-east-1	ratio de rebúfer		0,5
2021-09-07 21:48:44.000000000	12345678	iPhone 11	14.8	us-east-1	fallos en la reproducción de vídeo	0	
2021-09-07 21:48:44.000000000	12345678	iPhone 11	14.8	us-east-1	velocidad de fotogramas media		0.85

Tiempo	device_id	tipo_dispositivo	os_version	región	measure_name	measure_value::bigint	measure_value::double
2021-09-07 21:53:44.000000000	12345678	iPhone 11	14.8	us-east-1	video_startup_time	500	
2021-09-07 21:53:44.000000000	12345678	iPhone 11	14.8	us-east-1	ratio de rebúfer		1.5
2021-09-07 21:53:44.000000000	12345678	iPhone 11	14.8	us-east-1	fallos en la reproducción de vídeo	10	
2021-09-07 21:53:44.000000000	12345678	iPhone 11	14.8	us-east-1	velocidad de fotogramas media		0.2

Consultas: puede escribir una consulta que recupere todos los puntos de datos con la misma marca de tiempo recibidos en los últimos 15 minutos de la siguiente manera.

```
with cte_video_startup_time as ( SELECT time, device_id, device_type, os_version,
 region, measure_value::bigint as video_startup_time FROM table where time >= ago(15m)
 and measure_name=”video_startup_time”),
cte_rebuffering_ratio as ( SELECT time, device_id, device_type, os_version, region,
 measure_value::double as rebuffering_ratio FROM table where time >= ago(15m) and
 measure_name=”rebuffering_ratio”),
cte_video_playback_failures as ( SELECT time, device_id, device_type, os_version,
 region, measure_value::bigint as video_playback_failures FROM table where time >=
 ago(15m) and measure_name=”video_playback_failures”),
```

```
cte_average_frame_rate as ( SELECT time, device_id, device_type, os_version, region,
 measure_value::double as average_frame_rate FROM table where time >= ago(15m) and
 measure_name=”average_frame_rate”)
SELECT a.time, a.device_id, a.os_version, a.region, a.video_startup_time,
 b.rebuffering_ratio, c.video_playback_failures, d.average_frame_rate FROM
 cte_video_startup_time a, cte_buffering_ratio b, cte_video_playback_failures c,
 cte_average_frame_rate d WHERE
a.time = b.time AND a.device_id = b.device_id AND a.os_version = b.os_version AND
 a.region=b.region AND
a.time = c.time AND a.device_id = c.device_id AND a.os_version = c.os_version AND
 a.region=c.region AND
a.time = d.time AND a.device_id = d.device_id AND a.os_version = d.os_version AND
 a.region=d.region
```

Coste de la carga de trabajo: el coste de esta carga de trabajo se estima en 373,23$ al mes con registros de una sola medida

Uso de registros de múltiples medidas

Modelado de datos: con los registros de medidas múltiples, crearemos un registro que contenga las cuatro medidas (tiempo de inicio del vídeo, relación de realmacenamiento, errores de reproducción del vídeo y velocidad media de fotogramas), las cuatro dimensiones (device_id, device_type, os_version y región) y una marca de tiempo.

Escritura: al escribir datos en Amazon Timestream for Live Analytics, los registros se crean de la siguiente manera.

```
public void writeRecords() {
    System.out.println("Writing records");
    // Specify repeated values for all records
    List<Record> records = new ArrayList<>();
    final long time = System.currentTimeMillis();

    List<Dimension> dimensions = new ArrayList<>();

    final Dimension device_id = new
 Dimension().withName("device_id").withValue("12345678");
    final Dimension device_type = new
 Dimension().withName("device_type").withValue("iPhone 11");
    final Dimension os_version = new
 Dimension().withName("os_version").withValue("14.8");
    final Dimension region = new Dimension().withName("region").withValue("us-east-1");
```

```
    dimensions.add(device_id);
    dimensions.add(device_type);
    dimensions.add(os_version);
    dimensions.add(region);

    Record videoMetrics = new Record()
        .withDimensions(dimensions)
        .withMeasureName("video_metrics")
        .withTime(String.valueOf(time));
        .withMeasureValueType(MeasureValueType.MULTI)
        .withMeasureValues(
          new MeasureValue()
         .withName("video_startup_time")
         .withValue("0")
         .withValueType(MeasureValueType.BIGINT),
         new MeasureValue()
  .withName("rebuffering_ratio")
         .withValue("0.5")
         .withType(MeasureValueType.DOUBLE),
          new MeasureValue()
         .withName("video_playback_failures")
         .withValue("0")
         .withValueType(MeasureValueType.BIGINT),
   new MeasureValue()
          .withName("average_frame_rate")
         .withValue("0.5")
         .withValueType(MeasureValueType.DOUBLE))

    records.add(videoMetrics);

    WriteRecordsRequest writeRecordsRequest = new WriteRecordsRequest()
        .withDatabaseName(DATABASE_NAME)
        .withTableName(TABLE_NAME)
        .withRecords(records);

    try {
      WriteRecordsResult writeRecordsResult =
amazonTimestreamWrite.writeRecords(writeRecordsRequest);
      System.out.println("WriteRecords Status: " +
writeRecordsResult.getSdkHttpMetadata().getHttpStatusCode());
    } catch (RejectedRecordsException e) {
      System.out.println("RejectedRecords: " + e);
      for (RejectedRecord rejectedRecord : e.getRejectedRecords()) {
        System.out.println("Rejected Index " + rejectedRecord.getRecordIndex() + ": "
```

```
            + rejectedRecord.getReason());
      }
      System.out.println("Other records were written successfully. ");
    } catch (Exception e) {
      System.out.println("Error: " + e);
    }
  }
```

Al almacenar registros de múltiples medidas, los datos se representan de forma lógica de la siguiente manera.

Tiempo	device_id	tipo_dispositivo	os_version	región	measure_name	tiempo de inicio del vídeo	ratio de rebúfer	video_playback fallos	velocidad de fotogramas promedio
2021-09-7 21:48:44 0000000	1234567	iPhone 11	14.8	us-east-1	video_metrics	200	0,5	0	0.85
2021-09-7 21:53:44 0000000	1234567	iPhone 11	14.8	us-east-1	video_metrics	500	1.5	10	0.2

Consultas: puede escribir una consulta que recupere todos los puntos de datos con la misma marca de tiempo recibidos en los últimos 15 minutos de la siguiente manera.

```
SELECT time, device_id, device_type, os_version, region, video_startup_time,
 rebuffering_ratio, video_playback_failures, average_frame_rate FROM table where time
 >= ago(15m)
```

Coste de la carga de trabajo: el coste de la carga de trabajo se estima en 127,43$ con registros de múltiples medidas.

> **Note**
>
> En este caso, el uso de registros con múltiples medidas reduce 2,5 veces el gasto mensual total estimado: el coste de escritura de datos se reduce 3,3 veces, el coste de almacenamiento se reduce 3,3 veces y el coste de las consultas se reduce 1,2 veces.

Escribir datos con una marca de tiempo que existe en el pasado o en el futuro

Timestream for Live Analytics ofrece la posibilidad de escribir datos con una marca de tiempo que se encuentra fuera del intervalo de retención del almacén de memoria mediante un par de mecanismos diferentes.

- Grabaciones en almacenamiento magnético: puede escribir los datos que lleguen tarde directamente en el almacén magnético mediante escrituras en el almacén magnético. Para utilizar las escrituras por almacenamiento magnético, primero debe habilitar las escrituras por almacenamiento magnético para una tabla. A continuación, puede introducir datos en la tabla mediante el mismo mecanismo utilizado para escribir los datos en el almacén de memoria. Amazon Timestream for Live Analytics escribirá automáticamente los datos en el almacén magnético en función de su marca de tiempo.

  > **Note**
  >
  > La write-to-read latencia del almacén magnético puede ser de hasta 6 horas, a diferencia de la escritura de datos en el almacén de memoria, donde la write-to-read latencia se sitúa en un rango inferior a un segundo.

- TIMESTAMPtipo de datos para medidas: puede usar el TIMESTAMP tipo de datos para almacenar datos del pasado, presente o futuro. Un registro de múltiples medidas puede tener varios atributos de tipo de TIMESTAMP datos, además del campo de tiempo de un registro. TIMESTAMPLos atributos, en un registro de múltiples medidas, pueden tener marcas de tiempo en el futuro o en el pasado y comportarse como el campo de tiempo, excepto que Timestream for Live Analytics no indexa los valores de tipo TIMESTAMP en un registro de múltiples medidas.

  > **Note**
  >
  > El tipo de TIMESTAMP datos solo se admite para registros de múltiples medidas.

Coherencia eventual de las lecturas

Timestream for Live Analytics admite una semántica de coherencia eventual en las lecturas. Esto significa que cuando consulta datos inmediatamente después de escribir un lote de datos en Timestream for Live Analytics, es posible que los resultados de la consulta no reflejen los resultados de una operación de escritura completada recientemente. Si repite estas solicitudes de consulta después de un breve período de tiempo, los resultados deberían arrojar los datos más recientes.

El procesamiento por lotes escribe con WriteRecords API

Amazon Timestream para Live Analytics le permite escribir puntos de datos de una sola serie temporal o puntos de datos de varias series en una sola solicitud de escritura. La agrupación por lotes de varios puntos de datos en una sola operación de escritura resulta beneficiosa desde el punto de vista del rendimiento y los costes. Consulte Escribe la sección de medición y precios para obtener más información.

> **Note**
>
> Es posible que sus solicitudes de escritura a Timestream for Live Analytics se reduzcan a medida que Timestream for Live Analytics se amplíe para adaptarse a las necesidades de ingesta de datos de su aplicación. Si sus aplicaciones encuentran excepciones de limitación, debe seguir enviando datos con el mismo rendimiento (o superior) para permitir que Timestream for Live Analytics se adapte automáticamente a las necesidades de su aplicación.

Carga por lotes

Con la carga por lotes para Amazon Timestream, puede LiveAnalytics incorporar archivos almacenados en Amazon S3 CSV a Timestream por lotes. Con esta nueva funcionalidad, puede tener sus datos en Timestream LiveAnalytics sin tener que depender de otras herramientas ni escribir código personalizado. Puede utilizar la carga por lotes para rellenar los datos con tiempos de espera flexibles, como los datos que no se necesitan inmediatamente para consultarlos o analizarlos.

Puede crear tareas de carga por lotes mediante el AWS Management Console AWS CLI, el y el. AWS SDKs Para obtener más información, consulte Uso de la carga por lotes con la consola, Uso de la carga por lotes con el AWS CLI y Uso de la carga por lotes con el AWS SDKs.

Para obtener más información sobre la carga por lotes, consulteUso de la carga por lotes en Timestream para LiveAnalytics.

Elegir entre la WriteRecords API operación y la carga por lotes

Con esta WriteRecords API operación, puede escribir los datos de sus series temporales de transmisión en Timestream para LiveAnalytics que los genere su sistema. Al usarlo WriteRecords, puede ingerir continuamente un único punto de datos o lotes de datos más pequeños en tiempo real. Timestream for le LiveAnalytics ofrece un esquema flexible que detecta automáticamente los nombres de las columnas y los tipos de datos de su Timestream para LiveAnalytics tablas, en función de los nombres de las dimensiones y los tipos de datos de los puntos de datos que especifique al invocar las escrituras en la base de datos.

Por el contrario, la carga por lotes permite incorporar datos de series temporales por lotes de los archivos fuente (CSVarchivos) a Timestream, utilizando un modelo de datos que usted defina. LiveAnalytics Algunos ejemplos de cuándo utilizar la carga por lotes con un archivo fuente son la importación masiva de datos de series temporales para la evaluación de Timestream LiveAnalytics mediante una prueba de concepto, la importación masiva de datos de series temporales desde un dispositivo de IoT que estuvo fuera de línea durante algún tiempo y la migración de datos históricos de series temporales de Amazon S3 a Timestream for. LiveAnalytics Para obtener información sobre la carga por lotes, consulte. Uso de la carga por lotes en Timestream para LiveAnalytics

Ambas soluciones son seguras, fiables y eficaces.

Úselo WriteRecords cuando:

- Transmite cantidades de datos más pequeñas (menos de 10 MB) por solicitud.
- Rellenar las tablas existentes.
- Ingerir datos de un flujo de registro.
- Realización de análisis en tiempo real.
- Requiere una latencia más baja.

Utilice la carga por lotes cuando:

- Ingerir grandes cantidades de datos que se originan en Amazon S3 en CSV archivos. Para obtener más información acerca de los límites, consulte Cuotas.
- Rellenar tablas nuevas, como en el caso de una migración de datos.
- Enriquecer las bases de datos con datos históricos (incorporación a tablas nuevas).

- Tiene datos de origen que cambian lentamente o no cambian en absoluto.
- Dispone de tiempos de espera flexibles, ya que una tarea de carga por lotes puede estar pendiente hasta que haya recursos disponibles, especialmente si carga una gran cantidad de datos. La carga por lotes es adecuada para datos que no necesitan estar fácilmente disponibles para consultarlos o analizarlos para añadir más claridad.

Almacenamiento

Timestream for Live Analytics almacena y organiza los datos de las series temporales para optimizar el tiempo de procesamiento de las consultas y reducir los costes de almacenamiento. Ofrece almacenamiento de datos por niveles y admite dos niveles de almacenamiento: un almacén de memoria y un almacén magnético. El almacén de memoria está optimizado para escrituras de datos de alto rendimiento y consultas rápidas point-in-time. El almacén magnético está optimizado para reducir el rendimiento, la escritura tardía de datos, el almacenamiento de datos a largo plazo y la rapidez de las consultas analíticas.

Timestream for Live Analytics garantiza la durabilidad de los datos al replicar automáticamente los datos de la memoria y del almacén magnético en distintas zonas de disponibilidad en una sola zona. Región de AWS Todos los datos se graban en el disco antes de confirmar que la solicitud de escritura está completa.

Timestream para Live Analytics le permite configurar políticas de retención para mover los datos del almacén de memoria al almacén magnético. Cuando los datos alcanzan el valor configurado, Timestream for Live Analytics los mueve automáticamente al almacén magnético. También puede establecer un valor de retención en el almacén magnético. Cuando los datos caducan y salen del almacén magnético, se eliminan permanentemente.

Por ejemplo, imagine un escenario en el que se configura el almacén de memoria para almacenar datos de una semana y el almacén magnético para almacenar datos de un año. La antigüedad de los datos se calcula mediante la marca de tiempo asociada al punto de datos. Cuando los datos del almacén de memoria cumplen una semana, se mueven automáticamente al almacén magnético. A continuación, se conservan en el almacén magnético durante un año. Cuando los datos cumplen un año, se eliminan de Timestream for Live Analytics. Los valores de retención del almacén de memoria y del almacén magnético definen de forma acumulativa la cantidad de tiempo que los datos se almacenarán en Timestream for Live Analytics. Esto significa que, en el escenario anterior, desde el momento de la llegada de los datos, los datos se almacenan en Timestream for Live Analytics durante un período total de 1 año y 1 semana.

> **Note**
>
> Al actualizar el período de retención de la memoria o del almacén magnético, el cambio de retención se aplica a partir de ese momento. Por ejemplo, si el período de retención del almacén de memoria se estableció en 2 horas y luego se cambió a 24 horas al actualizar las políticas de retención de la tabla, el almacén de memoria podrá almacenar 24 horas de datos, pero se rellenará con 24 horas de datos 22 horas después de realizar este cambio. Timestream for Live Analytics no recupera datos del almacén magnético para rellenar el almacén de memoria.

Para garantizar la seguridad de los datos de series temporales, los datos de Timestream for Live Analytics siempre están cifrados de forma predeterminada. Esto se aplica a los datos en tránsito y en reposo. Además, Timestream for Live Analytics le permite utilizar claves gestionadas por el cliente para proteger sus datos en el almacén magnético. Para obtener más información sobre las claves administradas por el cliente, consulte. AWS KMS keys

Consultas

Con Timestream for Live Analytics, puede almacenar y analizar fácilmente métricas para datos de sensores para DevOps aplicaciones de IoT y datos de telemetría industrial para el mantenimiento de equipos, así como muchos otros casos de uso. El motor de consultas adaptativo y diseñado específicamente de Timestream for Live Analytics le permite acceder a los datos de todos los niveles de almacenamiento mediante una sola afirmación. SQL Accede a los datos de todos los niveles de almacenamiento y los combina de forma transparente sin necesidad de especificar la ubicación de los datos. Puede utilizar esta opción SQL para consultar datos en Timestream for Live Analytics para recuperar datos de series temporales de una o más tablas. Puede acceder a la información de metadatos de las bases de datos y las tablas. Timestream for Live Analytics SQL también admite funciones integradas para el análisis de series temporales. Puede consultar la Consulta el idioma de referencia referencia para obtener más detalles.

Timestream for Live Analytics está diseñado para tener una arquitectura de consulta, almacenamiento e ingesta de datos totalmente disociada, en la que cada componente puede escalarse de forma independiente de los demás componentes (lo que le permite ofrecer una escala prácticamente infinita para las necesidades de una aplicación). Esto significa que Timestream for Live Analytics no se interrumpe cuando las aplicaciones envían cientos de terabytes de datos al día o ejecutan millones de consultas para procesar cantidades pequeñas o grandes de datos. A medida que los datos aumentan con el tiempo, la latencia de las consultas en Timestream for Live Analytics

permanece prácticamente inalterada. Esto se debe a que la arquitectura de consultas de Timestream for Live Analytics puede aprovechar enormes cantidades de paralelismo para procesar volúmenes de datos más grandes y escalar automáticamente para adaptarse a las necesidades de rendimiento de las consultas de una aplicación.

Modelo de datos

Timestream admite dos modelos de datos para las consultas: el modelo plano y el modelo de series temporales.

> **Note**
>
> Los datos de Timestream se almacenan mediante el modelo plano y es el modelo predeterminado para consultar datos. El modelo de series temporales es un concepto de tiempo de consulta y se utiliza para el análisis de series temporales.

- Modelo plano
- Modelo de series temporales

Modelo plano

El modelo plano es el modelo de datos predeterminado de Timestream para las consultas. Representa datos de series temporales en formato tabular. Los nombres de las dimensiones, el tiempo, los nombres de las medidas y los valores de las medidas aparecen como columnas. Cada fila de la tabla es un punto de datos atómicos que corresponde a una medición en un momento específico dentro de una serie temporal. Las bases de datos, tablas y columnas de Timestream tienen algunas restricciones de nomenclatura. Estas se describen en. Límites de los servicios

La siguiente tabla muestra un ejemplo ilustrativo de cómo Timestream almacena los datos que representan la CPU utilización, la utilización de la memoria y la actividad de la red de las EC2 instancias, cuando los datos se envían como un registro de medida única. En este caso, las dimensiones son la región, la zona de disponibilidad, la nube privada virtual y la instancia IDs de las instancias. EC2 Las medidas son la CPU utilización, la utilización de la memoria y los datos de red entrantes de las EC2 instancias. Las columnas region, az, vpc e instance_id contienen los valores de las dimensiones. La columna time contiene la marca de tiempo de cada registro. La columna measure_name contiene los nombres de las medidas representadas por cpu-utilization, memory_utilization y network_bytes_in. Las columnas measure_value: :double contienen las medidas

emitidas como dobles (por ejemplo, la utilización y la utilización de la memoria). CPU La columna measure_value: :bigint contiene las medidas emitidas como números enteros, por ejemplo, los datos de red entrantes.

Tiempo	región	az	vpc	instance_id	measure_name	measure_value::double	measure_value::bigint
2019-12-04 19:00:00.000 000000	us-east-1	US-East-1d	vpc-1a2b3c4d	i-1234567890abcdef0	utilización de la CPU	35,0	null
2019-12-04 19:00:01.000 000000	us-east-1	US-East-1d	vpc-1a2b3c4d	i-1234567890abcdef0	utilización de la CPU	38.2	null
2019-12-04 19:00:02.000 000000	us-east-1	US-East-1d	vpc-1a2b3c4d	i-1234567890abcdef0	utilización de la CPU	45,3	null
2019-12-04 19:00:00.000 000000	us-east-1	US-East-1d	vpc-1a2b3c4d	i-1234567890abcdef0	memory_utilization	54,9	null
2019-12-04 19:00:01.000 000000	us-east-1	US-East-1d	vpc-1a2b3c4d	i-1234567890abcdef0	memory_utilization	42.6	null

Tiempo	región	az	vpc	instance_id	measure_name	measure_value::double	measure_value::bigint
2019-12-04 19:00:02.000 000000	us-east-1	US-East-1d	vpc-1a2b3c4d	i-1234567890abcdef0	memory_utilization	3.3	null
2019-12-04 19:00:00.000 000000	us-east-1	US-East-1d	vpc-1a2b3c4d	i-1234567890abcdef0	network_bytes	34.400	null
2019-12-04 19:00:01.000 000000	us-east-1	US-East-1d	vpc-1a2b3c4d	i-1234567890abcdef0	network_bytes	1500	null
2019-12-04 19:00:02.000 000000	us-east-1	US-East-1d	vpc-1a2b3c4d	i-1234567890abcdef0	network_bytes	6000	null

La siguiente tabla muestra un ejemplo ilustrativo de cómo Timestream almacena los datos que representan la CPU utilización, la utilización de la memoria y la actividad de la red de las EC2 instancias, cuando los datos se envían como un registro de múltiples medidas.

Tiempo	región	az	vpc	instance_id	measure_name	cpu_utilization	memory_utilization	bytes de red
04/12/2019 19:00:00.000 000000	us-east-1	US-East-1d	vpc-1a2b3c4d	i-1234567890abcde0	métricas	35,0	54,9	34.400
2019-12-04 19:00:01.000 000000	us-east-1	US-East-1d	vpc-1a2b3c4d	i-1234567890abcde0	métricas	38.2	42,6	1500
2019-12-04 19:00:02.000 000000	us-east-1	US-East-1d	vpc-1a2b3c4d	i-1234567890abcde0	métricas	45.3	3.3	6600

Modelo de series temporales

El modelo de series temporales es una construcción temporal de consulta que se utiliza para el análisis de series temporales. Representa los datos como una secuencia ordenada de pares (tiempo, valor de medición). Timestream admite funciones de series temporales, como la interpolación, para que pueda cubrir los vacíos en los datos. Para utilizar estas funciones, debe convertir los datos al modelo de series temporales mediante funciones como create_time_series. Consulte para obtener más información. Consulta el idioma de referencia

Utilizando el ejemplo anterior de la EC2 instancia, aquí están los datos de CPU uso expresados como una serie temporal.

región	az	vpc	instance_id	cpu_utilization
us-east-1	us-east-1d	vpc-1a2b3c4d	i-1234567890abcdef0	[{hora: 2019-12-04

región	az	vpc	instance_id	cpu_utilization
				19:00:00.000 000000, valor: 35}, {hora: 2019-12-04 19:00:01.000 000000, valor: 38,2}, {hora: 2019-12-04 19:00:02.000 000000, valor: 45,3}]

Consultas programadas

La función de consulta programada de Amazon Timestream for Live Analytics es una solución totalmente gestionada, escalable y sin servidor para calcular y almacenar agregados, acumulaciones y otros tipos de datos preprocesados que normalmente se utilizan para impulsar los paneles operativos, los informes empresariales, los análisis ad hoc y otras aplicaciones. Las consultas programadas permiten que los análisis en tiempo real sean más eficaces y rentables, lo que le permite obtener información adicional a partir de sus datos y seguir tomando mejores decisiones empresariales.

Para obtener más información sobre las consultas programadas, consulteUso de consultas programadas en Timestream para LiveAnalytics.

Unidad de cómputo Timestream () TCU

Amazon Timestream for Live Analytics mide la capacidad informática que se le asigna para sus necesidades de consulta en la unidad de cálculo de Timestream (). TCU Uno TCU consta de 4 vCPUs y 16 GB de memoria. Al ejecutar consultas en Timestream para Live Analytics, el servicio las asigna TCUs bajo demanda en función de la complejidad de las consultas y de la cantidad de datos que se están procesando. La cantidad TCUs que consume una consulta determina el costo asociado.

> **Note**
>
> Todo lo Cuentas de AWS que se incorpore al servicio después del 29 de abril de 2024 se utilizará de forma predeterminada TCUs para consultar los precios.

En este tema

MaxQuery TCU

Esta configuración especifica el número máximo de unidades de cómputo que el servicio utilizará en cualquier momento para atender sus consultas. Para ejecutar consultas, debes establecer la capacidad mínima en 4TCUs. Puede establecer el número máximo de TCUs múltiplos de 4, por ejemplo, 4, 8, 16, 32, etc. Solo se le cobrará por los recursos informáticos que utilice para su carga de trabajo. Por ejemplo, si estableces el máximo TCUs en 128, pero utilizas siempre solo 8TCUs. Solo se te cobrará por el tiempo durante el cual usaste el 8TCUs. El valor predeterminado `MaxQueryTCU` de tu cuenta es 200. Puede ajustar `MaxQueryTCU` de 4 a 1000, utilizando la UpdateAccountSettingsAPIoperación AWS Management Console o con la tecla AWS SDK o AWS CLI.

Te recomendamos configurar el `MaxQueryTCU` para tu cuenta. Establecer un TCU límite máximo ayuda a controlar los costes al restringir el número de unidades informáticas que el servicio puede utilizar para la carga de trabajo de las consultas. Esto le permite predecir y administrar mejor el gasto en consultas.

Facturación de TCU

Cada una TCU se factura por hora con una granularidad por segundo y durante un mínimo de 30 segundos. La unidad de uso de estas unidades de cómputo es de -hora. TCU

Cuando ejecutas consultas, se te factura por lo TCUs utilizado durante el tiempo de ejecución de la consulta, medido en TCU horas. Por ejemplo:

- Tu carga de trabajo utiliza 20 horas TCUs durante 3 horas. Se te facturan 60 TCU horas (20 TCUs x 3 horas).
- Tu carga de trabajo utiliza 10 TCUs durante 30 minutos y 20 TCUs durante los siguientes 30 minutos. Se te facturan 15 TCU horas (10 TCUs x 0,5 horas + 20 TCUs x 0,5 horas).

El precio por TCU hora varía según. Región de AWS Consulta los precios de Amazon Timestream para obtener más información. A medida que aumenta la carga de trabajo, el servicio escala automáticamente la capacidad de cómputo hasta el TCU límite máximo especificado (`MaxQueryTCU`) para mantener un rendimiento uniforme. La `MaxQueryTCU` configuración actúa como un límite máximo para la capacidad de cómputo a la que se puede escalar el servicio. Esta configuración le ayuda a controlar la cantidad de recursos informáticos y, en consecuencia, su coste.

Configurando TCU

Al contratar el servicio, cada uno Cuenta de AWS tiene un `MaxQueryTCU` límite predeterminado de 200. Puedes actualizar este límite según sea necesario en cualquier momento mediante la UpdateAccountSettingsAPIoperación AWS Management Console o con la tecla AWS SDK o AWS CLI.

Si no está seguro de los valores que debe configurar, controle la `QueryTCU` métrica de su cuenta. Esta métrica está disponible en Amazon AWS Management Console y Amazon CloudWatch. Esta métrica proporciona información sobre el número máximo de unidades TCUs utilizadas por minuto. En función de los datos históricos y de su estimación del crecimiento futuro, `MaxQueryTCU` configúrelo de manera que se adapte a los picos de consumo. Te recomendamos que tengas un margen de al menos 4 a 16 puntos por TCUs encima de tu consumo máximo. Por ejemplo, si tu pico `QueryTCU` en los últimos 30 días fue de 128, te recomendamos que lo hagas `MaxQueryTCU` entre 132 y 144.

Estimación de las unidades de cómputo requeridas

Las unidades de cómputo pueden procesar consultas de forma simultánea. Para determinar el número de unidades de cómputo necesarias, tenga en cuenta las pautas generales de la siguiente tabla:

Consultas simultáneas	TCUs
7	4
14	8
21	12

> **Note**
>
> - Estas son pautas generales y el número real de unidades de cómputo necesarias depende de varios factores, como:
>   - La simultaneidad efectiva de las consultas.
>   - Patrones de consulta.
>   - El número de particiones escaneadas.
>   - Otras características específicas de la carga de trabajo.
> - Esta guía se refiere a las consultas que buscan datos desde los últimos minutos hasta una hora y cumplen con las prácticas recomendadas para consultas de Timestream y las pautas de modelado de datos.
> - Supervisa el rendimiento de la aplicación y la `QueryTCU` métrica para ajustar las unidades de cómputo, según sea necesario.

¿Cuándo aumentar MaxQuery TCU

Debería considerar la posibilidad de `MaxQueryTCU` aumentarlo en los siguientes escenarios:

- Su consumo máximo de consultas se acerca o alcanza la consulta máxima configurada actualmenteTCU. Te recomendamos configurar la consulta máxima como TCU mínimo entre 4 y 16 veces TCUs más que tu consumo máximo.

- Sus consultas devuelven un error de 4 veces mayor y se ha superado el mensaje MaxQueryTCU. Si prevé un aumento planificado de su carga de trabajo, revise y ajuste la consulta TCU máxima configurada en consecuencia.

¿Cuándo disminuir MaxQuery TCU

Debería considerar la posibilidad de `MaxQueryTCU` reducirlo en los siguientes escenarios:

- Su carga de trabajo tiene un patrón de uso predecible y estable, y usted conoce bien sus requisitos de uso informático. Reducir la consulta TCU máxima entre 4 y 16 veces TCU por encima de los picos de consumo puede ayudar a evitar el uso y los costes no intencionados. Puede modificar el valor mediante la UpdateAccountSettingsAPIoperación.
- El uso máximo de la carga de trabajo ha disminuido con el tiempo, ya sea debido a cambios en la aplicación o a los patrones de comportamiento de los usuarios. Reducir el máximo TCU puede ayudar a mitigar los costes no intencionados.

> **Note**
>
> En función del uso actual, reducir el cambio de TCU límite máximo puede tardar hasta 24 horas en hacerse efectivo. Solo se le facturará lo TCUs que realmente consuman sus consultas. Tener un TCU límite máximo de consultas más alto no afecta a tus costes, a menos que los TCUs utilices tu carga de trabajo.

Supervisar el uso con CloudWatch métricas

Para monitorear su TCU uso, Timestream for Live Analytics proporciona la siguiente CloudWatch métrica:. `QueryTCU` Esta métrica especifica la cantidad de unidades de cómputo utilizadas en un minuto y se emite cada minuto. Puede elegir monitorizar el máximo y el mínimo TCUs utilizados en un minuto. También puedes configurar alarmas en esta métrica para hacer un seguimiento de los costes de las consultas en tiempo real.

Comprender las variaciones en el uso de las unidades de cómputo

La cantidad de recursos informáticos necesarios para las consultas puede aumentar o disminuir en función de varios parámetros. Por ejemplo, el volumen de datos, los patrones de ingesta de datos, la latencia de las consultas, la forma de las consultas, la eficiencia de las consultas y las

combinaciones de consultas que utilizan consultas analíticas y en tiempo real. Estos parámetros pueden hacer que se necesiten TCU unidades más altas o más bajas para su carga de trabajo. En un estado estable en el que estos parámetros no cambien, puede observar que la cantidad de unidades informáticas necesarias para la carga de trabajo disminuye. En consecuencia, esto puede reducir el coste mensual.

Además, si alguno de estos parámetros de la carga de trabajo o los datos cambia, la cantidad de unidades informáticas necesarias podría aumentar. Cuando Timestream recibe una consulta, en función de las particiones de datos a las que acceda la consulta, Timestream decide el número de recursos informáticos necesarios para abordar la consulta de forma eficaz.

A intervalos periódicos, en función de sus patrones de ingesta y acceso a las consultas, Timestream optimiza el diseño de los datos. Timestream realiza la optimización agrupando las particiones a las que menos se accede en una sola partición o dividiendo una partición activa en varias particiones para mejorar el rendimiento. En consecuencia, la capacidad de procesamiento utilizada por la misma consulta puede variar ligeramente en diferentes momentos.

> Optar por utilizar los TCU precios para sus consultas
>
> Como usuario actual, puedes registrarte una sola vez TCUs para poder gestionar mejor los costes y eliminar el número mínimo de bytes medidos por consulta. Puede activar la opción mediante la UpdateAccountSettingsAPIoperación AWS Management Console o con la tecla o. AWS SDK AWS CLI En la API operación, defina `COMPUTE_UNITS` el `QueryPricingModel` parámetro en.
> Optar por el modelo de precios basado en la computación es un cambio irreversible.

Acceder a Timestream para LiveAnalytics

Puede acceder a Timestream para LiveAnalytics usar la consola, o la. CLI API Para obtener información sobre cómo acceder a Timestream for LiveAnalytics, revise lo siguiente:

Temas

- Inscríbase en una Cuenta de AWS
- Creación de un usuario con acceso administrativo
- Proporcione Timestream para el acceso LiveAnalytics
- Concesión de acceso programático

Inscríbase en una Cuenta de AWS

Si no tiene uno Cuenta de AWS, complete los siguientes pasos para crearlo.

Para suscribirse a una Cuenta de AWS

1. Abrir https://portal.aws.amazon.com/billing/registro.
2. Siga las instrucciones que se le indiquen.

   Parte del procedimiento de registro consiste en recibir una llamada telefónica e indicar un código de verificación en el teclado del teléfono.

   Cuando te registras en un Cuenta de AWS, Usuario raíz de la cuenta de AWSse crea un. El usuario raíz tendrá acceso a todos los Servicios de AWS y recursos de esa cuenta. Como práctica recomendada de seguridad, asigne acceso administrativo a un usuario y utilice únicamente el usuario raíz para realizar tareas que requieren acceso de usuario raíz.

AWS te envía un correo electrónico de confirmación una vez finalizado el proceso de registro. En cualquier momento, puedes ver la actividad de tu cuenta actual y administrarla accediendo a https://aws.amazon.com/y seleccionando Mi cuenta.

Creación de un usuario con acceso administrativo

Después de crear un usuario administrativo Cuenta de AWS, asegúrelo Usuario raíz de la cuenta de AWS AWS IAM Identity Center, habilite y cree un usuario administrativo para no usar el usuario root en las tareas diarias.

Proteja su Usuario raíz de la cuenta de AWS

1. Inicie sesión AWS Management Consolecomo propietario de la cuenta seleccionando el usuario root e introduciendo su dirección de Cuenta de AWS correo electrónico. En la siguiente página, escriba su contraseña.

   Para obtener ayuda para iniciar sesión con el usuario raíz, consulte Iniciar sesión como usuario raíz en la Guía del usuario de AWS Sign-In .
2. Activa la autenticación multifactorial (MFA) para tu usuario root.

   Para obtener instrucciones, consulte Habilitar un MFA dispositivo virtual para el usuario Cuenta de AWS root (consola) en la Guía del IAM usuario.

Creación de un usuario con acceso administrativo

1. Habilite IAM Identity Center.

   Consulte las instrucciones en Activar AWS IAM Identity Center en la Guía del usuario de AWS IAM Identity Center .

2. En IAM Identity Center, conceda acceso administrativo a un usuario.

   Para ver un tutorial sobre cómo usar el Directorio de IAM Identity Center como fuente de identidad, consulte Configurar el acceso de los usuarios con la configuración predeterminada Directorio de IAM Identity Center en la Guía del AWS IAM Identity Center usuario.

Iniciar sesión como usuario con acceso de administrador

- Para iniciar sesión con su usuario de IAM Identity Center, utilice el inicio de sesión URL que se envió a su dirección de correo electrónico cuando creó el usuario de IAM Identity Center.

  Para obtener ayuda para iniciar sesión con un usuario de IAM Identity Center, consulte Iniciar sesión en el portal de AWS acceso en la Guía del AWS Sign-In usuario.

Concesión de acceso a usuarios adicionales

1. En IAM Identity Center, cree un conjunto de permisos que siga la práctica recomendada de aplicar permisos con privilegios mínimos.

   Para conocer las instrucciones, consulte Create a permission set en la Guía del usuario de AWS IAM Identity Center .

2. Asigne usuarios a un grupo y, a continuación, asigne el acceso de inicio de sesión único al grupo.

   Para conocer las instrucciones, consulte Add groups en la Guía del usuario de AWS IAM Identity Center .

Proporcione Timestream para el acceso LiveAnalytics

Los permisos necesarios para acceder a Timestream ya LiveAnalytics están concedidos al administrador. En el caso de los demás usuarios, debes concederles Timestream para el LiveAnalytics acceso mediante la siguiente política:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "timestream:*",
        "kms:DescribeKey",
        "kms:CreateGrant",
        "kms:Decrypt",
        "dbqms:CreateFavoriteQuery",
        "dbqms:DescribeFavoriteQueries",
        "dbqms:UpdateFavoriteQuery",
        "dbqms:DeleteFavoriteQueries",
        "dbqms:GetQueryString",
        "dbqms:CreateQueryHistory",
        "dbqms:UpdateQueryHistory",
        "dbqms:DeleteQueryHistory",
        "dbqms:DescribeQueryHistory",
        "s3:ListAllMyBuckets"
      ],
      "Resource": "*"
    }
  ]
}
```

> **Note**
>
> Para obtener más información`dbqms`, consulte las acciones, los recursos y las claves de condición del servicio de metadatos de consulta de bases de datos. Para obtener más información, `kms` consulte las acciones, los recursos y las claves de condición del Servicio de administración de AWS claves.

Concesión de acceso programático

Los usuarios necesitan acceso programático si quieren interactuar con personas AWS ajenas a. AWS Management Console La forma de conceder el acceso programático depende del tipo de usuario que acceda. AWS

Para conceder acceso programático a los usuarios, elija una de las siguientes opciones.

¿Qué usuario necesita acceso programático?	Para	Mediante
Identidad del personal (Los usuarios se administran en IAM Identity Center)	Utilice credenciales temporales para firmar las solicitudes programáticas dirigidas al AWS CLI AWS SDKs, o AWS APIs.	Siga las instrucciones de la interfaz que desea utilizar: • Para ello AWS CLI, consulte Configuración del AWS CLI uso AWS IAM Identity Center en la Guía del AWS Command Line Interface usuario. • Para AWS SDKs ver las herramientas y AWS APIs, consulte la autenticación de IAM Identity Center en la Guía de referencia de herramientas AWS SDKs y herramientas.
IAM	Utilice credenciales temporales para firmar las solicitudes programáticas dirigidas al AWS CLI AWS SDKs, o AWS APIs.	Siga las instrucciones de Uso de credenciales temporales con AWS recursos de la Guía del IAM usuario.
IAM	(No recomendado) Utilice credenciales de larga duración para firmar las solicitudes programáticas dirigidas al AWS CLI, AWS SDKs, o AWS APIs.	Siga las instrucciones de la interfaz que desea utilizar: • Para ello AWS CLI, consulte Autenticación con credencia les IAM de usuario en la Guía del AWS Command Line Interface usuario. • Para obtener AWS SDKs información sobre las herramientas, consulte

¿Qué usuario necesita acceso programático?	Para	Mediante
		Autenticarse con credencia les de larga duración en la Guía de referencia de herramientas AWS SDKs y herramientas. • Para ello AWS APIs, consulte Administrar las claves de acceso de IAM los usuarios en la Guía del IAM usuario.

Mediante la consola

Puede usar la consola AWS de administración de Timestream Live Analytics para crear, editar, eliminar, describir y enumerar bases de datos y tablas. También puede usar la consola para ejecutar consultas.

Temas

- Tutorial
- Creación de una base de datos de
- Creación de una tabla
- Ejecutar una consulta
- Cree una consulta programada
- Eliminar una consulta programada
- Eliminación de una tabla
- Eliminación de una base de datos
- Edita una tabla
- Edita una base de datos

Tutorial

En este tutorial, se muestra cómo crear una base de datos con conjuntos de datos de ejemplo y cómo ejecutar consultas de ejemplo. Los conjuntos de datos de muestra que se utilizan en este tutorial se ven con frecuencia en el IoT y en DevOps los escenarios. El conjunto de datos de IoT contiene datos de series temporales, como la velocidad, la ubicación y la carga de un camión, para agilizar la gestión de la flota e identificar oportunidades de optimización. El DevOps conjunto de datos contiene métricas de EC2 instanciaCPU, como la utilización de la red y la memoria, para mejorar el rendimiento y la disponibilidad de las aplicaciones. Este es un tutorial en vídeo con las instrucciones que se describen en esta sección

Siga estos pasos para crear una base de datos con los conjuntos de datos de ejemplo y ejecutar consultas de ejemplo con la AWS consola.

1. Abra la AWS consola.
2. En el panel de navegación, elija Bases de datos
3. Haga clic en Crear base de datos.
4. En la página de creación de base de datos, introduzca lo siguiente:
   - Elija la configuración: seleccione una base de datos de muestra.
   - Nombre: introduzca el nombre de la base de datos que desee.
   - Elija conjuntos de datos de muestra: seleccione IoT y. DevOps
   - Haga clic en Crear base de datos para crear una base de datos que contenga dos tablas: IoT y DevOps rellenada con datos de muestra.
5. En el panel de navegación, elija el editor de consultas
6. Seleccione Consultas de muestra en el menú superior.
7. Haga clic en una de las consultas de muestra. Esto le llevará de vuelta al editor de consultas con el editor lleno de la consulta de ejemplo.
8. Haga clic en Ejecutar para ejecutar la consulta y ver los resultados de la consulta.

Creación de una base de datos de

Siga estos pasos para crear una base de datos mediante la AWS consola.

1. Abre la AWS consola.
2. En el panel de navegación, elija Bases de datos

3. Haga clic en Crear base de datos.
4. En la página de creación de base de datos, introduzca lo siguiente.
   - Elija la configuración: seleccione la base de datos estándar.
   - Nombre: introduzca el nombre de la base de datos que desee.
   - Cifrado: elija una KMS clave o utilice la opción predeterminada, en la que Timestream Live Analytics creará una KMS clave en su cuenta si aún no existe ninguna.
5. Haga clic en Crear base de datos para crear una base de datos.

Creación de una tabla

Siga estos pasos para crear una tabla con la AWS consola.

1. Abre la AWS consola.
2. En el panel de navegación, seleccione Tablas
3. Haga clic en Crear tabla.
4. En la página de creación de tabla, introduzca lo siguiente.
   - Nombre de la base de datos: seleccione el nombre de la base de datos creada enCreación de una base de datos de .
   - Nombre de tabla: introduzca un nombre de tabla de su elección.
   - Retención del almacén de memoria: especifique durante cuánto tiempo desea conservar los datos en el almacén de memoria. El almacén de memoria procesa los datos entrantes, incluidos los que llegan tarde (datos con una marca de tiempo anterior a la hora actual) y está optimizado para realizar consultas rápidas. point-in-time
   - Retención en el almacén magnético: especifique durante cuánto tiempo desea conservar los datos en el almacén magnético. El almacén magnético está diseñado para el almacenamiento a largo plazo y está optimizado para consultas analíticas rápidas.
5. Haga clic en Crear tabla.

Ejecutar una consulta

Siga estos pasos para ejecutar consultas con la AWS consola.

1. Abre la AWS consola.

2. En el panel de navegación, elija el editor de consultas
3. En el panel izquierdo, seleccione la base de datos creada enCreación de una base de datos de .
4. En el panel izquierdo, seleccione la base de datos creada enCreación de una tabla.
5. En el editor de consultas, puede ejecutar una consulta. Para ver las últimas 10 filas de la tabla, ejecuta:

```
SELECT * FROM <database_name>.<table_name> ORDER BY time DESC LIMIT 10
```

6. (Opcional) Activa Enable Insights para obtener información sobre la eficacia de tus consultas.

Cree una consulta programada

Siga estos pasos para crear una consulta programada mediante la AWS consola.

1. Abre la AWS consola.
2. En el panel de navegación, seleccione Consultas programadas.
3. Haga clic en Crear consulta programada.
4. En las secciones Nombre de la consulta y Tabla de destino, introduzca lo siguiente.
   - Nombre: introduzca un nombre de consulta.
   - Nombre de la base de datos: seleccione el nombre de la base de datos creada en. Creación de una base de datos de
   - Nombre de la tabla: seleccione el nombre de la tabla creada en. Creación de una tabla
5. En la sección Declaración de consulta, introduzca una declaración de consulta válida. A continuación, haga clic en Validar consulta.
6. En el modelo de tabla de destino, defina el modelo para cualquier atributo indefinido. Puede usar Visual Builder oJSON.
7. En la sección Programación de ejecución, seleccione Velocidad fija o Expresión crónica. Para expresiones cronológicas, consulte Programar expresiones para consultas programadas para obtener más información sobre las expresiones programadas.
8. En la sección de SNStemas, introduzca el SNS tema que se utilizará para la notificación.
9. En la sección Informe del registro de errores, introduzca la ubicación S3 que se utilizará para informar de los errores.

   Elija el Encryption key type (Tipo de clave de cifrado).

10. En la sección Configuración de seguridad de la AWS KMSclave, elija el tipo de AWS KMS clave.

    Introduzca la IAMfunción que LiveAnalytics utilizará Timestream for para ejecutar la consulta programada. Consulte los ejemplos de IAM políticas para las consultas programadas para obtener más información sobre los permisos necesarios y la relación de confianza para el rol.
11. Haga clic en Crear consulta programada.

Eliminar una consulta programada

Siga estos pasos para eliminar o deshabilitar una consulta programada mediante la AWS consola.

1. Abre la AWS consola.
2. En el panel de navegación, seleccione Consultas programadas
3. Seleccione la consulta programada creada enCree una consulta programada.
4. Seleccione Acciones.
5. Seleccione Desactivar o Eliminar.
6. Si seleccionó Eliminar, confirme la acción y seleccione Eliminar.

Eliminación de una tabla

Siga estos pasos para eliminar una base de datos mediante la AWS consola.

1. Abre la AWS consola.
2. En el panel de navegación, seleccione Tablas
3. Seleccione la tabla en la que creóCreación de una tabla.
4. Haga clic en Delete.
5. Escribe eliminar en el cuadro de confirmación.

Eliminación de una base de datos

Siga estos pasos para eliminar una base de datos mediante la AWS consola:

1. Abre la AWS consola.
2. En el panel de navegación, elija Bases de datos
3. Seleccione la base de datos que creó en Crear una base de datos.

4. Haga clic en Delete.
5. Escriba eliminar en el cuadro de confirmación.

Edita una tabla

Siga estos pasos para editar una tabla mediante la AWS consola.

1. Abre la AWS consola.
2. En el panel de navegación, seleccione Tablas
3. Seleccione la tabla en la que creóCreación de una tabla.
4. Haga clic en Editar
5. Edite los detalles de la tabla y guárdela.

   - Retención del almacén de memoria: especifique durante cuánto tiempo desea conservar los datos en el almacén de memoria. El almacén de memoria procesa los datos entrantes, incluidos los que llegan tarde (datos con una marca de tiempo anterior a la hora actual) y está optimizado para realizar consultas rápidas. point-in-time
   - Retención en el almacén magnético: especifique durante cuánto tiempo desea conservar los datos en el almacén magnético. El almacén magnético está diseñado para el almacenamiento a largo plazo y está optimizado para consultas analíticas rápidas.

Edita una base de datos

Siga estos pasos para editar una base de datos mediante la AWS consola.

1. Abre la AWS consola.
2. En el panel de navegación, elija Bases de datos
3. Seleccione la base de datos que creó en Crear una base de datos.
4. Haga clic en Editar
5. Edite los detalles de la base de datos y guárdelos.

Acceso a Amazon Timestream LiveAnalytics para usar el AWS CLI

Puede usar el AWS Command Line Interface (AWS CLI) para controlar varios AWS servicios desde la línea de comandos y automatizarlos mediante scripts. Puede utilizar el AWS CLI para operaciones

ad hoc. También puede usarla para incrustar Amazon Timestream LiveAnalytics for operaciones en scripts de utilidades.

Antes de poder usarlo AWS CLI con Timestream LiveAnalytics, debe configurar el acceso mediante programación. Para obtener más información, consulte Concesión de acceso programático.

Para obtener una lista completa de todos los comandos disponibles para el flujo temporal de LiveAnalytics Query API en el AWS CLI, consulte la Referencia de comandos.AWS CLI

Para obtener una lista completa de todos los comandos disponibles para el flujo temporal de LiveAnalytics Write API in the AWS CLI, consulte la Referencia de comandos.AWS CLI

Temas

- Descarga y configuración de la AWS CLI
- Uso del AWS CLI con Timestream para LiveAnalytics

Descarga y configuración de la AWS CLI

Se AWS CLI ejecuta en Windows, macOS o Linux. Para descargarlo, instalarlo y configurarlo, sigue estos pasos:

1. AWS CLI Descárguelo en http://aws.amazon.com/cli.
2. Siga las instrucciones para instalar AWS CLI y configurar el que aparecen AWS CLI en la Guía del AWS Command Line Interface usuario.

Uso del AWS CLI con Timestream para LiveAnalytics

El formato de línea de comandos consiste en Amazon Timestream LiveAnalytics para el nombre de la operación, seguido de los parámetros de esa operación. AWS CLI Admite una sintaxis abreviada para los valores de los parámetros, además de. JSON

Se utiliza `help` para enumerar todos los comandos disponibles en Timestream para. LiveAnalytics Por ejemplo:

```
aws timestream-write help
```

```
aws timestream-query help
```

También puede utilizar la `help` para describir un comando específico y obtener más información sobre su uso:

```
aws timestream-write create-database help
```

Por ejemplo, para crear una base de datos:

```
aws timestream-write create-database --database-name myFirstDatabase
```

Para crear una tabla con la función de almacenamiento magnético habilitada:

```
aws timestream-write create-table \
--database-name metricsdb \
--table-name metrics \
--magnetic-store-write-properties "{\"EnableMagneticStoreWrites\": true}"
```

Para escribir datos mediante registros de una sola medida:

```
aws timestream-write write-records \
--database-name metricsdb \
--table-name metrics \
--common-attributes "{\"Dimensions\":[{\"Name\":\"asset_id\", \"Value\":\"100\"}],
 \"Time\":\"1631051324000\",\"TimeUnit\":\"MILLISECONDS\"}" \
--records "[{\"MeasureName\":\"temperature\", \"MeasureValueType\":\"DOUBLE\",
\"MeasureValue\":\"30\"},{\"MeasureName\":\"windspeed\", \"MeasureValueType\":\"DOUBLE
\",\"MeasureValue\":\"7\"},{\"MeasureName\":\"humidity\", \"MeasureValueType\":\"DOUBLE
\",\"MeasureValue\":\"15\"},{\"MeasureName\":\"brightness\", \"MeasureValueType\":
\"DOUBLE\",\"MeasureValue\":\"17\"}]"
```

Para escribir datos mediante registros de múltiples medidas:

```
# wide model helper method to create Multi-measure records
function ingest_multi_measure_records {
  epoch=`date +%s`
  epoch+=$i

  # multi-measure records
  aws timestream-write write-records \
  --database-name $src_db_wide \
  --table-name $src_tbl_wide \
  --common-attributes "{\"Dimensions\":[{\"Name\":\"device_id\", \
```

```
              \"Value\":\"12345678\"},\
            {\"Name\":\"device_type\", \"Value\":\"iPhone\"}, \
            {\"Name\":\"os_version\", \"Value\":\"14.8\"}, \
            {\"Name\":\"region\", \"Value\":\"us-east-1\"} ], \
            \"Time\":\"$epoch\",\"TimeUnit\":\"MILLISECONDS\"}" \
--records "[{\"MeasureName\":\"video_metrics\", \"MeasureValueType\":\"MULTI\", \
  \"MeasureValues\": \
  [{\"Name\":\"video_startup_time\",\"Value\":\"0\",\"Type\":\"BIGINT\"}, \
  {\"Name\":\"rebuffering_ratio\",\"Value\":\"0.5\",\"Type\":\"DOUBLE\"}, \
  {\"Name\":\"video_playback_failures\",\"Value\":\"0\",\"Type\":\"BIGINT\"}, \
  {\"Name\":\"average_frame_rate\",\"Value\":\"0.5\",\"Type\":\"DOUBLE\"}]}]" \
--endpoint-url $ingest_endpoint \
  --region  $region
}

# create 5 records
for i in {100..105};
  do ingest_multi_measure_records $i;
done
```

Para consultar una tabla:

```
aws timestream-query query \
--query-string "SELECT time, device_id, device_type, os_version,
region, video_startup_time, rebuffering_ratio, video_playback_failures, \
average_frame_rate \
FROM metricsdb.metrics \
where time >= ago (15m)"
```

Para crear una consulta programada:

```
aws timestream-query create-scheduled-query \
  --name scheduled_query_name \
  --query-string "select bin(time, 1m) as time, \
          avg(measure_value::double) as avg_cpu, min(measure_value::double) as min_cpu,
 region \
          from $src_db.$src_tbl where measure_name = 'cpu' \
          and time BETWEEN @scheduled_runtime - (interval '5' minute)  AND
 @scheduled_runtime \
          group by region, bin(time, 1m)" \
  --schedule-configuration "{\"ScheduleExpression\":\"$cron_exp\"}" \
  --notification-configuration "{\"SnsConfiguration\":{\"TopicArn\":\"$sns_topic_arn
\"}}" \
```

```
  --scheduled-query-execution-role-arn "arn:aws:iam::452360119086:role/
TimestreamSQExecutionRole" \
  --target-configuration "{\"TimestreamConfiguration\":{\
          \"DatabaseName\": \"$dest_db\",\
          \"TableName\": \"$dest_tbl\",\
          \"TimeColumn\":\"time\",\
          \"DimensionMappings\":[{\
            \"Name\": \"region\", \"DimensionValueType\": \"VARCHAR\"
          }],\
          \"MultiMeasureMappings\":{\
            \"TargetMultiMeasureName\": \"mma_name\",
            \"MultiMeasureAttributeMappings\":[{\
              \"SourceColumn\": \"avg_cpu\", \"MeasureValueType\": \"DOUBLE\",
 \"TargetMultiMeasureAttributeName\": \"target_avg_cpu\"
            },\
            { \
              \"SourceColumn\": \"min_cpu\", \"MeasureValueType\": \"DOUBLE\",
 \"TargetMultiMeasureAttributeName\": \"target_min_cpu\"
            }] \
          }\
          }}" \
  --error-report-configuration "{\"S3Configuration\": {\
        \"BucketName\": \"$s3_err_bucket\",\
        \"ObjectKeyPrefix\": \"scherrors\",\
        \"EncryptionOption\": \"SSE_S3\"\
        }\
      }"
```

Uso del API

Además SDKs, Amazon Timestream LiveAnalytics for proporciona acceso REST API directo a través del patrón de detección de puntos finales. El patrón de detección de puntos finales se describe a continuación, junto con sus casos de uso.

El patrón de detección de puntos finales

Dado que los Timestream Live Analytics SDKs están diseñados para funcionar de forma transparente con la arquitectura del servicio, incluida la administración y el mapeo de los puntos finales del servicio, se recomienda utilizarlos para la mayoría de las SDKs aplicaciones. Sin embargo, hay algunos casos en los que es necesario utilizar el patrón Timestream para LiveAnalytics REST API la detección de puntos finales:

- Está utilizando VPCendpoints (AWS PrivateLink) con Timestream para LiveAnalytics
- Su aplicación utiliza un lenguaje de programación que aún no es compatible SDK
- Necesita un mejor control sobre la implementación del lado del cliente

Esta sección incluye información sobre cómo funciona el patrón de detección de puntos finales, cómo implementar el patrón de descubrimiento de puntos finales y notas de uso. Seleccione uno de los temas siguientes para obtener más información.

Temas

Cómo funciona el patrón de detección de puntos finales

Timestream se creó con una arquitectura celular para garantizar una mejor escalabilidad y propiedades de aislamiento del tráfico. Dado que cada cuenta de cliente está asignada a una celda específica de una región, su aplicación debe utilizar los puntos finales específicos de celda correctos a los que se ha asignado su cuenta. Al utilizar elSDKs, este mapeo se gestiona de forma transparente y no es necesario que gestione los puntos finales específicos de cada celda. Sin embargo, al acceder directamente al RESTAPI, tendrá que gestionar y mapear usted mismo los puntos finales correctos. Este proceso, el patrón de detección de puntos finales, se describe a continuación:

1. El patrón de detección de puntos finales comienza con una llamada a la `DescribeEndpoints` acción (que se describe en la `DescribeEndpoints`sección).
2. El punto final debe almacenarse en caché y reutilizarse durante el tiempo especificado por el valor devuelto time-to-live (TTL) (the `CachePeriodInMinutes`). De este modo, se API pueden realizar llamadas a Timestream Live Analytics durante todo el. TTL
3. Una vez que TTL caduque, se DescribeEndpoints debe realizar una nueva llamada para actualizar el punto final (en otras palabras, volver a empezar por el paso 1).

> **Note**
>
> La sintaxis, los parámetros y otra información de uso de la `DescribeEndpoints` acción se describen en la APIReferencia. Tenga en cuenta que la `DescribeEndpoints` acción está disponible a través de ambas SDKs y es idéntica en cada una de ellas.

Para obtener información sobre la implementación del patrón de detección de puntos finales, consulteImplementación del patrón de descubrimiento de puntos finales.

Implementación del patrón de descubrimiento de puntos finales

Para implementar el patrón de detección de puntos finales, elija un punto API (escritura o consulta), cree una DescribeEndpointssolicitud y utilice los puntos finales devueltos mientras duren los TTL valores devueltos. El procedimiento de implementación se describe a continuación.

> **Note**
>
> Asegúrese de estar familiarizado con las notas de uso.

Procedimiento de implementación

1. Adquiera el punto final contra el API que desea realizar llamadas (escribir o consultar) mediante la solicitud. `DescribeEndpoints`

   a. Cree una solicitud `DescribeEndpoints`que se corresponda con lo que le interesa (escribir o consultar) utilizando uno de los dos puntos finales que se describen a continuación. API No hay parámetros de entrada para la solicitud. Asegúrese de leer las siguientes notas.

      EscribeSDK:

      ```
      ingest.timestream.<region>.amazonaws.com
      ```

      ConsultaSDK:

      ```
      query.timestream.<region>.amazonaws.com
      ```

A `us-east-1` continuación se muestra un ejemplo de CLI llamada para una región.

```
REGION_ENDPOINT="https://query.timestream.us-east-1.amazonaws.com"
REGION=us-east-1
aws timestream-write describe-endpoints \
--endpoint-url $REGION_ENDPOINT \
--region $REGION
```

> **Note**
>
> El encabezado HTTP «Host» también debe contener el API punto final. La solicitud fallará si no se completa el encabezado. Este es un requisito estándar para todas las solicitudes HTTP /1.1. Si utilizas una HTTP biblioteca compatible con la versión 1.1 o posterior, la HTTP biblioteca debería rellenar automáticamente el encabezado.

> **Note**
>
> Sustituya *<region>* por el identificador de región de la región en la que se realiza la solicitud, p. ej. `us-east-1`

b. Analice la respuesta para extraer los puntos finales y los TTL valores de la memoria caché. La respuesta es una matriz de uno o más `Endpoint`objetos. Cada `Endpoint` objeto contiene una dirección de punto final (`Address`) y la dirección TTL correspondiente a ese punto final (`CachePeriodInMinutes`).

2. Almacene en caché el punto final hasta el valor especificadoTTL.
3. Cuando TTL caduque, recupere un nuevo punto final empezando desde el paso 1 de la implementación.

Notas de uso del patrón de descubrimiento de puntos finales

- La DescribeEndpointsacción es la única acción que reconocen los puntos finales regionales de Timestream Live Analytics.
- La respuesta contiene una lista de puntos finales contra los que puede realizar llamadas de Timestream Live Analytics. API

- Si la respuesta es correcta, debe haber al menos un punto final en la lista. Si hay más de un punto final en la lista, cualquiera de ellos se puede utilizar por igual para las API llamadas, y la persona que llama puede elegir el punto final para usarlo de forma aleatoria.
- Además de la DNS dirección del punto final, cada punto final de la lista especificará el tiempo de vida (TTL) permitido para usar el punto final especificado en minutos.
- El punto final debe almacenarse en caché y reutilizarse durante el tiempo especificado en el TTL valor devuelto (en minutos). Una vez que TTL caduque, se DescribeEndpointsdebe realizar una nueva llamada a para actualizar el dispositivo de punto final que se vaya a utilizar, ya que el punto final dejará de funcionar cuando TTL haya caducado.

Uso del AWS SDKs

Puede acceder a Amazon Timestream mediante. AWS SDKs Timestream admite dos SDKs por idioma; a saber, la escritura SDK y la consulta. SDK La función Write SDK se utiliza para realizar CRUD operaciones e insertar los datos de series temporales en Timestream. La consulta SDK se utiliza para consultar los datos de series temporales existentes almacenados en Timestream.

Una vez que haya completado los requisitos previos necesarios para su SDK elección, puede empezar con. Ejemplos de código

Temas

- Java
- Java v2
- Go
- Python
- Node.js
- .NET

Java

Para empezar a utilizar Java 1.0 SDK y Amazon Timestream, complete los requisitos previos que se describen a continuación.

Una vez que haya completado los requisitos previos necesarios para JavaSDK, puede empezar con. Ejemplos de código

Requisitos previos

Antes de empezar con Java, debe hacer lo siguiente:

1. Siga las instrucciones de AWS configuración que se indican enAcceder a Timestream para LiveAnalytics.
2. Configure un entorno de desarrollo Java descargando e instalando lo siguiente:
   - Kit de desarrollo Java SE 8 (como Amazon Corretto 8).
   - Java IDE (como Eclipse o IntelliJ).

   Para obtener más información, consulte Cómo empezar con AWS SDK for Java
3. Configure sus AWS credenciales y su región para el desarrollo:
   - Configure sus credenciales AWS de seguridad para usarlas con AWS SDK for Java.
   - Configure su AWS región para determinar su flujo de tiempo predeterminado para LiveAnalytics el punto final.

Con Apache Maven

Puede usar Apache Maven para configurar y crear proyectos. AWS SDK for Java

> **Note**
>
> Para usar Apache Maven, asegúrese de que su entorno de ejecución SDK y Java sean 1.8 o superiores.

Puede configurarla AWS SDK como una dependencia de Maven, tal y como se describe en Uso de SDK con Apache Maven.

Puede ejecutar, compilar y ejecutar el código fuente con el siguiente comando:

```
mvn clean compile
mvn exec:java -Dexec.mainClass=<your source code Main class>
```

> **Note**
>
> `<your source code Main class>`es la ruta a la clase principal del código fuente de Java.

Configurando tus AWS credenciales

AWS SDK for JavaRequiere que proporciones AWS credenciales a tu aplicación en tiempo de ejecución. En los ejemplos de código de esta guía se supone que se utiliza un archivo de AWS credenciales, tal y como se describe en la sección Configuración de AWS credenciales y región para el desarrollo de la Guía para AWS SDK for Java desarrolladores.

A continuación se muestra un ejemplo de un archivo de AWS credenciales denominado`~/.aws/credentials`, en el que el carácter de tilde (~) representa su directorio principal.

```
[default]
aws_access_key_id = AWS access key ID goes here
aws_secret_access_key = Secret key goes here
```

Java v2

Para empezar a utilizar Java 2.0 SDK y Amazon Timestream, complete los requisitos previos que se describen a continuación.

Una vez que haya completado los requisitos previos necesarios para la versión 2.0 de JavaSDK, puede empezar con. Ejemplos de código

Requisitos previos

Antes de empezar con Java, debe hacer lo siguiente:

1. Siga las instrucciones de AWS configuración que se indican enAcceder a Timestream para LiveAnalytics.
2. Puede configurarla AWS SDK como una dependencia de Maven como se describe en Uso de SDK con Apache Maven.
3. Configure un entorno de desarrollo Java descargando e instalando lo siguiente:
   - Kit de desarrollo Java SE 8 (como Amazon Corretto 8).
   - Java IDE (como Eclipse o IntelliJ).

Para obtener más información, consulte Cómo empezar con AWS SDK for Java

Con Apache Maven

Puede usar Apache Maven para configurar y crear AWS SDK for Java proyectos.

> **Note**
>
> Para usar Apache Maven, asegúrese de que su entorno de ejecución SDK y Java sean 1.8 o superiores.

Puede configurarla AWS SDK como una dependencia de Maven, tal y como se describe en Uso de SDK con Apache Maven. Los cambios necesarios en el archivo pom.xml se describen aquí.

Puede ejecutar la compilación y ejecutar el código fuente con el siguiente comando:

```
mvn clean compile
mvn exec:java -Dexec.mainClass=<your source code Main class>
```

> **Note**
>
> `<your source code Main class>`es la ruta a la clase principal del código fuente de Java.

Go

Para empezar a usar Go SDK y Amazon Timestream, complete los requisitos previos que se describen a continuación.

Una vez que haya completado los requisitos previos necesarios para el GoSDK, puede empezar con el. Ejemplos de código

Requisitos previos

1. Descarga el GO SDK 1.14.
2. Configura el GO. SDK

3. Construye tu cliente.

Python

Para empezar a usar Python SDK y Amazon Timestream, complete los requisitos previos que se describen a continuación.

Una vez que haya completado los requisitos previos necesarios para PythonSDK, puede empezar con. Ejemplos de código

Requisitos previos

Para usar Python, instale y configure Boto3 siguiendo las instrucciones aquí.

Node.js

Para empezar a utilizar Node.js SDK y Amazon Timestream, complete los requisitos previos que se describen a continuación.

Una vez que haya completado los requisitos previos necesarios para el archivo Node.jsSDK, puede empezar con. Ejemplos de código

Requisitos previos

Antes de empezar con Node.js, debe hacer lo siguiente:

1. Instale Node.js.
2. Instale el AWS SDK formulario JavaScript.

.NET

Para empezar con. NETSDKy Amazon Timestream, cumplan los requisitos previos que se describen a continuación.

Una vez que haya completado los requisitos previos necesarios para el. NETSDK, puede empezar con. Ejemplos de código

Requisitos previos

Antes de empezar con. NET, instale los NuGet paquetes necesarios y asegúrese de que la AWSSDK versión.Core sea la 3.3.107 o posterior ejecutando los siguientes comandos:

```
dotnet add package AWSSDK.Core
dotnet add package AWSSDK.TimestreamWrite
dotnet add package AWSSDK.TimestreamQuery
```

Introducción

Esta sección incluye un tutorial para empezar a utilizar Amazon Timestream Live Analytics, así como instrucciones para configurar una aplicación de muestra totalmente funcional. Puede comenzar con el tutorial o la aplicación de muestra seleccionando uno de los enlaces siguientes.

Temas

- Tutorial
- Aplicación de muestra

Tutorial

En este tutorial se muestra cómo crear una base de datos con conjuntos de datos de ejemplo y cómo ejecutar consultas de ejemplo. Los conjuntos de datos de muestra que se utilizan en este tutorial se ven con frecuencia en el IoT y en DevOps los escenarios. El conjunto de datos de IoT contiene datos de series temporales, como la velocidad, la ubicación y la carga de un camión, para agilizar la gestión de la flota e identificar oportunidades de optimización. El conjunto de DevOps datos contiene métricas de EC2 instanciaCPU, como la utilización de la red y la memoria, para mejorar el rendimiento y la disponibilidad de las aplicaciones. Este es un tutorial en vídeo con las instrucciones que se describen en esta sección.

Siga estos pasos para crear una base de datos con los conjuntos de datos de ejemplo y ejecutar consultas de ejemplo con la AWS consola:

Mediante la consola

Siga estos pasos para crear una base de datos con los conjuntos de datos de ejemplo y ejecutar consultas de ejemplo con la AWS consola:

1. Abre la AWS consola.
2. En el panel de navegación, elija Bases de datos
3. Haga clic en Crear base de datos.
4. En la página de creación de base de datos, introduzca lo siguiente:

- Elija la configuración: seleccione una base de datos de muestra.
- Nombre: introduzca el nombre de la base de datos que desee.

> **Note**
>
> Tras crear una base de datos con conjuntos de datos de ejemplo, para utilizar las consultas de ejemplo disponibles en la consola, puede ajustar el nombre de la base de datos al que se hace referencia en la consulta para que coincida con el nombre de la base de datos que introduzca aquí. Hay consultas de ejemplo para cada combinación de conjunto de datos de muestra y tipo de registros de series temporales.

- Elija conjuntos de datos de muestra: seleccione IoT y DevOps.
- Elija el tipo de registros de series temporales: seleccione registros de medidas múltiples.
- Haga clic en Crear base de datos para crear una base de datos que contenga dos tablas rellenas con datos de muestra. Los nombres de las tablas para los conjuntos de datos de muestra con registros de múltiples medidas son `DevOpsMulti` y `IoTMulti`. Los nombres de las tablas para los conjuntos de datos de muestra con registros de una sola medida son `DevOps` y. `IoT`

5. En el panel de navegación, elija el editor de consultas
6. Seleccione Consultas de muestra en el menú superior.
7. Haga clic en una de las consultas de ejemplo del conjunto de datos que eligió al crear la base de datos de ejemplo. Esto le llevará de vuelta al editor de consultas con el editor lleno de la consulta de ejemplo.
8. Ajusta el nombre de la base de datos para la consulta de muestra.
9. Haga clic en Ejecutar para ejecutar la consulta y ver los resultados de la consulta.

Utilización del SDKs

Timestream Live Analytics proporciona una aplicación de ejemplo totalmente funcional que le muestra cómo crear una base de datos y una tabla, rellenar la tabla con aproximadamente 126 000 filas de datos de ejemplo y ejecutar consultas de muestra. La aplicación de ejemplo está disponible en GitHubJava, Python, Node.js, Go y. NET.

1. Clone las aplicaciones de ejemplo de Timestream Live Analytics del GitHub repositorio siguiendo las instrucciones de. GitHub

2. Configure el AWS SDK para conectarse a Amazon Timestream Live Analytics siguiendo las instrucciones que se describen en. Uso del AWS SDKs
3. Compila y ejecuta la aplicación de muestra siguiendo las instrucciones que aparecen a continuación:

   - Instrucciones para la aplicación de ejemplo de Java.
   - Instrucciones para la aplicación de ejemplo Java v2.
   - Instrucciones para la aplicación de ejemplo Go.
   - Instrucciones para la aplicación de ejemplo de Python.
   - Instrucciones para la aplicación de ejemplo Node.js.
   - Instrucciones para el. NETejemplo de aplicación.

Aplicación de muestra

Timestream incluye una aplicación de ejemplo totalmente funcional que muestra cómo crear una base de datos y una tabla, rellenar la tabla con aproximadamente 126 000 filas de datos de muestra y ejecutar consultas de muestra. Siga los pasos que se indican a continuación para empezar a utilizar la aplicación de ejemplo en cualquiera de los idiomas compatibles:

Java

1. Clone el GitHub repositorio Timestream para obtener aplicaciones LiveAnalytics de muestra siguiendo las instrucciones de. GitHub
2. Configure la conexión AWS SDK a Timestream LiveAnalytics siguiendo las instrucciones descritas en Cómo empezar con. Java
3. Ejecute la aplicación de ejemplo de Java siguiendo las instrucciones que se describen aquí

Java v2

1. Clone el GitHub repositorio Timestream para obtener aplicaciones LiveAnalytics de muestra siguiendo las instrucciones de. GitHub
2. Configure la conexión AWS SDK a Amazon Timestream LiveAnalytics siguiendo las instrucciones que se describen en Cómo empezar con. Java v2
3. Ejecute la aplicación de ejemplo Java 2.0 siguiendo las instrucciones que se describen aquí

Go

1. Clone el GitHub repositorio Timestream para obtener aplicaciones LiveAnalytics de muestra siguiendo las instrucciones de. GitHub
2. Configure la conexión AWS SDK a Amazon Timestream LiveAnalytics siguiendo las instrucciones que se describen en Cómo empezar con. Go
3. Ejecute la aplicación de muestra Go siguiendo las instrucciones que se describen aquí

Python

1. Clone el GitHub repositorio Timestream para obtener aplicaciones LiveAnalytics de muestra siguiendo las instrucciones de. GitHub
2. Configure la conexión AWS SDK a Amazon Timestream siguiendo las LiveAnalytics instrucciones descritas en. Python
3. Ejecute la aplicación de ejemplo de Python siguiendo las instrucciones que se describen aquí

Node.js

1. Clone el GitHub repositorio Timestream para obtener aplicaciones LiveAnalytics de muestra siguiendo las instrucciones de. GitHub
2. Configure la conexión AWS SDK a Amazon Timestream LiveAnalytics siguiendo las instrucciones que se describen en Cómo empezar con. Node.js
3. Ejecute la aplicación de ejemplo Node.js siguiendo las instrucciones que se describen aquí

.NET

1. Clone el GitHub repositorio Timestream para obtener aplicaciones LiveAnalytics de ejemplo siguiendo las instrucciones de. GitHub
2. Configure la conexión AWS SDK a Amazon Timestream LiveAnalytics siguiendo las instrucciones que se describen en Cómo empezar con. .NET
3. Ejecute el. NETejemplo de aplicación siguiendo las instrucciones que se describen aquí

Ejemplos de código

Puede acceder a Amazon Timestream mediante. AWS SDKs Timestream admite dos SDKs por idioma; a saber, la escritura SDK y la consulta. SDK La función Write SDK se utiliza para realizar CRUD operaciones e insertar los datos de series temporales en Timestream. La consulta SDK se utiliza para consultar los datos de series temporales existentes almacenados en Timestream. Seleccione un tema de la lista siguiente para obtener más información, incluidos ejemplos de código para cada uno de los temas admitidos. SDKs

Temas

- Listar consulta programada
- Describa la consulta programada
- Ejecutar consulta programada
- Actualizar consulta programada
- Eliminar consulta programada

Escribe un SDK cliente

Puede usar los siguientes fragmentos de código para crear un cliente Timestream para Write. SDK La escritura SDK se utiliza para realizar CRUD operaciones e insertar los datos de series temporales en Timestream.

> **Note**
>
> Estos fragmentos de código se basan en aplicaciones de muestra completas en. GitHub Para obtener más información sobre cómo empezar a utilizar las aplicaciones de ejemplo, consulte. Aplicación de muestra

Java

```
private static AmazonTimestreamWrite buildWriteClient() {
    final ClientConfiguration clientConfiguration = new ClientConfiguration()
            .withMaxConnections(5000)
            .withRequestTimeout(20 * 1000)
            .withMaxErrorRetry(10);

    return AmazonTimestreamWriteClientBuilder
            .standard()
            .withRegion("us-east-1")
            .withClientConfiguration(clientConfiguration)
            .build();
}
```

Java v2

```
private static TimestreamWriteClient buildWriteClient() {
    ApacheHttpClient.Builder httpClientBuilder =
```

```
                ApacheHttpClient.builder();
        httpClientBuilder.maxConnections(5000);

        RetryPolicy.Builder retryPolicy =
                RetryPolicy.builder();
        retryPolicy.numRetries(10);

        ClientOverrideConfiguration.Builder overrideConfig =
                ClientOverrideConfiguration.builder();
        overrideConfig.apiCallAttemptTimeout(Duration.ofSeconds(20));
        overrideConfig.retryPolicy(retryPolicy.build());

        return TimestreamWriteClient.builder()
                .httpClientBuilder(httpClientBuilder)
                .overrideConfiguration(overrideConfig.build())
                .region(Region.US_EAST_1)
                .build();
    }
```

Go

```
tr := &http.Transport{
        ResponseHeaderTimeout: 20 * time.Second,
        // Using DefaultTransport values for other parameters: https://golang.org/
pkg/net/http/#RoundTripper
        Proxy: http.ProxyFromEnvironment,
        DialContext: (&net.Dialer{
            KeepAlive: 30 * time.Second,
            DualStack: true,
            Timeout:   30 * time.Second,
        }).DialContext,
        MaxIdleConns:          100,
        IdleConnTimeout:       90 * time.Second,
        TLSHandshakeTimeout:   10 * time.Second,
        ExpectContinueTimeout: 1 * time.Second,
    }

    // So client makes HTTP/2 requests
    http2.ConfigureTransport(tr)

    sess, err := session.NewSession(&aws.Config{ Region: aws.String("us-east-1"),
 MaxRetries: aws.Int(10), HTTPClient: &http.Client{ Transport: tr }})
    writeSvc := timestreamwrite.New(sess)
```

Python

```
write_client = session.client('timestream-write', config=Config(read_timeout=20,
 max_pool_connections = 5000, retries={'max_attempts': 10}))
```

Node.js

El siguiente fragmento se utiliza AWS SDK para JavaScript la versión 3. Para obtener más información sobre cómo instalar el cliente y su uso, consulte Timestream Write Client: for v3. AWS SDK JavaScript

Aquí se muestra una importación de comandos adicional. La `CreateDatabaseCommand` importación no es necesaria para crear el cliente.

```
import { TimestreamWriteClient, CreateDatabaseCommand } from "@aws-sdk/client-
timestream-write";
const writeClient = new TimestreamWriteClient({ region: "us-east-1" });
```

En el siguiente fragmento se utiliza el estilo AWS SDK for JavaScript V2. Se basa en la aplicación de ejemplo de Node.js, ejemplo de Amazon Timestream LiveAnalytics para su aplicación en. GitHub

```
var https = require('https');
var agent = new https.Agent({
    maxSockets: 5000
});
writeClient = new AWS.TimestreamWrite({
        maxRetries: 10,
        httpOptions: {
            timeout: 20000,
            agent: agent
        }
    });
```

.NET

```
var writeClientConfig = new AmazonTimestreamWriteConfig
{
    RegionEndpoint = RegionEndpoint.USEast1,
    Timeout = TimeSpan.FromSeconds(20),
```

```
    MaxErrorRetry = 10
};

var writeClient = new AmazonTimestreamWriteClient(writeClientConfig);
```

Le recomendamos que utilice la siguiente configuración.

- Establezca el recuento SDK de reintentos en`10`.
- Utilice `SDK DEFAULT_BACKOFF_STRATEGY`.
- `RequestTimeout`Establézcalo en `20` segundos.
- Establezca el número máximo de conexiones en `5000` o más.

SDKCliente de consultas

Puede usar los siguientes fragmentos de código para crear un cliente Timestream para la consulta. SDK La consulta SDK se utiliza para consultar los datos de series temporales existentes almacenados en Timestream.

> **Note**
>
> Estos fragmentos de código se basan en aplicaciones de muestra completas en. GitHub Para obtener más información sobre cómo empezar a utilizar las aplicaciones de ejemplo, consulte. Aplicación de muestra

Java

```
    private static AmazonTimestreamQuery buildQueryClient() {
        AmazonTimestreamQuery client =
AmazonTimestreamQueryClient.builder().withRegion("us-east-1").build();
        return client;
    }
```

Java v2

```
    private static TimestreamQueryClient buildQueryClient() {
        return TimestreamQueryClient.builder()
```

```
                .region(Region.US_EAST_1)
                .build();
    }
```

Go

```
sess, err := session.NewSession(&aws.Config{Region: aws.String("us-east-1")})
```

Python

```
query_client = session.client('timestream-query')
```

Node.js

El siguiente fragmento se utiliza AWS SDK para JavaScript la versión 3. Para obtener más información sobre cómo instalar el cliente y su uso, consulte Timestream Query Client -, para la versión 3.AWS SDK JavaScript

Aquí se muestra una importación de comandos adicional. La `QueryCommand` importación no es necesaria para crear el cliente.

```
import { TimestreamQueryClient, QueryCommand } from "@aws-sdk/client-timestream-
query";
const queryClient = new TimestreamQueryClient({ region: "us-east-1" });
```

En el siguiente fragmento se utiliza el estilo AWS SDK for JavaScript V2. Se basa en la aplicación de ejemplo de Node.js, ejemplo de Amazon Timestream LiveAnalytics para su aplicación en. GitHub

```
queryClient = new AWS.TimestreamQuery();
```

.NET

```
var queryClientConfig = new AmazonTimestreamQueryConfig
{
    RegionEndpoint = RegionEndpoint.USEast1
};

var queryClient = new AmazonTimestreamQueryClient(queryClientConfig);
```

Crear base de datos

Puede utilizar los siguientes fragmentos de código para crear una base de datos.

> **Note**
>
> Estos fragmentos de código se basan en aplicaciones de muestra completas en. GitHub Para obtener más información sobre cómo empezar a utilizar las aplicaciones de ejemplo, consulte. Aplicación de muestra

Java

```
  public void createDatabase() {
       System.out.println("Creating database");
       CreateDatabaseRequest request = new CreateDatabaseRequest();
       request.setDatabaseName(DATABASE_NAME);
       try {
           amazonTimestreamWrite.createDatabase(request);
           System.out.println("Database [" + DATABASE_NAME + "] created
successfully");
       } catch (ConflictException e) {
           System.out.println("Database [" + DATABASE_NAME + "] exists. Skipping
database creation");
       }
   }
```

Java v2

```
   public void createDatabase() {
       System.out.println("Creating database");
       CreateDatabaseRequest request =
CreateDatabaseRequest.builder().databaseName(DATABASE_NAME).build();
       try {
           timestreamWriteClient.createDatabase(request);
           System.out.println("Database [" + DATABASE_NAME + "] created
successfully");
       } catch (ConflictException e) {
           System.out.println("Database [" + DATABASE_NAME + "] exists. Skipping
database creation");
       }
```

```
    }
```

Go

```
// Create database.
    createDatabaseInput := &timestreamwrite.CreateDatabaseInput{
        DatabaseName: aws.String(*databaseName),
    }

    _, err = writeSvc.CreateDatabase(createDatabaseInput)

    if err != nil {
        fmt.Println("Error:")
        fmt.Println(err)
    } else {
        fmt.Println("Database successfully created")
    }

    fmt.Println("Describing the database, hit enter to continue")
```

Python

```
    def create_database(self):
        print("Creating Database")
        try:
            self.client.create_database(DatabaseName=Constant.DATABASE_NAME)
            print("Database [%s] created successfully." % Constant.DATABASE_NAME)
        except self.client.exceptions.ConflictException:
            print("Database [%s] exists. Skipping database creation" %
 Constant.DATABASE_NAME)
        except Exception as err:
            print("Create database failed:", err)
```

Node.js

El siguiente fragmento se utiliza AWS SDK para JavaScript la versión 3. Para obtener más información sobre cómo instalar el cliente y su uso, consulte Timestream Write Client, para la versión 3. AWS SDK JavaScript

Consulte también Class y. CreateDatabaseCommand CreateDatabase

```
import { TimestreamWriteClient, CreateDatabaseCommand } from "@aws-sdk/client-
timestream-write";
```

```
const writeClient = new TimestreamWriteClient({ region: "us-east-1" });

const params = {
    DatabaseName: "testDbFromNode"
};

const command = new CreateDatabaseCommand(params);

try {
    const data = await writeClient.send(command);
    console.log(`Database ${data.Database.DatabaseName} created successfully`);
} catch (error) {
    if (error.code === 'ConflictException') {
        console.log(`Database ${params.DatabaseName} already exists. Skipping
 creation.`);
    } else {
        console.log("Error creating database", error);
    }
}
```

El siguiente fragmento usa el estilo AWS SDK for JavaScript V2. Se basa en la aplicación de ejemplo de Node.js, ejemplo de Amazon Timestream LiveAnalytics para su aplicación en. GitHub

```
async function createDatabase() {
    console.log("Creating Database");
    const params = {
        DatabaseName: constants.DATABASE_NAME
    };

    const promise = writeClient.createDatabase(params).promise();

    await promise.then(
        (data) => {
            console.log(`Database ${data.Database.DatabaseName} created
 successfully`);
        },
        (err) => {
            if (err.code === 'ConflictException') {
                console.log(`Database ${params.DatabaseName} already exists.
 Skipping creation.`);
            } else {
                console.log("Error creating database", err);
            }
```

```
        }
    );
}
```

.NET

```
        public async Task CreateDatabase()
        {
            Console.WriteLine("Creating Database");

            try
            {
                var createDatabaseRequest = new CreateDatabaseRequest
                {
                    DatabaseName = Constants.DATABASE_NAME
                };
                CreateDatabaseResponse response = await
 writeClient.CreateDatabaseAsync(createDatabaseRequest);
                Console.WriteLine($"Database {Constants.DATABASE_NAME} created");
            }
            catch (ConflictException)
            {
                Console.WriteLine("Database already exists.");
            }
            catch (Exception e)
            {
                Console.WriteLine("Create database failed:" + e.ToString());
            }

        }
```

Describir base de datos

Puede usar los siguientes fragmentos de código para obtener información sobre los atributos de la base de datos recién creada.

> **Note**
>
> Estos fragmentos de código se basan en aplicaciones de muestra completas en. GitHub Para obtener más información sobre cómo empezar a utilizar las aplicaciones de ejemplo, consulte. Aplicación de muestra

Java

```
   public void describeDatabase() {
       System.out.println("Describing database");
       final DescribeDatabaseRequest describeDatabaseRequest = new
DescribeDatabaseRequest();
       describeDatabaseRequest.setDatabaseName(DATABASE_NAME);
       try {
           DescribeDatabaseResult result =
amazonTimestreamWrite.describeDatabase(describeDatabaseRequest);
           final Database databaseRecord = result.getDatabase();
           final String databaseId = databaseRecord.getArn();
           System.out.println("Database " + DATABASE_NAME + " has id " +
databaseId);
       } catch (final Exception e) {
           System.out.println("Database doesn't exist = " + e);
           throw e;
       }
   }
```

Java v2

```
   public void describeDatabase() {
       System.out.println("Describing database");
       final DescribeDatabaseRequest describeDatabaseRequest =
DescribeDatabaseRequest.builder()
               .databaseName(DATABASE_NAME).build();
       try {
           DescribeDatabaseResponse response =
timestreamWriteClient.describeDatabase(describeDatabaseRequest);
           final Database databaseRecord = response.database();
           final String databaseId = databaseRecord.arn();
           System.out.println("Database " + DATABASE_NAME + " has id " +
databaseId);
       } catch (final Exception e) {
```

```
            System.out.println("Database doesn't exist = " + e);
            throw e;
        }
    }
```

Go

```
describeDatabaseOutput, err := writeSvc.DescribeDatabase(describeDatabaseInput)

    if err != nil {
        fmt.Println("Error:")
        fmt.Println(err)
    } else {
        fmt.Println("Describe database is successful, below is the output:")
        fmt.Println(describeDatabaseOutput)
    }
```

Python

```
    def describe_database(self):
        print("Describing database")
        try:
            result =
 self.client.describe_database(DatabaseName=Constant.DATABASE_NAME)
            print("Database [%s] has id [%s]" % (Constant.DATABASE_NAME,
 result['Database']['Arn']))
        except self.client.exceptions.ResourceNotFoundException:
            print("Database doesn't exist")
        except Exception as err:
            print("Describe database failed:", err)
```

Node.js

El siguiente fragmento se utiliza AWS SDK para JavaScript la versión 3. Para obtener más información sobre cómo instalar el cliente y su uso, consulte Timestream Write Client: for v3. AWS SDK JavaScript

Consulte también Class y. DescribeDatabaseCommand DescribeDatabase

```
import { TimestreamWriteClient, DescribeDatabaseCommand } from "@aws-sdk/client-
timestream-write";
const writeClient = new TimestreamWriteClient({ region: "us-east-1" });
```

```
const params = {
    DatabaseName: "testDbFromNode"
};

const command = new DescribeDatabaseCommand(params);

try {
    const data = await writeClient.send(command);
    console.log(`Database ${data.Database.DatabaseName} has id
 ${data.Database.Arn}`);
} catch (error) {
    if (error.code === 'ResourceNotFoundException') {
        console.log("Database doesn't exist.");
    } else {
        console.log("Describe database failed.", error);
        throw error;
    }
}
```

El siguiente fragmento usa el estilo AWS SDK for JavaScript V2. Se basa en la aplicación de ejemplo de Node.js, ejemplo de Amazon Timestream LiveAnalytics para su aplicación en. GitHub

```
async function describeDatabase () {
    console.log("Describing Database");
    const params = {
        DatabaseName: constants.DATABASE_NAME
    };

    const promise = writeClient.describeDatabase(params).promise();

    await promise.then(
        (data) => {
            console.log(`Database ${data.Database.DatabaseName} has id
 ${data.Database.Arn}`);
        },
        (err) => {
            if (err.code === 'ResourceNotFoundException') {
                console.log("Database doesn't exist.");
            } else {
                console.log("Describe database failed.", err);
                throw err;
            }
```

```
        }
    );
}
```

.NET

```
        public async Task DescribeDatabase()
        {
            Console.WriteLine("Describing Database");

            try
            {
                var describeDatabaseRequest = new DescribeDatabaseRequest
                {
                    DatabaseName = Constants.DATABASE_NAME
                };
                DescribeDatabaseResponse response = await
 writeClient.DescribeDatabaseAsync(describeDatabaseRequest);
                Console.WriteLine($"Database {Constants.DATABASE_NAME} has id:
{response.Database.Arn}");
            }
            catch (ResourceNotFoundException)
            {
                Console.WriteLine("Database does not exist.");
            }
            catch (Exception e)
            {
                Console.WriteLine("Describe database failed:" + e.ToString());
            }

        }
```

Actualizar base de datos

Puede usar los siguientes fragmentos de código para actualizar sus bases de datos.

> **Note**
>
> Estos fragmentos de código se basan en aplicaciones de muestra completas en. GitHub Para obtener más información sobre cómo empezar a utilizar las aplicaciones de ejemplo, consulte. Aplicación de muestra

Java

```
   public void updateDatabase(String kmsId) {
       System.out.println("Updating kmsId to " + kmsId);
       UpdateDatabaseRequest request = new UpdateDatabaseRequest();
       request.setDatabaseName(DATABASE_NAME);
       request.setKmsKeyId(kmsId);
       try {
           UpdateDatabaseResult result =
amazonTimestreamWrite.updateDatabase(request);
           System.out.println("Update Database complete");
       } catch (final ValidationException e) {
           System.out.println("Update database failed:");
           e.printStackTrace();
       } catch (final ResourceNotFoundException e) {
           System.out.println("Database " + DATABASE_NAME + " doesn't exist = " +
e);
       } catch (final Exception e) {
           System.out.println("Could not update Database " + DATABASE_NAME + " = "
+ e);
           throw e;
       }
   }
```

Java v2

```
   public void updateDatabase(String kmsKeyId) {

       if (kmsKeyId == null) {
           System.out.println("Skipping UpdateDatabase because KmsKeyId was not
given");
           return;
       }

       System.out.println("Updating database");
```

```
        UpdateDatabaseRequest request = UpdateDatabaseRequest.builder()
                .databaseName(DATABASE_NAME)
                .kmsKeyId(kmsKeyId)
                .build();
        try {
            timestreamWriteClient.updateDatabase(request);
            System.out.println("Database [" + DATABASE_NAME + "] updated
 successfully with kmsKeyId " + kmsKeyId);
        } catch (ResourceNotFoundException e) {
            System.out.println("Database [" + DATABASE_NAME + "] does not exist.
 Skipping UpdateDatabase");
        } catch (Exception e) {
            System.out.println("UpdateDatabase failed: " + e);
        }
    }
```

Go

```
// Update Database.
        updateDatabaseInput := &timestreamwrite.UpdateDatabaseInput {
            DatabaseName: aws.String(*databaseName),
            KmsKeyId: aws.String(*kmsKeyId),
        }

        updateDatabaseOutput, err := writeSvc.UpdateDatabase(updateDatabaseInput)

        if err != nil {
            fmt.Println("Error:")
            fmt.Println(err)
        } else {
            fmt.Println("Update database is successful, below is the output:")
            fmt.Println(updateDatabaseOutput)
        }
```

Python

```
    def update_database(self, kms_id):
        print("Updating database")
        try:
            result =
 self.client.update_database(DatabaseName=Constant.DATABASE_NAME, KmsKeyId=kms_id)
```

```
            print("Database [%s] was updated to use kms [%s] successfully" %
 (Constant.DATABASE_NAME,

 result['Database']['KmsKeyId']))
        except self.client.exceptions.ResourceNotFoundException:
            print("Database doesn't exist")
        except Exception as err:
            print("Update database failed:", err)
```

Node.js

El siguiente fragmento se utiliza AWS SDK para JavaScript la versión 3. Para obtener más información sobre cómo instalar el cliente y su uso, consulte Timestream Write Client, para la versión 3. AWS SDK JavaScript

Consulte también Class y. UpdateDatabaseCommand UpdateDatabase

```
import { TimestreamWriteClient, UpdateDatabaseCommand } from "@aws-sdk/client-
timestream-write";
const writeClient = new TimestreamWriteClient({ region: "us-east-1" });
let updatedKmsKeyId = "<updatedKmsKeyId>";

const params = {
    DatabaseName: "testDbFromNode",
    KmsKeyId: updatedKmsKeyId
};

const command = new UpdateDatabaseCommand(params);

try {
    const data = await writeClient.send(command);
    console.log(`Database ${data.Database.DatabaseName} updated kmsKeyId to
 ${updatedKmsKeyId}`);
} catch (error) {
    if (error.code === 'ResourceNotFoundException') {
        console.log("Database doesn't exist.");
    } else {
        console.log("Update database failed.", error);
    }
}
```

El siguiente fragmento usa el estilo AWS SDK for JavaScript V2. Se basa en la aplicación de ejemplo de Node.js, ejemplo de Amazon Timestream LiveAnalytics para su aplicación en. GitHub

```
async function updateDatabase(updatedKmsKeyId) {

    if (updatedKmsKeyId === undefined) {
        console.log("Skipping UpdateDatabase; KmsKeyId was not given");
        return;
    }
    console.log("Updating Database");
    const params = {
        DatabaseName: constants.DATABASE_NAME,
        KmsKeyId: updatedKmsKeyId
    }

    const promise = writeClient.updateDatabase(params).promise();

    await promise.then(
        (data) => {
            console.log(`Database ${data.Database.DatabaseName} updated kmsKeyId to
 ${updatedKmsKeyId}`);
        },
        (err) => {
            if (err.code === 'ResourceNotFoundException') {
                console.log("Database doesn't exist.");
            } else {
                console.log("Update database failed.", err);
            }
        }
    );
}
```

.NET

```
        public async Task UpdateDatabase(String updatedKmsKeyId)
        {
            Console.WriteLine("Updating Database");

            try
            {
                var updateDatabaseRequest = new UpdateDatabaseRequest
                {
                    DatabaseName = Constants.DATABASE_NAME,
                    KmsKeyId = updatedKmsKeyId
                };
```

```
                UpdateDatabaseResponse response = await
writeClient.UpdateDatabaseAsync(updateDatabaseRequest);
                Console.WriteLine($"Database {Constants.DATABASE_NAME} updated with
KmsKeyId {updatedKmsKeyId}");
            }
            catch (ResourceNotFoundException)
            {
                Console.WriteLine("Database does not exist.");
            }
            catch (Exception e)
            {
                Console.WriteLine("Update database failed: " + e.ToString());
            }

        }

        private void PrintDatabases(List<Database> databases)
        {
            foreach (Database database in databases)
                Console.WriteLine($"Database:{database.DatabaseName}");
        }
```

Eliminar base de datos

Puede usar el siguiente fragmento de código para eliminar una base de datos.

> **Note**
>
> Estos fragmentos de código se basan en aplicaciones de muestra completas en. GitHub Para obtener más información sobre cómo empezar a utilizar las aplicaciones de ejemplo, consulte. Aplicación de muestra

Java

```
   public void deleteDatabase() {
       System.out.println("Deleting database");
       final DeleteDatabaseRequest deleteDatabaseRequest = new
DeleteDatabaseRequest();
       deleteDatabaseRequest.setDatabaseName(DATABASE_NAME);
       try {
```

```
            DeleteDatabaseResult result =
                    amazonTimestreamWrite.deleteDatabase(deleteDatabaseRequest);
            System.out.println("Delete database status: " +
 result.getSdkHttpMetadata().getHttpStatusCode());
        } catch (final ResourceNotFoundException e) {
            System.out.println("Database " + DATABASE_NAME + " doesn't exist = " +
 e);
            throw e;
        } catch (final Exception e) {
            System.out.println("Could not delete Database " + DATABASE_NAME + " = "
 + e);
            throw e;
        }
    }
```

Java v2

```
    public void deleteDatabase() {
        System.out.println("Deleting database");
        final DeleteDatabaseRequest deleteDatabaseRequest = new
 DeleteDatabaseRequest();
        deleteDatabaseRequest.setDatabaseName(DATABASE_NAME);
        try {
            DeleteDatabaseResult result =
                    amazonTimestreamWrite.deleteDatabase(deleteDatabaseRequest);
            System.out.println("Delete database status: " +
 result.getSdkHttpMetadata().getHttpStatusCode());
        } catch (final ResourceNotFoundException e) {
            System.out.println("Database " + DATABASE_NAME + " doesn't exist = " +
 e);
            throw e;
        } catch (final Exception e) {
            System.out.println("Could not delete Database " + DATABASE_NAME + " = "
 + e);
            throw e;
        }
    }
```

Go

```
deleteDatabaseInput := &timestreamwrite.DeleteDatabaseInput{
        DatabaseName:   aws.String(*databaseName),
    }
```

```
    _, err = writeSvc.DeleteDatabase(deleteDatabaseInput)

    if err != nil {
        fmt.Println("Error:")
        fmt.Println(err)
    } else {
        fmt.Println("Database deleted:", *databaseName)
    }
```

Python

```
    def delete_database(self):
        print("Deleting Database")
        try:
            result =
 self.client.delete_database(DatabaseName=Constant.DATABASE_NAME)
            print("Delete database status [%s]" % result['ResponseMetadata']
['HTTPStatusCode'])
        except self.client.exceptions.ResourceNotFoundException:
            print("database [%s] doesn't exist" % Constant.DATABASE_NAME)
        except Exception as err:
            print("Delete database failed:", err)
```

Node.js

El siguiente fragmento se utiliza AWS SDK para JavaScript la versión 3. Para obtener más información sobre cómo instalar el cliente y su uso, consulte Timestream Write Client, para la versión 3. AWS SDK JavaScript

Consulte también Class y. DeleteDatabaseCommand DeleteDatabase

```
import { TimestreamWriteClient, DeleteDatabaseCommand } from "@aws-sdk/client-
timestream-write";
const writeClient = new TimestreamWriteClient({ region: "us-east-1" });

const params = {
    DatabaseName: "testDbFromNode"
};

const command = new DeleteDatabaseCommand(params);

try {
```

```
    const data = await writeClient.send(command);
    console.log("Deleted database");
} catch (error) {
    if (error.code === 'ResourceNotFoundException') {
        console.log(`Database ${params.DatabaseName} doesn't exists.`);
    } else {
        console.log("Delete database failed.", error);
        throw error;
    }
}
```

El siguiente fragmento usa el estilo AWS SDK for JavaScript V2. Se basa en la aplicación de ejemplo de Node.js, ejemplo de Amazon Timestream LiveAnalytics para su aplicación en. GitHub

```
async function deleteDatabase() {
    console.log("Deleting Database");
    const params = {
        DatabaseName: constants.DATABASE_NAME
    };

    const promise = writeClient.deleteDatabase(params).promise();

    await promise.then(
        function (data) {
            console.log("Deleted database");
         },
        function(err) {
            if (err.code === 'ResourceNotFoundException') {
                console.log(`Database ${params.DatabaseName} doesn't exists.`);
            } else {
                console.log("Delete database failed.", err);
                throw err;
            }
        }
    );
}
```

.NET

```
        public async Task DeleteDatabase()
        {
            Console.WriteLine("Deleting database");
            try
```

```
            {
                var deleteDatabaseRequest = new DeleteDatabaseRequest
                {
                    DatabaseName = Constants.DATABASE_NAME
                };
                DeleteDatabaseResponse response = await
writeClient.DeleteDatabaseAsync(deleteDatabaseRequest);
                Console.WriteLine($"Database {Constants.DATABASE_NAME} delete
request status:{response.HttpStatusCode}");
            }
            catch (ResourceNotFoundException)
            {
                Console.WriteLine($"Database {Constants.DATABASE_NAME} does not
exists");
            }
            catch (Exception e)
            {
                Console.WriteLine("Exception while deleting database:" +
e.ToString());
            }
        }
```

Enumeración de bases de datos

Puede usar los siguientes fragmentos de código para enumerar sus bases de datos.

> **Note**
>
> Estos fragmentos de código se basan en aplicaciones de muestra completas en. GitHub Para obtener más información sobre cómo empezar a utilizar las aplicaciones de ejemplo, consulte. Aplicación de muestra

Java

```
    public void listDatabases() {
        System.out.println("Listing databases");
        ListDatabasesRequest request = new ListDatabasesRequest();
        ListDatabasesResult result = amazonTimestreamWrite.listDatabases(request);
        final List<Database> databases = result.getDatabases();
        printDatabases(databases);
```

```
        String nextToken = result.getNextToken();
        while (nextToken != null && !nextToken.isEmpty()) {
            request.setNextToken(nextToken);
            ListDatabasesResult nextResult =
 amazonTimestreamWrite.listDatabases(request);
            final List<Database> nextDatabases = nextResult.getDatabases();
            printDatabases(nextDatabases);
            nextToken = nextResult.getNextToken();
        }
    }

    private void printDatabases(List<Database> databases) {
        for (Database db : databases) {
            System.out.println(db.getDatabaseName());
        }
    }
```

Java v2

```
    public void listDatabases() {
        System.out.println("Listing databases");
        ListDatabasesRequest request =
 ListDatabasesRequest.builder().maxResults(2).build();
        ListDatabasesIterable listDatabasesIterable =
 timestreamWriteClient.listDatabasesPaginator(request);
        for(ListDatabasesResponse listDatabasesResponse : listDatabasesIterable) {
            final List<Database> databases = listDatabasesResponse.databases();
            databases.forEach(database ->
 System.out.println(database.databaseName()));
        }
    }
```

Go

```
// List databases.
    listDatabasesMaxResult := int64(15)

    listDatabasesInput := &timestreamwrite.ListDatabasesInput{
        MaxResults: &listDatabasesMaxResult,
    }

    listDatabasesOutput, err := writeSvc.ListDatabases(listDatabasesInput)
```

```
    if err != nil {
        fmt.Println("Error:")
        fmt.Println(err)
    } else {
        fmt.Println("List databases is successful, below is the output:")
        fmt.Println(listDatabasesOutput)
    }
```

Python

```
    def list_databases(self):
        print("Listing databases")
        try:
            result = self.client.list_databases(MaxResults=5)
            self._print_databases(result['Databases'])
            next_token = result.get('NextToken', None)
            while next_token:
                result = self.client.list_databases(NextToken=next_token,
 MaxResults=5)
                self._print_databases(result['Databases'])
                next_token = result.get('NextToken', None)
        except Exception as err:
            print("List databases failed:", err)
```

Node.js

El siguiente fragmento se utiliza AWS SDK para JavaScript la versión 3. Para obtener más información sobre cómo instalar el cliente y su uso, consulte Timestream Write Client, para la versión 3. AWS SDK JavaScript

Consulte también Class y. ListDatabasesCommand ListDatabases

```
import { TimestreamWriteClient, ListDatabasesCommand } from "@aws-sdk/client-
timestream-write";
const writeClient = new TimestreamWriteClient({ region: "us-east-1" });

const params = {
    MaxResults: 15
};

const command = new ListDatabasesCommand(params);
```

```
getDatabasesList(null);

async function getDatabasesList(nextToken) {
    if (nextToken) {
        params.NextToken = nextToken;
    }

    try {
        const data = await writeClient.send(command);

        data.Databases.forEach(function (database) {
            console.log(database.DatabaseName);
        });

        if (data.NextToken) {
            return getDatabasesList(data.NextToken);
        }
    } catch (error) {
        console.log("Error while listing databases", error);
    }
}
```

El siguiente fragmento usa el estilo AWS SDK for JavaScript V2. Se basa en la aplicación de ejemplo de Node.js, ejemplo de Amazon Timestream LiveAnalytics para su aplicación en. GitHub

```
async function listDatabases() {
    console.log("Listing databases:");
    const databases = await getDatabasesList(null);
    databases.forEach(function(database){
        console.log(database.DatabaseName);
    });
}

function getDatabasesList(nextToken, databases = []) {
    var params = {
        MaxResults: 15
    };

    if(nextToken) {
        params.NextToken = nextToken;
    }

    return writeClient.listDatabases(params).promise()
```

```
        .then(
            (data) => {
                databases.push.apply(databases, data.Databases);
                if (data.NextToken) {
                    return getDatabasesList(data.NextToken, databases);
                } else {
                    return databases;
                }
            },
            (err) => {
                console.log("Error while listing databases", err);
            });
}
```

.NET

```
        public async Task ListDatabases()
        {
            Console.WriteLine("Listing Databases");

            try
            {
                var listDatabasesRequest = new ListDatabasesRequest
                {
                    MaxResults = 5
                };
                ListDatabasesResponse response = await
 writeClient.ListDatabasesAsync(listDatabasesRequest);
                PrintDatabases(response.Databases);
                var nextToken = response.NextToken;
                while (nextToken != null)
                {
                    listDatabasesRequest.NextToken = nextToken;
                    response = await
 writeClient.ListDatabasesAsync(listDatabasesRequest);
                    PrintDatabases(response.Databases);
                    nextToken = response.NextToken;
                }
            }
            catch (Exception e)
            {
                Console.WriteLine("List database failed:" + e.ToString());
            }
```

```
        }
```

Crear tablas

Temas

- El almacén de memoria escribe
- Magnetic Store escribe

El almacén de memoria escribe

Puede usar el siguiente fragmento de código para crear una tabla que tenga deshabilitada la escritura en el almacén magnético, por lo que solo podrá escribir datos en la ventana de retención del almacén de memoria.

> **Note**
>
> Estos fragmentos de código se basan en aplicaciones de muestra completas en. GitHub Para obtener más información sobre cómo empezar a utilizar las aplicaciones de ejemplo, consulte. Aplicación de muestra

Java

```
    public void createTable() {
        System.out.println("Creating table");
        CreateTableRequest createTableRequest = new CreateTableRequest();
        createTableRequest.setDatabaseName(DATABASE_NAME);
        createTableRequest.setTableName(TABLE_NAME);
        final RetentionProperties retentionProperties = new RetentionProperties()
                .withMemoryStoreRetentionPeriodInHours(HT_TTL_HOURS)
                .withMagneticStoreRetentionPeriodInDays(CT_TTL_DAYS);
        createTableRequest.setRetentionProperties(retentionProperties);

        try {
            amazonTimestreamWrite.createTable(createTableRequest);
            System.out.println("Table [" + TABLE_NAME + "] successfully created.");
        } catch (ConflictException e) {
```

```
            System.out.println("Table [" + TABLE_NAME + "] exists on database [" +
 DATABASE_NAME + "] . Skipping database creation");
        }
    }
```

Java v2

```
    public void createTable() {
        System.out.println("Creating table");

        final RetentionProperties retentionProperties =
 RetentionProperties.builder()
                .memoryStoreRetentionPeriodInHours(HT_TTL_HOURS)
                .magneticStoreRetentionPeriodInDays(CT_TTL_DAYS).build();
        final CreateTableRequest createTableRequest = CreateTableRequest.builder()

 .databaseName(DATABASE_NAME).tableName(TABLE_NAME).retentionProperties(retentionProperties)

        try {
            timestreamWriteClient.createTable(createTableRequest);
            System.out.println("Table [" + TABLE_NAME + "] successfully created.");
        } catch (ConflictException e) {
            System.out.println("Table [" + TABLE_NAME + "] exists on database [" +
 DATABASE_NAME + "] . Skipping database creation");
        }
    }
```

Go

```
// Create table.
    createTableInput := &timestreamwrite.CreateTableInput{
        DatabaseName: aws.String(*databaseName),
        TableName:    aws.String(*tableName),
    }
    _, err = writeSvc.CreateTable(createTableInput)

    if err != nil {
        fmt.Println("Error:")
        fmt.Println(err)
    } else {
        fmt.Println("Create table is successful")
    }
```

Python

```
    def create_table(self):
        print("Creating table")
        retention_properties = {
            'MemoryStoreRetentionPeriodInHours': Constant.HT_TTL_HOURS,
            'MagneticStoreRetentionPeriodInDays': Constant.CT_TTL_DAYS
        }
        try:
            self.client.create_table(DatabaseName=Constant.DATABASE_NAME,
 TableName=Constant.TABLE_NAME,
                                     RetentionProperties=retention_properties)
            print("Table [%s] successfully created." % Constant.TABLE_NAME)
        except self.client.exceptions.ConflictException:
            print("Table [%s] exists on database [%s]. Skipping table creation" % (
                Constant.TABLE_NAME, Constant.DATABASE_NAME))
        except Exception as err:
            print("Create table failed:", err)
```

Node.js

El siguiente fragmento se utiliza AWS SDK para JavaScript la versión 3. Para obtener más información sobre cómo instalar el cliente y su uso, consulte Timestream Write Client: for v3. AWS SDK JavaScript

Consulte también Class y. CreateTableCommand CreateTable

```
import { TimestreamWriteClient, CreateTableCommand } from "@aws-sdk/client-
timestream-write";
const writeClient = new TimestreamWriteClient({ region: "us-east-1" });

const params = {
    DatabaseName: "testDbFromNode",
    TableName: "testTableFromNode",
    RetentionProperties: {
        MemoryStoreRetentionPeriodInHours: 24,
        MagneticStoreRetentionPeriodInDays: 365
    }
};

const command = new CreateTableCommand(params);

try {
```

```
    const data = await writeClient.send(command);
    console.log(`Table ${data.Table.TableName} created successfully`);
} catch (error) {
    if (error.code === 'ConflictException') {
        console.log(`Table ${params.TableName} already exists on db
 ${params.DatabaseName}. Skipping creation.`);
    } else {
        console.log("Error creating table. ", error);
        throw error;
    }
}
```

El siguiente fragmento usa el estilo AWS SDK for JavaScript V2. Se basa en la aplicación de ejemplo de Node.js, ejemplo de Amazon Timestream LiveAnalytics para su aplicación en. GitHub

```
async function createTable() {
    console.log("Creating Table");
    const params = {
        DatabaseName: constants.DATABASE_NAME,
        TableName: constants.TABLE_NAME,
        RetentionProperties: {
            MemoryStoreRetentionPeriodInHours: constants.HT_TTL_HOURS,
            MagneticStoreRetentionPeriodInDays: constants.CT_TTL_DAYS
        }
    };

    const promise = writeClient.createTable(params).promise();

    await promise.then(
        (data) => {
            console.log(`Table ${data.Table.TableName} created successfully`);
        },
        (err) => {
            if (err.code === 'ConflictException') {
                console.log(`Table ${params.TableName} already exists on db
 ${params.DatabaseName}. Skipping creation.`);
            } else {
                console.log("Error creating table. ", err);
                throw err;
            }
        }
    );
}
```

.NET

```
        public async Task CreateTable()
        {
            Console.WriteLine("Creating Table");

            try
            {
                var createTableRequest = new CreateTableRequest
                {
                    DatabaseName = Constants.DATABASE_NAME,
                    TableName = Constants.TABLE_NAME,
                    RetentionProperties = new RetentionProperties
                    {
                        MagneticStoreRetentionPeriodInDays = Constants.CT_TTL_DAYS,
                        MemoryStoreRetentionPeriodInHours = Constants.HT_TTL_HOURS
                    }
                };
                CreateTableResponse response = await
writeClient.CreateTableAsync(createTableRequest);
                Console.WriteLine($"Table {Constants.TABLE_NAME} created");
            }
            catch (ConflictException)
            {
                Console.WriteLine("Table already exists.");
            }
            catch (Exception e)
            {
                Console.WriteLine("Create table failed:" + e.ToString());
            }

        }
```

Magnetic Store escribe

Puede usar el siguiente fragmento de código para crear una tabla con la función Magnetic Store Writs habilitada. Con las funciones de almacenamiento magnético, puede escribir datos tanto en la ventana de retención del almacén de memoria como en la ventana de retención del almacenamiento magnético.

> **Note**
>
> Estos fragmentos de código se basan en aplicaciones de muestra completas en. GitHub Para obtener más información sobre cómo empezar a utilizar las aplicaciones de ejemplo, consulte. Aplicación de muestra

Java

```
   public void createTable(String databaseName, String tableName) {
       System.out.println("Creating table");
       CreateTableRequest createTableRequest = new CreateTableRequest();
       createTableRequest.setDatabaseName(databaseName);
       createTableRequest.setTableName(tableName);
       final RetentionProperties retentionProperties = new RetentionProperties()
               .withMemoryStoreRetentionPeriodInHours(HT_TTL_HOURS)
               .withMagneticStoreRetentionPeriodInDays(CT_TTL_DAYS);
       createTableRequest.setRetentionProperties(retentionProperties);
       // Enable MagneticStoreWrite
       final MagneticStoreWriteProperties magneticStoreWriteProperties = new
MagneticStoreWriteProperties()
               .withEnableMagneticStoreWrites(true);

createTableRequest.setMagneticStoreWriteProperties(magneticStoreWriteProperties);
       try {
           amazonTimestreamWrite.createTable(createTableRequest);
           System.out.println("Table [" + tableName + "] successfully created.");
       } catch (ConflictException e) {
           System.out.println("Table [" + tableName + "] exists on database [" +
databaseName + "] . Skipping table creation");
           //We do not throw exception here, we use the existing table instead
       }
   }
```

Java v2

```
   public void createTable(String databaseName, String tableName) {
       System.out.println("Creating table");

       // Enable MagneticStoreWrite
       final MagneticStoreWriteProperties magneticStoreWriteProperties =
               MagneticStoreWriteProperties.builder()
```

```
                        .enableMagneticStoreWrites(true)
                        .build();

        CreateTableRequest createTableRequest =
                CreateTableRequest.builder()
                        .databaseName(databaseName)
                        .tableName(tableName)
                        .retentionProperties(RetentionProperties.builder()
                                .memoryStoreRetentionPeriodInHours(HT_TTL_HOURS)
                                .magneticStoreRetentionPeriodInDays(CT_TTL_DAYS)
                                .build())
                        .magneticStoreWriteProperties(magneticStoreWriteProperties)
                        .build();
        try {
            timestreamWriteClient.createTable(createTableRequest);
            System.out.println("Table [" + tableName + "] successfully created.");
        } catch (ConflictException e) {
            System.out.println("Table [" + tableName + "] exists in database [" +
 databaseName + "] . Skipping table creation");
        }
    }
```

Go

```
// Create table.
    createTableInput := &timestreamwrite.CreateTableInput{
        DatabaseName: aws.String(*databaseName),
        TableName:    aws.String(*tableName),
    // Enable MagneticStoreWrite
        MagneticStoreWriteProperties: &timestreamwrite.MagneticStoreWriteProperties{
            EnableMagneticStoreWrites: aws.Bool(true),
             },
      }
    _, err = writeSvc.CreateTable(createTableInput)
```

Python

```
    def create_table(self):
        print("Creating table")
        retention_properties = {
            'MemoryStoreRetentionPeriodInHours': Constant.HT_TTL_HOURS,
            'MagneticStoreRetentionPeriodInDays': Constant.CT_TTL_DAYS
        }
```

```
        magnetic_store_write_properties = {
            'EnableMagneticStoreWrites': True
        }
        try:
            self.client.create_table(DatabaseName=Constant.DATABASE_NAME,
 TableName=Constant.TABLE_NAME,
                                     RetentionProperties=retention_properties,

 MagneticStoreWriteProperties=magnetic_store_write_properties)
            print("Table [%s] successfully created." % Constant.TABLE_NAME)
        except self.client.exceptions.ConflictException:
            print("Table [%s] exists on database [%s]. Skipping table creation" % (
                Constant.TABLE_NAME, Constant.DATABASE_NAME))
        except Exception as err:
            print("Create table failed:", err)
```

Node.js

```
async function createTable() {
    console.log("Creating Table");

    const params = {
        DatabaseName: constants.DATABASE_NAME,
        TableName: constants.TABLE_NAME,
        RetentionProperties: {
            MemoryStoreRetentionPeriodInHours: constants.HT_TTL_HOURS,
            MagneticStoreRetentionPeriodInDays: constants.CT_TTL_DAYS
        },
        MagneticStoreWriteProperties: {
            EnableMagneticStoreWrites: true
        }
    };

    const promise = writeClient.createTable(params).promise();

    await promise.then(
        (data) => {
            console.log(`Table ${data.Table.TableName} created successfully`);
        },
        (err) => {
            if (err.code === 'ConflictException') {
                console.log(`Table ${params.TableName} already exists on db
 ${params.DatabaseName}. Skipping creation.`);
```

```
            } else {
                console.log("Error creating table. ", err);
                throw err;
            }
        }
    );
}
```

.NET

```
        public async Task CreateTable()
        {
            Console.WriteLine("Creating Table");

            try
            {
                var createTableRequest = new CreateTableRequest
                {
                    DatabaseName = Constants.DATABASE_NAME,
                    TableName = Constants.TABLE_NAME,
                    RetentionProperties = new RetentionProperties
                    {
                        MagneticStoreRetentionPeriodInDays = Constants.CT_TTL_DAYS,
                        MemoryStoreRetentionPeriodInHours = Constants.HT_TTL_HOURS
                    },
                    // Enable MagneticStoreWrite
                    MagneticStoreWriteProperties = new MagneticStoreWriteProperties
                    {
                        EnableMagneticStoreWrites = true,
                    }
                };
                CreateTableResponse response = await
 writeClient.CreateTableAsync(createTableRequest);
                Console.WriteLine($"Table {Constants.TABLE_NAME} created");
            }
            catch (ConflictException)
            {
                Console.WriteLine("Table already exists.");
            }
            catch (Exception e)
            {
                Console.WriteLine("Create table failed:" + e.ToString());
            }
```

```
        }
```

Describa la tabla

Puede utilizar los siguientes fragmentos de código para obtener información sobre los atributos de la tabla.

> **Note**
>
> Estos fragmentos de código se basan en aplicaciones de muestra completas en. GitHub Para obtener más información sobre cómo empezar a utilizar las aplicaciones de ejemplo, consulte. Aplicación de muestra

Java

```
   public void describeTable() {
       System.out.println("Describing table");
       final DescribeTableRequest describeTableRequest = new
DescribeTableRequest();
       describeTableRequest.setDatabaseName(DATABASE_NAME);
       describeTableRequest.setTableName(TABLE_NAME);
       try {
           DescribeTableResult result =
amazonTimestreamWrite.describeTable(describeTableRequest);
           String tableId = result.getTable().getArn();
           System.out.println("Table " + TABLE_NAME + " has id " + tableId);
       } catch (final Exception e) {
           System.out.println("Table " + TABLE_NAME + " doesn't exist = " + e);
           throw e;
       }
   }
```

Java v2

```
   public void describeTable() {
       System.out.println("Describing table");
       final DescribeTableRequest describeTableRequest =
DescribeTableRequest.builder()
```

```
                .databaseName(DATABASE_NAME).tableName(TABLE_NAME).build();
        try {
            DescribeTableResponse response =
 timestreamWriteClient.describeTable(describeTableRequest);
            String tableId = response.table().arn();
            System.out.println("Table " + TABLE_NAME + " has id " + tableId);
        } catch (final Exception e) {
            System.out.println("Table " + TABLE_NAME + " doesn't exist = " + e);
            throw e;
        }
    }
```

Go

```
// Describe table.
    describeTableInput := &timestreamwrite.DescribeTableInput{
        DatabaseName: aws.String(*databaseName),
        TableName:    aws.String(*tableName),
    }
    describeTableOutput, err := writeSvc.DescribeTable(describeTableInput)

    if err != nil {
        fmt.Println("Error:")
        fmt.Println(err)
    } else {
        fmt.Println("Describe table is successful, below is the output:")
        fmt.Println(describeTableOutput)
    }
```

Python

```
    def describe_table(self):
        print("Describing table")
        try:
            result = self.client.describe_table(DatabaseName=Constant.DATABASE_NAME,
 TableName=Constant.TABLE_NAME)
            print("Table [%s] has id [%s]" % (Constant.TABLE_NAME, result['Table']
['Arn']))
        except self.client.exceptions.ResourceNotFoundException:
            print("Table doesn't exist")
        except Exception as err:
            print("Describe table failed:", err)
```

Node.js

El siguiente fragmento se utiliza AWS SDK para JavaScript la versión 3. Para obtener más información sobre cómo instalar el cliente y su uso, consulte Timestream Write Client, para la versión 3. AWS SDK JavaScript

Consulte también Class y. DescribeTableCommand DescribeTable

```
import { TimestreamWriteClient, DescribeTableCommand } from "@aws-sdk/client-
timestream-write";
const writeClient = new TimestreamWriteClient({ region: "us-east-1" });

const params = {
    DatabaseName: "testDbFromNode",
    TableName: "testTableFromNode"
};

const command = new DescribeTableCommand(params);

try {
    const data = await writeClient.send(command);
    console.log(`Table ${data.Table.TableName} has id ${data.Table.Arn}`);
} catch (error) {
    if (error.code === 'ResourceNotFoundException') {
        console.log("Table or Database doesn't exist.");
    } else {
        console.log("Describe table failed.", error);
        throw error;
    }
}
```

El siguiente fragmento usa el estilo AWS SDK for JavaScript V2. Se basa en la aplicación de ejemplo de Node.js, ejemplo de Amazon Timestream LiveAnalytics para su aplicación en. GitHub

```
async function describeTable() {
    console.log("Describing Table");
    const params = {
        DatabaseName: constants.DATABASE_NAME,
        TableName: constants.TABLE_NAME
    };

    const promise = writeClient.describeTable(params).promise();
```

```
    await promise.then(
        (data) => {
            console.log(`Table ${data.Table.TableName} has id ${data.Table.Arn}`);
        },
        (err) => {
            if (err.code === 'ResourceNotFoundException') {
                console.log("Table or Database doesn't exists.");
            } else {
                console.log("Describe table failed.", err);
                throw err;
            }
        }
    );
}
```

.NET

```
        public async Task DescribeTable()
        {
            Console.WriteLine("Describing Table");

            try
            {
                var describeTableRequest = new DescribeTableRequest
                {
                    DatabaseName = Constants.DATABASE_NAME,
                    TableName = Constants.TABLE_NAME
                };
                DescribeTableResponse response = await
 writeClient.DescribeTableAsync(describeTableRequest);
                Console.WriteLine($"Table {Constants.TABLE_NAME} has id:
{response.Table.Arn}");
            }
            catch (ResourceNotFoundException)
            {
                Console.WriteLine("Table does not exist.");
            }
            catch (Exception e)
            {
                Console.WriteLine("Describe table failed:" + e.ToString());
            }

        }
```

Actualizar tabla

Puede utilizar los siguientes fragmentos de código para actualizar una tabla.

> **Note**
>
> Estos fragmentos de código se basan en aplicaciones de muestra completas en. GitHub Para obtener más información sobre cómo empezar a utilizar las aplicaciones de ejemplo, consulte. Aplicación de muestra

Java

```
    public void updateTable() {
        System.out.println("Updating table");
        UpdateTableRequest updateTableRequest = new UpdateTableRequest();
        updateTableRequest.setDatabaseName(DATABASE_NAME);
        updateTableRequest.setTableName(TABLE_NAME);

        final RetentionProperties retentionProperties = new RetentionProperties()
                .withMemoryStoreRetentionPeriodInHours(HT_TTL_HOURS)
                .withMagneticStoreRetentionPeriodInDays(CT_TTL_DAYS);

        updateTableRequest.setRetentionProperties(retentionProperties);

        amazonTimestreamWrite.updateTable(updateTableRequest);
        System.out.println("Table updated");
    }
```

Java v2

```
    public void updateTable() {
        System.out.println("Updating table");

        final RetentionProperties retentionProperties =
RetentionProperties.builder()
                .memoryStoreRetentionPeriodInHours(HT_TTL_HOURS)
                .magneticStoreRetentionPeriodInDays(CT_TTL_DAYS).build();
        final UpdateTableRequest updateTableRequest = UpdateTableRequest.builder()

.databaseName(DATABASE_NAME).tableName(TABLE_NAME).retentionProperties(retentionProperties)
```

```
        timestreamWriteClient.updateTable(updateTableRequest);
        System.out.println("Table updated");
    }
```

Go

```
// Update table.
    magneticStoreRetentionPeriodInDays := int64(7 * 365)
    memoryStoreRetentionPeriodInHours := int64(24)

    updateTableInput := &timestreamwrite.UpdateTableInput{
        DatabaseName: aws.String(*databaseName),
        TableName:    aws.String(*tableName),
        RetentionProperties: &timestreamwrite.RetentionProperties{
            MagneticStoreRetentionPeriodInDays: &magneticStoreRetentionPeriodInDays,
            MemoryStoreRetentionPeriodInHours:  &memoryStoreRetentionPeriodInHours,
        },
    }
    updateTableOutput, err := writeSvc.UpdateTable(updateTableInput)

    if err != nil {
        fmt.Println("Error:")
        fmt.Println(err)
    } else {
        fmt.Println("Update table is successful, below is the output:")
        fmt.Println(updateTableOutput)
    }
```

Python

```
    def update_table(self):
        print("Updating table")
        retention_properties = {
            'MemoryStoreRetentionPeriodInHours': Constant.HT_TTL_HOURS,
            'MagneticStoreRetentionPeriodInDays': Constant.CT_TTL_DAYS
        }
        try:
            self.client.update_table(DatabaseName=Constant.DATABASE_NAME,
 TableName=Constant.TABLE_NAME,
                                     RetentionProperties=retention_properties)
            print("Table updated.")
        except Exception as err:
```

```
            print("Update table failed:", err)
```

Node.js

El siguiente fragmento se utiliza AWS SDK para JavaScript la versión 3. Para obtener más información sobre cómo instalar el cliente y su uso, consulte Timestream Write Client, para la versión 3. AWS SDK JavaScript

Consulte también Class y. UpdateTableCommand UpdateTable

```
import { TimestreamWriteClient, UpdateTableCommand } from "@aws-sdk/client-
timestream-write";
const writeClient = new TimestreamWriteClient({ region: "us-east-1" });

const params = {
    DatabaseName: "testDbFromNode",
    TableName: "testTableFromNode",
    RetentionProperties: {
        MemoryStoreRetentionPeriodInHours: 24,
        MagneticStoreRetentionPeriodInDays: 180
    }
};

const command = new UpdateTableCommand(params);

try {
    const data = await writeClient.send(command);
    console.log("Table updated")
} catch (error) {
    console.log("Error updating table. ", error);
}
```

El siguiente fragmento usa el estilo AWS SDK for JavaScript V2. Se basa en la aplicación de ejemplo de Node.js, ejemplo de Amazon Timestream LiveAnalytics para su aplicación en. GitHub

```
async function updateTable() {
    console.log("Updating Table");
    const params = {
        DatabaseName: constants.DATABASE_NAME,
        TableName: constants.TABLE_NAME,
        RetentionProperties: {
            MemoryStoreRetentionPeriodInHours: constants.HT_TTL_HOURS,
```

```
            MagneticStoreRetentionPeriodInDays: constants.CT_TTL_DAYS
        }
    };

    const promise = writeClient.updateTable(params).promise();

    await promise.then(
        (data) => {
            console.log("Table updated")
        },
        (err) => {
            console.log("Error updating table. ", err);
            throw err;
        }
    );
}
```

.NET

```
        public async Task UpdateTable()
        {
            Console.WriteLine("Updating Table");

            try
            {
                var updateTableRequest = new UpdateTableRequest
                {
                    DatabaseName = Constants.DATABASE_NAME,
                    TableName = Constants.TABLE_NAME,
                    RetentionProperties = new RetentionProperties
                    {
                        MagneticStoreRetentionPeriodInDays = Constants.CT_TTL_DAYS,
                        MemoryStoreRetentionPeriodInHours = Constants.HT_TTL_HOURS
                    }
                };
                UpdateTableResponse response = await
 writeClient.UpdateTableAsync(updateTableRequest);
                Console.WriteLine($"Table {Constants.TABLE_NAME} updated");
            }
            catch (ResourceNotFoundException)
            {
                Console.WriteLine("Table does not exist.");
            }
```

```
            catch (Exception e)
            {
                Console.WriteLine("Update table failed:" + e.ToString());
            }

        }
```

Eliminar tabla

Puede utilizar los siguientes fragmentos de código para eliminar una tabla.

> **Note**
>
> Estos fragmentos de código se basan en aplicaciones de muestra completas en. GitHub Para obtener más información sobre cómo empezar a utilizar las aplicaciones de ejemplo, consulte. Aplicación de muestra

Java

```
   public void deleteTable() {
       System.out.println("Deleting table");
       final DeleteTableRequest deleteTableRequest = new DeleteTableRequest();
       deleteTableRequest.setDatabaseName(DATABASE_NAME);
       deleteTableRequest.setTableName(TABLE_NAME);
       try {
           DeleteTableResult result =
                   amazonTimestreamWrite.deleteTable(deleteTableRequest);
           System.out.println("Delete table status: " +
result.getSdkHttpMetadata().getHttpStatusCode());
       } catch (final ResourceNotFoundException e) {
           System.out.println("Table " + TABLE_NAME + " doesn't exist = " + e);
           throw e;
       } catch (final Exception e) {
           System.out.println("Could not delete table " + TABLE_NAME + " = " + e);
           throw e;
       }
   }
```

Java v2

```
    public void deleteTable() {
        System.out.println("Deleting table");
        final DeleteTableRequest deleteTableRequest = DeleteTableRequest.builder()
                .databaseName(DATABASE_NAME).tableName(TABLE_NAME).build();
        try {
            DeleteTableResponse response =
                    timestreamWriteClient.deleteTable(deleteTableRequest);
            System.out.println("Delete table status: " +
 response.sdkHttpResponse().statusCode());
        } catch (final ResourceNotFoundException e) {
            System.out.println("Table " + TABLE_NAME + " doesn't exist = " + e);
            throw e;
        } catch (final Exception e) {
            System.out.println("Could not delete table " + TABLE_NAME + " = " + e);
            throw e;
        }
    }
```

Go

```
deleteTableInput := &timestreamwrite.DeleteTableInput{
        DatabaseName:   aws.String(*databaseName),
        TableName:    aws.String(*tableName),
    }
    _, err = writeSvc.DeleteTable(deleteTableInput)

    if err != nil {
        fmt.Println("Error:")
        fmt.Println(err)
    } else {
        fmt.Println("Table deleted", *tableName)
    }
```

Python

```
    def delete_table(self):
        print("Deleting Table")
        try:
            result = self.client.delete_table(DatabaseName=Constant.DATABASE_NAME,
 TableName=Constant.TABLE_NAME)
```

```
            print("Delete table status [%s]" % result['ResponseMetadata']
['HTTPStatusCode'])
        except self.client.exceptions.ResourceNotFoundException:
            print("Table [%s] doesn't exist" % Constant.TABLE_NAME)
        except Exception as err:
            print("Delete table failed:", err)
```

Node.js

El siguiente fragmento se utiliza AWS SDK para JavaScript la versión 3. Para obtener más información sobre cómo instalar el cliente y su uso, consulte Timestream Write Client, para la versión 3. AWS SDK JavaScript

Consulte también Class y. DeleteTableCommand DeleteTable

```
import { TimestreamWriteClient, DeleteTableCommand } from "@aws-sdk/client-
timestream-write";
const writeClient = new TimestreamWriteClient({ region: "us-east-1" });

const params = {
    DatabaseName: "testDbFromNode",
    TableName: "testTableFromNode"
};

const command = new DeleteTableCommand(params);

try {
    const data = await writeClient.send(command);
    console.log("Deleted table");
} catch (error) {
    if (error.code === 'ResourceNotFoundException') {
        console.log(`Table ${params.TableName} or Database ${params.DatabaseName}
 doesn't exist.`);
    } else {
        console.log("Delete table failed.", error);
        throw error;
    }
}
```

El siguiente fragmento usa el estilo AWS SDK for JavaScript V2. Se basa en la aplicación de ejemplo de Node.js, ejemplo de Amazon Timestream LiveAnalytics para su aplicación en. GitHub

```
async function deleteTable() {
```

```
    console.log("Deleting Table");
    const params = {
        DatabaseName: constants.DATABASE_NAME,
        TableName: constants.TABLE_NAME
    };

    const promise = writeClient.deleteTable(params).promise();

    await promise.then(
        function (data) {
            console.log("Deleted table");
        },
        function(err) {
            if (err.code === 'ResourceNotFoundException') {
                console.log(`Table ${params.TableName} or Database
 ${params.DatabaseName} doesn't exists.`);
            } else {
                console.log("Delete table failed.", err);
                throw err;
            }
        }
    );
}
```

.NET

```
        public async Task DeleteTable()
        {
            Console.WriteLine("Deleting table");
            try
            {
                var deleteTableRequest = new DeleteTableRequest
                {
                    DatabaseName = Constants.DATABASE_NAME,
                    TableName = Constants.TABLE_NAME
                };
                DeleteTableResponse response = await
 writeClient.DeleteTableAsync(deleteTableRequest);
                Console.WriteLine($"Table {Constants.TABLE_NAME} delete request
 status: {response.HttpStatusCode}");
            }
            catch (ResourceNotFoundException)
            {
```

```
                Console.WriteLine($"Table {Constants.TABLE_NAME} does not exists");
            }
            catch (Exception e)
            {
                Console.WriteLine("Exception while deleting table:" + e.ToString());
            }
        }
```

Lista de tablas

Puede usar los siguientes fragmentos de código para enumerar tablas.

> **Note**
>
> Estos fragmentos de código se basan en aplicaciones de muestra completas en. GitHub Para obtener más información sobre cómo empezar a utilizar las aplicaciones de ejemplo, consulte. Aplicación de muestra

Java

```
    public void listTables() {
        System.out.println("Listing tables");
        ListTablesRequest request = new ListTablesRequest();
        request.setDatabaseName(DATABASE_NAME);
        ListTablesResult result = amazonTimestreamWrite.listTables(request);
        printTables(result.getTables());

        String nextToken = result.getNextToken();
        while (nextToken != null && !nextToken.isEmpty()) {
            request.setNextToken(nextToken);
            ListTablesResult nextResult = amazonTimestreamWrite.listTables(request);

            printTables(nextResult.getTables());
            nextToken = nextResult.getNextToken();
        }
    }

     private void printTables(List<Table> tables) {
        for (Table table : tables) {
            System.out.println(table.getTableName());
```

```
        }
    }
```

Java v2

```
    public void listTables() {
        System.out.println("Listing tables");
        ListTablesRequest request =
 ListTablesRequest.builder().databaseName(DATABASE_NAME).maxResults(2).build();
        ListTablesIterable listTablesIterable =
 timestreamWriteClient.listTablesPaginator(request);
        for(ListTablesResponse listTablesResponse : listTablesIterable) {
            final List<Table> tables = listTablesResponse.tables();
            tables.forEach(table -> System.out.println(table.tableName()));
        }
    }
```

Go

```
listTablesMaxResult := int64(15)

    listTablesInput := &timestreamwrite.ListTablesInput{
        DatabaseName: aws.String(*databaseName),
        MaxResults:   &listTablesMaxResult,
    }
    listTablesOutput, err := writeSvc.ListTables(listTablesInput)

    if err != nil {
        fmt.Println("Error:")
        fmt.Println(err)
    } else {
        fmt.Println("List tables is successful, below is the output:")
        fmt.Println(listTablesOutput)
    }
```

Python

```
    def list_tables(self):
        print("Listing tables")
        try:
            result = self.client.list_tables(DatabaseName=Constant.DATABASE_NAME,
 MaxResults=5)
            self.__print_tables(result['Tables'])
```

```
            next_token = result.get('NextToken', None)
            while next_token:
                result =
 self.client.list_tables(DatabaseName=Constant.DATABASE_NAME,
                                                    NextToken=next_token, MaxResults=5)
                self.__print_tables(result['Tables'])
                next_token = result.get('NextToken', None)
        except Exception as err:
            print("List tables failed:", err)
```

Node.js

El siguiente fragmento se utiliza AWS SDK para JavaScript la versión 3. Para obtener más información sobre cómo instalar el cliente y su uso, consulte Timestream Write Client, para la versión 3. AWS SDK JavaScript

Consulte también Class y. ListTablesCommand ListTables

```
import { TimestreamWriteClient, ListTablesCommand } from "@aws-sdk/client-
timestream-write";
const writeClient = new TimestreamWriteClient({ region: "us-east-1" });

const params = {
    DatabaseName: "testDbFromNode",
    MaxResults: 15
};

const command = new ListTablesCommand(params);

getTablesList(null);

async function getTablesList(nextToken) {
    if (nextToken) {
        params.NextToken = nextToken;
    }

    try {
        const data = await writeClient.send(command);

        data.Tables.forEach(function (table) {
            console.log(table.TableName);
        });
```

```
        if (data.NextToken) {
            return getTablesList(data.NextToken);
        }
    } catch (error) {
        console.log("Error while listing tables", error);
    }
}
```

El siguiente fragmento usa el estilo AWS SDK for JavaScript V2. Se basa en la aplicación de ejemplo de Node.js, ejemplo de Amazon Timestream LiveAnalytics para su aplicación en. GitHub

```
async function listTables() {
    console.log("Listing tables:");
    const tables = await getTablesList(null);
    tables.forEach(function(table){
        console.log(table.TableName);
    });
}

function getTablesList(nextToken, tables = []) {
    var params = {
        DatabaseName: constants.DATABASE_NAME,
        MaxResults: 15
    };

    if(nextToken) {
        params.NextToken = nextToken;
    }

    return writeClient.listTables(params).promise()
        .then(
            (data) => {
                tables.push.apply(tables, data.Tables);
                if (data.NextToken) {
                    return getTablesList(data.NextToken, tables);
                } else {
                    return tables;
                }
            },
            (err) => {
                console.log("Error while listing databases", err);
            });
}
```

.NET

```
        public async Task ListTables()
        {
            Console.WriteLine("Listing Tables");

            try
            {
                var listTablesRequest = new ListTablesRequest
                {
                    MaxResults = 5,
                    DatabaseName = Constants.DATABASE_NAME
                };
                ListTablesResponse response = await
writeClient.ListTablesAsync(listTablesRequest);
                PrintTables(response.Tables);
                string nextToken = response.NextToken;
                while (nextToken != null)
                {
                    listTablesRequest.NextToken = nextToken;
                    response = await writeClient.ListTablesAsync(listTablesRequest);
                    PrintTables(response.Tables);
                    nextToken = response.NextToken;
                }
            }
            catch (Exception e)
            {
                Console.WriteLine("List table failed:" + e.ToString());
            }

        }

        private void PrintTables(List<Table> tables)
        {
            foreach (Table table in tables)
                Console.WriteLine($"Table: {table.TableName}");
        }
```

Escribir datos (inserciones y ajustes)

Temas

Escribir lotes de registros

Puede usar los siguientes fragmentos de código para escribir datos en una tabla de Amazon Timestream. Escribir datos en lotes ayuda a optimizar el costo de las escrituras. Para obtener más información, consulte Calcular el número de escrituras.

> **Note**
>
> Estos fragmentos de código se basan en aplicaciones de muestra completas en. GitHub Para obtener más información sobre cómo empezar a utilizar las aplicaciones de ejemplo, consulte. Aplicación de muestra

Java

```
  public void writeRecords() {
    System.out.println("Writing records");
    // Specify repeated values for all records
    List<Record> records = new ArrayList<>();
    final long time = System.currentTimeMillis();

    List<Dimension> dimensions = new ArrayList<>();
    final Dimension region = new Dimension().withName("region").withValue("us-
east-1");
    final Dimension az = new Dimension().withName("az").withValue("az1");
    final Dimension hostname = new
 Dimension().withName("hostname").withValue("host1");

    dimensions.add(region);
    dimensions.add(az);
    dimensions.add(hostname);

    Record cpuUtilization = new Record()
```

```
        .withDimensions(dimensions)
        .withMeasureName("cpu_utilization")
        .withMeasureValue("13.5")
        .withMeasureValueType(MeasureValueType.DOUBLE)
        .withTime(String.valueOf(time));
    Record memoryUtilization = new Record()
        .withDimensions(dimensions)
        .withMeasureName("memory_utilization")
        .withMeasureValue("40")
        .withMeasureValueType(MeasureValueType.DOUBLE)
        .withTime(String.valueOf(time));

    records.add(cpuUtilization);
    records.add(memoryUtilization);

    WriteRecordsRequest writeRecordsRequest = new WriteRecordsRequest()
        .withDatabaseName(DATABASE_NAME)
        .withTableName(TABLE_NAME)
        .withRecords(records);

    try {
      WriteRecordsResult writeRecordsResult =
amazonTimestreamWrite.writeRecords(writeRecordsRequest);
      System.out.println("WriteRecords Status: " +
writeRecordsResult.getSdkHttpMetadata().getHttpStatusCode());
    } catch (RejectedRecordsException e) {
      System.out.println("RejectedRecords: " + e);
      for (RejectedRecord rejectedRecord : e.getRejectedRecords()) {
        System.out.println("Rejected Index " + rejectedRecord.getRecordIndex() + ":
"
            + rejectedRecord.getReason());
      }
      System.out.println("Other records were written successfully. ");
    } catch (Exception e) {
      System.out.println("Error: " + e);
    }
  }
```

Java v2

```
 public void writeRecords() {
   System.out.println("Writing records");
   // Specify repeated values for all records
```

```
    List<Record> records = new ArrayList<>();
    final long time = System.currentTimeMillis();

    List<Dimension> dimensions = new ArrayList<>();
    final Dimension region = Dimension.builder().name("region").value("us-
east-1").build();
    final Dimension az = Dimension.builder().name("az").value("az1").build();
    final Dimension hostname =
 Dimension.builder().name("hostname").value("host1").build();

    dimensions.add(region);
    dimensions.add(az);
    dimensions.add(hostname);

    Record cpuUtilization = Record.builder()
        .dimensions(dimensions)
        .measureValueType(MeasureValueType.DOUBLE)
        .measureName("cpu_utilization")
        .measureValue("13.5")
        .time(String.valueOf(time)).build();

    Record memoryUtilization = Record.builder()
        .dimensions(dimensions)
        .measureValueType(MeasureValueType.DOUBLE)
        .measureName("memory_utilization")
        .measureValue("40")
        .time(String.valueOf(time)).build();

    records.add(cpuUtilization);
    records.add(memoryUtilization);

    WriteRecordsRequest writeRecordsRequest = WriteRecordsRequest.builder()
        .databaseName(DATABASE_NAME).tableName(TABLE_NAME).records(records).build();

    try {
      WriteRecordsResponse writeRecordsResponse =
 timestreamWriteClient.writeRecords(writeRecordsRequest);
      System.out.println("WriteRecords Status: " +
 writeRecordsResponse.sdkHttpResponse().statusCode());
    } catch (RejectedRecordsException e) {
      System.out.println("RejectedRecords: " + e);
      for (RejectedRecord rejectedRecord : e.rejectedRecords()) {
        System.out.println("Rejected Index " + rejectedRecord.recordIndex() + ": "
            + rejectedRecord.reason());
```

```
      }
      System.out.println("Other records were written successfully. ");
    } catch (Exception e) {
      System.out.println("Error: " + e);
    }
  }
```

Go

```
now := time.Now()
currentTimeInSeconds := now.Unix()
writeRecordsInput := &timestreamwrite.WriteRecordsInput{
  DatabaseName: aws.String(*databaseName),
  TableName:  aws.String(*tableName),
  Records: []*timestreamwrite.Record{
    &timestreamwrite.Record{
      Dimensions: []*timestreamwrite.Dimension{
        &timestreamwrite.Dimension{
          Name:  aws.String("region"),
          Value: aws.String("us-east-1"),
        },
        &timestreamwrite.Dimension{
          Name:  aws.String("az"),
          Value: aws.String("az1"),
        },
        &timestreamwrite.Dimension{
          Name:  aws.String("hostname"),
          Value: aws.String("host1"),
        },
      },
      MeasureName:    aws.String("cpu_utilization"),
      MeasureValue:   aws.String("13.5"),
      MeasureValueType: aws.String("DOUBLE"),
      Time:       aws.String(strconv.FormatInt(currentTimeInSeconds, 10)),
      TimeUnit:  aws.String("SECONDS"),
    },
    &timestreamwrite.Record{
      Dimensions: []*timestreamwrite.Dimension{
        &timestreamwrite.Dimension{
          Name:  aws.String("region"),
          Value: aws.String("us-east-1"),
        },
        &timestreamwrite.Dimension{
```

```
          Name:  aws.String("az"),
          Value: aws.String("az1"),
        },
        &timestreamwrite.Dimension{
          Name:  aws.String("hostname"),
          Value: aws.String("host1"),
        },
      },
      MeasureName:    aws.String("memory_utilization"),
      MeasureValue:   aws.String("40"),
      MeasureValueType: aws.String("DOUBLE"),
      Time:       aws.String(strconv.FormatInt(currentTimeInSeconds, 10)),
      TimeUnit:  aws.String("SECONDS"),
    },
  },
}

_, err = writeSvc.WriteRecords(writeRecordsInput)

if err != nil {
  fmt.Println("Error:")
  fmt.Println(err)
} else {
  fmt.Println("Write records is successful")
}
```

Python

```
  def write_records(self):
    print("Writing records")
    current_time = self._current_milli_time()

    dimensions = [
      {'Name': 'region', 'Value': 'us-east-1'},
      {'Name': 'az', 'Value': 'az1'},
      {'Name': 'hostname', 'Value': 'host1'}
    ]

    cpu_utilization = {
      'Dimensions': dimensions,
      'MeasureName': 'cpu_utilization',
      'MeasureValue': '13.5',
      'MeasureValueType': 'DOUBLE',
```

```
      'Time': current_time
    }

    memory_utilization = {
      'Dimensions': dimensions,
      'MeasureName': 'memory_utilization',
      'MeasureValue': '40',
      'MeasureValueType': 'DOUBLE',
      'Time': current_time
    }

    records = [cpu_utilization, memory_utilization]

    try:
      result = self.client.write_records(DatabaseName=Constant.DATABASE_NAME,
 TableName=Constant.TABLE_NAME,
                         Records=records, CommonAttributes={})
      print("WriteRecords Status: [%s]" % result['ResponseMetadata']
['HTTPStatusCode'])
    except self.client.exceptions.RejectedRecordsException as err:
      self._print_rejected_records_exceptions(err)
    except Exception as err:
      print("Error:", err)

  @staticmethod
  def _print_rejected_records_exceptions(err):
    print("RejectedRecords: ", err)
    for rr in err.response["RejectedRecords"]:
      print("Rejected Index " + str(rr["RecordIndex"]) + ": " + rr["Reason"])
      if "ExistingVersion" in rr:
        print("Rejected record existing version: ", rr["ExistingVersion"])

  @staticmethod
  def _current_milli_time():
    return str(int(round(time.time() * 1000)))
```

Node.js

En el siguiente fragmento se utiliza el estilo AWS SDK for JavaScript V2. Se basa en la aplicación de ejemplo de Node.js, ejemplo de Amazon Timestream LiveAnalytics para su aplicación en. GitHub

```
async function writeRecords() {
```

```
  console.log("Writing records");
  const currentTime = Date.now().toString(); // Unix time in milliseconds

  const dimensions = [
    {'Name': 'region', 'Value': 'us-east-1'},
    {'Name': 'az', 'Value': 'az1'},
    {'Name': 'hostname', 'Value': 'host1'}
  ];

  const cpuUtilization = {
    'Dimensions': dimensions,
    'MeasureName': 'cpu_utilization',
    'MeasureValue': '13.5',
    'MeasureValueType': 'DOUBLE',
    'Time': currentTime.toString()
  };

  const memoryUtilization = {
    'Dimensions': dimensions,
    'MeasureName': 'memory_utilization',
    'MeasureValue': '40',
    'MeasureValueType': 'DOUBLE',
    'Time': currentTime.toString()
  };

  const records = [cpuUtilization, memoryUtilization];

  const params = {
    DatabaseName: constants.DATABASE_NAME,
    TableName: constants.TABLE_NAME,
    Records: records
  };

  const request = writeClient.writeRecords(params);

  await request.promise().then(
    (data) => {
      console.log("Write records successful");
    },
    (err) => {
      console.log("Error writing records:", err);
      if (err.code === 'RejectedRecordsException') {
        const responsePayload =
JSON.parse(request.response.httpResponse.body.toString());
```

```
        console.log("RejectedRecords: ", responsePayload.RejectedRecords);
        console.log("Other records were written successfully. ");
      }
    }
  );
}
```

.NET

```
  public async Task WriteRecords()
  {
    Console.WriteLine("Writing records");

    DateTimeOffset now = DateTimeOffset.UtcNow;
    string currentTimeString = (now.ToUnixTimeMilliseconds()).ToString();

    List<Dimension> dimensions = new List<Dimension>{
      new Dimension { Name = "region", Value = "us-east-1" },
      new Dimension { Name = "az", Value = "az1" },
      new Dimension { Name = "hostname", Value = "host1" }
    };

    var cpuUtilization = new Record
    {
      Dimensions = dimensions,
      MeasureName = "cpu_utilization",
      MeasureValue = "13.6",
      MeasureValueType = MeasureValueType.DOUBLE,
      Time = currentTimeString
    };

    var memoryUtilization = new Record
    {
      Dimensions = dimensions,
      MeasureName = "memory_utilization",
      MeasureValue = "40",
      MeasureValueType = MeasureValueType.DOUBLE,
      Time = currentTimeString
    };


    List<Record> records = new List<Record> {
      cpuUtilization,
```

```
      memoryUtilization
    };

    try
    {
      var writeRecordsRequest = new WriteRecordsRequest
      {
        DatabaseName = Constants.DATABASE_NAME,
        TableName = Constants.TABLE_NAME,
        Records = records
      };
      WriteRecordsResponse response = await
writeClient.WriteRecordsAsync(writeRecordsRequest);
      Console.WriteLine($"Write records status code:
{response.HttpStatusCode.ToString()}");
    }
    catch (RejectedRecordsException e) {
      Console.WriteLine("RejectedRecordsException:" + e.ToString());
      foreach (RejectedRecord rr in e.RejectedRecords) {
        Console.WriteLine("RecordIndex " + rr.RecordIndex + " : " + rr.Reason);
      }
      Console.WriteLine("Other records were written successfully. ");
    }
    catch (Exception e)
    {
      Console.WriteLine("Write records failure:" + e.ToString());
    }
  }
```

Escribir lotes de registros con atributos comunes

Si los datos de sus series temporales tienen medidas o dimensiones comunes en muchos puntos de datos, también puede utilizar la siguiente versión optimizada writeRecords API para insertar datos en Timestream. LiveAnalytics El uso de atributos comunes con el procesamiento por lotes puede optimizar aún más el costo de las escrituras, como se describe en. Calcular el número de escrituras

> **Note**
>
> Estos fragmentos de código se basan en aplicaciones de muestra completas en. GitHub Para obtener más información sobre cómo empezar a utilizar las aplicaciones de ejemplo, consulte. Aplicación de muestra

Java

```
  public void writeRecordsWithCommonAttributes() {
    System.out.println("Writing records with extracting common attributes");
    // Specify repeated values for all records
    List<Record> records = new ArrayList<>();
    final long time = System.currentTimeMillis();

    List<Dimension> dimensions = new ArrayList<>();
    final Dimension region = new Dimension().withName("region").withValue("us-
east-1");
    final Dimension az = new Dimension().withName("az").withValue("az1");
    final Dimension hostname = new
 Dimension().withName("hostname").withValue("host1");

    dimensions.add(region);
    dimensions.add(az);
    dimensions.add(hostname);

    Record commonAttributes = new Record()
        .withDimensions(dimensions)
        .withMeasureValueType(MeasureValueType.DOUBLE)
        .withTime(String.valueOf(time));

    Record cpuUtilization = new Record()
        .withMeasureName("cpu_utilization")
        .withMeasureValue("13.5");
    Record memoryUtilization = new Record()
        .withMeasureName("memory_utilization")
        .withMeasureValue("40");

    records.add(cpuUtilization);
    records.add(memoryUtilization);

    WriteRecordsRequest writeRecordsRequest = new WriteRecordsRequest()
```

```
        .withDatabaseName(DATABASE_NAME)
        .withTableName(TABLE_NAME)
        .withCommonAttributes(commonAttributes);
    writeRecordsRequest.setRecords(records);

    try {
      WriteRecordsResult writeRecordsResult =
 amazonTimestreamWrite.writeRecords(writeRecordsRequest);
      System.out.println("writeRecordsWithCommonAttributes Status: " +
 writeRecordsResult.getSdkHttpMetadata().getHttpStatusCode());
    } catch (RejectedRecordsException e) {
      System.out.println("RejectedRecords: " + e);
      for (RejectedRecord rejectedRecord : e.getRejectedRecords()) {
        System.out.println("Rejected Index " + rejectedRecord.getRecordIndex() + ":
 "
            + rejectedRecord.getReason());
      }
      System.out.println("Other records were written successfully. ");
    } catch (Exception e) {
      System.out.println("Error: " + e);
    }
  }
```

Java v2

```
  public void writeRecordsWithCommonAttributes() {
    System.out.println("Writing records with extracting common attributes");
    // Specify repeated values for all records
    List<Record> records = new ArrayList<>();
    final long time = System.currentTimeMillis();

    List<Dimension> dimensions = new ArrayList<>();
    final Dimension region = Dimension.builder().name("region").value("us-
east-1").build();
    final Dimension az = Dimension.builder().name("az").value("az1").build();
    final Dimension hostname =
 Dimension.builder().name("hostname").value("host1").build();

    dimensions.add(region);
    dimensions.add(az);
    dimensions.add(hostname);

    Record commonAttributes = Record.builder()
```

```
        .dimensions(dimensions)
        .measureValueType(MeasureValueType.DOUBLE)
        .time(String.valueOf(time)).build();

    Record cpuUtilization = Record.builder()
        .measureName("cpu_utilization")
        .measureValue("13.5").build();
    Record memoryUtilization = Record.builder()
        .measureName("memory_utilization")
        .measureValue("40").build();

    records.add(cpuUtilization);
    records.add(memoryUtilization);

    WriteRecordsRequest writeRecordsRequest = WriteRecordsRequest.builder()
        .databaseName(DATABASE_NAME)
        .tableName(TABLE_NAME)
        .commonAttributes(commonAttributes)
        .records(records).build();

    try {
      WriteRecordsResponse writeRecordsResponse =
 timestreamWriteClient.writeRecords(writeRecordsRequest);
      System.out.println("writeRecordsWithCommonAttributes Status: " +
 writeRecordsResponse.sdkHttpResponse().statusCode());
    } catch (RejectedRecordsException e) {
      System.out.println("RejectedRecords: " + e);
      for (RejectedRecord rejectedRecord : e.rejectedRecords()) {
        System.out.println("Rejected Index " + rejectedRecord.recordIndex() + ": "
            + rejectedRecord.reason());
      }
      System.out.println("Other records were written successfully. ");
    } catch (Exception e) {
      System.out.println("Error: " + e);
    }
  }
```

Go

```
now = time.Now()
currentTimeInSeconds = now.Unix()
writeRecordsCommonAttributesInput := &timestreamwrite.WriteRecordsInput{
 DatabaseName: aws.String(*databaseName),
```

```
 TableName:  aws.String(*tableName),
 CommonAttributes: &timestreamwrite.Record{
  Dimensions: []*timestreamwrite.Dimension{
   &timestreamwrite.Dimension{
    Name:  aws.String("region"),
    Value: aws.String("us-east-1"),
   },
   &timestreamwrite.Dimension{
    Name:  aws.String("az"),
    Value: aws.String("az1"),
   },
   &timestreamwrite.Dimension{
    Name:  aws.String("hostname"),
    Value: aws.String("host1"),
   },
  },
  MeasureValueType: aws.String("DOUBLE"),
  Time:       aws.String(strconv.FormatInt(currentTimeInSeconds, 10)),
  TimeUnit:     aws.String("SECONDS"),
 },
 Records: []*timestreamwrite.Record{
  &timestreamwrite.Record{
   MeasureName:  aws.String("cpu_utilization"),
   MeasureValue: aws.String("13.5"),
  },
  &timestreamwrite.Record{
   MeasureName:  aws.String("memory_utilization"),
   MeasureValue: aws.String("40"),
  },
 },
}

_, err = writeSvc.WriteRecords(writeRecordsCommonAttributesInput)

if err != nil {
 fmt.Println("Error:")
 fmt.Println(err)
} else {
 fmt.Println("Ingest records is successful")
}
```

Python

```
  def write_records_with_common_attributes(self):
    print("Writing records extracting common attributes")
    current_time = self._current_milli_time()

    dimensions = [
      {'Name': 'region', 'Value': 'us-east-1'},
      {'Name': 'az', 'Value': 'az1'},
      {'Name': 'hostname', 'Value': 'host1'}
    ]

    common_attributes = {
      'Dimensions': dimensions,
      'MeasureValueType': 'DOUBLE',
      'Time': current_time
    }

    cpu_utilization = {
      'MeasureName': 'cpu_utilization',
      'MeasureValue': '13.5'
    }

    memory_utilization = {
      'MeasureName': 'memory_utilization',
      'MeasureValue': '40'
    }

    records = [cpu_utilization, memory_utilization]

    try:
      result = self.client.write_records(DatabaseName=Constant.DATABASE_NAME,
 TableName=Constant.TABLE_NAME,
                         Records=records, CommonAttributes=common_attributes)
      print("WriteRecords Status: [%s]" % result['ResponseMetadata']
['HTTPStatusCode'])
    except self.client.exceptions.RejectedRecordsException as err:
      self._print_rejected_records_exceptions(err)
    except Exception as err:
      print("Error:", err)

  @staticmethod
  def _print_rejected_records_exceptions(err):
    print("RejectedRecords: ", err)
```

```
    for rr in err.response["RejectedRecords"]:
      print("Rejected Index " + str(rr["RecordIndex"]) + ": " + rr["Reason"])
      if "ExistingVersion" in rr:
        print("Rejected record existing version: ", rr["ExistingVersion"])

  @staticmethod
  def _current_milli_time():
    return str(int(round(time.time() * 1000)))
```

Node.js

En el siguiente fragmento se utiliza el estilo AWS SDK for JavaScript V2. Se basa en la aplicación de ejemplo de Node.js, ejemplo de Amazon Timestream LiveAnalytics para su aplicación en. GitHub

```
async function writeRecordsWithCommonAttributes() {
  console.log("Writing records with common attributes");
  const currentTime = Date.now().toString(); // Unix time in milliseconds

  const dimensions = [
    {'Name': 'region', 'Value': 'us-east-1'},
    {'Name': 'az', 'Value': 'az1'},
    {'Name': 'hostname', 'Value': 'host1'}
  ];

  const commonAttributes = {
    'Dimensions': dimensions,
    'MeasureValueType': 'DOUBLE',
    'Time': currentTime.toString()
  };

  const cpuUtilization = {
    'MeasureName': 'cpu_utilization',
    'MeasureValue': '13.5'
  };

  const memoryUtilization = {
    'MeasureName': 'memory_utilization',
    'MeasureValue': '40'
  };

  const records = [cpuUtilization, memoryUtilization];
```

```
  const params = {
    DatabaseName: constants.DATABASE_NAME,
    TableName: constants.TABLE_NAME,
    Records: records,
    CommonAttributes: commonAttributes
  };

  const request = writeClient.writeRecords(params);

  await request.promise().then(
    (data) => {
      console.log("Write records successful");
    },
    (err) => {
      console.log("Error writing records:", err);
      if (err.code === 'RejectedRecordsException') {
        const responsePayload =
 JSON.parse(request.response.httpResponse.body.toString());
        console.log("RejectedRecords: ", responsePayload.RejectedRecords);
        console.log("Other records were written successfully. ");
      }
    }
  );
}
```

.NET

```
  public async Task WriteRecordsWithCommonAttributes()
  {
    Console.WriteLine("Writing records with common attributes");

    DateTimeOffset now = DateTimeOffset.UtcNow;
    string currentTimeString = (now.ToUnixTimeMilliseconds()).ToString();

    List<Dimension> dimensions = new List<Dimension>{
      new Dimension { Name = "region", Value = "us-east-1" },
      new Dimension { Name = "az", Value = "az1" },
      new Dimension { Name = "hostname", Value = "host1" }
    };

    var commonAttributes = new Record
    {
      Dimensions = dimensions,
```

```
      MeasureValueType = MeasureValueType.DOUBLE,
      Time = currentTimeString
    };

    var cpuUtilization = new Record
    {
      MeasureName = "cpu_utilization",
      MeasureValue = "13.6"
    };

    var memoryUtilization = new Record
    {
      MeasureName = "memory_utilization",
      MeasureValue = "40"
    };


    List<Record> records = new List<Record>();
    records.Add(cpuUtilization);
    records.Add(memoryUtilization);

    try
    {
      var writeRecordsRequest = new WriteRecordsRequest
      {
        DatabaseName = Constants.DATABASE_NAME,
        TableName = Constants.TABLE_NAME,
        Records = records,
        CommonAttributes = commonAttributes
      };
      WriteRecordsResponse response = await
writeClient.WriteRecordsAsync(writeRecordsRequest);
      Console.WriteLine($"Write records status code:
{response.HttpStatusCode.ToString()}");
    }
    catch (RejectedRecordsException e) {
      Console.WriteLine("RejectedRecordsException:" + e.ToString());
      foreach (RejectedRecord rr in e.RejectedRecords) {
        Console.WriteLine("RecordIndex " + rr.RecordIndex + " : " + rr.Reason);
      }
      Console.WriteLine("Other records were written successfully. ");
    }
    catch (Exception e)
    {
```

```
      Console.WriteLine("Write records failure:" + e.ToString());
    }
  }
```

Alterando los registros

Si bien las escrituras predeterminadas en Amazon Timestream siguen la semántica del primer escritor gana, donde los datos se almacenan solo como datos adjuntos y se rechazan los registros duplicados, hay aplicaciones que requieren la capacidad de escribir datos en Amazon Timestream utilizando la semántica del último escritor gana, donde el registro con la versión más alta se almacena en el sistema. También hay aplicaciones que requieren la capacidad de actualizar los registros existentes. Para abordar estos escenarios, Amazon Timestream ofrece la posibilidad de alterar los datos. Upsert es una operación que inserta un registro en el sistema cuando el registro no existe o lo actualiza, cuando existe.

Puede alterar los registros incluyendo la definición del registro al `Version` enviar una solicitud. `WriteRecords` Amazon Timestream almacenará el registro con el registro más alto. `Version` El siguiente ejemplo de código muestra cómo se pueden alterar los datos:

> **Note**
>
> Estos fragmentos de código se basan en aplicaciones de muestra completas en. GitHub Para obtener más información sobre cómo empezar a utilizar las aplicaciones de ejemplo, consulte. Aplicación de muestra

Java

```
  public void writeRecordsWithUpsert() {
    System.out.println("Writing records with upsert");
    // Specify repeated values for all records
    List<Record> records = new ArrayList<>();
    final long time = System.currentTimeMillis();
    // To achieve upsert (last writer wins) semantic, one example is to use current
 time as the version if you are writing directly from the data source
    long version = System.currentTimeMillis();

    List<Dimension> dimensions = new ArrayList<>();
    final Dimension region = new Dimension().withName("region").withValue("us-
east-1");
```

```
   final Dimension az = new Dimension().withName("az").withValue("az1");
   final Dimension hostname = new
Dimension().withName("hostname").withValue("host1");

   dimensions.add(region);
   dimensions.add(az);
   dimensions.add(hostname);

   Record commonAttributes = new Record()
       .withDimensions(dimensions)
       .withMeasureValueType(MeasureValueType.DOUBLE)
       .withTime(String.valueOf(time))
       .withVersion(version);

   Record cpuUtilization = new Record()
       .withMeasureName("cpu_utilization")
       .withMeasureValue("13.5");
   Record memoryUtilization = new Record()
       .withMeasureName("memory_utilization")
       .withMeasureValue("40");

   records.add(cpuUtilization);
   records.add(memoryUtilization);

   WriteRecordsRequest writeRecordsRequest = new WriteRecordsRequest()
       .withDatabaseName(DATABASE_NAME)
       .withTableName(TABLE_NAME)
       .withCommonAttributes(commonAttributes);
   writeRecordsRequest.setRecords(records);

   // write records for first time
   try {
     WriteRecordsResult writeRecordsResult =
amazonTimestreamWrite.writeRecords(writeRecordsRequest);
     System.out.println("WriteRecords Status for first time: " +
writeRecordsResult.getSdkHttpMetadata().getHttpStatusCode());
   } catch (RejectedRecordsException e) {
     printRejectedRecordsException(e);
   } catch (Exception e) {
     System.out.println("Error: " + e);
   }

   // Successfully retry same writeRecordsRequest with same records and versions,
because writeRecords API is idempotent.
```

```
   try {
     WriteRecordsResult writeRecordsResult =
amazonTimestreamWrite.writeRecords(writeRecordsRequest);
     System.out.println("WriteRecords Status for retry: " +
writeRecordsResult.getSdkHttpMetadata().getHttpStatusCode());
   } catch (RejectedRecordsException e) {
     printRejectedRecordsException(e);
   } catch (Exception e) {
     System.out.println("Error: " + e);
   }

   // upsert with lower version, this would fail because a higher version is
required to update the measure value.
   version -= 1;
   commonAttributes.setVersion(version);

   cpuUtilization.setMeasureValue("14.5");
   memoryUtilization.setMeasureValue("50");

   List<Record> upsertedRecords = new ArrayList<>();
   upsertedRecords.add(cpuUtilization);
   upsertedRecords.add(memoryUtilization);

   WriteRecordsRequest writeRecordsUpsertRequest = new WriteRecordsRequest()
       .withDatabaseName(DATABASE_NAME)
       .withTableName(TABLE_NAME)
       .withCommonAttributes(commonAttributes);
   writeRecordsUpsertRequest.setRecords(upsertedRecords);

   try {
     WriteRecordsResult writeRecordsUpsertResult =
amazonTimestreamWrite.writeRecords(writeRecordsUpsertRequest);
     System.out.println("WriteRecords Status for upsert with lower version: " +
writeRecordsUpsertResult.getSdkHttpMetadata().getHttpStatusCode());
   } catch (RejectedRecordsException e) {
     System.out.println("WriteRecords Status for upsert with lower version: ");
     printRejectedRecordsException(e);
   } catch (Exception e) {
     System.out.println("Error: " + e);
   }

   // upsert with higher version as new data in generated
   version = System.currentTimeMillis();
   commonAttributes.setVersion(version);
```

```
    writeRecordsUpsertRequest = new WriteRecordsRequest()
        .withDatabaseName(DATABASE_NAME)
        .withTableName(TABLE_NAME)
        .withCommonAttributes(commonAttributes);
    writeRecordsUpsertRequest.setRecords(upsertedRecords);

    try {
      WriteRecordsResult writeRecordsUpsertResult =
 amazonTimestreamWrite.writeRecords(writeRecordsUpsertRequest);
      System.out.println("WriteRecords Status for upsert with higher version: " +
 writeRecordsUpsertResult.getSdkHttpMetadata().getHttpStatusCode());
    } catch (RejectedRecordsException e) {
      printRejectedRecordsException(e);
    } catch (Exception e) {
      System.out.println("Error: " + e);
    }
  }
```

Java v2

```
  public void writeRecordsWithUpsert() {
    System.out.println("Writing records with upsert");
    // Specify repeated values for all records
    List<Record> records = new ArrayList<>();
    final long time = System.currentTimeMillis();
    // To achieve upsert (last writer wins) semantic, one example is to use current
 time as the version if you are writing directly from the data source
    long version = System.currentTimeMillis();

    List<Dimension> dimensions = new ArrayList<>();
    final Dimension region = Dimension.builder().name("region").value("us-
east-1").build();
    final Dimension az = Dimension.builder().name("az").value("az1").build();
    final Dimension hostname =
 Dimension.builder().name("hostname").value("host1").build();

    dimensions.add(region);
    dimensions.add(az);
    dimensions.add(hostname);

    Record commonAttributes = Record.builder()
        .dimensions(dimensions)
```

```
        .measureValueType(MeasureValueType.DOUBLE)
        .time(String.valueOf(time))
        .version(version)
        .build();

    Record cpuUtilization = Record.builder()
        .measureName("cpu_utilization")
        .measureValue("13.5").build();
    Record memoryUtilization = Record.builder()
        .measureName("memory_utilization")
        .measureValue("40").build();

    records.add(cpuUtilization);
    records.add(memoryUtilization);

    WriteRecordsRequest writeRecordsRequest = WriteRecordsRequest.builder()
        .databaseName(DATABASE_NAME)
        .tableName(TABLE_NAME)
        .commonAttributes(commonAttributes)
        .records(records).build();

    // write records for first time
    try {
      WriteRecordsResponse writeRecordsResponse =
timestreamWriteClient.writeRecords(writeRecordsRequest);
      System.out.println("WriteRecords Status for first time: " +
writeRecordsResponse.sdkHttpResponse().statusCode());
    } catch (RejectedRecordsException e) {
      printRejectedRecordsException(e);
    } catch (Exception e) {
      System.out.println("Error: " + e);
    }

    // Successfully retry same writeRecordsRequest with same records and versions,
because writeRecords API is idempotent.
    try {
      WriteRecordsResponse writeRecordsResponse =
timestreamWriteClient.writeRecords(writeRecordsRequest);
      System.out.println("WriteRecords Status for retry: " +
writeRecordsResponse.sdkHttpResponse().statusCode());
    } catch (RejectedRecordsException e) {
      printRejectedRecordsException(e);
    } catch (Exception e) {
      System.out.println("Error: " + e);
```

```
    }

    // upsert with lower version, this would fail because a higher version is
 required to update the measure value.
    version -= 1;
    commonAttributes = Record.builder()
        .dimensions(dimensions)
        .measureValueType(MeasureValueType.DOUBLE)
        .time(String.valueOf(time))
        .version(version)
        .build();

    cpuUtilization = Record.builder()
        .measureName("cpu_utilization")
        .measureValue("14.5").build();
    memoryUtilization = Record.builder()
        .measureName("memory_utilization")
        .measureValue("50").build();

    List<Record> upsertedRecords = new ArrayList<>();
    upsertedRecords.add(cpuUtilization);
    upsertedRecords.add(memoryUtilization);

    WriteRecordsRequest writeRecordsUpsertRequest = WriteRecordsRequest.builder()
        .databaseName(DATABASE_NAME)
        .tableName(TABLE_NAME)
        .commonAttributes(commonAttributes)
        .records(upsertedRecords).build();

    try {
      WriteRecordsResponse writeRecordsResponse =
 timestreamWriteClient.writeRecords(writeRecordsUpsertRequest);
      System.out.println("WriteRecords Status for upsert with lower version: " +
 writeRecordsResponse.sdkHttpResponse().statusCode());
    } catch (RejectedRecordsException e) {
      System.out.println("WriteRecords Status for upsert with lower version: ");
      printRejectedRecordsException(e);
    } catch (Exception e) {
      System.out.println("Error: " + e);
    }

    // upsert with higher version as new data in generated
    version = System.currentTimeMillis();
    commonAttributes = Record.builder()
```

```
        .dimensions(dimensions)
        .measureValueType(MeasureValueType.DOUBLE)
        .time(String.valueOf(time))
        .version(version)
        .build();

    writeRecordsUpsertRequest = WriteRecordsRequest.builder()
        .databaseName(DATABASE_NAME)
        .tableName(TABLE_NAME)
        .commonAttributes(commonAttributes)
        .records(upsertedRecords).build();

    try {
      WriteRecordsResponse writeRecordsUpsertResponse =
 timestreamWriteClient.writeRecords(writeRecordsUpsertRequest);
      System.out.println("WriteRecords Status for upsert with higher version: " +
 writeRecordsUpsertResponse.sdkHttpResponse().statusCode());
    } catch (RejectedRecordsException e) {
      printRejectedRecordsException(e);
    } catch (Exception e) {
      System.out.println("Error: " + e);
    }
  }
```

Go

```
// Below code will ingest and upsert cpu_utilization and memory_utilization metric
 for a host on
// region=us-east-1, az=az1, and hostname=host1
fmt.Println("Ingesting records and set version as currentTimeInMills, hit enter to
 continue")
reader.ReadString('\n')

// Get current time in seconds.
now = time.Now()
currentTimeInSeconds = now.Unix()
// To achieve upsert (last writer wins) semantic, one example is to use current time
 as the version if you are writing directly from the data source
version := time.Now().Round(time.Millisecond).UnixNano() / 1e6   // set version as
 currentTimeInMills

writeRecordsCommonAttributesUpsertInput := &timestreamwrite.WriteRecordsInput{
 DatabaseName: aws.String(*databaseName),
```

```
 TableName:  aws.String(*tableName),
 CommonAttributes: &timestreamwrite.Record{
  Dimensions: []*timestreamwrite.Dimension{
   &timestreamwrite.Dimension{
    Name:  aws.String("region"),
    Value: aws.String("us-east-1"),
   },
   &timestreamwrite.Dimension{
    Name:  aws.String("az"),
    Value: aws.String("az1"),
   },
   &timestreamwrite.Dimension{
    Name:  aws.String("hostname"),
    Value: aws.String("host1"),
   },
  },
  MeasureValueType: aws.String("DOUBLE"),
  Time:       aws.String(strconv.FormatInt(currentTimeInSeconds, 10)),
  TimeUnit:  aws.String("SECONDS"),
  Version:      &version,
 },
 Records: []*timestreamwrite.Record{
  &timestreamwrite.Record{
   MeasureName:  aws.String("cpu_utilization"),
   MeasureValue: aws.String("13.5"),
  },
  &timestreamwrite.Record{
   MeasureName:  aws.String("memory_utilization"),
   MeasureValue: aws.String("40"),
  },
 },
}

// write records for first time
_, err = writeSvc.WriteRecords(writeRecordsCommonAttributesUpsertInput)

if err != nil {
 fmt.Println("Error:")
 fmt.Println(err)
} else {
 fmt.Println("Frist-time write records is successful")
}
```

```
fmt.Println("Retry same writeRecordsRequest with same records and versions. Because
 writeRecords API is idempotent, this will success. hit enter to continue")
reader.ReadString('\n')

_, err = writeSvc.WriteRecords(writeRecordsCommonAttributesUpsertInput)

if err != nil {
 fmt.Println("Error:")
 fmt.Println(err)
} else {
 fmt.Println("Retry write records for same request is successful")
}

fmt.Println("Upsert with lower version, this would fail because a higher version is
 required to update the measure value. hit enter to continue")
reader.ReadString('\n')
version -= 1
writeRecordsCommonAttributesUpsertInput.CommonAttributes.Version = &version

updated_cpu_utilization := &timestreamwrite.Record{
 MeasureName:    aws.String("cpu_utilization"),
 MeasureValue:   aws.String("14.5"),
}
updated_memory_utilization := &timestreamwrite.Record{
 MeasureName:    aws.String("memory_utilization"),
 MeasureValue:   aws.String("50"),
}


writeRecordsCommonAttributesUpsertInput.Records = []*timestreamwrite.Record{
 updated_cpu_utilization,
 updated_memory_utilization,
}

_, err = writeSvc.WriteRecords(writeRecordsCommonAttributesUpsertInput)

if err != nil {
 fmt.Println("Error:")
 fmt.Println(err)
} else {
 fmt.Println("Write records with lower version is successful")
}
```

```
fmt.Println("Upsert with higher version as new data in generated, this would
 success. hit enter to continue")
reader.ReadString('\n')

version = time.Now().Round(time.Millisecond).UnixNano() / 1e6  // set version as
 currentTimeInMills
writeRecordsCommonAttributesUpsertInput.CommonAttributes.Version = &version

_, err = writeSvc.WriteRecords(writeRecordsCommonAttributesUpsertInput)

if err != nil {
 fmt.Println("Error:")
 fmt.Println(err)
} else {
 fmt.Println("Write records with higher version is successful")
}
```

Python

```
  def write_records_with_upsert(self):
    print("Writing records with upsert")
    current_time = self._current_milli_time()
    # To achieve upsert (last writer wins) semantic, one example is to use current
 time as the version if you are writing directly from the data source
    version = int(self._current_milli_time())

    dimensions = [
          {'Name': 'region', 'Value': 'us-east-1'},
          {'Name': 'az', 'Value': 'az1'},
          {'Name': 'hostname', 'Value': 'host1'}
        ]

    common_attributes = {
      'Dimensions': dimensions,
      'MeasureValueType': 'DOUBLE',
      'Time': current_time,
      'Version': version
    }

    cpu_utilization = {
      'MeasureName': 'cpu_utilization',
      'MeasureValue': '13.5'
    }
```

```
    memory_utilization = {
      'MeasureName': 'memory_utilization',
      'MeasureValue': '40'
    }

    records = [cpu_utilization, memory_utilization]

    # write records for first time
    try:
      result = self.client.write_records(DatabaseName=Constant.DATABASE_NAME,
 TableName=Constant.TABLE_NAME,
                         Records=records, CommonAttributes=common_attributes)
      print("WriteRecords Status for first time: [%s]" % result['ResponseMetadata']
['HTTPStatusCode'])
    except self.client.exceptions.RejectedRecordsException as err:
      self._print_rejected_records_exceptions(err)
    except Exception as err:
      print("Error:", err)

    # Successfully retry same writeRecordsRequest with same records and versions,
 because writeRecords API is idempotent.
    try:
      result = self.client.write_records(DatabaseName=Constant.DATABASE_NAME,
 TableName=Constant.TABLE_NAME,
                         Records=records, CommonAttributes=common_attributes)
      print("WriteRecords Status for retry: [%s]" % result['ResponseMetadata']
['HTTPStatusCode'])
    except self.client.exceptions.RejectedRecordsException as err:
      self._print_rejected_records_exceptions(err)
    except Exception as err:
      print("Error:", err)

    # upsert with lower version, this would fail because a higher version is
 required to update the measure value.
    version -= 1
    common_attributes["Version"] = version

    cpu_utilization["MeasureValue"] = '14.5'
    memory_utilization["MeasureValue"] = '50'

    upsertedRecords = [cpu_utilization, memory_utilization]

    try:
```

```
      upsertedResult =
 self.client.write_records(DatabaseName=Constant.DATABASE_NAME,
 TableName=Constant.TABLE_NAME,
                           Records=upsertedRecords,
 CommonAttributes=common_attributes)
      print("WriteRecords Status for upsert with lower version: [%s]" %
 upsertedResult['ResponseMetadata']['HTTPStatusCode'])
    except self.client.exceptions.RejectedRecordsException as err:
      self._print_rejected_records_exceptions(err)
    except Exception as err:
      print("Error:", err)


    # upsert with higher version as new data is generated
    version = int(self._current_milli_time())
    common_attributes["Version"] = version

    try:
      upsertedResult =
 self.client.write_records(DatabaseName=Constant.DATABASE_NAME,
 TableName=Constant.TABLE_NAME,
                           Records=upsertedRecords,
 CommonAttributes=common_attributes)
      print("WriteRecords Upsert Status: [%s]" % upsertedResult['ResponseMetadata']
['HTTPStatusCode'])
    except self.client.exceptions.RejectedRecordsException as err:
      self._print_rejected_records_exceptions(err)
    except Exception as err:
      print("Error:", err)

  @staticmethod
  def _current_milli_time():
    return str(int(round(time.time() * 1000)))
```

Node.js

En el siguiente fragmento se utiliza el estilo AWS SDK for JavaScript V2. Se basa en la aplicación de ejemplo de Node.js, ejemplo de Amazon Timestream LiveAnalytics para su aplicación en. GitHub

```
async function writeRecordsWithUpsert() {
  console.log("Writing records with upsert");
  const currentTime = Date.now().toString(); // Unix time in milliseconds
```

```
 // To achieve upsert (last writer wins) semantic, one example is to use current
time as the version if you are writing directly from the data source
 let version = Date.now();

 const dimensions = [
   {'Name': 'region', 'Value': 'us-east-1'},
   {'Name': 'az', 'Value': 'az1'},
   {'Name': 'hostname', 'Value': 'host1'}
 ];

 const commonAttributes = {
   'Dimensions': dimensions,
   'MeasureValueType': 'DOUBLE',
   'Time': currentTime.toString(),
   'Version': version
 };

 const cpuUtilization = {
   'MeasureName': 'cpu_utilization',
   'MeasureValue': '13.5'
 };

 const memoryUtilization = {
   'MeasureName': 'memory_utilization',
   'MeasureValue': '40'
 };

 const records = [cpuUtilization, memoryUtilization];

 const params = {
   DatabaseName: constants.DATABASE_NAME,
   TableName: constants.TABLE_NAME,
   Records: records,
   CommonAttributes: commonAttributes
 };

 const request = writeClient.writeRecords(params);

 // write records for first time
 await request.promise().then(
   (data) => {
     console.log("Write records successful for first time.");
   },
   (err) => {
```

```
      console.log("Error writing records:", err);
      if (err.code === 'RejectedRecordsException') {
        printRejectedRecordsException(request);
      }
    }
  );

  // Successfully retry same writeRecordsRequest with same records and versions,
because writeRecords API is idempotent.
  await request.promise().then(
    (data) => {
      console.log("Write records successful for retry.");
    },
    (err) => {
      console.log("Error writing records:", err);
      if (err.code === 'RejectedRecordsException') {
        printRejectedRecordsException(request);
      }
    }
  );

  // upsert with lower version, this would fail because a higher version is required
to update the measure value.
  version--;

  const commonAttributesWithLowerVersion = {
    'Dimensions': dimensions,
    'MeasureValueType': 'DOUBLE',
    'Time': currentTime.toString(),
    'Version': version
  };

  const updatedCpuUtilization = {
    'MeasureName': 'cpu_utilization',
    'MeasureValue': '14.5'
  };

  const updatedMemoryUtilization = {
    'MeasureName': 'memory_utilization',
    'MeasureValue': '50'
  };

  const upsertedRecords = [updatedCpuUtilization, updatedMemoryUtilization];
```

```
 const upsertedParamsWithLowerVersion = {
   DatabaseName: constants.DATABASE_NAME,
   TableName: constants.TABLE_NAME,
   Records: upsertedRecords,
   CommonAttributes: commonAttributesWithLowerVersion
 };

 const upsertRequestWithLowerVersion =
writeClient.writeRecords(upsertedParamsWithLowerVersion);

 await upsertRequestWithLowerVersion.promise().then(
   (data) => {
     console.log("Write records for upsert with lower version successful");
   },
   (err) => {
     console.log("Error writing records:", err);
     if (err.code === 'RejectedRecordsException') {
       printRejectedRecordsException(upsertRequestWithLowerVersion);
     }
   }
 );

 // upsert with higher version as new data in generated
 version = Date.now();

 const commonAttributesWithHigherVersion = {
   'Dimensions': dimensions,
   'MeasureValueType': 'DOUBLE',
   'Time': currentTime.toString(),
   'Version': version
 };

 const upsertedParamsWithHigherVerion = {
   DatabaseName: constants.DATABASE_NAME,
   TableName: constants.TABLE_NAME,
   Records: upsertedRecords,
   CommonAttributes: commonAttributesWithHigherVersion
 };

 const upsertRequestWithHigherVersion =
writeClient.writeRecords(upsertedParamsWithHigherVerion);

 await upsertRequestWithHigherVersion.promise().then(
   (data) => {
```

```
      console.log("Write records upsert successful with higher version");
    },
    (err) => {
      console.log("Error writing records:", err);
      if (err.code === 'RejectedRecordsException') {
        printRejectedRecordsException(upsertedParamsWithHigherVerion);
      }
    }
  );

}
```

.NET

```
  public async Task WriteRecordsWithUpsert()
  {
    Console.WriteLine("Writing records with upsert");

    DateTimeOffset now = DateTimeOffset.UtcNow;
    string currentTimeString = (now.ToUnixTimeMilliseconds()).ToString();
    // To achieve upsert (last writer wins) semantic, one example is to use current
 time as the version if you are writing directly from the data source
    long version = now.ToUnixTimeMilliseconds();

    List<Dimension> dimensions = new List<Dimension>{
      new Dimension { Name = "region", Value = "us-east-1" },
      new Dimension { Name = "az", Value = "az1" },
      new Dimension { Name = "hostname", Value = "host1" }
    };

    var commonAttributes = new Record
    {
      Dimensions = dimensions,
      MeasureValueType = MeasureValueType.DOUBLE,
      Time = currentTimeString,
      Version = version
    };

    var cpuUtilization = new Record
    {
      MeasureName = "cpu_utilization",
      MeasureValue = "13.6"
    };
```

```
   var memoryUtilization = new Record
   {
     MeasureName = "memory_utilization",
     MeasureValue = "40"
   };


   List<Record> records = new List<Record>();
   records.Add(cpuUtilization);
   records.Add(memoryUtilization);

   // write records for first time
   try
   {
     var writeRecordsRequest = new WriteRecordsRequest
     {
       DatabaseName = Constants.DATABASE_NAME,
       TableName = Constants.TABLE_NAME,
       Records = records,
       CommonAttributes = commonAttributes
     };
     WriteRecordsResponse response = await
writeClient.WriteRecordsAsync(writeRecordsRequest);
     Console.WriteLine($"WriteRecords Status for first time:
{response.HttpStatusCode.ToString()}");
   }
   catch (RejectedRecordsException e) {
     PrintRejectedRecordsException(e);
   }
   catch (Exception e)
   {
     Console.WriteLine("Write records failure:" + e.ToString());
   }

   // Successfully retry same writeRecordsRequest with same records and versions,
because writeRecords API is idempotent.
   try
   {
     var writeRecordsRequest = new WriteRecordsRequest
     {
       DatabaseName = Constants.DATABASE_NAME,
       TableName = Constants.TABLE_NAME,
       Records = records,
```

```
        CommonAttributes = commonAttributes
      };
      WriteRecordsResponse response = await
writeClient.WriteRecordsAsync(writeRecordsRequest);
      Console.WriteLine($"WriteRecords Status for retry:
{response.HttpStatusCode.ToString()}");
    }
    catch (RejectedRecordsException e) {
      PrintRejectedRecordsException(e);
    }
    catch (Exception e)
    {
      Console.WriteLine("Write records failure:" + e.ToString());
    }

    // upsert with lower version, this would fail because a higher version is
required to update the measure value.
    version--;
    Type recordType = typeof(Record);
    recordType.GetProperty("Version").SetValue(commonAttributes, version);
    recordType.GetProperty("MeasureValue").SetValue(cpuUtilization, "14.6");
    recordType.GetProperty("MeasureValue").SetValue(memoryUtilization, "50");

    List<Record> upsertedRecords = new List<Record> {
      cpuUtilization,
      memoryUtilization
    };

    try
    {
      var writeRecordsUpsertRequest = new WriteRecordsRequest
      {
        DatabaseName = Constants.DATABASE_NAME,
        TableName = Constants.TABLE_NAME,
        Records = upsertedRecords,
        CommonAttributes = commonAttributes
      };
      WriteRecordsResponse upsertResponse = await
writeClient.WriteRecordsAsync(writeRecordsUpsertRequest);
      Console.WriteLine($"WriteRecords Status for upsert with lower version:
{upsertResponse.HttpStatusCode.ToString()}");
    }
    catch (RejectedRecordsException e) {
      PrintRejectedRecordsException(e);
```

```
    }
    catch (Exception e)
    {
      Console.WriteLine("Write records failure:" + e.ToString());
    }

    // upsert with higher version as new data in generated
    now = DateTimeOffset.UtcNow;
    version = now.ToUnixTimeMilliseconds();
    recordType.GetProperty("Version").SetValue(commonAttributes, version);

    try
    {
      var writeRecordsUpsertRequest = new WriteRecordsRequest
      {
        DatabaseName = Constants.DATABASE_NAME,
        TableName = Constants.TABLE_NAME,
        Records = upsertedRecords,
        CommonAttributes = commonAttributes
      };
      WriteRecordsResponse upsertResponse = await
writeClient.WriteRecordsAsync(writeRecordsUpsertRequest);
      Console.WriteLine($"WriteRecords Status for upsert with higher version:
{upsertResponse.HttpStatusCode.ToString()}");
    }
    catch (RejectedRecordsException e) {
      PrintRejectedRecordsException(e);
    }
    catch (Exception e)
    {
      Console.WriteLine("Write records failure:" + e.ToString());
    }
  }
```

Ejemplo de atributo de medidas múltiples

En este ejemplo, se ilustra la escritura de atributos de múltiples mediciones. Los atributos de medición múltiple son útiles cuando un dispositivo o una aplicación que se está rastreando emite varias métricas o eventos al mismo tiempo.

> **Note**
>
> Estos fragmentos de código se basan en ejemplos completos de aplicaciones. GitHub
> Para obtener más información sobre cómo empezar a utilizar las aplicaciones de ejemplo, consulte. Aplicación de muestra

Java

```
package com.amazonaws.services.timestream;

import static com.amazonaws.services.timestream.Main.DATABASE_NAME;
import static com.amazonaws.services.timestream.Main.REGION;
import static com.amazonaws.services.timestream.Main.TABLE_NAME;

import java.util.ArrayList;
import java.util.List;

import com.amazonaws.services.timestreamwrite.AmazonTimestreamWrite;
import com.amazonaws.services.timestreamwrite.model.Dimension;
import com.amazonaws.services.timestreamwrite.model.MeasureValue;
import com.amazonaws.services.timestreamwrite.model.MeasureValueType;
import com.amazonaws.services.timestreamwrite.model.Record;
import com.amazonaws.services.timestreamwrite.model.RejectedRecordsException;
import com.amazonaws.services.timestreamwrite.model.WriteRecordsRequest;
import com.amazonaws.services.timestreamwrite.model.WriteRecordsResult;


public class multimeasureAttributeExample {
  AmazonTimestreamWrite timestreamWriteClient;

  public multimeasureAttributeExample(AmazonTimestreamWrite client) {
    this.timestreamWriteClient = client;
  }

  public void writeRecordsMultiMeasureValueSingleRecord() {
    System.out.println("Writing records with multi value attributes");

    List<Record> records = new ArrayList<>();
    final long time = System.currentTimeMillis();
    long version = System.currentTimeMillis();
```

```
    List<Dimension> dimensions = new ArrayList<>();
    final Dimension region = new Dimension().withName("region").withValue(REGION);
    final Dimension az = new Dimension().withName("az").withValue("az1");
    final Dimension hostname = new
Dimension().withName("hostname").withValue("host1");

    dimensions.add(region);
    dimensions.add(az);
    dimensions.add(hostname);

    Record commonAttributes = new Record()
        .withDimensions(dimensions)
        .withTime(String.valueOf(time))
        .withVersion(version);

    MeasureValue cpuUtilization = new MeasureValue()
        .withName("cpu_utilization")
        .withType(MeasureValueType.DOUBLE)
        .withValue("13.5");
    MeasureValue memoryUtilization = new MeasureValue()
        .withName("memory_utilization")
        .withType(MeasureValueType.DOUBLE)
        .withValue("40");
    Record computationalResources = new Record()
        .withMeasureName("cpu_memory")
        .withMeasureValues(cpuUtilization, memoryUtilization)
        .withMeasureValueType(MeasureValueType.MULTI);

    records.add(computationalResources);

    WriteRecordsRequest writeRecordsRequest = new WriteRecordsRequest()
        .withDatabaseName(DATABASE_NAME)
        .withTableName(TABLE_NAME)
        .withCommonAttributes(commonAttributes)
        .withRecords(records);

    // write records for first time
    try {
      WriteRecordsResult writeRecordResult =
timestreamWriteClient.writeRecords(writeRecordsRequest);
     System.out.println(
         "WriteRecords Status for multi value attributes: " + writeRecordResult
             .getSdkHttpMetadata().getHttpStatusCode());
    } catch (RejectedRecordsException e) {
```

```
      printRejectedRecordsException(e);
    } catch (Exception e) {
      System.out.println("Error: " + e);
    }
  }

  public void writeRecordsMultiMeasureValueMultipleRecords() {
    System.out.println(
        "Writing records with multi value attributes mixture type");

    List<Record> records = new ArrayList<>();
    final long time = System.currentTimeMillis();
    long version = System.currentTimeMillis();

    List<Dimension> dimensions = new ArrayList<>();
    final Dimension region = new Dimension().withName("region").withValue(REGION);
    final Dimension az = new Dimension().withName("az").withValue("az1");
    final Dimension hostname = new
Dimension().withName("hostname").withValue("host1");

    dimensions.add(region);
    dimensions.add(az);
    dimensions.add(hostname);

    Record commonAttributes = new Record()
        .withDimensions(dimensions)
        .withTime(String.valueOf(time))
        .withVersion(version);

    MeasureValue cpuUtilization = new MeasureValue()
        .withName("cpu_utilization")
        .withType(MeasureValueType.DOUBLE)
        .withValue("13");
    MeasureValue memoryUtilization =new MeasureValue()
        .withName("memory_utilization")
        .withType(MeasureValueType.DOUBLE)
        .withValue("40");
    MeasureValue activeCores = new MeasureValue()
        .withName("active_cores")
        .withType(MeasureValueType.BIGINT)
        .withValue("4");


    Record computationalResources = new Record()
```

```
        .withMeasureName("computational_utilization")
        .withMeasureValues(cpuUtilization, memoryUtilization, activeCores)
        .withMeasureValueType(MeasureValueType.MULTI);

    records.add(computationalResources);

    WriteRecordsRequest writeRecordsRequest = new WriteRecordsRequest()
        .withDatabaseName(DATABASE_NAME)
        .withTableName(TABLE_NAME)
        .withCommonAttributes(commonAttributes)
        .withRecords(records);

    // write records for first time
    try {
      WriteRecordsResult writeRecordResult =
 timestreamWriteClient.writeRecords(writeRecordsRequest);
      System.out.println(
          "WriteRecords Status for multi value attributes: " + writeRecordResult
              .getSdkHttpMetadata().getHttpStatusCode());
    } catch (RejectedRecordsException e) {
      printRejectedRecordsException(e);
    } catch (Exception e) {
      System.out.println("Error: " + e);
    }
  }

  private void printRejectedRecordsException(RejectedRecordsException e) {
    System.out.println("RejectedRecords: " + e);
    e.getRejectedRecords().forEach(System.out::println);
  }
}
```

Java v2

```
package com.amazonaws.services.timestream;

import java.util.ArrayList;
import java.util.List;

import software.amazon.awssdk.services.timestreamwrite.TimestreamWriteClient;
import software.amazon.awssdk.services.timestreamwrite.model.Dimension;
import software.amazon.awssdk.services.timestreamwrite.model.MeasureValue;
import software.amazon.awssdk.services.timestreamwrite.model.MeasureValueType;
```

```
import software.amazon.awssdk.services.timestreamwrite.model.Record;
import
 software.amazon.awssdk.services.timestreamwrite.model.RejectedRecordsException;
import software.amazon.awssdk.services.timestreamwrite.model.WriteRecordsRequest;
import software.amazon.awssdk.services.timestreamwrite.model.WriteRecordsResponse;

import static com.amazonaws.services.timestream.Main.DATABASE_NAME;
import static com.amazonaws.services.timestream.Main.TABLE_NAME;


public class multimeasureAttributeExample {

  TimestreamWriteClient timestreamWriteClient;

  public multimeasureAttributeExample(TimestreamWriteClient client) {
    this.timestreamWriteClient = client;
  }

  public void writeRecordsMultiMeasureValueSingleRecord() {
    System.out.println("Writing records with multi value attributes");

    List<Record> records = new ArrayList<>();
    final long time = System.currentTimeMillis();
    long version = System.currentTimeMillis();

    List<Dimension> dimensions = new ArrayList<>();
    final Dimension region =
        Dimension.builder().name("region").value("us-east-1").build();
    final Dimension az = Dimension.builder().name("az").value("az1").build();
    final Dimension hostname =
        Dimension.builder().name("hostname").value("host1").build();

    dimensions.add(region);
    dimensions.add(az);
    dimensions.add(hostname);

    Record commonAttributes = Record.builder()
        .dimensions(dimensions)
        .time(String.valueOf(time))
        .version(version)
        .build();

    MeasureValue cpuUtilization = MeasureValue.builder()
        .name("cpu_utilization")
```

```
        .type(MeasureValueType.DOUBLE)
        .value("13.5").build();
    MeasureValue memoryUtilization = MeasureValue.builder()
        .name("memory_utilization")
        .type(MeasureValueType.DOUBLE)
        .value("40").build();
    Record computationalResources = Record
        .builder()
        .measureName("cpu_memory")
        .measureValues(cpuUtilization, memoryUtilization)
        .measureValueType(MeasureValueType.MULTI)
        .build();

    records.add(computationalResources);

    WriteRecordsRequest writeRecordsRequest = WriteRecordsRequest.builder()
        .databaseName(DATABASE_NAME)
        .tableName(TABLE_NAME)
        .commonAttributes(commonAttributes)
        .records(records).build();

    // write records for first time
    try {
      WriteRecordsResponse writeRecordsResponse =
timestreamWriteClient.writeRecords(writeRecordsRequest);
      System.out.println(
          "WriteRecords Status for multi value attributes: " + writeRecordsResponse
              .sdkHttpResponse()
              .statusCode());
    } catch (RejectedRecordsException e) {
      printRejectedRecordsException(e);
    } catch (Exception e) {
      System.out.println("Error: " + e);
    }
  }

  public void writeRecordsMultiMeasureValueMultipleRecords() {
    System.out.println(
        "Writing records with multi value attributes mixture type");

    List<Record> records = new ArrayList<>();
    final long time = System.currentTimeMillis();
    long version = System.currentTimeMillis();
```

```
    List<Dimension> dimensions = new ArrayList<>();
    final Dimension region =
        Dimension.builder().name("region").value("us-east-1").build();
    final Dimension az = Dimension.builder().name("az").value("az1").build();
    final Dimension hostname =
        Dimension.builder().name("hostname").value("host1").build();

    dimensions.add(region);
    dimensions.add(az);
    dimensions.add(hostname);

    Record commonAttributes = Record.builder()
        .dimensions(dimensions)
        .time(String.valueOf(time))
        .version(version)
        .build();

    MeasureValue cpuUtilization = MeasureValue.builder()
        .name("cpu_utilization")
        .type(MeasureValueType.DOUBLE)
        .value("13.5").build();
    MeasureValue memoryUtilization = MeasureValue.builder()
        .name("memory_utilization")
        .type(MeasureValueType.DOUBLE)
        .value("40").build();
    MeasureValue activeCores = MeasureValue.builder()
        .name("active_cores")
        .type(MeasureValueType.BIGINT)
        .value("4").build();


    Record computationalResources = Record
        .builder()
        .measureName("computational_utilization")
        .measureValues(cpuUtilization, memoryUtilization, activeCores)
        .measureValueType(MeasureValueType.MULTI)
        .build();

    records.add(computationalResources);

    WriteRecordsRequest writeRecordsRequest = WriteRecordsRequest.builder()
        .databaseName(DATABASE_NAME)
        .tableName(TABLE_NAME)
        .commonAttributes(commonAttributes)
```

```
        .records(records).build();

    // write records for first time
    try {
      WriteRecordsResponse writeRecordsResponse =
 timestreamWriteClient.writeRecords(writeRecordsRequest);
      System.out.println(
          "WriteRecords Status for multi value attributes: " + writeRecordsResponse
              .sdkHttpResponse()
              .statusCode());
    } catch (RejectedRecordsException e) {
      printRejectedRecordsException(e);
    } catch (Exception e) {
      System.out.println("Error: " + e);
    }
  }

  private void printRejectedRecordsException(RejectedRecordsException e) {
    System.out.println("RejectedRecords: " + e);
    e.rejectedRecords().forEach(System.out::println);
  }
}
```

Go

```
  now := time.Now()
  currentTimeInSeconds := now.Unix()
  writeRecordsInput := &timestreamwrite.WriteRecordsInput{
    DatabaseName: aws.String(*databaseName),
    TableName:  aws.String(*tableName),
    Records: []*timestreamwrite.Record{
    &timestreamwrite.Record{
      Dimensions: []*timestreamwrite.Dimension{
      &timestreamwrite.Dimension{
        Name:  aws.String("region"),
        Value: aws.String("us-east-1"),
      },
      &timestreamwrite.Dimension{
        Name:  aws.String("az"),
        Value: aws.String("az1"),
      },
      &timestreamwrite.Dimension{
        Name:  aws.String("hostname"),
```

```
        Value: aws.String("host1"),
      },
      },
      MeasureName:  aws.String("metrics"),
      MeasureValueType: aws.String("MULTI"),
      Time:     aws.String(strconv.FormatInt(currentTimeInSeconds, 10)),
      TimeUnit:  aws.String("SECONDS"),
      MeasureValues: []*timestreamwrite.MeasureValue{
      &timestreamwrite.MeasureValue{
        Name:  aws.String("cpu_utilization"),
        Value: aws.String("13.5"),
        Type:  aws.String("DOUBLE"),
      },
      &timestreamwrite.MeasureValue{
        Name:  aws.String("memory_utilization"),
        Value: aws.String("40"),
        Type:  aws.String("DOUBLE"),
      },
      },
    },
    },
  }

  _, err = writeSvc.WriteRecords(writeRecordsInput)

  if err != nil {
    fmt.Println("Error:")
    fmt.Println(err)
  } else {
    fmt.Println("Write records is successful")
  }
```

Python

```
import time
import boto3
import psutil
import os

from botocore.config import Config

DATABASE_NAME = os.environ['DATABASE_NAME']
TABLE_NAME = os.environ['TABLE_NAME']
```

```
COUNTRY = "UK"
CITY = "London"
HOSTNAME = "MyHostname" # You can make it dynamic using socket.gethostname()

INTERVAL = 1 # Seconds

def prepare_common_attributes():
  common_attributes = {
    'Dimensions': [
      {'Name': 'country', 'Value': COUNTRY},
      {'Name': 'city', 'Value': CITY},
      {'Name': 'hostname', 'Value': HOSTNAME}
    ],
    'MeasureName': 'utilization',
    'MeasureValueType': 'MULTI'
  }
  return common_attributes


def prepare_record(current_time):
  record = {
    'Time': str(current_time),
    'MeasureValues': []
  }
  return record


def prepare_measure(measure_name, measure_value):
  measure = {
    'Name': measure_name,
    'Value': str(measure_value),
    'Type': 'DOUBLE'
  }
  return measure


def write_records(records, common_attributes):
  try:
    result = write_client.write_records(DatabaseName=DATABASE_NAME,
                                        TableName=TABLE_NAME,
                                        CommonAttributes=common_attributes,
                                        Records=records)
    status = result['ResponseMetadata']['HTTPStatusCode']
```

```
    print("Processed %d records. WriteRecords HTTPStatusCode: %s" %
        (len(records), status))
  except Exception as err:
    print("Error:", err)


if __name__ == '__main__':

  print("writing data to database {} table {}".format(
    DATABASE_NAME, TABLE_NAME))

  session = boto3.Session()
  write_client = session.client('timestream-write', config=Config(
    read_timeout=20, max_pool_connections=5000, retries={'max_attempts': 10}))
  query_client = session.client('timestream-query') # Not used

  common_attributes = prepare_common_attributes()

  records = []

  while True:

    current_time = int(time.time() * 1000)
    cpu_utilization = psutil.cpu_percent()
    memory_utilization = psutil.virtual_memory().percent
    swap_utilization = psutil.swap_memory().percent
    disk_utilization = psutil.disk_usage('/').percent

    record = prepare_record(current_time)
    record['MeasureValues'].append(prepare_measure('cpu', cpu_utilization))
    record['MeasureValues'].append(prepare_measure('memory', memory_utilization))
    record['MeasureValues'].append(prepare_measure('swap', swap_utilization))
    record['MeasureValues'].append(prepare_measure('disk', disk_utilization))

    records.append(record)

    print("records {} - cpu {} - memory {} - swap {} - disk {}".format(
      len(records), cpu_utilization, memory_utilization,
      swap_utilization, disk_utilization))

    if len(records) == 100:
      write_records(records, common_attributes)
      records = []
```

```
    time.sleep(INTERVAL)
```

Node.js

En el siguiente fragmento se utiliza el estilo AWS SDK for JavaScript V2. Se basa en la aplicación de ejemplo de Node.js, ejemplo de Amazon Timestream LiveAnalytics para su aplicación en. GitHub

```
async function writeRecords() {
  console.log("Writing records");
  const currentTime = Date.now().toString(); // Unix time in milliseconds

  const dimensions = [
  {'Name': 'region', 'Value': 'us-east-1'},
  {'Name': 'az', 'Value': 'az1'},
  {'Name': 'hostname', 'Value': 'host1'}
  ];

  const record = {
  'Dimensions': dimensions,
  'MeasureName': 'metrics',
  'MeasureValues': [
    {
      'Name': 'cpu_utilization',
      'Value': '40',
      'Type': 'DOUBLE',
    },
    {
      'Name': 'memory_utilization',
      'Value': '13.5',
      'Type': 'DOUBLE',
    },
    ],
    'MeasureValueType': 'MULTI',
    'Time': currentTime.toString()
  }

  const records = [record];

  const params = {
  DatabaseName: 'DatabaseName',
  TableName: 'TableName',
  Records: records
```

```
    };

    const response = await writeClient.writeRecords(params);

    console.log(response);
  }
```

.NET

```
using System;
using System.IO;
using System.Collections.Generic;
using Amazon.TimestreamWrite;
using Amazon.TimestreamWrite.Model;
using System.Threading.Tasks;

namespace TimestreamDotNetSample
{
  static class MultiMeasureValueConstants
  {
    public const string MultiMeasureValueSampleDb = "multiMeasureValueSampleDb";
    public const string MultiMeasureValueSampleTable =
 "multiMeasureValueSampleTable";
  }

  public class MultiValueAttributesExample
  {
    private readonly AmazonTimestreamWriteClient writeClient;

    public MultiValueAttributesExample(AmazonTimestreamWriteClient writeClient)
    {
      this.writeClient = writeClient;
    }

    public async Task WriteRecordsMultiMeasureValueSingleRecord()
    {
      Console.WriteLine("Writing records with multi value attributes");

      DateTimeOffset now = DateTimeOffset.UtcNow;
      string currentTimeString = (now.ToUnixTimeMilliseconds()).ToString();

      List<Dimension> dimensions = new List<Dimension>{
        new Dimension { Name = "region", Value = "us-east-1" },
```

```
        new Dimension { Name = "az", Value = "az1" },
        new Dimension { Name = "hostname", Value = "host1" }
      };

      var commonAttributes = new Record
      {
        Dimensions = dimensions,
        Time = currentTimeString
      };

      var cpuUtilization = new MeasureValue
      {
        Name = "cpu_utilization",
        Value = "13.6",
        Type = "DOUBLE"
      };

      var memoryUtilization = new MeasureValue
      {
        Name = "memory_utilization",
        Value = "40",
        Type = "DOUBLE"
      };

      var computationalRecord = new Record
      {
        MeasureName = "cpu_memory",
        MeasureValues = new List<MeasureValue> {cpuUtilization, memoryUtilization},
        MeasureValueType = "MULTI"
      };


      List<Record> records = new List<Record>();
      records.Add(computationalRecord);

      try
      {
        var writeRecordsRequest = new WriteRecordsRequest
        {
          DatabaseName = MultiMeasureValueConstants.MultiMeasureValueSampleDb,
          TableName = MultiMeasureValueConstants.MultiMeasureValueSampleTable,
          Records = records,
          CommonAttributes = commonAttributes
        };
```

```
        WriteRecordsResponse response = await
writeClient.WriteRecordsAsync(writeRecordsRequest);
        Console.WriteLine($"Write records status code:
{response.HttpStatusCode.ToString()}");
      }
      catch (Exception e)
      {
        Console.WriteLine("Write records failure:" + e.ToString());
      }
    }

    public async Task WriteRecordsMultiMeasureValueMultipleRecords()
    {
      Console.WriteLine("Writing records with multi value attributes mixture type");

      DateTimeOffset now = DateTimeOffset.UtcNow;
      string currentTimeString = (now.ToUnixTimeMilliseconds()).ToString();

      List<Dimension> dimensions = new List<Dimension>{
        new Dimension { Name = "region", Value = "us-east-1" },
        new Dimension { Name = "az", Value = "az1" },
        new Dimension { Name = "hostname", Value = "host1" }
      };

      var commonAttributes = new Record
      {
        Dimensions = dimensions,
        Time = currentTimeString
      };

      var cpuUtilization = new MeasureValue
      {
        Name = "cpu_utilization",
        Value = "13.6",
        Type = "DOUBLE"
      };

      var memoryUtilization = new MeasureValue
      {
        Name = "memory_utilization",
        Value = "40",
        Type = "DOUBLE"
      };
```

```
      var activeCores = new MeasureValue
      {
        Name = "active_cores",
        Value = "4",
        Type = "BIGINT"
      };

      var computationalRecord = new Record
      {
        MeasureName = "computational_utilization",
        MeasureValues = new List<MeasureValue> {cpuUtilization, memoryUtilization,
activeCores},
        MeasureValueType = "MULTI"
      };

      var aliveRecord = new Record
      {
        MeasureName = "is_healthy",
        MeasureValue = "true",
        MeasureValueType = "BOOLEAN"
      };

      List<Record> records = new List<Record>();
      records.Add(computationalRecord);
      records.Add(aliveRecord);

      try
      {
        var writeRecordsRequest = new WriteRecordsRequest
        {
          DatabaseName = MultiMeasureValueConstants.MultiMeasureValueSampleDb,
          TableName = MultiMeasureValueConstants.MultiMeasureValueSampleTable,
          Records = records,
          CommonAttributes = commonAttributes
        };
        WriteRecordsResponse response = await
writeClient.WriteRecordsAsync(writeRecordsRequest);
        Console.WriteLine($"Write records status code:
{response.HttpStatusCode.ToString()}");
      }
      catch (Exception e)
      {
        Console.WriteLine("Write records failure:" + e.ToString());
      }
```

```
    }
  }
}
```

Manejo de errores de escritura

Las escrituras en Amazon Timestream pueden fallar por uno o varios de los siguientes motivos:

- Hay registros con marcas de tiempo que se encuentran fuera del período de retención del almacén de memoria.
- Hay registros que contienen dimensiones o medidas que superan los límites definidos por Timestream.
- Amazon Timestream ha detectado registros duplicados. Los registros se marcan como duplicados cuando hay varios registros con las mismas dimensiones, marcas de tiempo y nombres de medidas, pero:
  - Los valores de las medidas son diferentes.
  - La versión no está presente en la solicitud o el valor de la versión en el nuevo registro es igual o inferior al valor existente. Si Amazon Timestream rechaza los datos por este motivo, `ExistingVersion` el campo contendrá `RejectedRecords` la versión actual del registro almacenada en Amazon Timestream. Para forzar una actualización, puede volver a enviar la solicitud con una versión del registro establecida en un valor superior a. `ExistingVersion`

Para obtener más información sobre los errores y los registros rechazados, consulte Errores y RejectedRecord.

Si su aplicación recibe un mensaje `RejectedRecordsException` al intentar escribir registros en Timestream, puede analizar los registros rechazados para obtener más información sobre los errores de escritura, tal y como se muestra a continuación.

> **Note**
>
> Estos fragmentos de código se basan en ejemplos completos de aplicaciones en. GitHub Para obtener más información sobre cómo empezar a utilizar las aplicaciones de ejemplo, consulte. Aplicación de muestra

Java

```
 try {
   WriteRecordsResult writeRecordsResult =
amazonTimestreamWrite.writeRecords(writeRecordsRequest);
   System.out.println("WriteRecords Status: " +
writeRecordsResult.getSdkHttpMetadata().getHttpStatusCode());
 } catch (RejectedRecordsException e) {
   System.out.println("RejectedRecords: " + e);
   for (RejectedRecord rejectedRecord : e.getRejectedRecords()) {
     System.out.println("Rejected Index " + rejectedRecord.getRecordIndex() + ": "
         + rejectedRecord.getReason());
   }
   System.out.println("Other records were written successfully. ");
 } catch (Exception e) {
   System.out.println("Error: " + e);
 }
```

Java v2

```
   try {
     WriteRecordsResponse writeRecordsResponse =
timestreamWriteClient.writeRecords(writeRecordsRequest);
     System.out.println("writeRecordsWithCommonAttributes Status: " +
writeRecordsResponse.sdkHttpResponse().statusCode());
   } catch (RejectedRecordsException e) {
     System.out.println("RejectedRecords: " + e);
     for (RejectedRecord rejectedRecord : e.rejectedRecords()) {
       System.out.println("Rejected Index " + rejectedRecord.recordIndex() + ": "
           + rejectedRecord.reason());
     }
     System.out.println("Other records were written successfully. ");
   } catch (Exception e) {
     System.out.println("Error: " + e);
   }
```

Go

```
_, err = writeSvc.WriteRecords(writeRecordsInput)

if err != nil {
  fmt.Println("Error:")
  fmt.Println(err)
```

```
} else {
  fmt.Println("Write records is successful")
}
```

Python

```
try:
  result = self.client.write_records(DatabaseName=Constant.DATABASE_NAME,
 TableName=Constant.TABLE_NAME, Records=records, CommonAttributes=common_attributes)
  print("WriteRecords Status: [%s]" % result['ResponseMetadata']['HTTPStatusCode'])
except self.client.exceptions.RejectedRecordsException as err:
  print("RejectedRecords: ", err)
  for rr in err.response["RejectedRecords"]:
    print("Rejected Index " + str(rr["RecordIndex"]) + ": " + rr["Reason"])
  print("Other records were written successfully. ")
except Exception as err:
  print("Error:", err)
```

Node.js

En el siguiente fragmento se utiliza el estilo AWS SDK for JavaScript V2. Se basa en la aplicación de ejemplo de Node.js, ejemplo de Amazon Timestream LiveAnalytics para su aplicación en. GitHub

```
await request.promise().then(
    (data) => {
      console.log("Write records successful");
    },
    (err) => {
      console.log("Error writing records:", err);
      if (err.code === 'RejectedRecordsException') {
        const responsePayload =
 JSON.parse(request.response.httpResponse.body.toString());
        console.log("RejectedRecords: ", responsePayload.RejectedRecords);
        console.log("Other records were written successfully. ");
      }
    }
  );
```

.NET

```
  try
```

```
 {
   var writeRecordsRequest = new WriteRecordsRequest
   {
     DatabaseName = Constants.DATABASE_NAME,
     TableName = Constants.TABLE_NAME,
     Records = records,
     CommonAttributes = commonAttributes
   };
   WriteRecordsResponse response = await
writeClient.WriteRecordsAsync(writeRecordsRequest);
   Console.WriteLine($"Write records status code:
{response.HttpStatusCode.ToString()}");
 }
 catch (RejectedRecordsException e) {
   Console.WriteLine("RejectedRecordsException:" + e.ToString());
   foreach (RejectedRecord rr in e.RejectedRecords) {
     Console.WriteLine("RecordIndex " + rr.RecordIndex + " : " + rr.Reason);
   }
   Console.WriteLine("Other records were written successfully. ");
 }
 catch (Exception e)
 {
   Console.WriteLine("Write records failure:" + e.ToString());
 }
```

Ejecutar consulta

Temas

- Paginación de resultados
- Analizar conjuntos de resultados
- Acceder al estado de la consulta

Paginación de resultados

Al ejecutar una consulta, Timestream devuelve el conjunto de resultados de forma paginada para optimizar la capacidad de respuesta de las aplicaciones. El siguiente fragmento de código muestra cómo se puede paginar el conjunto de resultados. Debe recorrer todas las páginas del conjunto de resultados hasta encontrar un valor nulo. Los tokens de paginación caducan a las 3 horas de haber sido emitidos por Timestream para. LiveAnalytics

> **Note**
>
> Estos fragmentos de código se basan en aplicaciones de muestra completas en. GitHub Para obtener más información sobre cómo empezar a utilizar las aplicaciones de ejemplo, consulte. Aplicación de muestra

Java

```
   private void runQuery(String queryString) {
       try {
           QueryRequest queryRequest = new QueryRequest();
           queryRequest.setQueryString(queryString);
           QueryResult queryResult = queryClient.query(queryRequest);
           while (true) {
               parseQueryResult(queryResult);
               if (queryResult.getNextToken() == null) {
                   break;
               }
               queryRequest.setNextToken(queryResult.getNextToken());
               queryResult = queryClient.query(queryRequest);
           }
       } catch (Exception e) {
           // Some queries might fail with 500 if the result of a sequence function
has more than 10000 entries
           e.printStackTrace();
       }
   }
```

Java v2

```
   private void runQuery(String queryString) {
       try {
           QueryRequest queryRequest =
QueryRequest.builder().queryString(queryString).build();
           final QueryIterable queryResponseIterator =
timestreamQueryClient.queryPaginator(queryRequest);
           for(QueryResponse queryResponse : queryResponseIterator) {
               parseQueryResult(queryResponse);
           }
       } catch (Exception e) {
```

```
            // Some queries might fail with 500 if the result of a sequence function
 has more than 10000 entries
            e.printStackTrace();
        }
    }
```

Go

```
func runQuery(queryPtr *string, querySvc *timestreamquery.TimestreamQuery, f
 *os.File) {
    queryInput := &timestreamquery.QueryInput{
        QueryString: aws.String(*queryPtr),
    }
    fmt.Println("QueryInput:")
    fmt.Println(queryInput)
    // execute the query
    err := querySvc.QueryPages(queryInput,
        func(page *timestreamquery.QueryOutput, lastPage bool) bool {
            // process query response
            queryStatus := page.QueryStatus
            fmt.Println("Current query status:", queryStatus)
            // query response metadata
            // includes column names and types
            metadata := page.ColumnInfo
            // fmt.Println("Metadata:")
            fmt.Println(metadata)
            header := ""
            for i := 0; i < len(metadata); i++ {
                header += *metadata[i].Name
                if i != len(metadata)-1 {
                    header += ", "
                }
            }
            write(f, header)

            // query response data
            fmt.Println("Data:")
            // process rows
            rows := page.Rows
            for i := 0; i < len(rows); i++ {
                data := rows[i].Data
                value := processRowType(data, metadata)
                fmt.Println(value)
```

```
                write(f, value)
            }
            fmt.Println("Number of rows:", len(page.Rows))
            return true
        })
    if err != nil {
        fmt.Println("Error:")
        fmt.Println(err)
    }
}
```

Python

```
    def run_query(self, query_string):
        try:
            page_iterator = self.paginator.paginate(QueryString=query_string)
            for page in page_iterator:
                self._parse_query_result(page)
        except Exception as err:
            print("Exception while running query:", err)
```

Node.js

En el siguiente fragmento se utiliza el estilo AWS SDK for JavaScript V2. Se basa en la aplicación de ejemplo de Node.js, ejemplo de Amazon Timestream LiveAnalytics para su aplicación en. GitHub

```
async function getAllRows(query, nextToken) {
    const params = {
        QueryString: query
    };

    if (nextToken) {
        params.NextToken = nextToken;
    }

    await queryClient.query(params).promise()
        .then(
            (response) => {
                parseQueryResult(response);
                if (response.NextToken) {
                    getAllRows(query, response.NextToken);
```

```
                }
            },
            (err) => {
                console.error("Error while querying:", err);
            });
}
```

.NET

```
        private async Task RunQueryAsync(string queryString)
        {
            try
            {
                QueryRequest queryRequest = new QueryRequest();
                queryRequest.QueryString = queryString;
                QueryResponse queryResponse = await
 queryClient.QueryAsync(queryRequest);
                while (true)
                {
                    ParseQueryResult(queryResponse);
                    if (queryResponse.NextToken == null)
                    {
                        break;
                    }
                    queryRequest.NextToken = queryResponse.NextToken;
                    queryResponse = await queryClient.QueryAsync(queryRequest);
                }
            } catch(Exception e)
            {
                // Some queries might fail with 500 if the result of a sequence
 function has more than 10000 entries
                Console.WriteLine(e.ToString());
            }
        }
```

Analizar conjuntos de resultados

Puede utilizar los siguientes fragmentos de código para extraer datos del conjunto de resultados. Se puede acceder a los resultados de la consulta durante un máximo de 24 horas después de que se complete la consulta.

> **Note**
>
> Estos fragmentos de código se basan en aplicaciones de muestra completas en. GitHub Para obtener más información sobre cómo empezar a utilizar las aplicaciones de ejemplo, consulte. Aplicación de muestra

Java

```
   private static final DateTimeFormatter TIMESTAMP_FORMATTER =
DateTimeFormatter.ofPattern("yyyy-MM-dd HH:mm:ss.SSSSSSSSS");
   private static final DateTimeFormatter DATE_FORMATTER =
DateTimeFormatter.ofPattern("yyyy-MM-dd");
   private static final DateTimeFormatter TIME_FORMATTER =
DateTimeFormatter.ofPattern("HH:mm:ss.SSSSSSSSS");

   private static final long ONE_GB_IN_BYTES = 1073741824L;

   private void parseQueryResult(QueryResult response) {
       final QueryStatus currentStatusOfQuery = queryResult.getQueryStatus();

       System.out.println("Query progress so far: " +
currentStatusOfQuery.getProgressPercentage() + "%");

       double bytesScannedSoFar = ((double)
currentStatusOfQuery.getCumulativeBytesScanned() / ONE_GB_IN_BYTES);
       System.out.println("Bytes scanned so far: " + bytesScannedSoFar + " GB");

       double bytesMeteredSoFar = ((double)
currentStatusOfQuery.getCumulativeBytesMetered() / ONE_GB_IN_BYTES);
       System.out.println("Bytes metered so far: " + bytesMeteredSoFar + " GB");

       List<ColumnInfo> columnInfo = response.getColumnInfo();
       List<Row> rows = response.getRows();

       System.out.println("Metadata: " + columnInfo);
       System.out.println("Data: ");

       // iterate every row
       for (Row row : rows) {
           System.out.println(parseRow(columnInfo, row));
       }
```

```
    }

    private String parseRow(List<ColumnInfo> columnInfo, Row row) {
        List<Datum> data = row.getData();
        List<String> rowOutput = new ArrayList<>();
        // iterate every column per row
        for (int j = 0; j < data.size(); j++) {
            ColumnInfo info = columnInfo.get(j);
            Datum datum = data.get(j);
            rowOutput.add(parseDatum(info, datum));
        }
        return String.format("{%s}",
rowOutput.stream().map(Object::toString).collect(Collectors.joining(",")));
    }

    private String parseDatum(ColumnInfo info, Datum datum) {
        if (datum.isNullValue() != null && datum.isNullValue()) {
            return info.getName() + "=" + "NULL";
        }
        Type columnType = info.getType();
        // If the column is of TimeSeries Type
        if (columnType.getTimeSeriesMeasureValueColumnInfo() != null) {
            return parseTimeSeries(info, datum);
        }
        // If the column is of Array Type
        else if (columnType.getArrayColumnInfo() != null) {
            List<Datum> arrayValues = datum.getArrayValue();
            return info.getName() + "=" +
parseArray(info.getType().getArrayColumnInfo(), arrayValues);
        }
        // If the column is of Row Type
        else if (columnType.getRowColumnInfo() != null) {
            List<ColumnInfo> rowColumnInfo = info.getType().getRowColumnInfo();
            Row rowValues = datum.getRowValue();
            return parseRow(rowColumnInfo, rowValues);
        }
        // If the column is of Scalar Type
        else {
            return parseScalarType(info, datum);
        }
    }

    private String parseTimeSeries(ColumnInfo info, Datum datum) {
        List<String> timeSeriesOutput = new ArrayList<>();
```

```
        for (TimeSeriesDataPoint dataPoint : datum.getTimeSeriesValue()) {
            timeSeriesOutput.add("{time=" + dataPoint.getTime() + ", value=" +
                    parseDatum(info.getType().getTimeSeriesMeasureValueColumnInfo(),
dataPoint.getValue()) + "}");
        }
        return String.format("[%s]",
timeSeriesOutput.stream().map(Object::toString).collect(Collectors.joining(",")));
    }

    private String parseScalarType(ColumnInfo info, Datum datum) {
        switch (ScalarType.fromValue(info.getType().getScalarType())) {
            case VARCHAR:
                return parseColumnName(info) + datum.getScalarValue();
            case BIGINT:
                Long longValue = Long.valueOf(datum.getScalarValue());
                return parseColumnName(info) + longValue;
            case INTEGER:
                Integer intValue = Integer.valueOf(datum.getScalarValue());
                return parseColumnName(info) + intValue;
            case BOOLEAN:
                Boolean booleanValue = Boolean.valueOf(datum.getScalarValue());
                return parseColumnName(info) + booleanValue;
            case DOUBLE:
                Double doubleValue = Double.valueOf(datum.getScalarValue());
                return parseColumnName(info) + doubleValue;
            case TIMESTAMP:
                return parseColumnName(info) +
LocalDateTime.parse(datum.getScalarValue(), TIMESTAMP_FORMATTER);
            case DATE:
                return parseColumnName(info) +
LocalDate.parse(datum.getScalarValue(), DATE_FORMATTER);
            case TIME:
                return parseColumnName(info) +
LocalTime.parse(datum.getScalarValue(), TIME_FORMATTER);
            case INTERVAL_DAY_TO_SECOND:
            case INTERVAL_YEAR_TO_MONTH:
                return parseColumnName(info) + datum.getScalarValue();
            case UNKNOWN:
                return parseColumnName(info) + datum.getScalarValue();
            default:
                throw new IllegalArgumentException("Given type is not valid: " +
info.getType().getScalarType());
        }
    }
```

```
   private String parseColumnName(ColumnInfo info) {
       return info.getName() == null ? "" : info.getName() + "=";
   }

   private String parseArray(ColumnInfo arrayColumnInfo, List<Datum> arrayValues) {
       List<String> arrayOutput = new ArrayList<>();
       for (Datum datum : arrayValues) {
           arrayOutput.add(parseDatum(arrayColumnInfo, datum));
       }
       return String.format("[%s]",
arrayOutput.stream().map(Object::toString).collect(Collectors.joining(",")));
   }
```

Java v2

```
   private static final long ONE_GB_IN_BYTES = 1073741824L;

   private void parseQueryResult(QueryResponse response) {
       final QueryStatus currentStatusOfQuery = response.queryStatus();

       System.out.println("Query progress so far: " +
currentStatusOfQuery.progressPercentage() + "%");

       double bytesScannedSoFar = ((double)
currentStatusOfQuery.cumulativeBytesScanned() / ONE_GB_IN_BYTES);
       System.out.println("Bytes scanned so far: " + bytesScannedSoFar + " GB");

       double bytesMeteredSoFar = ((double)
currentStatusOfQuery.cumulativeBytesMetered() / ONE_GB_IN_BYTES);
       System.out.println("Bytes metered so far: " + bytesMeteredSoFar + " GB");

       List<ColumnInfo> columnInfo = response.columnInfo();
       List<Row> rows = response.rows();

       System.out.println("Metadata: " + columnInfo);
       System.out.println("Data: ");

       // iterate every row
       for (Row row : rows) {
           System.out.println(parseRow(columnInfo, row));
       }
   }
```

```
   private String parseRow(List<ColumnInfo> columnInfo, Row row) {
       List<Datum> data = row.data();
       List<String> rowOutput = new ArrayList<>();
       // iterate every column per row
       for (int j = 0; j < data.size(); j++) {
           ColumnInfo info = columnInfo.get(j);
           Datum datum = data.get(j);
           rowOutput.add(parseDatum(info, datum));
       }
       return String.format("{%s}",
rowOutput.stream().map(Object::toString).collect(Collectors.joining(",")));
   }

   private String parseDatum(ColumnInfo info, Datum datum) {
       if (datum.nullValue() != null && datum.nullValue()) {
           return info.name() + "=" + "NULL";
       }
       Type columnType = info.type();
       // If the column is of TimeSeries Type
       if (columnType.timeSeriesMeasureValueColumnInfo() != null) {
           return parseTimeSeries(info, datum);
       }
       // If the column is of Array Type
       else if (columnType.arrayColumnInfo() != null) {
           List<Datum> arrayValues = datum.arrayValue();
           return info.name() + "=" + parseArray(info.type().arrayColumnInfo(),
arrayValues);
       }
       // If the column is of Row Type
       else if (columnType.rowColumnInfo() != null &&
columnType.rowColumnInfo().size() > 0) {
           List<ColumnInfo> rowColumnInfo = info.type().rowColumnInfo();
           Row rowValues = datum.rowValue();
           return parseRow(rowColumnInfo, rowValues);
       }
       // If the column is of Scalar Type
       else {
           return parseScalarType(info, datum);
       }
   }

   private String parseTimeSeries(ColumnInfo info, Datum datum) {
       List<String> timeSeriesOutput = new ArrayList<>();
```

```
        for (TimeSeriesDataPoint dataPoint : datum.timeSeriesValue()) {
            timeSeriesOutput.add("{time=" + dataPoint.time() + ", value=" +
                    parseDatum(info.type().timeSeriesMeasureValueColumnInfo(),
 dataPoint.value()) + "}");
        }
        return String.format("[%s]",
 timeSeriesOutput.stream().map(Object::toString).collect(Collectors.joining(",")));
    }

    private String parseScalarType(ColumnInfo info, Datum datum) {
        return parseColumnName(info) + datum.scalarValue();
    }

    private String parseColumnName(ColumnInfo info) {
        return info.name() == null ? "" : info.name() + "=";
    }

    private String parseArray(ColumnInfo arrayColumnInfo, List<Datum> arrayValues) {
        List<String> arrayOutput = new ArrayList<>();
        for (Datum datum : arrayValues) {
            arrayOutput.add(parseDatum(arrayColumnInfo, datum));
        }
        return String.format("[%s]",
 arrayOutput.stream().map(Object::toString).collect(Collectors.joining(",")));
    }
```

Go

```
func processScalarType(data *timestreamquery.Datum) string {
    return *data.ScalarValue
}

func processTimeSeriesType(data []*timestreamquery.TimeSeriesDataPoint, columnInfo
 *timestreamquery.ColumnInfo) string {
    value := ""
    for k := 0; k < len(data); k++ {
        time := data[k].Time
        value += *time + ":"
        if columnInfo.Type.ScalarType != nil {
            value += processScalarType(data[k].Value)
        } else if columnInfo.Type.ArrayColumnInfo != nil {
            value += processArrayType(data[k].Value.ArrayValue,
 columnInfo.Type.ArrayColumnInfo)
```

```
        } else if columnInfo.Type.RowColumnInfo != nil {
            value += processRowType(data[k].Value.RowValue.Data,
 columnInfo.Type.RowColumnInfo)
        } else {
            fail("Bad data type")
        }
        if k != len(data)-1 {
            value += ", "
        }
    }
    return value
}

func processArrayType(datumList []*timestreamquery.Datum, columnInfo
 *timestreamquery.ColumnInfo) string {
    value := ""
    for k := 0; k < len(datumList); k++ {
        if columnInfo.Type.ScalarType != nil {
            value += processScalarType(datumList[k])
        } else if columnInfo.Type.TimeSeriesMeasureValueColumnInfo != nil {
            value += processTimeSeriesType(datumList[k].TimeSeriesValue,
 columnInfo.Type.TimeSeriesMeasureValueColumnInfo)
        } else if columnInfo.Type.ArrayColumnInfo != nil {
            value += "["
            value += processArrayType(datumList[k].ArrayValue,
 columnInfo.Type.ArrayColumnInfo)
            value += "]"
        } else if columnInfo.Type.RowColumnInfo != nil {
            value += "["
            value += processRowType(datumList[k].RowValue.Data,
 columnInfo.Type.RowColumnInfo)
            value += "]"
        } else {
            fail("Bad column type")
        }

        if k != len(datumList)-1 {
            value += ", "
        }
    }
    return value
}
```

```
func processRowType(data []*timestreamquery.Datum, metadata
 []*timestreamquery.ColumnInfo) string {
    value := ""
    for j := 0; j < len(data); j++ {
        if metadata[j].Type.ScalarType != nil {
            // process simple data types
            value += processScalarType(data[j])
        } else if metadata[j].Type.TimeSeriesMeasureValueColumnInfo != nil {
            // fmt.Println("Timeseries measure value column info")
            // fmt.Println(metadata[j].Type.TimeSeriesMeasureValueColumnInfo.Type)
            datapointList := data[j].TimeSeriesValue
            value += "["
            value += processTimeSeriesType(datapointList,
 metadata[j].Type.TimeSeriesMeasureValueColumnInfo)
            value += "]"
        } else if metadata[j].Type.ArrayColumnInfo != nil {
            columnInfo := metadata[j].Type.ArrayColumnInfo
            // fmt.Println("Array column info")
            // fmt.Println(columnInfo)
            datumList := data[j].ArrayValue
            value += "["
            value += processArrayType(datumList, columnInfo)
            value += "]"
        } else if metadata[j].Type.RowColumnInfo != nil {
            columnInfo := metadata[j].Type.RowColumnInfo
            datumList := data[j].RowValue.Data
            value += "["
            value += processRowType(datumList, columnInfo)
            value += "]"
        } else {
            panic("Bad column type")
        }
        // comma seperated column values
        if j != len(data)-1 {
            value += ", "
        }
    }
    return value
}
```

Python

```
    def _parse_query_result(self, query_result):
```

```
        query_status = query_result["QueryStatus"]

        progress_percentage = query_status["ProgressPercentage"]
        print(f"Query progress so far: {progress_percentage}%")

        bytes_scanned = float(query_status["CumulativeBytesScanned"]) /
ONE_GB_IN_BYTES
        print(f"Data scanned so far: {bytes_scanned} GB")

        bytes_metered = float(query_status["CumulativeBytesMetered"]) /
ONE_GB_IN_BYTES
        print(f"Data metered so far: {bytes_metered} GB")

        column_info = query_result['ColumnInfo']

        print("Metadata: %s" % column_info)
        print("Data: ")
        for row in query_result['Rows']:
            print(self._parse_row(column_info, row))

    def _parse_row(self, column_info, row):
        data = row['Data']
        row_output = []
        for j in range(len(data)):
            info = column_info[j]
            datum = data[j]
            row_output.append(self._parse_datum(info, datum))

        return "{%s}" % str(row_output)

    def _parse_datum(self, info, datum):
        if datum.get('NullValue', False):
            return "%s=NULL" % info['Name'],

        column_type = info['Type']

        # If the column is of TimeSeries Type
        if 'TimeSeriesMeasureValueColumnInfo' in column_type:
            return self._parse_time_series(info, datum)

        # If the column is of Array Type
        elif 'ArrayColumnInfo' in column_type:
            array_values = datum['ArrayValue']
```

```
            return "%s=%s" % (info['Name'], self._parse_array(info['Type']
['ArrayColumnInfo'], array_values))

        # If the column is of Row Type
        elif 'RowColumnInfo' in column_type:
            row_column_info = info['Type']['RowColumnInfo']
            row_values = datum['RowValue']
            return self._parse_row(row_column_info, row_values)

        # If the column is of Scalar Type
        else:
            return self._parse_column_name(info) + datum['ScalarValue']

    def _parse_time_series(self, info, datum):
        time_series_output = []
        for data_point in datum['TimeSeriesValue']:
            time_series_output.append("{time=%s, value=%s}"
                                      % (data_point['Time'],
                                         self._parse_datum(info['Type']
['TimeSeriesMeasureValueColumnInfo'],
                                                           data_point['Value'])))
        return "[%s]" % str(time_series_output)

    def _parse_array(self, array_column_info, array_values):
        array_output = []
        for datum in array_values:
            array_output.append(self._parse_datum(array_column_info, datum))

        return "[%s]" % str(array_output)

    @staticmethod
    def _parse_column_name(info):
        if 'Name' in info:
            return info['Name'] + "="
        else:
            return ""
```

Node.js

En el siguiente fragmento se utiliza el estilo AWS SDK for JavaScript V2. Se basa en la aplicación de ejemplo de Node.js, ejemplo de Amazon Timestream LiveAnalytics para su aplicación en. GitHub

```
function parseQueryResult(response) {
    const queryStatus = response.QueryStatus;
    console.log("Current query status: " + JSON.stringify(queryStatus));

    const columnInfo = response.ColumnInfo;
    const rows = response.Rows;

    console.log("Metadata: " + JSON.stringify(columnInfo));
    console.log("Data: ");

    rows.forEach(function (row) {
        console.log(parseRow(columnInfo, row));
    });
}

function parseRow(columnInfo, row) {
    const data = row.Data;
    const rowOutput = [];

    var i;
    for ( i = 0; i < data.length; i++ ) {
        info = columnInfo[i];
        datum = data[i];
        rowOutput.push(parseDatum(info, datum));
    }

    return `{${rowOutput.join(", ")}}`
}

function parseDatum(info, datum) {
    if (datum.NullValue != null && datum.NullValue === true) {
        return `${info.Name}=NULL`;
    }

    const columnType = info.Type;

    // If the column is of TimeSeries Type
    if (columnType.TimeSeriesMeasureValueColumnInfo != null) {
        return parseTimeSeries(info, datum);
    }
    // If the column is of Array Type
    else if (columnType.ArrayColumnInfo != null) {
        const arrayValues = datum.ArrayValue;
```

```
        return `${info.Name}=${parseArray(info.Type.ArrayColumnInfo, arrayValues)}`;
    }
    // If the column is of Row Type
    else if (columnType.RowColumnInfo != null) {
        const rowColumnInfo = info.Type.RowColumnInfo;
        const rowValues = datum.RowValue;
        return parseRow(rowColumnInfo, rowValues);
    }
    // If the column is of Scalar Type
    else {
        return parseScalarType(info, datum);
    }
}

function parseTimeSeries(info, datum) {
    const timeSeriesOutput = [];
    datum.TimeSeriesValue.forEach(function (dataPoint) {
        timeSeriesOutput.push(`{time=${dataPoint.Time}, value=
${parseDatum(info.Type.TimeSeriesMeasureValueColumnInfo, dataPoint.Value)}}`)
    });

    return `[${timeSeriesOutput.join(", ")}]`
}

function parseScalarType(info, datum) {
    return parseColumnName(info) + datum.ScalarValue;
}

function parseColumnName(info) {
    return info.Name == null ? "" : `${info.Name}=`;
}

function parseArray(arrayColumnInfo, arrayValues) {
    const arrayOutput = [];
    arrayValues.forEach(function (datum) {
        arrayOutput.push(parseDatum(arrayColumnInfo, datum));
    });
    return `[${arrayOutput.join(", ")}]`
}
```

.NET

```
        private void ParseQueryResult(QueryResponse response)
```

```
        {
            List<ColumnInfo> columnInfo = response.ColumnInfo;
            var options = new JsonSerializerOptions
            {
                IgnoreNullValues = true
            };
            List<String> columnInfoStrings = columnInfo.ConvertAll(x =>
JsonSerializer.Serialize(x, options));
            List<Row> rows = response.Rows;

            QueryStatus queryStatus = response.QueryStatus;
            Console.WriteLine("Current Query status:" +
JsonSerializer.Serialize(queryStatus, options));

            Console.WriteLine("Metadata:" + string.Join(",", columnInfoStrings));
            Console.WriteLine("Data:");

            foreach (Row row in rows)
            {
                Console.WriteLine(ParseRow(columnInfo, row));
            }
        }

        private string ParseRow(List<ColumnInfo> columnInfo, Row row)
        {
            List<Datum> data = row.Data;
            List<string> rowOutput = new List<string>();
            for (int j = 0; j < data.Count; j++)
            {
                ColumnInfo info = columnInfo[j];
                Datum datum = data[j];
                rowOutput.Add(ParseDatum(info, datum));
            }
            return $"{{{string.Join(",", rowOutput)}}}";
        }

        private string ParseDatum(ColumnInfo info, Datum datum)
        {
            if (datum.NullValue)
            {
                return $"{info.Name}=NULL";
            }

            Amazon.TimestreamQuery.Model.Type columnType = info.Type;
```

```
            if (columnType.TimeSeriesMeasureValueColumnInfo != null)
            {
                return ParseTimeSeries(info, datum);
            }
            else if (columnType.ArrayColumnInfo != null)
            {
                List<Datum> arrayValues = datum.ArrayValue;
                return $"{info.Name}={ParseArray(info.Type.ArrayColumnInfo,
arrayValues)}";
            }
            else if (columnType.RowColumnInfo != null &&
columnType.RowColumnInfo.Count > 0)
            {
                List<ColumnInfo> rowColumnInfo = info.Type.RowColumnInfo;
                Row rowValue = datum.RowValue;
                return ParseRow(rowColumnInfo, rowValue);
            }
            else
            {
                return ParseScalarType(info, datum);
            }
        }

        private string ParseTimeSeries(ColumnInfo info, Datum datum)
        {
            var timeseriesString = datum.TimeSeriesValue
                .Select(value => $"{{time={value.Time},
value={ParseDatum(info.Type.TimeSeriesMeasureValueColumnInfo, value.Value)}}}")
                .Aggregate((current, next) => current + "," + next);

            return $"[{timeseriesString}]";
        }

        private string ParseScalarType(ColumnInfo info, Datum datum)
        {
            return ParseColumnName(info) + datum.ScalarValue;
        }

        private string ParseColumnName(ColumnInfo info)
        {
            return info.Name == null ? "" : (info.Name + "=");
        }
```

```
        private string ParseArray(ColumnInfo arrayColumnInfo, List<Datum>
arrayValues)
        {
            return $"[{arrayValues.Select(value => ParseDatum(arrayColumnInfo,
value)).Aggregate((current, next) => current + "," + next)}]";
        }
```

Acceder al estado de la consulta

Para acceder al estado de la consulta`QueryResponse`, que contiene información sobre el progreso de la consulta, los bytes escaneados por la consulta y los bytes medidos por la consulta. Los `bytesScanned` valores `bytesMetered` y son acumulativos y se actualizan continuamente al paginar los resultados de la consulta. Puede utilizar esta información para comprender los bytes escaneados por una consulta individual y también para tomar determinadas decisiones. Por ejemplo, suponiendo que el precio de la consulta sea de 0,01$ por GB escaneado, es posible que desee cancelar las consultas que superen los 25$ por consulta, o X GB. En el siguiente fragmento de código se muestra cómo se puede hacer esto.

> **Note**
>
> Estos fragmentos de código se basan en aplicaciones de muestra completas en. GitHub Para obtener más información sobre cómo empezar a utilizar las aplicaciones de ejemplo, consulte. Aplicación de muestra

Java

```
   private static final long ONE_GB_IN_BYTES = 1073741824L;
   private static final double QUERY_COST_PER_GB_IN_DOLLARS = 0.01; // Assuming the
price of query is $0.01 per GB

   public void cancelQueryBasedOnQueryStatus() {
       System.out.println("Starting query: " + SELECT_ALL_QUERY);
       QueryRequest queryRequest = new QueryRequest();
       queryRequest.setQueryString(SELECT_ALL_QUERY);
       QueryResult queryResult = queryClient.query(queryRequest);

       while (true) {
           final QueryStatus currentStatusOfQuery = queryResult.getQueryStatus();
```

```
            System.out.println("Query progress so far: " +
currentStatusOfQuery.getProgressPercentage() + "%");
            double bytesMeteredSoFar = ((double)
currentStatusOfQuery.getCumulativeBytesMetered() / ONE_GB_IN_BYTES);
            System.out.println("Bytes metered so far: " + bytesMeteredSoFar + "
GB");
            // Cancel query if its costing more than 1 cent
            if (bytesMeteredSoFar * QUERY_COST_PER_GB_IN_DOLLARS > 0.01) {
                cancelQuery(queryResult);
                break;
            }

            if (queryResult.getNextToken() == null) {
                break;
            }
            queryRequest.setNextToken(queryResult.getNextToken());
            queryResult = queryClient.query(queryRequest);
        }
    }
```

Java v2

```
    private static final long ONE_GB_IN_BYTES = 1073741824L;
    private static final double QUERY_COST_PER_GB_IN_DOLLARS = 0.01; // Assuming the
price of query is $0.01 per GB

    public void cancelQueryBasedOnQueryStatus() {
        System.out.println("Starting query: " + SELECT_ALL_QUERY);
        QueryRequest queryRequest =
QueryRequest.builder().queryString(SELECT_ALL_QUERY).build();

        final QueryIterable queryResponseIterator =
timestreamQueryClient.queryPaginator(queryRequest);
        for(QueryResponse queryResponse : queryResponseIterator) {
            final QueryStatus currentStatusOfQuery = queryResponse.queryStatus();
            System.out.println("Query progress so far: " +
currentStatusOfQuery.progressPercentage() + "%");
            double bytesMeteredSoFar = ((double)
currentStatusOfQuery.cumulativeBytesMetered() / ONE_GB_IN_BYTES);
            System.out.println("Bytes metered so far: " + bytesMeteredSoFar + "GB");
            // Cancel query if its costing more than 1 cent
            if (bytesMeteredSoFar * QUERY_COST_PER_GB_IN_DOLLARS > 0.01) {
                cancelQuery(queryResponse);
```

```
                break;
            }
        }
    }
```

Go

```
const OneGbInBytes = 1073741824
// Assuming the price of query is $0.01 per GB
const QueryCostPerGbInDollars = 0.01

func cancelQueryBasedOnQueryStatus(queryPtr *string, querySvc
 *timestreamquery.TimestreamQuery, f *os.File) {
    queryInput := &timestreamquery.QueryInput{
        QueryString: aws.String(*queryPtr),
    }
    fmt.Println("QueryInput:")
    fmt.Println(queryInput)
    // execute the query
    err := querySvc.QueryPages(queryInput,
        func(page *timestreamquery.QueryOutput, lastPage bool) bool {
            // process query response
            queryStatus := page.QueryStatus
            fmt.Println("Current query status:", queryStatus)
            bytes_metered := float64(*queryStatus.CumulativeBytesMetered) /
 float64(ONE_GB_IN_BYTES)
            if bytes_metered * QUERY_COST_PER_GB_IN_DOLLARS > 0.01 {
                cancelQuery(page, querySvc)
                return true
            }
            // query response metadata
            // includes column names and types
            metadata := page.ColumnInfo
            // fmt.Println("Metadata:")
            fmt.Println(metadata)
            header := ""
            for i := 0; i < len(metadata); i++ {
                header += *metadata[i].Name
                if i != len(metadata)-1 {
                    header += ", "
                }
            }
            write(f, header)
```

```
            // query response data
            fmt.Println("Data:")
            // process rows
            rows := page.Rows
            for i := 0; i < len(rows); i++ {
                data := rows[i].Data
                value := processRowType(data, metadata)
                fmt.Println(value)
                write(f, value)
            }
            fmt.Println("Number of rows:", len(page.Rows))
            return true
        })
    if err != nil {
        fmt.Println("Error:")
        fmt.Println(err)
    }
}
```

Python

```
ONE_GB_IN_BYTES = 1073741824
# Assuming the price of query is $0.01 per GB
QUERY_COST_PER_GB_IN_DOLLARS = 0.01

    def cancel_query_based_on_query_status(self):
        try:
            print("Starting query: " + self.SELECT_ALL)
            page_iterator = self.paginator.paginate(QueryString=self.SELECT_ALL)
            for page in page_iterator:
                query_status = page["QueryStatus"]
                progress_percentage = query_status["ProgressPercentage"]
                print("Query progress so far: " + str(progress_percentage) + "%")
                bytes_metered = query_status["CumulativeBytesMetered"] /
 self.ONE_GB_IN_BYTES
                print("Bytes Metered so far: " + str(bytes_metered) + " GB")
                if bytes_metered * self.QUERY_COST_PER_GB_IN_DOLLARS > 0.01:
                    self.cancel_query_for(page)
                    break
        except Exception as err:
            print("Exception while running query:", err)
            traceback.print_exc(file=sys.stderr)
```

Node.js

En el siguiente fragmento se utiliza el estilo AWS SDK for JavaScript V2. Se basa en la aplicación de ejemplo de Node.js, ejemplo de Amazon Timestream LiveAnalytics para su aplicación en. GitHub

```
function parseQueryResult(response) {
    const queryStatus = response.QueryStatus;
    console.log("Current query status: " + JSON.stringify(queryStatus));

    const columnInfo = response.ColumnInfo;
    const rows = response.Rows;

    console.log("Metadata: " + JSON.stringify(columnInfo));
    console.log("Data: ");

    rows.forEach(function (row) {
        console.log(parseRow(columnInfo, row));
    });
}

function parseRow(columnInfo, row) {
    const data = row.Data;
    const rowOutput = [];

    var i;
    for ( i = 0; i < data.length; i++ ) {
        info = columnInfo[i];
        datum = data[i];
        rowOutput.push(parseDatum(info, datum));
    }

    return `{${rowOutput.join(", ")}}`
}

function parseDatum(info, datum) {
    if (datum.NullValue != null && datum.NullValue === true) {
        return `${info.Name}=NULL`;
    }

    const columnType = info.Type;

    // If the column is of TimeSeries Type
```

```
    if (columnType.TimeSeriesMeasureValueColumnInfo != null) {
        return parseTimeSeries(info, datum);
    }
    // If the column is of Array Type
    else if (columnType.ArrayColumnInfo != null) {
        const arrayValues = datum.ArrayValue;
        return `${info.Name}=${parseArray(info.Type.ArrayColumnInfo, arrayValues)}`;
    }
    // If the column is of Row Type
    else if (columnType.RowColumnInfo != null) {
        const rowColumnInfo = info.Type.RowColumnInfo;
        const rowValues = datum.RowValue;
        return parseRow(rowColumnInfo, rowValues);
    }
    // If the column is of Scalar Type
    else {
        return parseScalarType(info, datum);
    }
}

function parseTimeSeries(info, datum) {
    const timeSeriesOutput = [];
    datum.TimeSeriesValue.forEach(function (dataPoint) {
        timeSeriesOutput.push(`{time=${dataPoint.Time}, value=
${parseDatum(info.Type.TimeSeriesMeasureValueColumnInfo, dataPoint.Value)}}`)
    });

    return `[${timeSeriesOutput.join(", ")}]`
}

function parseScalarType(info, datum) {
    return parseColumnName(info) + datum.ScalarValue;
}

function parseColumnName(info) {
    return info.Name == null ? "" : `${info.Name}=`;
}

function parseArray(arrayColumnInfo, arrayValues) {
    const arrayOutput = [];
    arrayValues.forEach(function (datum) {
        arrayOutput.push(parseDatum(arrayColumnInfo, datum));
    });
    return `[${arrayOutput.join(", ")}]`
```

```
}
```

.NET

```
private static readonly long ONE_GB_IN_BYTES = 1073741824L;
private static readonly double QUERY_COST_PER_GB_IN_DOLLARS = 0.01; // Assuming the
 price of query is $0.01 per GB

private async Task CancelQueryBasedOnQueryStatus(string queryString)
{
    try
    {
        QueryRequest queryRequest = new QueryRequest();
        queryRequest.QueryString = queryString;
        QueryResponse queryResponse = await queryClient.QueryAsync(queryRequest);
        while (true)
        {
            QueryStatus queryStatus = queryResponse.QueryStatus;
            double bytesMeteredSoFar = ((double)
 queryStatus.CumulativeBytesMetered / ONE_GB_IN_BYTES);
            // Cancel query if its costing more than 1 cent
            if (bytesMeteredSoFar * QUERY_COST_PER_GB_IN_DOLLARS > 0.01)
            {
                await CancelQuery(queryResponse);
                break;
            }

            ParseQueryResult(queryResponse);
            if (queryResponse.NextToken == null)
            {
                break;
            }
            queryRequest.NextToken = queryResponse.NextToken;
            queryResponse = await queryClient.QueryAsync(queryRequest);
       }
    } catch(Exception e)
    {
        // Some queries might fail with 500 if the result of a sequence function has
 more than 10000 entries
        Console.WriteLine(e.ToString());
    }
}
```

Para obtener información adicional sobre cómo cancelar una consulta, consulte. Cancelar consulta

Ejecutar UNLOAD consulta

Los siguientes ejemplos de código llaman a una UNLOAD consulta. Para obtener más información sobre UNLOAD, consulte Se utiliza UNLOAD para exportar los resultados de las consultas a S3 desde Timestream para LiveAnalytics. Para ver ejemplos de UNLOAD consultas, consulteEjemplo de caso de uso de Timestream para UNLOAD LiveAnalytics.

Temas

- Cree y ejecute una UNLOAD consulta
- Analice UNLOAD la respuesta y obtenga el recuento de filas, el enlace al manifiesto y el enlace a los metadatos
- Lee y analiza el contenido del manifiesto
- Lea y analice el contenido de los metadatos

Cree y ejecute una UNLOAD consulta

Java

```
// When you have a SELECT like below

String QUERY_1 = "SELECT user_id, ip_address, event, session_id, measure_name, time,
 query, quantity, product_id, channel FROM "
        + DATABASE_NAME + "." + UNLOAD_TABLE_NAME
        + " WHERE time BETWEEN ago(2d) AND now()";

// You can construct UNLOAD query as follows
UnloadQuery unloadQuery = UnloadQuery.builder()
        .selectQuery(QUERY_1)
        .bucketName("timestream-sample-<region>-<accountId>")
        .resultsPrefix("without_partition")
        .format(CSV)
        .compression(UnloadQuery.Compression.GZIP)
        .build();
QueryResult unloadResult = runQuery(unloadQuery.getUnloadQuery());

// Run UNLOAD query (Similar to how you run SELECT query)
// https://docs.aws.amazon.com/timestream/latest/developerguide/code-samples.run-
query.html#code-samples.run-query.pagination
```

```
    private QueryResult runQuery(String queryString) {
        QueryResult queryResult = null;
        try {
            QueryRequest queryRequest = new QueryRequest();
            queryRequest.setQueryString(queryString);
            queryResult = queryClient.query(queryRequest);
            while (true) {
                parseQueryResult(queryResult);
                if (queryResult.getNextToken() == null) {
                    break;
                }
                queryRequest.setNextToken(queryResult.getNextToken());
                queryResult = queryClient.query(queryRequest);
            }
        } catch (Exception e) {
            // Some queries might fail with 500 if the result of a sequence function
 has more than 10000 entries
            e.printStackTrace();
        }
        return queryResult;
    }

// Utility that helps to construct UNLOAD query

@Builder
static class UnloadQuery {
    private String selectQuery;
    private String bucketName;
    private String resultsPrefix;
    private Format format;
    private Compression compression;
    private EncryptionType encryptionType;
    private List<String> partitionColumns;
    private String kmsKey;
    private Character csvFieldDelimiter;
    private Character csvEscapeCharacter;

    public String getUnloadQuery() {
        String destination = constructDestination();
        String withClause = constructOptionalParameters();
        return String.format("UNLOAD (%s) TO '%s' %s", selectQuery, destination,
 withClause);
    }
```

```
    private String constructDestination() {
        return "s3://" + this.bucketName + "/" + this.resultsPrefix + "/";
    }

    private String constructOptionalParameters() {
        boolean isOptionalParametersPresent = Objects.nonNull(format)
                || Objects.nonNull(compression)
                || Objects.nonNull(encryptionType)
                || Objects.nonNull(partitionColumns)
                || Objects.nonNull(kmsKey)
                || Objects.nonNull(csvFieldDelimiter)
                || Objects.nonNull(csvEscapeCharacter);

        String withClause = "";
        if (isOptionalParametersPresent) {
            StringJoiner optionalParameters = new StringJoiner(",");
            if (Objects.nonNull(format)) {
                optionalParameters.add("format = '" + format + "'");
            }
            if (Objects.nonNull(compression)) {
                optionalParameters.add("compression = '" + compression + "'");
            }
            if (Objects.nonNull(encryptionType)) {
                optionalParameters.add("encryption = '" + encryptionType + "'");
            }
            if (Objects.nonNull(kmsKey)) {
                optionalParameters.add("kms_key = '" + kmsKey + "'");
            }
            if (Objects.nonNull(csvFieldDelimiter)) {
                optionalParameters.add("field_delimiter = '" + csvFieldDelimiter +
"'");
            }
            if (Objects.nonNull(csvEscapeCharacter)) {
                optionalParameters.add("escaped_by = '" + csvEscapeCharacter + "'");
            }
            if (Objects.nonNull(partitionColumns) && !partitionColumns.isEmpty()) {
                final StringJoiner partitionedByList = new StringJoiner(",");
                partitionColumns.forEach(column -> partitionedByList.add("'" +
column + "'"));
                optionalParameters.add(String.format("partitioned_by = ARRAY[%s]",
partitionedByList));
            }
            withClause = String.format("WITH (%s)", optionalParameters);
        }
```

```
        return withClause;
    }

    public enum Format {
        CSV, PARQUET
    }

    public enum Compression {
        GZIP, NONE
    }

    public enum EncryptionType {
        SSE_S3, SSE_KMS
    }

    @Override
    public String toString() {
        return getUnloadQuery();
    }
}
```

Java v2

```
// When you have a SELECT like below

String QUERY_1 = "SELECT user_id, ip_address, event, session_id, measure_name, time,
 query, quantity, product_id, channel FROM "
        + DATABASE_NAME + "." + UNLOAD_TABLE_NAME
        + " WHERE time BETWEEN ago(2d) AND now()";

//You can construct UNLOAD query as follows
UnloadQuery unloadQuery = UnloadQuery.builder()
        .selectQuery(QUERY_1)
        .bucketName("timestream-sample-<region>-<accountId>")
        .resultsPrefix("without_partition")
        .format(CSV)
        .compression(UnloadQuery.Compression.GZIP)
        .build();

QueryResponse unloadResponse = runQuery(unloadQuery.getUnloadQuery());


// Run UNLOAD query (Similar to how you run SELECT query)
```

```
// https://docs.aws.amazon.com/timestream/latest/developerguide/code-samples.run-
query.html#code-samples.run-query.pagination
private QueryResponse runQuery(String queryString) {
   QueryResponse finalResponse = null;
    try {
        QueryRequest queryRequest =
 QueryRequest.builder().queryString(queryString).build();
        final QueryIterable queryResponseIterator =
 timestreamQueryClient.queryPaginator(queryRequest);
        for(QueryResponse queryResponse : queryResponseIterator) {
            parseQueryResult(queryResponse);
           finalResponse = queryResponse;
        }
    } catch (Exception e) {
        // Some queries might fail with 500 if the result of a sequence function has
 more than 10000 entries
        e.printStackTrace();
    }
   return finalResponse;
}

// Utility that helps to construct UNLOAD query
@Builder
static class UnloadQuery {
    private String selectQuery;
    private String bucketName;
    private String resultsPrefix;
    private Format format;
    private Compression compression;
    private EncryptionType encryptionType;
    private List<String> partitionColumns;
    private String kmsKey;
    private Character csvFieldDelimiter;
    private Character csvEscapeCharacter;

    public String getUnloadQuery() {
        String destination = constructDestination();
        String withClause = constructOptionalParameters();
        return String.format("UNLOAD (%s) TO '%s' %s", selectQuery, destination,
 withClause);
    }

    private String constructDestination() {
        return "s3://" + this.bucketName + "/" + this.resultsPrefix + "/";
```

```
    }

    private String constructOptionalParameters() {
        boolean isOptionalParametersPresent = Objects.nonNull(format)
                || Objects.nonNull(compression)
                || Objects.nonNull(encryptionType)
                || Objects.nonNull(partitionColumns)
                || Objects.nonNull(kmsKey)
                || Objects.nonNull(csvFieldDelimiter)
                || Objects.nonNull(csvEscapeCharacter);

        String withClause = "";
        if (isOptionalParametersPresent) {
            StringJoiner optionalParameters = new StringJoiner(",");
            if (Objects.nonNull(format)) {
                optionalParameters.add("format = '" + format + "'");
            }
            if (Objects.nonNull(compression)) {
                optionalParameters.add("compression = '" + compression + "'");
            }
            if (Objects.nonNull(encryptionType)) {
                optionalParameters.add("encryption = '" + encryptionType + "'");
            }
            if (Objects.nonNull(kmsKey)) {
                optionalParameters.add("kms_key = '" + kmsKey + "'");
            }
            if (Objects.nonNull(csvFieldDelimiter)) {
                optionalParameters.add("field_delimiter = '" + csvFieldDelimiter +
"'");
            }
            if (Objects.nonNull(csvEscapeCharacter)) {
                optionalParameters.add("escaped_by = '" + csvEscapeCharacter + "'");
            }
            if (Objects.nonNull(partitionColumns) && !partitionColumns.isEmpty()) {
                final StringJoiner partitionedByList = new StringJoiner(",");
                partitionColumns.forEach(column -> partitionedByList.add("'" +
column + "'"));
                optionalParameters.add(String.format("partitioned_by = ARRAY[%s]",
partitionedByList));
            }
            withClause = String.format("WITH (%s)", optionalParameters);
        }
        return withClause;
    }
```

```
    public enum Format {
        CSV, PARQUET
    }

    public enum Compression {
        GZIP, NONE
    }

    public enum EncryptionType {
        SSE_S3, SSE_KMS
    }

    @Override
    public String toString() {
        return getUnloadQuery();
    }
}
```

Go

```
// When you have a SELECT like below
var Query = "SELECT user_id, ip_address, event, session_id, measure_name, time,
 query, quantity, product_id, channel FROM "
+ *databaseName + "." + *tableName + " WHERE time BETWEEN ago(2d) AND now()"

// You can construct UNLOAD query as follows
var unloadQuery = UnloadQuery{
    Query: "SELECT user_id, ip_address, session_id, measure_name, time, query,
 quantity, product_id, channel, event FROM " + *databaseName + "." + *tableName +
    " WHERE time BETWEEN ago(2d) AND now()",
    Partitioned_by: []string{},
    Compression: "GZIP",
    Format: "CSV",
    S3Location: bucketName,
    ResultPrefix: "without_partition",
}


// Run UNLOAD query (Similar to how you run SELECT query)
// https://docs.aws.amazon.com/timestream/latest/developerguide/code-samples.run-
query.html#code-samples.run-query.pagination
```

```
queryInput := &timestreamquery.QueryInput{
    QueryString: build_query(unloadQuery),
}

err := querySvc.QueryPages(queryInput,
    func(page *timestreamquery.QueryOutput, lastPage bool) bool {
        if (lastPage) {
            var response = parseQueryResult(page)
            var unloadFiles = getManifestAndMetadataFiles(s3Svc, response)
            displayColumns(unloadFiles, unloadQuery.Partitioned_by)
            displayResults(s3Svc, unloadFiles)
        }
        return true
    })

if err != nil {
    fmt.Println("Error:")
    fmt.Println(err)
}

// Utility that helps to construct UNLOAD query
type UnloadQuery struct {
    Query string
    Partitioned_by []string
    Format string
    S3Location string
    ResultPrefix string
    Compression string
}

func build_query(unload_query UnloadQuery) *string {
    var query_results_s3_path = "'s3://" + unload_query.S3Location + "/" +
 unload_query.ResultPrefix + "/'"
    var query = "UNLOAD(" + unload_query.Query + ") TO " + query_results_s3_path + "
 WITH ( "
    if (len(unload_query.Partitioned_by) > 0) {
        query = query + "partitioned_by=ARRAY["
        for i, column := range unload_query.Partitioned_by {
            if i == 0 {
                query = query + "'" + column + "'"
            } else {
                query = query + ",'" + column + "'"
            }
        }
```

```
        query = query + "],"
    }
    query = query + " format='" + unload_query.Format + "', "
    query = query + "  compression='" + unload_query.Compression + "')"
    fmt.Println(query)
    return aws.String(query)
}
```

Python

```
# When you have a SELECT like below
QUERY_1 = "SELECT user_id, ip_address, event, session_id, measure_name, time, query,
 quantity, product_id, channel FROM "
        + database_name + "." + table_name + " WHERE time BETWEEN ago(2d) AND now()"
# You can construct UNLOAD query as follows
UNLOAD_QUERY_1 = UnloadQuery(QUERY_1, "timestream-sample-<region>-<accountId>",
 "without_partition", "CSV", "GZIP", "")

# Run UNLOAD query (Similar to how you run SELECT query)
# https://docs.aws.amazon.com/timestream/latest/developerguide/code-samples.run-
query.html#code-samples.run-query.pagination
def run_query(self, query_string):
    try:
        page_iterator = self.paginator.paginate(QueryString=UNLOAD_QUERY_1)
    except Exception as err:
        print("Exception while running query:", err)

# Utility that helps to construct UNLOAD query
class UnloadQuery:
    def __init__(self, query, s3_bucket_location, results_prefix, format,
 compression , partition_by):
        self.query = query
        self.s3_bucket_location = s3_bucket_location
        self.results_prefix = results_prefix
        self.format = format
        self.compression = compression
        self.partition_by = partition_by

    def build_query(self):
        query_results_s3_path = "'s3://" + self.s3_bucket_location + "/" +
 self.results_prefix + "/'"
        unload_query = "UNLOAD("
        unload_query = unload_query + self.query
```

```
        unload_query = unload_query + ") "
        unload_query = unload_query + " TO " + query_results_s3_path
        unload_query = unload_query + " WITH ( "

        if(len(self.partition_by) > 0) :
            unload_query = unload_query + " partitioned_by = ARRAY" +
 str(self.partition_by) + ","

        unload_query = unload_query + " format='" + self.format  + "', "
        unload_query = unload_query + "  compression='" + self.compression + "')"

        return unload_query
```

Node.js

```
// When you have a SELECT like below
QUERY_1 = "SELECT user_id, ip_address, event, session_id, measure_name, time, query,
 quantity, product_id, channel FROM "
        + database_name + "." + table_name + " WHERE time BETWEEN ago(2d) AND now()"
// You can construct UNLOAD query as follows
UNLOAD_QUERY_1 = new UnloadQuery(QUERY_1, "timestream-sample-<region>-<accountId>",
 "without_partition", "CSV", "GZIP", "")


// Run UNLOAD query (Similar to how you run SELECT query)
// https://docs.aws.amazon.com/timestream/latest/developerguide/code-samples.run-
query.html#code-samples.run-query.pagination

async runQuery(query = UNLOAD_QUERY_1, nextToken) {
    const params = new QueryCommand({
        QueryString: query
    });

    if (nextToken) {
        params.NextToken = nextToken;
    }

    await queryClient.send(params).then(
            (response) => {
                if (response.NextToken) {
                    runQuery(queryClient, query, response.NextToken);
                } else {
                    await parseAndDisplayResults(response);
```

```
                }
            },
            (err) => {
                console.error("Error while querying:", err);
            });
}


class UnloadQuery {
    constructor(query, s3_bucket_location, results_prefix, format, compression ,
 partition_by) {
        this.query = query;
        this.s3_bucket_location = s3_bucket_location
        this.results_prefix = results_prefix
        this.format = format
        this.compression = compression
        this.partition_by = partition_by
    }

    buildQuery() {
        const query_results_s3_path = "'s3://" + this.s3_bucket_location + "/" +
 this.results_prefix + "/'"
        let unload_query = "UNLOAD("
        unload_query = unload_query + this.query
        unload_query = unload_query + ") "
        unload_query = unload_query + " TO " + query_results_s3_path
        unload_query = unload_query + " WITH ( "

        if(this.partition_by.length > 0) {
            let partitionBy = ""
            this.partition_by.forEach((str, i) => {
                partitionBy = partitionBy + (i ? ",'" : "'") + str + "'"
            })
            unload_query = unload_query + " partitioned_by = ARRAY[" + partitionBy +
 "],"
        }
        unload_query = unload_query + " format='" + this.format  + "', "
        unload_query = unload_query + "  compression='" + this.compression + "')"

        return unload_query
    }
}
```

Analice UNLOAD la respuesta y obtenga el recuento de filas, el enlace al manifiesto y el enlace a los metadatos

Java

```
// Parsing UNLOAD query response is similar to how you parse SELECT query response:
// https://docs.aws.amazon.com/timestream/latest/developerguide/code-samples.run-
query.html#code-samples.run-query.parsing

// But unlike SELECT, UNLOAD only has 1 row * 3 columns outputed
// (rows, metadataFile, manifestFile) => (BIGINT, VARCHAR, VARCHAR)

public UnloadResponse parseResult(QueryResult queryResult) {
    Map<String, String> outputMap = new HashMap<>();
    for (int i = 0; i < queryResult.getColumnInfo().size(); i++) {
        outputMap.put(queryResult.getColumnInfo().get(i).getName(),
                queryResult.getRows().get(0).getData().get(i).getScalarValue());

    }
    return new UnloadResponse(outputMap);
}

@Getter
class UnloadResponse {
    private final String metadataFile;
    private final String manifestFile;
    private final int rows;

    public UnloadResponse(Map<String, String> unloadResponse) {
        this.metadataFile = unloadResponse.get("metadataFile");
        this.manifestFile = unloadResponse.get("manifestFile");
        this.rows = Integer.parseInt(unloadResponse.get("rows"));
    }
}
```

Java v2

```
// Parsing UNLOAD query response is similar to how you parse SELECT query response:
// https://docs.aws.amazon.com/timestream/latest/developerguide/code-samples.run-
query.html#code-samples.run-query.parsing

// But unlike SELECT, UNLOAD only has 1 row * 3 columns outputed
// (rows, metadataFile, manifestFile) => (BIGINT, VARCHAR, VARCHAR)
```

```
public UnloadResponse parseResult(QueryResponse queryResponse) {
    Map<String, String> outputMap = new HashMap<>();
    for (int i = 0; i < queryResponse.columnInfo().size(); i++) {
        outputMap.put(queryResponse.columnInfo().get(i).name(),
                queryResponse.rows().get(0).data().get(i).scalarValue());

    }
    return new UnloadResponse(outputMap);
}

@Getter
class UnloadResponse {
    private final String metadataFile;
    private final String manifestFile;
    private final int rows;

    public UnloadResponse(Map<String, String> unloadResponse) {
        this.metadataFile = unloadResponse.get("metadataFile");
        this.manifestFile = unloadResponse.get("manifestFile");
        this.rows = Integer.parseInt(unloadResponse.get("rows"));
    }
}
```

Go

```
// Parsing UNLOAD query response is similar to how you parse SELECT query response:
// https://docs.aws.amazon.com/timestream/latest/developerguide/code-samples.run-
query.html#code-samples.run-query.parsing

// But unlike SELECT, UNLOAD only has 1 row * 3 columns outputed
// (rows, metadataFile, manifestFile) => (BIGINT, VARCHAR, VARCHAR)

func parseQueryResult(queryOutput *timestreamquery.QueryOutput) map[string]string {
    var columnInfo = queryOutput.ColumnInfo;
    fmt.Println("ColumnInfo", columnInfo)
    fmt.Println("QueryId", queryOutput.QueryId)
    fmt.Println("QueryStatus", queryOutput.QueryStatus)
    return parseResponse(columnInfo, queryOutput.Rows[0])
}

func parseResponse(columnInfo []*timestreamquery.ColumnInfo, row
 *timestreamquery.Row) map[string]string {
```

```
    var datum = row.Data
    response := make(map[string]string)
    for i, column := range columnInfo {
        response[*column.Name] = *datum[i].ScalarValue
    }
    return response
}
```

Python

```
# Parsing UNLOAD query response is similar to how you parse SELECT query response:
# https://docs.aws.amazon.com/timestream/latest/developerguide/code-samples.run-
query.html#code-samples.run-query.parsing

# But unlike SELECT, UNLOAD only has 1 row * 3 columns outputed
# (rows, metadataFile, manifestFile) => (BIGINT, VARCHAR, VARCHAR)

for page in page_iterator:
  last_page = page
response = self._parse_unload_query_result(last_page)

def _parse_unload_query_result(self, query_result):
    column_info = query_result['ColumnInfo']

    print("ColumnInfo: %s" % column_info)
    print("QueryId: %s" % query_result['QueryId'])
    print("QueryStatus:%s" % query_result['QueryStatus'])
    return self.parse_unload_response(column_info, query_result['Rows'][0])

def parse_unload_response(self, column_info, row):
    response = {}
    data = row['Data']
    for i, column in enumerate(column_info):
        response[column['Name']] = data[i]['ScalarValue']
   print("Rows: %s" % response['rows'])
   print("Metadata File location: %s" % response['metadataFile'])
   print("Manifest File location: %s" % response['manifestFile'])
   return response
```

Node.js

```
# Parsing UNLOAD query response is similar to how you parse SELECT query response:
```

```
# https://docs.aws.amazon.com/timestream/latest/developerguide/code-samples.run-
query.html#code-samples.run-query.parsing

# But unlike SELECT, UNLOAD only has 1 row * 3 columns outputed
# (rows, metadataFile, manifestFile) => (BIGINT, VARCHAR, VARCHAR)

async parseAndDisplayResults(data, query) {
    const columnInfo = data['ColumnInfo'];
    console.log("ColumnInfo:", columnInfo)
    console.log("QueryId: %s", data['QueryId'])
    console.log("QueryStatus:", data['QueryStatus'])
    await this.parseResponse(columnInfo, data['Rows'][0], query)
}

async parseResponse(columnInfo, row, query) {
    let response = {}
    const data = row['Data']
    columnInfo.forEach((column, i) => {
        response[column['Name']] = data[i]['ScalarValue']
    })

   console.log("Manifest file", response['manifestFile']);
   console.log("Metadata file", response['metadataFile']);

   return response
}
```

Lee y analiza el contenido del manifiesto

Java

```
// Read and parse manifest content
public UnloadManifest getUnloadManifest(UnloadResponse unloadResponse) throws
 IOException {
   AmazonS3URI s3URI = new AmazonS3URI(unloadResponse.getManifestFile());
    S3Object s3Object = s3Client.getObject(s3URI.getBucket(), s3URI.getKey());
    String manifestFileContent = new
 String(IOUtils.toByteArray(s3Object.getObjectContent()), StandardCharsets.UTF_8);
    return new Gson().fromJson(manifestFileContent, UnloadManifest.class);
}

class UnloadManifest {
```

```
    @Getter
    public class FileMetadata {
        long content_length_in_bytes;
        long row_count;
    }

    @Getter
    public class ResultFile {
        String url;
        FileMetadata file_metadata;
    }

    @Getter
    public class QueryMetadata {
        long total_content_length_in_bytes;
        long total_row_count;
        String result_format;
        String result_version;
    }

    @Getter
    public class Author {
        String name;
        String manifest_file_version;
    }

    @Getter
    private List<ResultFile> result_files;
    @Getter
    private QueryMetadata query_metadata;
    @Getter
    private Author author;
}
```

Java v2

```
// Read and parse manifest content
public UnloadManifest getUnloadManifest(UnloadResponse unloadResponse) throws
 URISyntaxException {
    // Space needs to encoded to use S3 parseUri function
    S3Uri s3Uri =
 s3Utilities.parseUri(URI.create(unloadResponse.getManifestFile().replace(" ",
 "%20")));
```

```
    ResponseBytes<GetObjectResponse> objectBytes =
 s3Client.getObjectAsBytes(GetObjectRequest.builder()
            .bucket(s3Uri.bucket().orElseThrow(() -> new
 URISyntaxException(unloadResponse.getManifestFile(), "Invalid S3 URI")))
            .key(s3Uri.key().orElseThrow(() -> new
 URISyntaxException(unloadResponse.getManifestFile(), "Invalid S3 URI")))
            .build());
    String manifestFileContent = new String(objectBytes.asByteArray(),
 StandardCharsets.UTF_8);
    return new Gson().fromJson(manifestFileContent, UnloadManifest.class);
}

class UnloadManifest {
    @Getter
    public class FileMetadata {
        long content_length_in_bytes;
        long row_count;
    }

    @Getter
    public class ResultFile {
        String url;
        FileMetadata file_metadata;
    }

    @Getter
    public class QueryMetadata {
        long total_content_length_in_bytes;
        long total_row_count;
        String result_format;
        String result_version;
    }

    @Getter
    public class Author {
        String name;
        String manifest_file_version;
    }

    @Getter
    private List<ResultFile> result_files;
    @Getter
    private QueryMetadata query_metadata;
    @Getter
```

```
    private Author author;
}
```

Go

```
// Read and parse manifest content

func getManifestFile(s3Svc *s3.S3, response map[string]string) Manifest {
    var manifestBuf = getObject(s3Svc, response["manifestFile"])
    var manifest Manifest
    json.Unmarshal(manifestBuf.Bytes(), &manifest)
    return manifest
}

func getObject(s3Svc *s3.S3, s3Uri string) *bytes.Buffer {
    u,_ := url.Parse(s3Uri)
    getObjectInput := &s3.GetObjectInput{
        Key:    aws.String(u.Path),
        Bucket: aws.String(u.Host),
    }
    getObjectOutput, err := s3Svc.GetObject(getObjectInput)
    if err != nil {
        fmt.Println("Error: %s\n", err.Error())
    }
    buf := new(bytes.Buffer)
    buf.ReadFrom(getObjectOutput.Body)
    return buf
}

// Unload's Manifest structure

type Manifest struct {
    Author interface{}
    Query_metadata map[string]any
    Result_files  []struct {
        File_metadata interface{}
        Url string
    }
}}
```

Python

```
def __get_manifest_file(self, response):
```

```
    manifest = self.get_object(response['manifestFile']).read().decode('utf-8')
    parsed_manifest = json.loads(manifest)
    print("Manifest contents: \n%s" % parsed_manifest)

def get_object(self, uri):
    try:
        bucket, key = uri.replace("s3://", "").split("/", 1)
        s3_client = boto3.client('s3', region_name=<region>)
        response = s3_client.get_object(Bucket=bucket, Key=key)
        return response['Body']
    except Exception as err:
        print("Failed to get the object for URI:", uri)
        raise err
```

Node.js

```
// Read and parse manifest content

async getManifestFile(response) {
    let manifest;
    await this.getS3Object(response['manifestFile']).then(
        (data) => {
            manifest = JSON.parse(data);
        }
    );
    return manifest;
}

async getS3Object(uri) {
    const {bucketName, key} = this.getBucketAndKey(uri);
    const params = new GetObjectCommand({
        Bucket: bucketName,
        Key: key
    })
    const response = await this.s3Client.send(params);
    return await response.Body.transformToString();
}

getBucketAndKey(uri) {
    const [bucketName] = uri.replace("s3://", "").split("/", 1);
    const key = uri.replace("s3://", "").split('/').slice(1).join('/');
    return {bucketName, key};
}
```

Lea y analice el contenido de los metadatos

Java

```
// Read and parse metadata content
public UnloadMetadata getUnloadMetadata(UnloadResponse unloadResponse) throws
 IOException {
   AmazonS3URI s3URI = new AmazonS3URI(unloadResponse.getMetadataFile());
   S3Object s3Object = s3Client.getObject(s3URI.getBucket(), s3URI.getKey());
    String metadataFileContent = new
 String(IOUtils.toByteArray(s3Object.getObjectContent()), StandardCharsets.UTF_8);
    final Gson gson = new GsonBuilder()
            .setFieldNamingPolicy(FieldNamingPolicy.UPPER_CAMEL_CASE)
            .create();
    return gson.fromJson(metadataFileContent, UnloadMetadata.class);
}

class UnloadMetadata {
    @JsonProperty("ColumnInfo")
    List<ColumnInfo> columnInfo;
    @JsonProperty("Author")
    Author author;

    @Data
    public class Author {
        @JsonProperty("Name")
        String name;
        @JsonProperty("MetadataFileVersion")
        String metadataFileVersion;
    }
}
```

Java v2

```
// Read and parse metadata content

public UnloadMetadata getUnloadMetadata(UnloadResponse unloadResponse) throws
 URISyntaxException {
   // Space needs to encoded to use S3 parseUri function
    S3Uri s3Uri =
 s3Utilities.parseUri(URI.create(unloadResponse.getMetadataFile().replace(" ",
 "%20")));
```

```
    ResponseBytes<GetObjectResponse> objectBytes =
 s3Client.getObjectAsBytes(GetObjectRequest.builder()
            .bucket(s3Uri.bucket().orElseThrow(() -> new
 URISyntaxException(unloadResponse.getMetadataFile(), "Invalid S3 URI")))
            .key(s3Uri.key().orElseThrow(() -> new
 URISyntaxException(unloadResponse.getMetadataFile(), "Invalid S3 URI")))
            .build());
    String metadataFileContent = new String(objectBytes.asByteArray(),
 StandardCharsets.UTF_8);
    final Gson gson = new GsonBuilder()
            .setFieldNamingPolicy(FieldNamingPolicy.UPPER_CAMEL_CASE)
            .create();
    return gson.fromJson(metadataFileContent, UnloadMetadata.class);
}

class UnloadMetadata {
    @JsonProperty("ColumnInfo")
    List<ColumnInfo> columnInfo;
    @JsonProperty("Author")
    Author author;

    @Data
    public class Author {
        @JsonProperty("Name")
        String name;
        @JsonProperty("MetadataFileVersion")
        String metadataFileVersion;
    }
}
```

Go

```
// Read and parse metadata content

func getMetadataFile(s3Svc *s3.S3, response map[string]string) Metadata {
    var metadataBuf = getObject(s3Svc, response["metadataFile"])
    var metadata Metadata
    json.Unmarshal(metadataBuf.Bytes(), &metadata)
    return metadata
}

func getObject(s3Svc *s3.S3, s3Uri string) *bytes.Buffer {
    u,_ := url.Parse(s3Uri)
```

```
    getObjectInput := &s3.GetObjectInput{
        Key:    aws.String(u.Path),
        Bucket: aws.String(u.Host),
    }
    getObjectOutput, err := s3Svc.GetObject(getObjectInput)
    if err != nil {
        fmt.Println("Error: %s\n", err.Error())
    }
    buf := new(bytes.Buffer)
    buf.ReadFrom(getObjectOutput.Body)
    return buf
}

// Unload's Metadata structure

type Metadata struct {
    Author interface{}
    ColumnInfo []struct {
        Name string
        Type map[string]string
    }
}
```

Python

```
def __get_metadata_file(self, response):
    metadata = self.get_object(response['metadataFile']).read().decode('utf-8')
    parsed_metadata = json.loads(metadata)
   print("Metadata contents: \n%s" % parsed_metadata)

def get_object(self, uri):
    try:
        bucket, key = uri.replace("s3://", "").split("/", 1)
        s3_client = boto3.client('s3', region_name=<region>)
        response = s3_client.get_object(Bucket=bucket, Key=key)
        return response['Body']
    except Exception as err:
        print("Failed to get the object for URI:", uri)
        raise err
```

Node.js

```
// Read and parse metadata content
async getMetadataFile(response) {
    let metadata;
    await this.getS3Object(response['metadataFile']).then(
        (data) => {
            metadata = JSON.parse(data);
        }
    );
    return metadata;
}

async getS3Object(uri) {
    const {bucketName, key} = this.getBucketAndKey(uri);
    const params = new GetObjectCommand({
        Bucket: bucketName,
        Key: key
    })
    const response = await this.s3Client.send(params);
    return await response.Body.transformToString();
}

getBucketAndKey(uri) {
    const [bucketName] = uri.replace("s3://", "").split("/", 1);
    const key = uri.replace("s3://", "").split('/').slice(1).join('/');
    return {bucketName, key};
}
```

Cancelar consulta

Puede usar los siguientes fragmentos de código para cancelar una consulta.

> **Note**
>
> Estos fragmentos de código se basan en aplicaciones de muestra completas en. GitHub Para obtener más información sobre cómo empezar a utilizar las aplicaciones de ejemplo, consulte. Aplicación de muestra

Java

```
   public void cancelQuery() {
       System.out.println("Starting query: " + SELECT_ALL_QUERY);
       QueryRequest queryRequest = new QueryRequest();
       queryRequest.setQueryString(SELECT_ALL_QUERY);
       QueryResult queryResult = queryClient.query(queryRequest);

       System.out.println("Cancelling the query: " + SELECT_ALL_QUERY);
       final CancelQueryRequest cancelQueryRequest = new CancelQueryRequest();
       cancelQueryRequest.setQueryId(queryResult.getQueryId());
       try {
           queryClient.cancelQuery(cancelQueryRequest);
           System.out.println("Query has been successfully cancelled");
       } catch (Exception e) {
           System.out.println("Could not cancel the query: " + SELECT_ALL_QUERY + "
= " + e);
       }
   }
```

Java v2

```
   public void cancelQuery() {
       System.out.println("Starting query: " + SELECT_ALL_QUERY);
       QueryRequest queryRequest =
QueryRequest.builder().queryString(SELECT_ALL_QUERY).build();
       QueryResponse queryResponse = timestreamQueryClient.query(queryRequest);

       System.out.println("Cancelling the query: " + SELECT_ALL_QUERY);
       final CancelQueryRequest cancelQueryRequest = CancelQueryRequest.builder()
               .queryId(queryResponse.queryId()).build();
       try {
           timestreamQueryClient.cancelQuery(cancelQueryRequest);
           System.out.println("Query has been successfully cancelled");
       } catch (Exception e) {
           System.out.println("Could not cancel the query: " + SELECT_ALL_QUERY + "
= " + e);
       }
   }
```

Go

```
cancelQueryInput := &timestreamquery.CancelQueryInput{
```

```
    QueryId: aws.String(*queryOutput.QueryId),
}

fmt.Println("Submitting cancellation for the query")
fmt.Println(cancelQueryInput)

// submit the query
cancelQueryOutput, err := querySvc.CancelQuery(cancelQueryInput)

if err != nil {
    fmt.Println("Error:")
    fmt.Println(err)
} else {
  fmt.Println("Query has been cancelled successfully")
  fmt.Println(cancelQueryOutput)
}
```

Python

```
  def cancel_query(self):
      print("Starting query: " + self.SELECT_ALL)
      result = self.client.query(QueryString=self.SELECT_ALL)
      print("Cancelling query: " + self.SELECT_ALL)
      try:
          self.client.cancel_query(QueryId=result['QueryId'])
          print("Query has been successfully cancelled")
      except Exception as err:
          print("Cancelling query failed:", err)
```

Node.js

En el siguiente fragmento se utiliza el estilo AWS SDK for JavaScript V2. Se basa en la aplicación de ejemplo de Node.js, ejemplo de Amazon Timestream LiveAnalytics para su aplicación en. GitHub

```
async function tryCancelQuery() {
    const params = {
        QueryString: SELECT_ALL_QUERY
    };
    console.log(`Running query: ${SELECT_ALL_QUERY}`);

    await queryClient.query(params).promise()
        .then(
```

```
            async (response) => {
                await cancelQuery(response.QueryId);
            },
            (err) => {
                console.error("Error while executing select all query:", err);
            });
}

async function cancelQuery(queryId) {
    const cancelParams = {
        QueryId: queryId
    };
    console.log(`Sending cancellation for query: ${SELECT_ALL_QUERY}`);
    await queryClient.cancelQuery(cancelParams).promise()
        .then(
            (response) => {
                console.log("Query has been cancelled successfully");
            },
            (err) => {
                console.error("Error while cancelling select all:", err);
            });
}
```

.NET

```
        public async Task CancelQuery()
        {
            Console.WriteLine("Starting query: " + SELECT_ALL_QUERY);
            QueryRequest queryRequest = new QueryRequest();
            queryRequest.QueryString = SELECT_ALL_QUERY;
            QueryResponse queryResponse = await
 queryClient.QueryAsync(queryRequest);

            Console.WriteLine("Cancelling query: " + SELECT_ALL_QUERY);
            CancelQueryRequest cancelQueryRequest = new CancelQueryRequest();
            cancelQueryRequest.QueryId = queryResponse.QueryId;

            try
            {
                await queryClient.CancelQueryAsync(cancelQueryRequest);
                Console.WriteLine("Query has been successfully cancelled.");
            } catch(Exception e)
            {
```

```
                Console.WriteLine("Could not cancel the query: " + SELECT_ALL_QUERY
 + " = " + e);
            }
        }
```

Crear tarea de carga por lotes

Puede usar los siguientes fragmentos de código para crear tareas de carga por lotes.

Java

```
package com.example.tryit;

import java.util.Arrays;

import
 software.amazon.awssdk.services.timestreamwrite.model.CreateBatchLoadTaskRequest;
import
 software.amazon.awssdk.services.timestreamwrite.model.CreateBatchLoadTaskResponse;
import software.amazon.awssdk.services.timestreamwrite.model.DataModel;
import software.amazon.awssdk.services.timestreamwrite.model.DataModelConfiguration;
import
 software.amazon.awssdk.services.timestreamwrite.model.DataSourceConfiguration;
import
 software.amazon.awssdk.services.timestreamwrite.model.DataSourceS3Configuration;
import software.amazon.awssdk.services.timestreamwrite.model.DimensionMapping;
import
 software.amazon.awssdk.services.timestreamwrite.model.MultiMeasureAttributeMapping;
import software.amazon.awssdk.services.timestreamwrite.model.MultiMeasureMappings;
import software.amazon.awssdk.services.timestreamwrite.model.ReportConfiguration;
import software.amazon.awssdk.services.timestreamwrite.model.ReportS3Configuration;
import software.amazon.awssdk.services.timestreamwrite.model.ScalarMeasureValueType;
import software.amazon.awssdk.services.timestreamwrite.model.TimeUnit;
import software.amazon.awssdk.services.timestreamwrite.TimestreamWriteClient;

public class BatchLoadExample {
    public static final String DATABASE_NAME = <database name>;
    public static final String TABLE_NAME = <table name>;
    public static final String INPUT_BUCKET = <S3 location>;
    public static final String INPUT_OBJECT_KEY_PREFIX = <CSV filename>;
    public static final String REPORT_BUCKET = <S3 location>;
    public static final long HT_TTL_HOURS = 24L;
```

```
    public static final long CT_TTL_DAYS = 7L;

    TimestreamWriteClient amazonTimestreamWrite;

    public BatchLoadExample(TimestreamWriteClient client) {
        this.amazonTimestreamWrite = client;
    }

    public String createBatchLoadTask() {
        System.out.println("Creating batch load task");

        CreateBatchLoadTaskRequest request = CreateBatchLoadTaskRequest.builder()
                .dataModelConfiguration(DataModelConfiguration.builder()
                        .dataModel(DataModel.builder()
                                .timeColumn("timestamp")
                                .timeUnit(TimeUnit.SECONDS)
                                .dimensionMappings(Arrays.asList(
                                        DimensionMapping.builder()
                                                .sourceColumn("vehicle")
                                                .build(),
                                        DimensionMapping.builder()
                                                .sourceColumn("registration")
                                                .destinationColumn("license")
                                                .build()))
                                .multiMeasureMappings(MultiMeasureMappings.builder()
                                        .targetMultiMeasureName("mva_measure_name")

.multiMeasureAttributeMappings(Arrays.asList(

MultiMeasureAttributeMapping.builder()
                                                        .sourceColumn("wgt")

.targetMultiMeasureAttributeName("weight")

.measureValueType(ScalarMeasureValueType.DOUBLE)
                                                        .build(),

MultiMeasureAttributeMapping.builder()
                                                        .sourceColumn("spd")

.targetMultiMeasureAttributeName("speed")

.measureValueType(ScalarMeasureValueType.DOUBLE)
                                                        .build(),
```

```
 MultiMeasureAttributeMapping.builder()
                                                        .sourceColumn("fuel")

 .measureValueType(ScalarMeasureValueType.DOUBLE)
                                                        .build(),

 MultiMeasureAttributeMapping.builder()
                                                        .sourceColumn("miles")

 .measureValueType(ScalarMeasureValueType.DOUBLE)
                                                        .build()))
                                        .build())
                                .build())
                        .build())
                .dataSourceConfiguration(DataSourceConfiguration.builder()
                        .dataSourceS3Configuration(
                                DataSourceS3Configuration.builder()
                                        .bucketName(INPUT_BUCKET)
                                        .objectKeyPrefix(INPUT_OBJECT_KEY_PREFIX)
                                        .build())
                        .dataFormat("CSV")
                        .build())
                .reportConfiguration(ReportConfiguration.builder()
                        .reportS3Configuration(ReportS3Configuration.builder()
                                .bucketName(REPORT_BUCKET)
                                .build())
                        .build())
                .targetDatabaseName(DATABASE_NAME)
                .targetTableName(TABLE_NAME)
                .build();
        try {
            final CreateBatchLoadTaskResponse createBatchLoadTaskResponse =
 amazonTimestreamWrite.createBatchLoadTask(request);
            String taskId = createBatchLoadTaskResponse.taskId();
            System.out.println("Successfully created batch load task: " + taskId);
            return taskId;
        } catch (Exception e) {
            System.out.println("Failed to create batch load task: " + e);
            throw e;
        }
    }
}
```

Go

```
package main

import (
 "fmt"
 "context"
 "log"
 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/service/timestreamwrite"
 "github.com/aws/aws-sdk-go-v2/service/timestreamwrite/types"
)

func main() {
 customResolver := aws.EndpointResolverWithOptionsFunc(func(service, region string,
 options ...interface{})(aws.Endpoint, error) {
  if service == timestreamwrite.ServiceID &&  region == "us-west-2" {
      return aws.Endpoint{
          PartitionID:   "aws",
          URL:           <URL>,
          SigningRegion: "us-west-2",
      }, nil
  }
  return aws.Endpoint{}, &  aws.EndpointNotFoundError{}
 })

 cfg, err := config.LoadDefaultConfig(context.TODO(),
 config.WithEndpointResolverWithOptions(customResolver), config.WithRegion("us-
west-2"))

 if err != nil {
    log.Fatalf("failed to load configuration, %v", err)
 }

 client := timestreamwrite.NewFromConfig(cfg)

 response, err := client.CreateBatchLoadTask(context.TODO(), &
 timestreamwrite.CreateBatchLoadTaskInput{
            TargetDatabaseName: aws.String("BatchLoadExampleDatabase"),
            TargetTableName: aws.String("BatchLoadExampleTable"),
            RecordVersion: aws.Int64(1),
            DataModelConfiguration: &  types.DataModelConfiguration{
                DataModel: &  types.DataModel{
```

```
                    TimeColumn: aws.String("timestamp"),
                    TimeUnit: types.TimeUnitMilliseconds,
                    DimensionMappings: []types.DimensionMapping{
                        {
                            SourceColumn: aws.String("registration"),
                            DestinationColumn: aws.String("license"),
                        },
                    },
                    MultiMeasureMappings: &  types.MultiMeasureMappings{
                        TargetMultiMeasureName: aws.String("mva_measure_name"),
                        MultiMeasureAttributeMappings:
[]types.MultiMeasureAttributeMapping{
                            {
                                SourceColumn: aws.String("wgt"),
                                TargetMultiMeasureAttributeName:
aws.String("weight"),
                                MeasureValueType:
types.ScalarMeasureValueTypeDouble,
                            },
                            {
                                SourceColumn: aws.String("spd"),
                                TargetMultiMeasureAttributeName:
aws.String("speed"),
                                MeasureValueType:
types.ScalarMeasureValueTypeDouble,
                            },
                            {
                                SourceColumn: aws.String("fuel_consumption"),
                                TargetMultiMeasureAttributeName: aws.String("fuel"),
                                MeasureValueType:
types.ScalarMeasureValueTypeDouble,
                            },
                        },
                    },
                },
            },
          DataSourceConfiguration: &  types.DataSourceConfiguration{
               DataSourceS3Configuration: &  types.DataSourceS3Configuration{
                    BucketName: aws.String("test-batch-load-west-2"),
                    ObjectKeyPrefix: aws.String("sample.csv"),
                },
                DataFormat: types.BatchLoadDataFormatCsv,
            },
          ReportConfiguration: & types.ReportConfiguration{
```

```
                ReportS3Configuration: & types.ReportS3Configuration{
                    BucketName: aws.String("test-batch-load-report-west-2"),
                    EncryptionOption: types.S3EncryptionOptionSseS3,
                },
            },
 })

 fmt.Println(aws.ToString(response.TaskId))
}
```

Python

```
import boto3
from botocore.config import Config

INGEST_ENDPOINT = "<URL>"
REGION = "us-west-2"
HT_TTL_HOURS = 24
CT_TTL_DAYS = 7
DATABASE_NAME = "<database name>"
TABLE_NAME = "<table name>"
INPUT_BUCKET_NAME = "<S3 location>"
INPUT_OBJECT_KEY_PREFIX = "<CSV file name>"
REPORT_BUCKET_NAME = "<S3 location>"


def create_batch_load_task(client, database_name, table_name, input_bucket_name,
 input_object_key_prefix, report_bucket_name):
    try:
        result = client.create_batch_load_task(TargetDatabaseName=database_name,
 TargetTableName=table_name,
                                               DataModelConfiguration={"DataModel":
 {
                                                   "TimeColumn": "timestamp",
                                                   "TimeUnit": "SECONDS",
                                                   "DimensionMappings": [
                                                       {
                                                           "SourceColumn": "vehicle"
                                                       },
                                                       {
                                                           "SourceColumn":
 "registration",
```

```
                                                        "DestinationColumn":
"license"
                                                    }
                                                ],
                                                "MultiMeasureMappings": {
                                                    "TargetMultiMeasureName":
"metrics",

"MultiMeasureAttributeMappings": [
                                                        {
                                                            "SourceColumn":
"wgt",
                                                            "MeasureValueType":
"DOUBLE"
                                                        },
                                                        {
                                                            "SourceColumn":
"spd",
                                                            "MeasureValueType":
"DOUBLE"
                                                        },
                                                        {
                                                            "SourceColumn":
"fuel_consumption",

"TargetMultiMeasureAttributeName": "fuel",
                                                            "MeasureValueType":
"DOUBLE"
                                                        },
                                                        {
                                                            "SourceColumn":
"miles",
                                                            "MeasureValueType":
"DOUBLE"
                                                        }
                                                    ]}
                                            }
                                            },
                                            DataSourceConfiguration={
                                                "DataSourceS3Configuration": {
                                                    "BucketName":
input_bucket_name,
                                                    "ObjectKeyPrefix":
input_object_key_prefix
```

```
                                                    },
                                                    "DataFormat": "CSV"
                                                },
                                                ReportConfiguration={
                                                    "ReportS3Configuration": {
                                                        "BucketName":
 report_bucket_name,
                                                        "EncryptionOption": "SSE_S3"
                                                    }
                                                }
                                                )

        task_id = result["TaskId"]
        print("Successfully created batch load task: ", task_id)
        return task_id
    except Exception as err:
        print("Create batch load task job failed:", err)
        return None


if __name__ == '__main__':
    session = boto3.Session()

    write_client = session.client('timestream-write',
                                  endpoint_url=INGEST_ENDPOINT, region_name=REGION,
                                  config=Config(read_timeout=20,
 max_pool_connections=5000, retries={'max_attempts': 10}))

    task_id = create_batch_load_task(write_client, DATABASE_NAME, TABLE_NAME,
                                     INPUT_BUCKET_NAME, INPUT_OBJECT_KEY_PREFIX,
 REPORT_BUCKET_NAME)
```

Node.js

El siguiente fragmento se usa para la versión 3. AWS SDK JavaScript Para obtener más información sobre cómo instalar el cliente y su uso, consulte Timestream Write Client, para la versión 3. AWS SDK JavaScript

Para obtener API más información, consulte Clase y. CreateBatchLoadCommand CreateBatchLoadTask

```
import { TimestreamWriteClient, CreateBatchLoadTaskCommand } from "@aws-sdk/client-
timestream-write";
```

```
const writeClient = new TimestreamWriteClient({ region: "us-west-2", endpoint:
 "https://gamma-ingest-cell3.timestream.us-west-2.amazonaws.com" });

const params = {
    TargetDatabaseName: "BatchLoadExampleDatabase",
    TargetTableName: "BatchLoadExampleTable",
 RecordVersion: 1,
 DataModelConfiguration: {
  DataModel: {
   TimeColumn: "timestamp",
            TimeUnit: "MILLISECONDS",
            DimensionMappings: [
                {
                    SourceColumn: "registration",
                    DestinationColumn: "license"
                }
            ],
            MultiMeasureMappings: {
                TargetMultiMeasureName: "mva_measure_name",
                MultiMeasureAttributeMappings: [
                    {
                        SourceColumn: "wgt",
                        TargetMultiMeasureAttributeName: "weight",
                        MeasureValueType: "DOUBLE"
                    },
                    {
                        SourceColumn: "spd",
                        TargetMultiMeasureAttributeName: "speed",
                        MeasureValueType: "DOUBLE"
                    },
                    {
                        SourceColumn: "fuel_consumption",
                        TargetMultiMeasureAttributeName: "fuel",
                        MeasureValueType: "DOUBLE"
                    }
                ]
            }
        }
    },
 DataSourceConfiguration: {
        DataSourceS3Configuration: {
            BucketName: "test-batch-load-west-2",
            ObjectKeyPrefix: "sample.csv"
        },
```

```
        DataFormat: "CSV"
    },
    ReportConfiguration: {
        ReportS3Configuration: {
            BucketName: "test-batch-load-report-west-2",
            EncryptionOption: "SSE_S3"
        }
    }
};

const command = new CreateBatchLoadTaskCommand(params);

try {
    const data = await writeClient.send(command);
    console.log(`Created batch load task ` + data.TaskId);
} catch (error) {
 console.log("Error creating table. ", error);
 throw error;
}
```

.NET

```
using System;
using System.IO;
using System.Collections.Generic;
using Amazon.TimestreamWrite;
using Amazon.TimestreamWrite.Model;
using System.Threading.Tasks;

namespace TimestreamDotNetSample
{
    public class CreateBatchLoadTaskExample
    {
        public const string DATABASE_NAME = "<database name>";
        public const string TABLE_NAME = "<table name>";
        public const string INPUT_BUCKET = "<input bucket name>";
        public const string INPUT_OBJECT_KEY_PREFIX = "<CSV file name>";
        public const string REPORT_BUCKET = "<report bucket name>";
        public const long HT_TTL_HOURS = 24L;
        public const long CT_TTL_DAYS = 7L;
        private readonly AmazonTimestreamWriteClient writeClient;

        public CreateBatchLoadTaskExample(AmazonTimestreamWriteClient writeClient)
```

```
        {
            this.writeClient = writeClient;
        }

        public async Task CreateBatchLoadTask()
        {
            try
            {
                var createBatchLoadTaskRequest = new CreateBatchLoadTaskRequest
                {
                    DataModelConfiguration = new DataModelConfiguration
                    {
                        DataModel = new DataModel
                        {
                            TimeColumn = "timestamp",
                            TimeUnit = TimeUnit.SECONDS,
                            DimensionMappings = new List<DimensionMapping>()
                            {
                                new()
                                {
                                        SourceColumn = "vehicle"
                                },
                                new()
                                {
                                        SourceColumn = "registration",
                                        DestinationColumn = "license"
                                }
                            },
                            MultiMeasureMappings = new MultiMeasureMappings
                            {
                                TargetMultiMeasureName = "mva_measure_name",
                                MultiMeasureAttributeMappings = new
List<MultiMeasureAttributeMapping>()
                                {
                                        new()
                                        {
                                                SourceColumn = "wgt",
                                                TargetMultiMeasureAttributeName =
"weight",
                                                MeasureValueType =
ScalarMeasureValueType.DOUBLE
                                        },
                                        new()
                                        {
```

```
                                                SourceColumn = "spd",
                                                TargetMultiMeasureAttributeName =
"speed",
                                                MeasureValueType =
ScalarMeasureValueType.DOUBLE
                                            },
                                            new()
                                            {
                                                SourceColumn = "fuel",
                                                TargetMultiMeasureAttributeName =
"fuel",
                                                MeasureValueType =
ScalarMeasureValueType.DOUBLE
                                            },
                                            new()
                                            {
                                                SourceColumn = "miles",
                                                TargetMultiMeasureAttributeName =
"miles",
                                                MeasureValueType =
ScalarMeasureValueType.DOUBLE
                                            }
                                        }
                                    }
                                }
                            },
                            DataSourceConfiguration = new DataSourceConfiguration
                            {
                                DataSourceS3Configuration = new DataSourceS3Configuration
                                {
                                    BucketName = INPUT_BUCKET,
                                    ObjectKeyPrefix = INPUT_OBJECT_KEY_PREFIX
                                },
                                DataFormat = "CSV"
                            },
                            ReportConfiguration = new ReportConfiguration
                            {
                                ReportS3Configuration = new ReportS3Configuration
                                {
                                    BucketName = REPORT_BUCKET
                                }
                            },
                            TargetDatabaseName = DATABASE_NAME,
                            TargetTableName = TABLE_NAME
```

```
                };

                CreateBatchLoadTaskResponse response = await
 writeClient.CreateBatchLoadTaskAsync(createBatchLoadTaskRequest);
                Console.WriteLine($"Task created: " + response.TaskId);
            }
            catch (Exception e)
            {
                Console.WriteLine("Create batch load task failed:" + e.ToString());
            }
        }
    }
}
```

```
using Amazon.TimestreamWrite;
using Amazon.TimestreamWrite.Model;
using Amazon;
using Amazon.TimestreamQuery;
using System.Threading.Tasks;
using System;
using CommandLine;
static class Constants
{

}
namespace TimestreamDotNetSample
{
    class MainClass
    {
        public class Options
        {

        }
        public static void Main(string[] args)
        {
            Parser.Default.ParseArguments<Options>(args)
                .WithParsed<Options>(o => {
                    MainAsync().GetAwaiter().GetResult();
                });
        }

        static async Task MainAsync()
        {
```

```
            var writeClientConfig = new AmazonTimestreamWriteConfig
            {
                ServiceURL =  "<service URL>",
                Timeout = TimeSpan.FromSeconds(20),
                MaxErrorRetry = 10
            };

            var writeClient = new AmazonTimestreamWriteClient(writeClientConfig);
            var example = new CreateBatchLoadTaskExample(writeClient);
            await example.CreateBatchLoadTask();
        }
    }
}
```

Describa la tarea de carga por lotes

Puede utilizar los siguientes fragmentos de código para describir las tareas de carga por lotes.

Java

```
   public void describeBatchLoadTask(String taskId) {
           final DescribeBatchLoadTaskResponse batchLoadTaskResponse =
 amazonTimestreamWrite

 .describeBatchLoadTask(DescribeBatchLoadTaskRequest.builder()
                                          .taskId(taskId)
                                          .build());

           System.out.println("Task id: " +
 batchLoadTaskResponse.batchLoadTaskDescription().taskId());
           System.out.println("Status: " +
 batchLoadTaskResponse.batchLoadTaskDescription().taskStatusAsString());
           System.out.println("Records processed: "
                           +
 batchLoadTaskResponse.batchLoadTaskDescription().progressReport().recordsProcessed());
   }
```

Go

```
package main

import (
```

```
 "fmt"
 "context"
 "log"
 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/service/timestreamwrite"
)

func main() {
 customResolver := aws.EndpointResolverWithOptionsFunc(func(service, region string,
 options ...interface{}) (aws.Endpoint, error) {
  if service == timestreamwrite.ServiceID && region == "us-west-2" {
      return aws.Endpoint{
          PartitionID:   "aws",
          URL:           <URL>,
          SigningRegion: "us-west-2",
      }, nil
  }
  return aws.Endpoint{}, &aws.EndpointNotFoundError{}
 })

 cfg, err := config.LoadDefaultConfig(context.TODO(),
 config.WithEndpointResolverWithOptions(customResolver), config.WithRegion("us-
west-2"))

 if err != nil {
    log.Fatalf("failed to load configuration, %v", err)
 }

 client := timestreamwrite.NewFromConfig(cfg)

 response, err := client.DescribeBatchLoadTask(context.TODO(),
 &timestreamwrite.DescribeBatchLoadTaskInput{
  TaskId: aws.String("<TaskId>"),
 })

 fmt.Println(aws.ToString(response.BatchLoadTaskDescription.TaskId))
}
```

Python

```
import boto3
from botocore.config import Config
```

```
INGEST_ENDPOINT="<url>"
REGION="us-west-2"
HT_TTL_HOURS = 24
CT_TTL_DAYS = 7
TASK_ID = "<task id>"

def describe_batch_load_task(client, task_id):
    try:
        result = client.describe_batch_load_task(TaskId=task_id)
        print("Successfully described batch load task: ", result)
    except Exception as err:
        print("Describe batch load task job failed:", err)


if __name__ == '__main__':
    session = boto3.Session()

    write_client = session.client('timestream-write', \
        endpoint_url=INGEST_ENDPOINT, region_name=REGION, \
        config=Config(read_timeout=20, max_pool_connections = 5000,
 retries={'max_attempts': 10}))

    describe_batch_load_task(write_client, TASK_ID)
```

Node.js

El siguiente fragmento se usa para la versión 3. AWS SDK JavaScript Para obtener más información sobre cómo instalar el cliente y su uso, consulte Timestream Write Client, para la versión 3. AWS SDK JavaScript

Para obtener API más información, consulte Clase y. DescribeBatchLoadCommand DescribeBatchLoadTask

```
import { TimestreamWriteClient, DescribeBatchLoadTaskCommand } from "@aws-sdk/
client-timestream-write";
const writeClient = new TimestreamWriteClient({ region: "<region>", endpoint:
 "<endpoint>" });

const params = {
    TaskId: "<TaskId>"
};
```

```
const command = new DescribeBatchLoadTaskCommand(params);

try {
    const data = await writeClient.send(command);
    console.log(`Batch load task has id ` + data.BatchLoadTaskDescription.TaskId);
} catch (error) {
    if (error.code === 'ResourceNotFoundException') {
        console.log("Batch load task doesn't exist.");
    } else {
        console.log("Describe batch load task failed.", error);
        throw error;
    }
}
```

.NET

```
using System;
using System.IO;
using System.Collections.Generic;
using Amazon.TimestreamWrite;
using Amazon.TimestreamWrite.Model;
using System.Threading.Tasks;

namespace TimestreamDotNetSample
{
    public class DescribeBatchLoadTaskExample
    {
        private readonly AmazonTimestreamWriteClient writeClient;

        public DescribeBatchLoadTaskExample(AmazonTimestreamWriteClient writeClient)
        {
            this.writeClient = writeClient;
        }

        public async Task DescribeBatchLoadTask(String taskId)
        {
            try
            {
                var describeBatchLoadTaskRequest = new DescribeBatchLoadTaskRequest
                {
                    TaskId = taskId
                };
```

```
                DescribeBatchLoadTaskResponse response = await
 writeClient.DescribeBatchLoadTaskAsync(describeBatchLoadTaskRequest);
                Console.WriteLine($"Task has id:
{response.BatchLoadTaskDescription.TaskId}");
            }
            catch (ResourceNotFoundException)
            {
                Console.WriteLine("Batch load task does not exist.");
            }
            catch (Exception e)
            {
                Console.WriteLine("Describe batch load task failed:" +
 e.ToString());
            }
        }
    }
}
```

```
using Amazon.TimestreamWrite;
using Amazon.TimestreamWrite.Model;
using Amazon;
using Amazon.TimestreamQuery;
using System.Threading.Tasks;
using System;
using CommandLine;
static class Constants
{

}
namespace TimestreamDotNetSample
{
    class MainClass
    {
        public class Options
        {

        }
        public static void Main(string[] args)
        {
            Parser.Default.ParseArguments<Options>(args)
                .WithParsed<Options>(o => {
                    MainAsync().GetAwaiter().GetResult();
                });
```

```
        }

        static async Task MainAsync()
        {
            var writeClientConfig = new AmazonTimestreamWriteConfig
            {
                ServiceURL =  "<service URL>",
                Timeout = TimeSpan.FromSeconds(20),
                MaxErrorRetry = 10
            };

            var writeClient = new AmazonTimestreamWriteClient(writeClientConfig);
            var example = new DescribeBatchLoadTaskExample(writeClient);
            await example.DescribeBatchLoadTask("<batch load task id>");
        }
    }
}
```

Listar tareas de carga por lotes

Puede usar los siguientes fragmentos de código para enumerar las tareas de carga por lotes.

Java

```
    public void listBatchLoadTasks() {
            final ListBatchLoadTasksResponse listBatchLoadTasksResponse =
 amazonTimestreamWrite
                            .listBatchLoadTasks(ListBatchLoadTasksRequest.builder()
                                    .maxResults(15)
                                    .build());

            for (BatchLoadTask batchLoadTask :
 listBatchLoadTasksResponse.batchLoadTasks()) {
                    System.out.println(batchLoadTask.taskId());
            }
    }
```

Go

```
package main

import (
```

```
 "fmt"
 "context"
 "log"
 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/service/timestreamwrite"
)

func main() {
 customResolver := aws.EndpointResolverWithOptionsFunc(func(service, region string,
 options ...interface{}) (aws.Endpoint, error) {
  if service == timestreamwrite.ServiceID && region == "us-west-2" {
      return aws.Endpoint{
          PartitionID:   "aws",
          URL:           <URL>,
          SigningRegion: "us-west-2",
      }, nil
  }
  return aws.Endpoint{}, &aws.EndpointNotFoundError{}
 })

 cfg, err := config.LoadDefaultConfig(context.TODO(),
 config.WithEndpointResolverWithOptions(customResolver), config.WithRegion("us-
west-2"))

 if err != nil {
    log.Fatalf("failed to load configuration, %v", err)
 }

 client := timestreamwrite.NewFromConfig(cfg)
 listBatchLoadTasksMaxResult := int32(15)

 response, err := client.ListBatchLoadTasks(context.TODO(),
 &timestreamwrite.ListBatchLoadTasksInput{
  MaxResults: &listBatchLoadTasksMaxResult,
 })

 for i, task := range response.BatchLoadTasks {
  fmt.Println(i, aws.ToString(task.TaskId))
 }
}
```

Python

```
import boto3
from botocore.config import Config

INGEST_ENDPOINT = "<url>"
REGION = "us-west-2"
HT_TTL_HOURS = 24
CT_TTL_DAYS = 7


def print_batch_load_tasks(batch_load_tasks):
    for batch_load_task in batch_load_tasks:
        print(batch_load_task['TaskId'])


def list_batch_load_tasks(client):
    print("\nListing batch load tasks")
    try:
        response = client.list_batch_load_tasks(MaxResults=10)
        print_batch_load_tasks(response['BatchLoadTasks'])
        next_token = response.get('NextToken', None)
        while next_token:
            response = client.list_batch_load_tasks(
                NextToken=next_token, MaxResults=10)
            print_batch_load_tasks(response['BatchLoadTasks'])
            next_token = response.get('NextToken', None)
    except Exception as err:
        print("List batch load tasks failed:", err)
        raise err


if __name__ == '__main__':
    session = boto3.Session()

    write_client = session.client('timestream-write',
                                  endpoint_url=INGEST_ENDPOINT, region_name=REGION,
                                  config=Config(read_timeout=20,
 max_pool_connections=5000, retries={'max_attempts': 10}))

    list_batch_load_tasks(write_client)
```

Node.js

El siguiente fragmento se usa para la versión 3. AWS SDK JavaScript Para obtener más información sobre cómo instalar el cliente y su uso, consulte Timestream Write Client, para la versión 3. AWS SDK JavaScript

Para obtener API más información, consulte Clase y. DescribeBatchLoadCommand DescribeBatchLoadTask

```
import { TimestreamWriteClient, ListBatchLoadTasksCommand } from "@aws-sdk/client-
timestream-write";
const writeClient = new TimestreamWriteClient({ region: "<region>", endpoint:
 "<endpoint>" });

const params = {
    MaxResults: <15>
};

const command = new ListBatchLoadTasksCommand(params);

getBatchLoadTasksList(null);

async function getBatchLoadTasksList(nextToken) {
    if (nextToken) {
        params.NextToken = nextToken;
    }

    try {
        const data = await writeClient.send(command);

        data.BatchLoadTasks.forEach(function (task) {
            console.log(task.TaskId);
        });

        if (data.NextToken) {
            return getBatchLoadTasksList(data.NextToken);
        }
    } catch (error) {
        console.log("Error while listing batch load tasks", error);
    }
}
```

.NET

```
using System;
using System.IO;
using System.Collections.Generic;
using Amazon.TimestreamWrite;
using Amazon.TimestreamWrite.Model;
using System.Threading.Tasks;

namespace TimestreamDotNetSample
{
    public class ListBatchLoadTasksExample
    {
        private readonly AmazonTimestreamWriteClient writeClient;

        public ListBatchLoadTasksExample(AmazonTimestreamWriteClient writeClient)
        {
            this.writeClient = writeClient;
        }

        public async Task ListBatchLoadTasks()
        {
            Console.WriteLine("Listing batch load tasks");

            try
            {
                var listBatchLoadTasksRequest = new ListBatchLoadTasksRequest
                {
                    MaxResults = 15
                };

                ListBatchLoadTasksResponse response = await
 writeClient.ListBatchLoadTasksAsync(listBatchLoadTasksRequest);

                PrintBatchLoadTasks(response.BatchLoadTasks);
                var nextToken = response.NextToken;

                while (nextToken != null)
                {
                    listBatchLoadTasksRequest.NextToken = nextToken;
                    response = await
 writeClient.ListBatchLoadTasksAsync(listBatchLoadTasksRequest);
                    PrintBatchLoadTasks(response.BatchLoadTasks);
                    nextToken = response.NextToken;
```

```
                }
            }
            catch (Exception e)
            {
                Console.WriteLine("List batch load tasks failed:" + e.ToString());
            }
        }

        private void PrintBatchLoadTasks(List<BatchLoadTask> tasks)
        {
            foreach (BatchLoadTask task in tasks)
                Console.WriteLine($"Task:{task.TaskId}");
        }
    }
}
```

```
using Amazon.TimestreamWrite;
using Amazon.TimestreamWrite.Model;
using Amazon;
using Amazon.TimestreamQuery;
using System.Threading.Tasks;
using System;
using CommandLine;
static class Constants
{

}
namespace TimestreamDotNetSample
{
    class MainClass
    {
        public class Options
        {

        }
        public static void Main(string[] args)
        {
            Parser.Default.ParseArguments<Options>(args)
                .WithParsed<Options>(o => {
                    MainAsync().GetAwaiter().GetResult();
                });
        }
```

```
        static async Task MainAsync()
        {
            var writeClientConfig = new AmazonTimestreamWriteConfig
            {
                ServiceURL =  "<service URL>",
                Timeout = TimeSpan.FromSeconds(20),
                MaxErrorRetry = 10
            };

            var writeClient = new AmazonTimestreamWriteClient(writeClientConfig);
            var example = new ListBatchLoadTasksExample(writeClient);
            await example.ListBatchLoadTasks();
        }
    }
}
```

Reanudar tarea de carga por lotes

Puede utilizar los siguientes fragmentos de código para reanudar las tareas de carga por lotes.

Java

```
    public void resumeBatchLoadTask(String taskId) {
            try {
                    amazonTimestreamWrite

 .resumeBatchLoadTask(ResumeBatchLoadTaskRequest.builder()
                                                    .taskId(taskId)
                                                    .build());

                    System.out.println("Successfully resumed batch load task.");
            } catch (ValidationException validationException) {
                    System.out.println(validationException.getMessage());
            }
    }
```

Go

```
package main

import (
 "fmt"
```

```
 "context"
 "log"
 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/service/timestreamwrite"
)

func main() {
 customResolver := aws.EndpointResolverWithOptionsFunc(func(service, region string,
 options ...interface{}) (aws.Endpoint, error) {
  if service == timestreamwrite.ServiceID && region == "us-west-2" {
      return aws.Endpoint{
          PartitionID:   "aws",
          URL:           <URL>,
          SigningRegion: "us-west-2",
      }, nil
  }
  return aws.Endpoint{}, &aws.EndpointNotFoundError{}
 })

 cfg, err := config.LoadDefaultConfig(context.TODO(),
 config.WithEndpointResolverWithOptions(customResolver), config.WithRegion("us-
west-2"))

 if err != nil {
    log.Fatalf("failed to load configuration, %v", err)
 }

 client := timestreamwrite.NewFromConfig(cfg)

 response, err := client.ResumeBatchLoadTask(context.TODO(),
 &timestreamwrite.ResumeBatchLoadTaskInput{
  TaskId: aws.String("TaskId"),
 })

 if err != nil {
  fmt.Println("Error:")
  fmt.Println(err)
 } else {
  fmt.Println("Resume batch load task is successful")
  fmt.Println(response)
 }
}
```

Python

```
import boto3
from botocore.config import Config

INGEST_ENDPOINT="<url>"
REGION="us-west-2"
HT_TTL_HOURS = 24
CT_TTL_DAYS = 7
TASK_ID = "<TaskId>"

def resume_batch_load_task(client, task_id):
    try:
        result = client.resume_batch_load_task(TaskId=task_id)
        print("Successfully resumed batch load task: ", result)
    except Exception as err:
        print("Resume batch load task failed:", err)


if __name__ == '__main__':
    session = boto3.Session()

    write_client = session.client('timestream-write', \
        endpoint_url=INGEST_ENDPOINT, region_name=REGION, \
        config=Config(read_timeout=20, max_pool_connections = 5000,
 retries={'max_attempts': 10}))

    resume_batch_load_task(write_client, TASK_ID)
```

Node.js

El siguiente fragmento se usa para la versión 3. AWS SDK JavaScript Para obtener más información sobre cómo instalar el cliente y su uso, consulte Timestream Write Client: for v3. AWS SDK JavaScript

Para obtener API más información, consulte Clase y. CreateBatchLoadCommand CreateBatchLoadTask

```
import { TimestreamWriteClient, ResumeBatchLoadTaskCommand } from "@aws-sdk/client-
timestream-write";
const writeClient = new TimestreamWriteClient({ region: "<region>", endpoint:
 "<endpoint>" });
```

```
const params = {
    TaskId: "<TaskId>"
};

const command = new ResumeBatchLoadTaskCommand(params);

try {
    const data = await writeClient.send(command);
    console.log("Resumed batch load task");
} catch (error) {
    console.log("Resume batch load task failed.", error);
 throw error;
}
```

.NET

```
using System;
using System.IO;
using System.Collections.Generic;
using Amazon.TimestreamWrite;
using Amazon.TimestreamWrite.Model;
using System.Threading.Tasks;

namespace TimestreamDotNetSample
{
    public class ResumeBatchLoadTaskExample
    {
        private readonly AmazonTimestreamWriteClient writeClient;

        public ResumeBatchLoadTaskExample(AmazonTimestreamWriteClient writeClient)
        {
            this.writeClient = writeClient;
        }

        public async Task ResumeBatchLoadTask(String taskId)
        {
            try
            {
                var resumeBatchLoadTaskRequest = new ResumeBatchLoadTaskRequest
                {
                    TaskId = taskId
                };
```

```
                ResumeBatchLoadTaskResponse response = await
 writeClient.ResumeBatchLoadTaskAsync(resumeBatchLoadTaskRequest);
                Console.WriteLine("Successfully resumed batch load task.");
            }
            catch (ResourceNotFoundException)
            {
                Console.WriteLine("Batch load task does not exist.");
            }
            catch (Exception e)
            {
                Console.WriteLine("Resume batch load task failed: " + e.ToString());
            }
        }
    }
}
```

Crear consulta programada

Puede usar los siguientes fragmentos de código para crear una consulta programada con un mapeo de múltiples medidas.

Java

```
public static String DATABASE_NAME = "devops_sample_application";
public static String TABLE_NAME = "host_metrics_sample_application";
public static String HOSTNAME = "host-24Gju";
public static String SQ_NAME = "daily-sample";
public static String SCHEDULE_EXPRESSION = "cron(0/2 * * * ? *)";

// Find the average, p90, p95, and p99 CPU utilization for a specific EC2 host over
 the past 2 hours.
public static String QUERY = "SELECT region, az, hostname, BIN(time, 15s) AS
 binned_timestamp, " +
"ROUND(AVG(cpu_utilization), 2) AS avg_cpu_utilization, " +
"ROUND(APPROX_PERCENTILE(cpu_utilization, 0.9), 2) AS p90_cpu_utilization, " +
"ROUND(APPROX_PERCENTILE(cpu_utilization, 0.95), 2) AS p95_cpu_utilization, " +
"ROUND(APPROX_PERCENTILE(cpu_utilization, 0.99), 2) AS p99_cpu_utilization " +
"FROM " +  DATABASE_NAME + "." +  TABLE_NAME + " " +
"WHERE measure_name = 'metrics' " +
"AND hostname = '" + HOSTNAME + "' " +
"AND time > ago(2h) " +
"GROUP BY region, hostname, az, BIN(time, 15s) " +
```

```
"ORDER BY binned_timestamp ASC " +
"LIMIT 5";


public String createScheduledQuery(String topic_arn,
    String role_arn,
    String database_name,
    String table_name) {
    System.out.println("Creating Scheduled Query");

    List<Pair<String, MeasureValueType>> sourceColToMeasureValueTypes =
 Arrays.asList(
        Pair.of("avg_cpu_utilization", DOUBLE),
        Pair.of("p90_cpu_utilization", DOUBLE),
        Pair.of("p95_cpu_utilization", DOUBLE),
        Pair.of("p99_cpu_utilization", DOUBLE));

    CreateScheduledQueryRequest createScheduledQueryRequest = new
 CreateScheduledQueryRequest()
            .withName(SQ_NAME)
            .withQueryString(QUERY)
            .withScheduleConfiguration(new ScheduleConfiguration()
                    .withScheduleExpression(SCHEDULE_EXPRESSION))
            .withNotificationConfiguration(new NotificationConfiguration()
                    .withSnsConfiguration(new SnsConfiguration()
                            .withTopicArn(topic_arn)))
            .withTargetConfiguration(new
 TargetConfiguration().withTimestreamConfiguration(new TimestreamConfiguration()
                    .withDatabaseName(database_name)
                    .withTableName(table_name)
                    .withTimeColumn("binned_timestamp")
                    .withDimensionMappings(Arrays.asList(
                            new DimensionMapping()
                                    .withName("region")
                                    .withDimensionValueType("VARCHAR"),
                            new DimensionMapping()
                                    .withName("az")
                                    .withDimensionValueType("VARCHAR"),
                            new DimensionMapping()
                                    .withName("hostname")
                                    .withDimensionValueType("VARCHAR")
                    ))
                    .withMultiMeasureMappings(new MultiMeasureMappings()
                        .withTargetMultiMeasureName("multi-metrics")
```

```
                        .withMultiMeasureAttributeMappings(
                            sourceColToMeasureValueTypes.stream()
                            .map(pair -> new MultiMeasureAttributeMapping()
                                .withMeasureValueType(pair.getValue().name())
                                .withSourceColumn(pair.getKey()))
                            .collect(Collectors.toList()))))
            .withErrorReportConfiguration(new ErrorReportConfiguration()
                    .withS3Configuration(new S3Configuration()

 .withBucketName(timestreamDependencyHelper.getS3ErrorReportBucketName())))
            .withScheduledQueryExecutionRoleArn(role_arn);

    try {
        final CreateScheduledQueryResult createScheduledQueryResult =
 queryClient.createScheduledQuery(createScheduledQueryRequest);
        final String scheduledQueryArn = createScheduledQueryResult.getArn();
        System.out.println("Successfully created scheduled query : " +
 scheduledQueryArn);
        return scheduledQueryArn;
    }
    catch (Exception e) {
        System.out.println("Scheduled Query creation failed: " + e);
        throw e;
    }
}
```

Java v2

```
public static String DATABASE_NAME = "testJavaV2DB";
public static String TABLE_NAME = "testJavaV2Table";
public static String HOSTNAME = "host-24Gju";
public static String SQ_NAME = "daily-sample";
public static String SCHEDULE_EXPRESSION = "cron(0/2 * * * ? *)";

// Find the average, p90, p95, and p99 CPU utilization for a specific EC2 host over
 the past 2 hours.
public static String VALID_QUERY = "SELECT region, az, hostname, BIN(time, 15s) AS
 binned_timestamp, " +
"ROUND(AVG(cpu_utilization), 2) AS avg_cpu_utilization, " +
"ROUND(APPROX_PERCENTILE(cpu_utilization, 0.9), 2) AS p90_cpu_utilization, " +
"ROUND(APPROX_PERCENTILE(cpu_utilization, 0.95), 2) AS p95_cpu_utilization, " +
"ROUND(APPROX_PERCENTILE(cpu_utilization, 0.99), 2) AS p99_cpu_utilization " +
"FROM " +  DATABASE_NAME + "." +  TABLE_NAME + " " +
```

```
"WHERE measure_name = 'metrics' " +
"AND hostname = '" + HOSTNAME + "' " +
"AND time > ago(2h) " +
"GROUP BY region, hostname, az, BIN(time, 15s) " +
"ORDER BY binned_timestamp ASC " +
"LIMIT 5";


private String createScheduledQueryHelper(String topicArn, String roleArn,
        String s3ErrorReportBucketName, String query,
        TargetConfiguration targetConfiguration) {
    System.out.println("Creating Scheduled Query");

    CreateScheduledQueryRequest createScheduledQueryRequest =
 CreateScheduledQueryRequest.builder()
            .name(SQ_NAME)
            .queryString(query)
            .scheduleConfiguration(ScheduleConfiguration.builder()
                    .scheduleExpression(SCHEDULE_EXPRESSION)
                    .build())
            .notificationConfiguration(NotificationConfiguration.builder()
                    .snsConfiguration(SnsConfiguration.builder()
                            .topicArn(topicArn)
                            .build())
                    .build())
            .targetConfiguration(targetConfiguration)
            .errorReportConfiguration(ErrorReportConfiguration.builder()
                    .s3Configuration(S3Configuration.builder()
                            .bucketName(s3ErrorReportBucketName)
                            .objectKeyPrefix(SCHEDULED_QUERY_EXAMPLE)
                            .build())
                    .build())
            .scheduledQueryExecutionRoleArn(roleArn)
            .build();

    try {
        final CreateScheduledQueryResponse response =
 queryClient.createScheduledQuery(createScheduledQueryRequest);
        final String scheduledQueryArn = response.arn();
        System.out.println("Successfully created scheduled query : " +
 scheduledQueryArn);
        return scheduledQueryArn;
    }
    catch (Exception e) {
```

```
        System.out.println("Scheduled Query creation failed: " + e);
        throw e;
    }
}

public String createScheduledQuery(String topicArn, String roleArn,
        String databaseName, String tableName, String s3ErrorReportBucketName) {
    List<Pair<String, MeasureValueType>> sourceColToMeasureValueTypes =
 Arrays.asList(
            Pair.of("avg_cpu_utilization", DOUBLE),
            Pair.of("p90_cpu_utilization", DOUBLE),
            Pair.of("p95_cpu_utilization", DOUBLE),
            Pair.of("p99_cpu_utilization", DOUBLE));

    TargetConfiguration targetConfiguration = TargetConfiguration.builder()
            .timestreamConfiguration(TimestreamConfiguration.builder()
            .databaseName(databaseName)
            .tableName(tableName)
            .timeColumn("binned_timestamp")
            .dimensionMappings(Arrays.asList(
                    DimensionMapping.builder()
                            .name("region")
                            .dimensionValueType("VARCHAR")
                            .build(),
                    DimensionMapping.builder()
                            .name("az")
                            .dimensionValueType("VARCHAR")
                            .build(),
                    DimensionMapping.builder()
                            .name("hostname")
                            .dimensionValueType("VARCHAR")
                            .build()
            ))
            .multiMeasureMappings(MultiMeasureMappings.builder()
                    .targetMultiMeasureName("multi-metrics")
                    .multiMeasureAttributeMappings(
                            sourceColToMeasureValueTypes.stream()
                                    .map(pair ->
 MultiMeasureAttributeMapping.builder()

 .measureValueType(pair.getValue().name())
                                            .sourceColumn(pair.getKey())
                                            .build())
                                    .collect(Collectors.toList()))
```

```
                    .build())
            .build())
            .build();

    return createScheduledQueryHelper(topicArn, roleArn, s3ErrorReportBucketName,
 VALID_QUERY, targetConfiguration);
}}
```

Go

```
SQ_ERROR_CONFIGURATION_S3_BUCKET_NAME_PREFIX = "sq-error-configuration-sample-s3-
bucket-"
HOSTNAME            = "host-24Gju"
SQ_NAME             = "daily-sample"
SCHEDULE_EXPRESSION = "cron(0/1 * * * ? *)"
QUERY               = "SELECT region, az, hostname, BIN(time, 15s) AS
 binned_timestamp, " +
    "ROUND(AVG(cpu_utilization), 2) AS avg_cpu_utilization, " +
    "ROUND(APPROX_PERCENTILE(cpu_utilization, 0.9), 2) AS p90_cpu_utilization, " +
    "ROUND(APPROX_PERCENTILE(cpu_utilization, 0.95), 2) AS p95_cpu_utilization, " +
    "ROUND(APPROX_PERCENTILE(cpu_utilization, 0.99), 2) AS p99_cpu_utilization " +
    "FROM %s.%s " +
    "WHERE measure_name = 'metrics' " +
    "AND hostname = '" + HOSTNAME + "' " +
    "AND time > ago(2h) " +
    "GROUP BY region, hostname, az, BIN(time, 15s) " +
    "ORDER BY binned_timestamp ASC " +
    "LIMIT 5"
s3BucketName = utils.SQ_ERROR_CONFIGURATION_S3_BUCKET_NAME_PREFIX +
 generateRandomStringWithSize(5)

func generateRandomStringWithSize(size int) string {
     rand.Seed(time.Now().UnixNano())
     alphaNumericList := []rune("abcdefghijklmnopqrstuvwxyz0123456789")
     randomPrefix := make([]rune, size)
     for i := range randomPrefix {
         randomPrefix[i] = alphaNumericList[rand.Intn(len(alphaNumericList))]
     }
     return string(randomPrefix)
 }

func (timestreamBuilder TimestreamBuilder) createScheduledQuery(topicArn string,
 roleArn string, s3ErrorReportBucketName string,
```

```
query string, targetConfiguration timestreamquery.TargetConfiguration) (string,
 error) {

createScheduledQueryInput := &timestreamquery.CreateScheduledQueryInput{
    Name:        aws.String(SQ_NAME),
    QueryString: aws.String(query),
    ScheduleConfiguration: &timestreamquery.ScheduleConfiguration{
        ScheduleExpression: aws.String(SCHEDULE_EXPRESSION),
    },
    NotificationConfiguration: &timestreamquery.NotificationConfiguration{
        SnsConfiguration: &timestreamquery.SnsConfiguration{
            TopicArn: aws.String(topicArn),
        },
    },
    TargetConfiguration: &targetConfiguration,
    ErrorReportConfiguration: &timestreamquery.ErrorReportConfiguration{
        S3Configuration: &timestreamquery.S3Configuration{
            BucketName: aws.String(s3ErrorReportBucketName),
        },
    },
    ScheduledQueryExecutionRoleArn: aws.String(roleArn),
}

createScheduledQueryOutput, err :=
 timestreamBuilder.QuerySvc.CreateScheduledQuery(createScheduledQueryInput)

if err != nil {
    fmt.Printf("Error: %s", err.Error())
} else {
    fmt.Println("createScheduledQueryResult is successful")
    return *createScheduledQueryOutput.Arn, nil
}
return "", err
}

 func (timestreamBuilder TimestreamBuilder) CreateValidScheduledQuery(topicArn
 string, roleArn string, s3ErrorReportBucketName string,
     sqDatabaseName string, sqTableName string, databaseName string, tableName
 string) (string, error) {

     targetConfiguration := timestreamquery.TargetConfiguration{
         TimestreamConfiguration: &timestreamquery.TimestreamConfiguration{
             DatabaseName: aws.String(sqDatabaseName),
             TableName:    aws.String(sqTableName),
```

```
            TimeColumn:   aws.String("binned_timestamp"),
            DimensionMappings: []*timestreamquery.DimensionMapping{
                {
                    Name:               aws.String("region"),
                    DimensionValueType: aws.String("VARCHAR"),
                },
                {
                    Name:               aws.String("az"),
                    DimensionValueType: aws.String("VARCHAR"),
                },
                {
                    Name:               aws.String("hostname"),
                    DimensionValueType: aws.String("VARCHAR"),
                },
            },
            MultiMeasureMappings: &timestreamquery.MultiMeasureMappings{
                TargetMultiMeasureName: aws.String("multi-metrics"),
                MultiMeasureAttributeMappings:
[]*timestreamquery.MultiMeasureAttributeMapping{
                    {
                        SourceColumn:     aws.String("avg_cpu_utilization"),
                        MeasureValueType:
aws.String(timestreamquery.MeasureValueTypeDouble),
                    },
                    {
                        SourceColumn:     aws.String("p90_cpu_utilization"),
                        MeasureValueType:
aws.String(timestreamquery.MeasureValueTypeDouble),
                    },
                    {
                        SourceColumn:     aws.String("p95_cpu_utilization"),
                        MeasureValueType:
aws.String(timestreamquery.MeasureValueTypeDouble),
                    },
                    {
                        SourceColumn:     aws.String("p99_cpu_utilization"),
                        MeasureValueType:
aws.String(timestreamquery.MeasureValueTypeDouble),
                    },
                },
            },
        },
    }
```

```
    return timestreamBuilder.createScheduledQuery(topicArn, roleArn,
 s3ErrorReportBucketName,
        fmt.Sprintf(QUERY, databaseName, tableName), targetConfiguration)
 }
```

Python

```
HOSTNAME = "host-24Gju"
SQ_NAME = "daily-sample"
ERROR_BUCKET_NAME = "scheduledquerysamplerrorbucket" +
 ''.join([choice(ascii_lowercase) for _ in range(5)])
QUERY = \
    "SELECT region, az, hostname, BIN(time, 15s) AS binned_timestamp, " \
    "    ROUND(AVG(cpu_utilization), 2) AS avg_cpu_utilization, " \
    "    ROUND(APPROX_PERCENTILE(cpu_utilization, 0.9), 2) AS p90_cpu_utilization, "
 \
    "    ROUND(APPROX_PERCENTILE(cpu_utilization, 0.95), 2) AS p95_cpu_utilization,
 " \
    "    ROUND(APPROX_PERCENTILE(cpu_utilization, 0.99), 2) AS p99_cpu_utilization "
 \
    "FROM " + database_name + "." + table_name + " " \
    "WHERE measure_name = 'metrics' " \
    "AND hostname = '" + self.HOSTNAME + "' " \
    "AND time > ago(2h) " \
    "GROUP BY region, hostname, az, BIN(time, 15s) " \
    "ORDER BY binned_timestamp ASC " \
    "LIMIT 5"

def create_scheduled_query_helper(self, topic_arn, role_arn, query,
 target_configuration):
    print("\nCreating Scheduled Query")
    schedule_configuration = {
        'ScheduleExpression': 'cron(0/2 * * * ? *)'
    }
    notification_configuration = {
        'SnsConfiguration': {
            'TopicArn': topic_arn
        }
    }
    error_report_configuration = {
        'S3Configuration': {
            'BucketName': ERROR_BUCKET_NAME
        }
```

```
    }

    try:
        create_scheduled_query_response = \
            query_client.create_scheduled_query(Name=self.SQ_NAME,
                 QueryString=query,
                 ScheduleConfiguration=schedule_configuration,
                 NotificationConfiguration=notification_configuration,
                 TargetConfiguration=target_configuration,
                 ScheduledQueryExecutionRoleArn=role_arn,
                 ErrorReportConfiguration=error_report_configuration
                 )
        print("Successfully created scheduled query : ",
 create_scheduled_query_response['Arn'])
        return create_scheduled_query_response['Arn']
    except Exception as err:
        print("Scheduled Query creation failed:", err)
        raise err

def create_valid_scheduled_query(self, topic_arn, role_arn):
    target_configuration = {
        'TimestreamConfiguration': {
            'DatabaseName': self.sq_database_name,
            'TableName': self.sq_table_name,
            'TimeColumn': 'binned_timestamp',
            'DimensionMappings': [
                {'Name': 'region', 'DimensionValueType': 'VARCHAR'},
                {'Name': 'az', 'DimensionValueType': 'VARCHAR'},
                {'Name': 'hostname', 'DimensionValueType': 'VARCHAR'}
            ],
            'MultiMeasureMappings': {
                'TargetMultiMeasureName': 'target_name',
                'MultiMeasureAttributeMappings': [
                    {'SourceColumn': 'avg_cpu_utilization', 'MeasureValueType':
 'DOUBLE',
                     'TargetMultiMeasureAttributeName': 'avg_cpu_utilization'},
                    {'SourceColumn': 'p90_cpu_utilization', 'MeasureValueType':
 'DOUBLE',
                     'TargetMultiMeasureAttributeName': 'p90_cpu_utilization'},
                    {'SourceColumn': 'p95_cpu_utilization', 'MeasureValueType':
 'DOUBLE',
                     'TargetMultiMeasureAttributeName': 'p95_cpu_utilization'},
                    {'SourceColumn': 'p99_cpu_utilization', 'MeasureValueType':
 'DOUBLE',
```

```
                    'TargetMultiMeasureAttributeName': 'p99_cpu_utilization'},
                ]
            }
        }
    }

    return self.create_scheduled_query_helper(topic_arn, role_arn, QUERY,
 target_configuration)
```

Node.js

En el siguiente fragmento se utiliza el estilo for V2. AWS SDK JavaScript Se basa en la aplicación de ejemplo de Node.js, ejemplo de Amazon Timestream LiveAnalytics para su aplicación en. GitHub

```
const DATABASE_NAME = 'devops_sample_application';
const TABLE_NAME = 'host_metrics_sample_application';
const SQ_DATABASE_NAME = 'sq_result_database';
const SQ_TABLE_NAME = 'sq_result_table';
const HOSTNAME = "host-24Gju";
const SQ_NAME = "daily-sample";
const SCHEDULE_EXPRESSION = "cron(0/1 * * * ? *)";

// Find the average, p90, p95, and p99 CPU utilization for a specific EC2 host over
 the past 2 hours.
const VALID_QUERY = "SELECT region, az, hostname, BIN(time, 15s) AS
 binned_timestamp, " +
    " ROUND(AVG(cpu_utilization), 2) AS avg_cpu_utilization, " +
    " ROUND(APPROX_PERCENTILE(cpu_utilization, 0.9), 2) AS p90_cpu_utilization, " +
    " ROUND(APPROX_PERCENTILE(cpu_utilization, 0.95), 2) AS p95_cpu_utilization, " +
    " ROUND(APPROX_PERCENTILE(cpu_utilization, 0.99), 2) AS p99_cpu_utilization " +
    "FROM " + DATABASE_NAME + "." + TABLE_NAME + " " +
    "WHERE measure_name = 'metrics' " +
    " AND hostname = '" + HOSTNAME + "' " +
    " AND time > ago(2h) " +
    "GROUP BY region, hostname, az, BIN(time, 15s) " +
    "ORDER BY binned_timestamp ASC " +
    "LIMIT 5";

async function createScheduledQuery(topicArn, roleArn, s3ErrorReportBucketName) {
    console.log("Creating Valid Scheduled Query");
    const DimensionMappingList = [{
            'Name': 'region',
```

```
            'DimensionValueType': 'VARCHAR'
        },
        {
            'Name': 'az',
            'DimensionValueType': 'VARCHAR'
        },
        {
            'Name': 'hostname',
            'DimensionValueType': 'VARCHAR'
        }
    ];

    const MultiMeasureMappings = {
        TargetMultiMeasureName: "multi-metrics",
        MultiMeasureAttributeMappings: [{
                'SourceColumn': 'avg_cpu_utilization',
                'MeasureValueType': 'DOUBLE'
            },
            {
                'SourceColumn': 'p90_cpu_utilization',
                'MeasureValueType': 'DOUBLE'
            },
            {
                'SourceColumn': 'p95_cpu_utilization',
                'MeasureValueType': 'DOUBLE'
            },
            {
                'SourceColumn': 'p99_cpu_utilization',
                'MeasureValueType': 'DOUBLE'
            },
        ]
    }

    const timestreamConfiguration = {
        DatabaseName: SQ_DATABASE_NAME,
        TableName: SQ_TABLE_NAME,
        TimeColumn: "binned_timestamp",
        DimensionMappings: DimensionMappingList,
        MultiMeasureMappings: MultiMeasureMappings
    }

    const createScheduledQueryRequest = {
        Name: SQ_NAME,
        QueryString: VALID_QUERY,
```

```
        ScheduleConfiguration: {
            ScheduleExpression: SCHEDULE_EXPRESSION
        },
        NotificationConfiguration: {
            SnsConfiguration: {
                TopicArn: topicArn
            }
        },
        TargetConfiguration: {
            TimestreamConfiguration: timestreamConfiguration
        },
        ScheduledQueryExecutionRoleArn: roleArn,
        ErrorReportConfiguration: {
            S3Configuration: {
                BucketName: s3ErrorReportBucketName
            }
        }
    };
    try {
        const data = await
 queryClient.createScheduledQuery(createScheduledQueryRequest).promise();
        console.log("Successfully created scheduled query: " + data.Arn);
        return data.Arn;
    } catch (err) {
        console.log("Scheduled Query creation failed: ", err);
        throw err;
    }
}
```

.NET

```
public const string Hostname = "host-24Gju";
public const string SqName = "timestream-sample";
public const string SqDatabaseName = "sq_result_database";
public const string SqTableName = "sq_result_table";

public const string ErrorConfigurationS3BucketNamePrefix = "error-configuration-
sample-s3-bucket-";
public const string ScheduleExpression = "cron(0/2 * * * ? *)";

// Find the average, p90, p95, and p99 CPU utilization for a specific EC2 host over
 the past 2 hours.
```

```
public const string ValidQuery = "SELECT region, az, hostname, BIN(time, 15s) AS
 binned_timestamp, " +
      "ROUND(AVG(cpu_utilization), 2) AS avg_cpu_utilization, " +
      "ROUND(APPROX_PERCENTILE(cpu_utilization, 0.9), 2) AS p90_cpu_utilization, " +
      "ROUND(APPROX_PERCENTILE(cpu_utilization, 0.95), 2) AS p95_cpu_utilization, "
 +
      "ROUND(APPROX_PERCENTILE(cpu_utilization, 0.99), 2) AS p99_cpu_utilization " +
      "FROM " + Constants.DATABASE_NAME + "." + Constants.TABLE_NAME + " " +
      "WHERE measure_name = 'metrics' " +
      "AND hostname = '" + Hostname + "' " +
      "AND time > ago(2h) " +
      "GROUP BY region, hostname, az, BIN(time, 15s) " +
      "ORDER BY binned_timestamp ASC " +
      "LIMIT 5";

private async Task<String> CreateValidScheduledQuery(string topicArn, string
 roleArn,
             string databaseName, string tableName, string s3ErrorReportBucketName)
 {
     List<MultiMeasureAttributeMapping> sourceColToMeasureValueTypes =
         new List<MultiMeasureAttributeMapping>()
         {
             new()
             {
                 SourceColumn = "avg_cpu_utilization",
                 MeasureValueType = MeasureValueType.DOUBLE.Value
             },
             new()
             {
                 SourceColumn = "p90_cpu_utilization",
                 MeasureValueType = MeasureValueType.DOUBLE.Value
             },
             new()
             {
                 SourceColumn = "p95_cpu_utilization",
                 MeasureValueType = MeasureValueType.DOUBLE.Value
             },
             new()
             {
                 SourceColumn = "p99_cpu_utilization",
                 MeasureValueType = MeasureValueType.DOUBLE.Value
             }
         };
```

```
    TargetConfiguration targetConfiguration = new TargetConfiguration()
    {
        TimestreamConfiguration = new TimestreamConfiguration()
        {
            DatabaseName = databaseName,
            TableName = tableName,
            TimeColumn = "binned_timestamp",
            DimensionMappings = new List<DimensionMapping>()
            {
                new()
                {
                    Name = "region",
                    DimensionValueType = "VARCHAR"
                },
                new()
                {
                    Name = "az",
                    DimensionValueType = "VARCHAR"
                },
                new()
                {
                    Name = "hostname",
                    DimensionValueType = "VARCHAR"
                }
            },
            MultiMeasureMappings = new MultiMeasureMappings()
            {
                TargetMultiMeasureName = "multi-metrics",
                MultiMeasureAttributeMappings = sourceColToMeasureValueTypes
            }
        }
    };
    return await CreateScheduledQuery(topicArn, roleArn, s3ErrorReportBucketName,
        ScheduledQueryConstants.ValidQuery, targetConfiguration);
}

private async Task<String> CreateScheduledQuery(string topicArn, string roleArn,
            string s3ErrorReportBucketName, string query, TargetConfiguration
 targetConfiguration)
{
    try
    {
        Console.WriteLine("Creating Scheduled Query");
```

```
        CreateScheduledQueryResponse response = await
_amazonTimestreamQuery.CreateScheduledQueryAsync(
            new CreateScheduledQueryRequest()
            {
                Name = ScheduledQueryConstants.SqName,
                QueryString = query,
                ScheduleConfiguration = new ScheduleConfiguration()
                {
                    ScheduleExpression = ScheduledQueryConstants.ScheduleExpression
                },
                NotificationConfiguration = new NotificationConfiguration()
                {
                    SnsConfiguration = new SnsConfiguration()
                    {
                        TopicArn = topicArn
                    }
                },
                TargetConfiguration = targetConfiguration,
                ErrorReportConfiguration = new ErrorReportConfiguration()
                {
                    S3Configuration = new S3Configuration()
                    {
                        BucketName = s3ErrorReportBucketName
                    }
                },
                ScheduledQueryExecutionRoleArn = roleArn
            });
        Console.WriteLine($"Successfully created scheduled query :
{response.Arn}");
        return response.Arn;
    }
    catch (Exception e)
    {
        Console.WriteLine($"Scheduled Query creation failed: {e}");
        throw;
    }
}
```

Listar consulta programada

Puede usar los siguientes fragmentos de código para enumerar las consultas programadas.

Java

```
public void listScheduledQueries() {
    System.out.println("Listing Scheduled Query");
    try {
        String nextToken = null;
        List<String> scheduledQueries = new ArrayList<>();

        do {
            ListScheduledQueriesResult listScheduledQueriesResult =
                    queryClient.listScheduledQueries(new
 ListScheduledQueriesRequest()
                            .withNextToken(nextToken).withMaxResults(10));
            List<ScheduledQuery> scheduledQueryList =
 listScheduledQueriesResult.getScheduledQueries();

            printScheduledQuery(scheduledQueryList);
            nextToken = listScheduledQueriesResult.getNextToken();
        } while (nextToken != null);
    }
    catch (Exception e) {
        System.out.println("List Scheduled Query failed: " + e);
        throw e;
    }
}

public void printScheduledQuery(List<ScheduledQuery> scheduledQueryList) {
    for (ScheduledQuery scheduledQuery: scheduledQueryList) {
        System.out.println(scheduledQuery.getArn());
    }
}
```

Java v2

```
public void listScheduledQueries() {
    System.out.println("Listing Scheduled Query");
    try {
        String nextToken = null;

        do {
            ListScheduledQueriesResponse listScheduledQueriesResult =

 queryClient.listScheduledQueries(ListScheduledQueriesRequest.builder()
```

```
                            .nextToken(nextToken).maxResults(10)
                            .build());
            List<ScheduledQuery> scheduledQueryList =
 listScheduledQueriesResult.scheduledQueries();

            printScheduledQuery(scheduledQueryList);
            nextToken = listScheduledQueriesResult.nextToken();
        } while (nextToken != null);
    }
    catch (Exception e) {
        System.out.println("List Scheduled Query failed: " + e);
        throw e;
    }
}

public void printScheduledQuery(List<ScheduledQuery> scheduledQueryList) {
    for (ScheduledQuery scheduledQuery: scheduledQueryList) {
        System.out.println(scheduledQuery.arn());
    }
}
```

Go

```
func (timestreamBuilder TimestreamBuilder) ListScheduledQueries()
 ([]*timestreamquery.ScheduledQuery, error) {

     var nextToken *string = nil
     var scheduledQueries []*timestreamquery.ScheduledQuery
     for ok := true; ok; ok = nextToken != nil {
         listScheduledQueriesInput := &timestreamquery.ListScheduledQueriesInput{
             MaxResults: aws.Int64(15),
         }
         if nextToken != nil {
             listScheduledQueriesInput.NextToken = aws.String(*nextToken)
         }

         listScheduledQueriesOutput, err :=
 timestreamBuilder.QuerySvc.ListScheduledQueries(listScheduledQueriesInput)
         if err != nil {
             fmt.Printf("Error: %s", err.Error())
             return nil, err
         }
```

```
        scheduledQueries = append(scheduledQueries,
 listScheduledQueriesOutput.ScheduledQueries...)
        nextToken = listScheduledQueriesOutput.NextToken
    }
    return scheduledQueries, nil
}
```

Python

```
def list_scheduled_queries(self):
    print("\nListing Scheduled Queries")
    try:
        response = self.query_client.list_scheduled_queries(MaxResults=10)
        self.print_scheduled_queries(response['ScheduledQueries'])
        next_token = response.get('NextToken', None)
        while next_token:
            response =
 self.query_client.list_scheduled_queries(NextToken=next_token, MaxResults=10)
            self.print_scheduled_queries(response['ScheduledQueries'])
            next_token = response.get('NextToken', None)
    except Exception as err:
        print("List scheduled queries failed:", err)
        raise err

@staticmethod
def print_scheduled_queries(scheduled_queries):
    for scheduled_query in scheduled_queries:
        print(scheduled_query['Arn'])
```

Node.js

En el siguiente fragmento se utiliza el estilo for V2. AWS SDK JavaScript Se basa en la aplicación de ejemplo de Node.js, ejemplo de Amazon Timestream LiveAnalytics para su aplicación en. GitHub

```
async function listScheduledQueries() {
    console.log("Listing Scheduled Query");
    try {
        var nextToken = null;
        do {
            var params = {
                MaxResults: 10,
                NextToken: nextToken
```

```
            }
            var data = await queryClient.listScheduledQueries(params).promise();
            var scheduledQueryList = data.ScheduledQueries;
            printScheduledQuery(scheduledQueryList);
            nextToken = data.NextToken;
        }
        while (nextToken != null);
    }  catch (err) {
        console.log("List Scheduled Query failed: ", err);
        throw err;
    }
}

async function printScheduledQuery(scheduledQueryList) {
    scheduledQueryList.forEach(element => console.log(element.Arn));
}
```

.NET

```
private async Task ListScheduledQueries()
{
    try
    {
        Console.WriteLine("Listing Scheduled Query");
        string nextToken;
        do
        {
            ListScheduledQueriesResponse response =
                await _amazonTimestreamQuery.ListScheduledQueriesAsync(new
ListScheduledQueriesRequest());
            foreach (var scheduledQuery in response.ScheduledQueries)
            {
                Console.WriteLine($"{scheduledQuery.Arn}");
            }

            nextToken = response.NextToken;
        } while (nextToken != null);
    }
    catch (Exception e)
    {
        Console.WriteLine($"List Scheduled Query failed: {e}");
        throw;
    }
```

```
 }
```

Describa la consulta programada

Puede usar los siguientes fragmentos de código para describir una consulta programada.

Java

```
public void describeScheduledQueries(String scheduledQueryArn) {
    System.out.println("Describing Scheduled Query");
    try {
        DescribeScheduledQueryResult describeScheduledQueryResult =
 queryClient.describeScheduledQuery(new
 DescribeScheduledQueryRequest().withScheduledQueryArn(scheduledQueryArn));
        System.out.println(describeScheduledQueryResult);
    }
    catch (ResourceNotFoundException e) {
        System.out.println("Scheduled Query doesn't exist");
        throw e;
    }
    catch (Exception e) {
        System.out.println("Describe Scheduled Query failed: " + e);
        throw e;
    }
}
```

Java v2

```
public void describeScheduledQueries(String scheduledQueryArn) {
    System.out.println("Describing Scheduled Query");
    try {
        DescribeScheduledQueryResponse describeScheduledQueryResult =

 queryClient.describeScheduledQuery(DescribeScheduledQueryRequest.builder()
                        .scheduledQueryArn(scheduledQueryArn)
                        .build());
        System.out.println(describeScheduledQueryResult);
    }
    catch (ResourceNotFoundException e) {
        System.out.println("Scheduled Query doesn't exist");
        throw e;
    }
```

```
    catch (Exception e) {
        System.out.println("Describe Scheduled Query failed: " + e);
        throw e;
    }
}
```

Go

```
func (timestreamBuilder TimestreamBuilder) DescribeScheduledQuery(scheduledQueryArn
 string) error {

    describeScheduledQueryInput := &timestreamquery.DescribeScheduledQueryInput{
        ScheduledQueryArn: aws.String(scheduledQueryArn),
    }
    describeScheduledQueryOutput, err :=
 timestreamBuilder.QuerySvc.DescribeScheduledQuery(describeScheduledQueryInput)

    if err != nil {
        if aerr, ok := err.(awserr.Error); ok {
            switch aerr.Code() {
            case timestreamquery.ErrCodeResourceNotFoundException:
                fmt.Println(timestreamquery.ErrCodeResourceNotFoundException,
 aerr.Error())
            default:
                fmt.Printf("Error: %s", err.Error())
            }
        } else {
            fmt.Printf("Error: %s", aerr.Error())
        }
        return err
    } else {
        fmt.Println("DescribeScheduledQuery is successful, below is the output:")
        fmt.Println(describeScheduledQueryOutput.ScheduledQuery)
        return nil
    }
}
```

Python

```
def describe_scheduled_query(self, scheduled_query_arn):
    print("\nDescribing Scheduled Query")
    try:
```

```
        response =
 self.query_client.describe_scheduled_query(ScheduledQueryArn=scheduled_query_arn)
        if 'ScheduledQuery' in response:
            response = response['ScheduledQuery']
            for key in response:
                print("{} :{}".format(key, response[key]))
    except self.query_client.exceptions.ResourceNotFoundException as err:
        print("Scheduled Query doesn't exist")
        raise err
    except Exception as err:
        print("Scheduled Query describe failed:", err)
        raise err
```

Node.js

En el siguiente fragmento se utiliza el estilo for V2. AWS SDK JavaScript Se basa en la aplicación de ejemplo de Node.js, ejemplo de Amazon Timestream LiveAnalytics para su aplicación en. GitHub

```
async function describeScheduledQuery(scheduledQueryArn) {
     console.log("Describing Scheduled Query");
     var params = {
         ScheduledQueryArn: scheduledQueryArn
     }
     try {
         const data = await queryClient.describeScheduledQuery(params).promise();
         console.log(data.ScheduledQuery);
     } catch (err) {
         console.log("Describe Scheduled Query failed: ", err);
         throw err;
     }
 }
```

.NET

```
private async Task DescribeScheduledQuery(string scheduledQueryArn)
 {
     try
     {
         Console.WriteLine("Describing Scheduled Query");
         DescribeScheduledQueryResponse response = await
 _amazonTimestreamQuery.DescribeScheduledQueryAsync(
             new DescribeScheduledQueryRequest()
```

```
            {
                ScheduledQueryArn = scheduledQueryArn
            });

 Console.WriteLine($"{JsonConvert.SerializeObject(response.ScheduledQuery)}");
    }
    catch (ResourceNotFoundException e)
    {
        Console.WriteLine($"Scheduled Query doesn't exist: {e}");
        throw;
    }
    catch (Exception e)
    {
        Console.WriteLine($"Describe Scheduled Query failed: {e}");
        throw;
    }
}
```

Ejecutar consulta programada

Puede usar los siguientes fragmentos de código para ejecutar una consulta programada.

Java

```
public void executeScheduledQueries(String scheduledQueryArn, Date invocationTime) {
    System.out.println("Executing Scheduled Query");
    try {
        ExecuteScheduledQueryResult executeScheduledQueryResult =
 queryClient.executeScheduledQuery(new ExecuteScheduledQueryRequest()
                .withScheduledQueryArn(scheduledQueryArn)
                .withInvocationTime(invocationTime)
        );

    }
    catch (ResourceNotFoundException e) {
        System.out.println("Scheduled Query doesn't exist");
        throw e;
    }
    catch (Exception e) {
        System.out.println("Execution Scheduled Query failed: " + e);
        throw e;
    }
```

```
}
```

Java v2

```
public void executeScheduledQuery(String scheduledQueryArn) {
    System.out.println("Executing Scheduled Query");
    try {
        ExecuteScheduledQueryResponse executeScheduledQueryResult =
 queryClient.executeScheduledQuery(ExecuteScheduledQueryRequest.builder()
                .scheduledQueryArn(scheduledQueryArn)
                .invocationTime(Instant.now())
                .build()
        );

        System.out.println("Execute ScheduledQuery response code: " +
 executeScheduledQueryResult.sdkHttpResponse().statusCode());

    }
    catch (ResourceNotFoundException e) {
        System.out.println("Scheduled Query doesn't exist");
        throw e;
    }
    catch (Exception e) {
        System.out.println("Execution Scheduled Query failed: " + e);
        throw e;
    }
}
```

Go

```
func (timestreamBuilder TimestreamBuilder) ExecuteScheduledQuery(scheduledQueryArn
 string, invocationTime time.Time) error {

    executeScheduledQueryInput := &timestreamquery.ExecuteScheduledQueryInput{
        ScheduledQueryArn: aws.String(scheduledQueryArn),
        InvocationTime:    aws.Time(invocationTime),
    }
    executeScheduledQueryOutput, err :=
 timestreamBuilder.QuerySvc.ExecuteScheduledQuery(executeScheduledQueryInput)

    if err != nil {
        if aerr, ok := err.(awserr.Error); ok {
```

```
            switch aerr.Code() {
            case timestreamquery.ErrCodeResourceNotFoundException:
                fmt.Println(timestreamquery.ErrCodeResourceNotFoundException,
 aerr.Error())
            default:
                fmt.Printf("Error: %s", aerr.Error())
            }
        } else {
            fmt.Printf("Error: %s", err.Error())
        }
        return err
    } else {
        fmt.Println("ExecuteScheduledQuery is successful, below is the output:")
        fmt.Println(executeScheduledQueryOutput.GoString())
        return nil
    }
}
```

Python

```
def execute_scheduled_query(self, scheduled_query_arn, invocation_time):
    print("\nExecuting Scheduled Query")
    try:

 self.query_client.execute_scheduled_query(ScheduledQueryArn=scheduled_query_arn,
 InvocationTime=invocation_time)
        print("Successfully started executing scheduled query")
    except self.query_client.exceptions.ResourceNotFoundException as err:
        print("Scheduled Query doesn't exist")
        raise err
    except Exception as err:
        print("Scheduled Query execution failed:", err)
        raise err
```

Node.js

En el siguiente fragmento se utiliza el estilo for V2. AWS SDK JavaScript Se basa en la aplicación de ejemplo de Node.js, ejemplo de Amazon Timestream LiveAnalytics para su aplicación en. GitHub

```
async function executeScheduledQuery(scheduledQueryArn, invocationTime) {
     console.log("Executing Scheduled Query");
     var params = {
```

```
        ScheduledQueryArn: scheduledQueryArn,
        InvocationTime: invocationTime
    }
    try {
        await queryClient.executeScheduledQuery(params).promise();
    } catch (err) {
        console.log("Execute Scheduled Query failed: ", err);
        throw err;
    }
}
```

.NET

```
private async Task ExecuteScheduledQuery(string scheduledQueryArn, DateTime
 invocationTime)
 {
     try
     {
         Console.WriteLine("Running Scheduled Query");
         await _amazonTimestreamQuery.ExecuteScheduledQueryAsync(new
 ExecuteScheduledQueryRequest()
         {
             ScheduledQueryArn = scheduledQueryArn,
             InvocationTime = invocationTime
         });
         Console.WriteLine("Successfully started manual run of scheduled query");
     }
     catch (ResourceNotFoundException e)
     {
         Console.WriteLine($"Scheduled Query doesn't exist: {e}");
         throw;
     }
     catch (Exception e)
     {
         Console.WriteLine($"Execute Scheduled Query failed: {e}");
         throw;
     }
 }
```

Actualizar consulta programada

Puede usar los siguientes fragmentos de código para actualizar una consulta programada.

Java

```
public void updateScheduledQueries(String scheduledQueryArn) {
    System.out.println("Updating Scheduled Query");
    try {
        queryClient.updateScheduledQuery(new UpdateScheduledQueryRequest()
                .withScheduledQueryArn(scheduledQueryArn)
                .withState(ScheduledQueryState.DISABLED));
        System.out.println("Successfully update scheduled query state");
    }
    catch (ResourceNotFoundException e) {
        System.out.println("Scheduled Query doesn't exist");
        throw e;
    }
    catch (Exception e) {
        System.out.println("Execution Scheduled Query failed: " + e);
        throw e;
    }
}
```

Java v2

```
public void updateScheduledQuery(String scheduledQueryArn, ScheduledQueryState
 state) {
    System.out.println("Updating Scheduled Query");
    try {
        queryClient.updateScheduledQuery(UpdateScheduledQueryRequest.builder()
                .scheduledQueryArn(scheduledQueryArn)
                .state(state)
                .build());
        System.out.println("Successfully update scheduled query state");
    }
    catch (ResourceNotFoundException e) {
        System.out.println("Scheduled Query doesn't exist");
        throw e;
    }
    catch (Exception e) {
        System.out.println("Execution Scheduled Query failed: " + e);
        throw e;
    }
}
```

Go

```
func (timestreamBuilder TimestreamBuilder) UpdateScheduledQuery(scheduledQueryArn
 string) error {

    updateScheduledQueryInput := &timestreamquery.UpdateScheduledQueryInput{
        ScheduledQueryArn: aws.String(scheduledQueryArn),
        State:             aws.String(timestreamquery.ScheduledQueryStateDisabled),
    }
    _, err :=
 timestreamBuilder.QuerySvc.UpdateScheduledQuery(updateScheduledQueryInput)

    if err != nil {
        if aerr, ok := err.(awserr.Error); ok {
            switch aerr.Code() {
            case timestreamquery.ErrCodeResourceNotFoundException:
                fmt.Println(timestreamquery.ErrCodeResourceNotFoundException,
 aerr.Error())
            default:
                fmt.Printf("Error: %s", aerr.Error())
            }
        } else {
            fmt.Printf("Error: %s", err.Error())
        }
        return err
    } else {
        fmt.Println("UpdateScheduledQuery is successful")
        return nil
    }
}
```

Python

```
def update_scheduled_query(self, scheduled_query_arn, state):
    print("\nUpdating Scheduled Query")
    try:

 self.query_client.update_scheduled_query(ScheduledQueryArn=scheduled_query_arn,
                                                 State=state)
        print("Successfully update scheduled query state to", state)
    except self.query_client.exceptions.ResourceNotFoundException as err:
        print("Scheduled Query doesn't exist")
        raise err
```

```
    except Exception as err:
        print("Scheduled Query deletion failed:", err)
        raise err
```

Node.js

En el siguiente fragmento se utiliza el estilo for V2. AWS SDK JavaScript Se basa en la aplicación de ejemplo de Node.js, ejemplo de Amazon Timestream LiveAnalytics para su aplicación en. GitHub

```
async function updateScheduledQueries(scheduledQueryArn) {
     console.log("Updating Scheduled Query");
     var params = {
         ScheduledQueryArn: scheduledQueryArn,
         State: "DISABLED"
     }
     try {
         await queryClient.updateScheduledQuery(params).promise();
         console.log("Successfully update scheduled query state");
     } catch (err) {
         console.log("Update Scheduled Query failed: ", err);
         throw err;
     }
 }
```

.NET

```
private async Task UpdateScheduledQuery(string scheduledQueryArn,
 ScheduledQueryState state)
 {
     try
     {
         Console.WriteLine("Updating Scheduled Query");
         await _amazonTimestreamQuery.UpdateScheduledQueryAsync(new
 UpdateScheduledQueryRequest()
         {
             ScheduledQueryArn = scheduledQueryArn,
             State = state
         });
         Console.WriteLine("Successfully update scheduled query state");
     }
     catch (ResourceNotFoundException e)
     {
```

```
            Console.WriteLine($"Scheduled Query doesn't exist: {e}");
            throw;
        }
        catch (Exception e)
        {
            Console.WriteLine($"Update Scheduled Query failed: {e}");
            throw;
        }
    }
```

Eliminar consulta programada

Puede usar los siguientes fragmentos de código para eliminar una consulta programada.

Java

```
public void deleteScheduledQuery(String scheduledQueryArn) {
    System.out.println("Deleting Scheduled Query");

    try {
        queryClient.deleteScheduledQuery(new
 DeleteScheduledQueryRequest().withScheduledQueryArn(scheduledQueryArn));
        System.out.println("Successfully deleted scheduled query");
    }
    catch (Exception e) {
        System.out.println("Scheduled Query deletion failed: " + e);
    }
}
```

Java v2

```
public void deleteScheduledQuery(String scheduledQueryArn) {
    System.out.println("Deleting Scheduled Query");

    try {
        queryClient.deleteScheduledQuery(DeleteScheduledQueryRequest.builder()
                .scheduledQueryArn(scheduledQueryArn).build());
        System.out.println("Successfully deleted scheduled query");
    }
    catch (Exception e) {
        System.out.println("Scheduled Query deletion failed: " + e);
    }
```

```
}
```

Go

```
func (timestreamBuilder TimestreamBuilder) DeleteScheduledQuery(scheduledQueryArn
 string) error {

    deleteScheduledQueryInput := &timestreamquery.DeleteScheduledQueryInput{
        ScheduledQueryArn: aws.String(scheduledQueryArn),
    }
    _, err :=
 timestreamBuilder.QuerySvc.DeleteScheduledQuery(deleteScheduledQueryInput)

    if err != nil {
        fmt.Println("Error:")
        if aerr, ok := err.(awserr.Error); ok {
            switch aerr.Code() {
            case timestreamquery.ErrCodeResourceNotFoundException:
                fmt.Println(timestreamquery.ErrCodeResourceNotFoundException,
 aerr.Error())
            default:
                fmt.Printf("Error: %s", aerr.Error())
            }
        } else {
            fmt.Printf("Error: %s", err.Error())
        }
        return err
    } else {
        fmt.Println("DeleteScheduledQuery is successful")
        return nil
    }
 }
```

Python

```
def delete_scheduled_query(self, scheduled_query_arn):
    print("\nDeleting Scheduled Query")
    try:

 self.query_client.delete_scheduled_query(ScheduledQueryArn=scheduled_query_arn)
        print("Successfully deleted scheduled query :", scheduled_query_arn)
    except Exception as err:
        print("Scheduled Query deletion failed:", err)
```

```
        raise err
```

Node.js

En el siguiente fragmento se utiliza el estilo for V2. AWS SDK JavaScript Se basa en la aplicación de ejemplo de Node.js, ejemplo de Amazon Timestream LiveAnalytics para su aplicación en. GitHub

```
async function deleteScheduleQuery(scheduledQueryArn) {
     console.log("Deleting Scheduled Query");
     const params = {
         ScheduledQueryArn: scheduledQueryArn
     }
     try {
         await queryClient.deleteScheduledQuery(params).promise();
         console.log("Successfully deleted scheduled query");
     } catch (err) {
         console.log("Scheduled Query deletion failed: ", err);
     }
 }
```

.NET

```
private async Task DeleteScheduledQuery(string scheduledQueryArn)
 {
     try
     {
         Console.WriteLine("Deleting Scheduled Query");
         await _amazonTimestreamQuery.DeleteScheduledQueryAsync(new
 DeleteScheduledQueryRequest()
         {
             ScheduledQueryArn = scheduledQueryArn
         });
         Console.WriteLine($"Successfully deleted scheduled query :
 {scheduledQueryArn}");
     }
     catch (Exception e)
     {
         Console.WriteLine($"Scheduled Query deletion failed: {e}");
         throw;
     }
 }
```

Uso de la carga por lotes en Timestream para LiveAnalytics

Con la carga por lotes para Amazon Timestream, puede LiveAnalytics incorporar archivos almacenados en Amazon S3 CSV a Timestream por lotes. Con esta nueva funcionalidad, puede tener sus datos en Timestream LiveAnalytics sin tener que depender de otras herramientas ni escribir código personalizado. Puede utilizar la carga por lotes para rellenar los datos con tiempos de espera flexibles, como los datos que no se necesitan inmediatamente para consultarlos o analizarlos.

Puede crear tareas de carga por lotes mediante el AWS Management Console AWS CLI, el y el. AWS SDKs Para obtener más información, consulte Uso de la carga por lotes con la consola, Uso de la carga por lotes con el AWS CLI y Uso de la carga por lotes con el AWS SDKs.

Además de la carga por lotes, puede escribir varios registros al mismo tiempo con la WriteRecords API operación. Para obtener instrucciones sobre qué usar, consulteElegir entre la WriteRecords API operación y la carga por lotes.

Temas

Conceptos de carga por lotes en Timestream

Revise los siguientes conceptos para comprender mejor la funcionalidad de carga por lotes.

Tarea de carga por lotes: la tarea que define los datos de origen y el destino en Amazon Timestream. Al crear la tarea de carga por lotes, debe especificar una configuración adicional, como el modelo de datos. Puede crear tareas de carga por lotes a través de AWS Management Console AWS CLI, el y el AWS SDKs.

Destino de importación: la base de datos y la tabla de destino en Timestream. Para obtener información sobre la creación de bases de datos y tablas, consulte Creación de una base de datos de y. Creación de una tabla

Fuente de datos: el CSV archivo fuente que se almacena en un bucket de S3. Para obtener información sobre cómo preparar el archivo de datos, consultePreparación de un archivo de datos de carga por lotes. Para obtener información sobre los precios de S3, consulte los precios de Amazon S3.

Informe de errores de carga por lotes: informe que almacena información sobre los errores de una tarea de carga por lotes. La ubicación S3 para los informes de errores de carga por lotes se define como parte de una tarea de carga por lotes. Para obtener información sobre la información de los informes, consulteUso de informes de errores de carga por lotes.

Mapeo de modelos de datos: mapeo de carga por lotes de tiempo, dimensiones y medidas desde una fuente de datos en una ubicación de S3 hasta un flujo temporal de destino para LiveAnalytics una tabla. Para obtener más información, consulte Mapeos de modelos de datos para carga por lotes.

Requisitos previos de carga por lotes

Esta es una lista de requisitos previos para utilizar la carga por lotes. Para obtener las prácticas recomendadas, consulte Mejores prácticas de carga por lotes.

- Los datos de origen de carga por lotes se almacenan en Amazon S3 en CSV formato con encabezados.
- Para cada bucket de origen de Amazon S3, debe tener los siguientes permisos en una política adjunta:

```
"s3:GetObject",
"s3:GetBucketAcl"
"s3:ListBucket"
```

  Del mismo modo, para cada segmento de salida de Amazon S3 en el que se escriban los informes, debe tener los siguientes permisos en una política adjunta:

```
"s3:PutObject",
"s3:GetBucketAcl"
```

Por ejemplo:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "s3:GetObject",
        "s3:GetBucketAcl"
        "s3:ListBucket"
      ],
      "Resource": [
        "arn:aws:s3:::inputs-source-bucket-name-A”
        "arn:aws:s3:::inputs-source-bucket-name-B”
      ],
      "Effect": "Allow"
    },
    {
      "Action": [
        "s3:PutObject",
        "s3:GetBucketAcl"
      ],
      "Resource": [
        "arn:aws:s3:::reports-output-bucket-name”
      ]
      "Effect": "Allow"
    }
  ]
}
```

- Timestream for LiveAnalytics los analiza CSV mapeando la información que se proporciona en el modelo de datos a los encabezados. CSV Los datos deben tener una columna que represente la marca de tiempo, al menos una columna de dimensiones y al menos una columna de medida.
- Los cubos S3 utilizados con la carga por lotes deben estar en la misma región y proceder de la misma cuenta que la LiveAnalytics tabla Timestream for que se utiliza en la carga por lotes.
- La `timestamp` columna debe ser un tipo de datos largo que represente el tiempo transcurrido desde la época de Unix. Por ejemplo, la marca de tiempo se `2021-03-25T08:45:21Z` representaría como. `1616661921` Timestream admite segundos, milisegundos, microsegundos y nanosegundos para la precisión de la marca de tiempo. Al utilizar el lenguaje de consulta, puede

convertir entre formatos con funciones como. `to_unixtime` Para obtener más información, consulte Funciones de fecha y hora.

- Timestream admite el tipo de datos de cadena para los valores de dimensión. Admite los tipos de datos largos, dobles, de cadena y booleanos para las columnas de medida.

Para conocer los límites y las cuotas de carga de lotes, consulte. Carga por lotes

Mejores prácticas de carga por lotes

La carga por lotes funciona mejor (alto rendimiento) cuando se cumplen las siguientes condiciones y recomendaciones:

1. CSVLos archivos que se envían para su ingesta son pequeños, específicamente con un tamaño de archivo de 100 MB a 1 GB, para mejorar el paralelismo y la velocidad de ingesta.
2. Evite introducir datos simultáneamente en la misma tabla (por ejemplo, mediante la WriteRecords API operación o una consulta programada) cuando la carga del lote esté en curso. Esto podría provocar retrasos y la tarea de carga por lotes fallaría.
3. No añada, modifique ni elimine archivos del depósito de S3 utilizado en la carga por lotes mientras la tarea de carga por lotes esté en ejecución.
4. No elimine ni revoque los permisos de las tablas o la fuente, ni denuncie los depósitos de S3 que tengan tareas de carga por lotes programadas o en curso.
5. Cuando ingiera datos con un conjunto de valores de dimensión de alta cardinalidad, siga las instrucciones que se indican en. Recomendaciones para particionar registros de múltiples medidas
6. Asegúrese de comprobar la exactitud de los datos enviando un archivo pequeño. Se le cobrará por los datos que se envíen por lotes, independientemente de si son correctos. Para obtener más información sobre los precios, consulta los precios de Amazon Timestream.
7. No reanude una tarea de carga por lotes a menos que `ActiveMagneticStorePartitions` estén por debajo de 250. Es posible que el trabajo se acelere y falle. El envío de varios trabajos al mismo tiempo para la misma base de datos debería reducir el número.

Las siguientes son las prácticas recomendadas para consolas:

1. Use el generador solo para un modelado de datos más simple que use solo un nombre de medida para los registros de múltiples medidas.

2. Para un modelado de datos más complejo, utiliceJSON. Por ejemplo, JSON utilícelo cuando utilice varios nombres de medidas cuando utilice registros de varias medidas.

Para obtener información adicional sobre las LiveAnalytics mejores prácticas de Timestream, consulte. Prácticas recomendadas

Preparación de un archivo de datos de carga por lotes

Un archivo de datos de origen tiene valores separados por delimitadores. El término más específico, valores separados por comas (CSV), se utiliza de forma genérica. Los separadores de columnas válidos incluyen comas y barras verticales. Los registros están separados por líneas nuevas. Los archivos deben almacenarse en Amazon S3. Al crear una nueva tarea de carga por lotes, la ubicación de los datos de origen se especifica mediante una ARN para el archivo. Un archivo contiene encabezados. Una columna representa la marca de tiempo. Al menos otra columna representa una medida.

Los cubos S3 utilizados con la carga de lotes deben estar en la misma región que la cadena temporal de la LiveAnalytics tabla que se utiliza en la carga de lotes. No añada ni elimine archivos del depósito de S3 utilizado en la carga por lotes una vez que se haya enviado la tarea de carga por lotes. Para obtener información sobre cómo trabajar con buckets de S3, consulte Introducción a Amazon S3.

> **Note**
>
> CSVLos archivos que generan algunas aplicaciones, como Excel, pueden contener una marca de orden de bytes (BOM) que no coincide con la codificación esperada. El flujo temporal de las tareas de carga por LiveAnalytics lotes que hacen referencia a un CSV archivo BOM con un error cuando se procesan mediante programación. Para evitarlo, puedes eliminar elBOM, que es un carácter invisible.
> Por ejemplo, puede guardar el archivo desde una aplicación como Notepad++, que le permite especificar una nueva codificación. También puede utilizar una opción programática que lea la primera línea, elimine el carácter de la línea y escriba el nuevo valor sobre la primera línea del archivo.
> Al guardar desde Excel, hay varias CSV opciones. Guardar con una CSV opción diferente puede evitar el problema descrito. Sin embargo, deberías comprobar el resultado, ya que un cambio en la codificación puede afectar a algunos caracteres.

CSVparámetros de formato

Se utilizan caracteres de escape cuando se representa un valor que, de otro modo, estaría reservado por los parámetros de formato. Por ejemplo, si el carácter de comilla es una comilla doble, para representar una comilla doble en los datos, coloque el carácter de escape antes de la comilla doble.

Para obtener información sobre cuándo especificarlos al crear una tarea de carga por lotes, consulteCree una tarea de carga por lotes.

Parámetro	Opciones
Separador de columnas	(Coma (',') | Barra vertical ('|') | Punto y coma (';') | Tab ('/t') | Espacio en blanco (''))
Personaje de escape	Ninguno
Cita el personaje	Consola: (Comilla doble («) | Comilla simple (''))
Valor nulo	Espacio en blanco ('')
Recorta los espacios en blanco	Consola: (No | Sí)

Mapeos de modelos de datos para carga por lotes

A continuación se analiza el esquema de las asignaciones de modelos de datos y se proporciona un ejemplo.

Esquema de mapeo de modelos de datos

La sintaxis de la `CreateBatchLoadTask` solicitud y un `BatchLoadTaskDescription` objeto devuelto por una llamada para `DescribeBatchLoadTask` incluir un `DataModelConfiguration` objeto que incluye el objeto `DataModel` para la carga por lotes. `DataModel`Define las asignaciones desde los datos de origen que se almacenan en CSV formato en una ubicación de S3 hasta una cadena temporal de destino para la base de datos y la tabla. LiveAnalytics

El `TimeColumn` campo indica la ubicación de los datos de origen para el valor que se va a asignar a la columna de la tabla de destino en Timestream for. `time` LiveAnalytics `TimeUnit`Especifica la unidad de`TimeColumn`, y puede ser una de`MILLISECONDS`,`SECONDS`, `MICROSECONDS` o.

NANOSECONDS También hay mapeos de dimensiones y medidas. Los mapeos de dimensiones se componen de columnas de origen y campos de destino.

Para obtener más información, consulte. DimensionMapping Los mapeos de las medidas tienen dos opciones: yMixedMeasureMappings. MultiMeasureMappings

En resumen, a DataModel contiene las asignaciones de una fuente de datos en una ubicación de S3 a una cadena temporal de destino para la siguiente tabla. LiveAnalytics

- Tiempo
- Dimensiones
- Medidas

Si es posible, le recomendamos que asigne los datos de medición a registros de medidas múltiples en Timestream for. LiveAnalytics Para obtener información sobre las ventajas de los registros de medidas múltiples, consulte. Registros de medidas múltiples

Si se almacenan varias medidas en los datos de origen en una fila, puede mapear esas múltiples medidas a registros de múltiples medidas en Timestream para su uso. LiveAnalytics MultiMeasureMappings Si hay valores que deben asignarse a un registro de una sola medida, puede utilizarlos. MixedMeasureMappings

MixedMeasureMappingsy MultiMeasureMappings ambos incluyenMultiMeasureAttributeMappings. Se admiten registros de medidas múltiples independientemente de si se necesitan registros de una sola medida.

Si solo se necesitan registros objetivo de medidas múltiples en Timestream LiveAnalytics, puede definir los mapeos de medidas en la siguiente estructura.

```
CreateBatchLoadTask
    MeasureNameColumn
    MultiMeasureMappings
        TargetMultiMeasureName
        MultiMeasureAttributeMappings array
```

> **Note**
>
> Recomendamos utilizarlos siempre que sea posible. MultiMeasureMappings

Si se necesitan registros de objetivos de medida única en Timestream LiveAnalytics, puede definir las asignaciones de medidas en la siguiente estructura.

```
CreateBatchLoadTask
    MeasureNameColumn
    MixedMeasureMappings array
        MixedMeasureMapping
            MeasureName
            MeasureValueType
            SourceColumn
            TargetMeasureName
            MultiMeasureAttributeMappings array
```

Cuando se utiliza, la matriz siempre es `MultiMeasureMappings` necesaria. `MultiMeasureAttributeMappings` Cuando se usa la `MixedMeasureMappings` matriz, si `MeasureValueType` es `MULTI` para algo dado`MixedMeasureMapping`, `MultiMeasureAttributeMappings` es necesario para eso`MixedMeasureMapping`. De lo contrario, `MeasureValueType` indica el tipo de medida para el registro de medida única.

De cualquier forma, hay una variedad de opciones `MultiMeasureAttributeMapping` disponibles. Las asignaciones a los registros de medidas múltiples en cada `MultiMeasureAttributeMapping` una de ellas se definen de la siguiente manera:

`SourceColumn`

La columna de los datos de origen que se encuentra en Amazon S3.

`TargetMultiMeasureAttributeName`

El nombre del nombre de la multimedida de destino en la tabla de destino. Esta entrada es obligatoria cuando no `MeasureNameColumn` se proporciona. Si `MeasureNameColumn` se proporciona, el valor de esa columna se utiliza como nombre de las múltiples medidas.

`MeasureValueType`

Uno de`DOUBLE`, `BIGINT` `BOOLEAN``VARCHAR`, o`TIMESTAMP`.

Mapeos de modelos de datos con un ejemplo **`MultiMeasureMappings`**

Este ejemplo muestra el mapeo a registros de múltiples medidas, el enfoque preferido, que almacena cada valor de medida en una columna dedicada. Puede descargar una muestra CSV en sample

CSV. El ejemplo tiene los siguientes encabezados para asignarlos a una columna de destino de una tabla Timestream for. LiveAnalytics

- `time`
- `measure_name`
- `region`
- `location`
- `hostname`
- `memory_utilization`
- `cpu_utilization`

Identifique las `measure_name` columnas `time` y del archivo. CSV En este caso, se asignan directamente al Timestream para las columnas de LiveAnalytics la tabla con los mismos nombres.

- `time`ese asigna a `time`
- `measure_name`se asigna a `measure_name` (o al valor que elija)

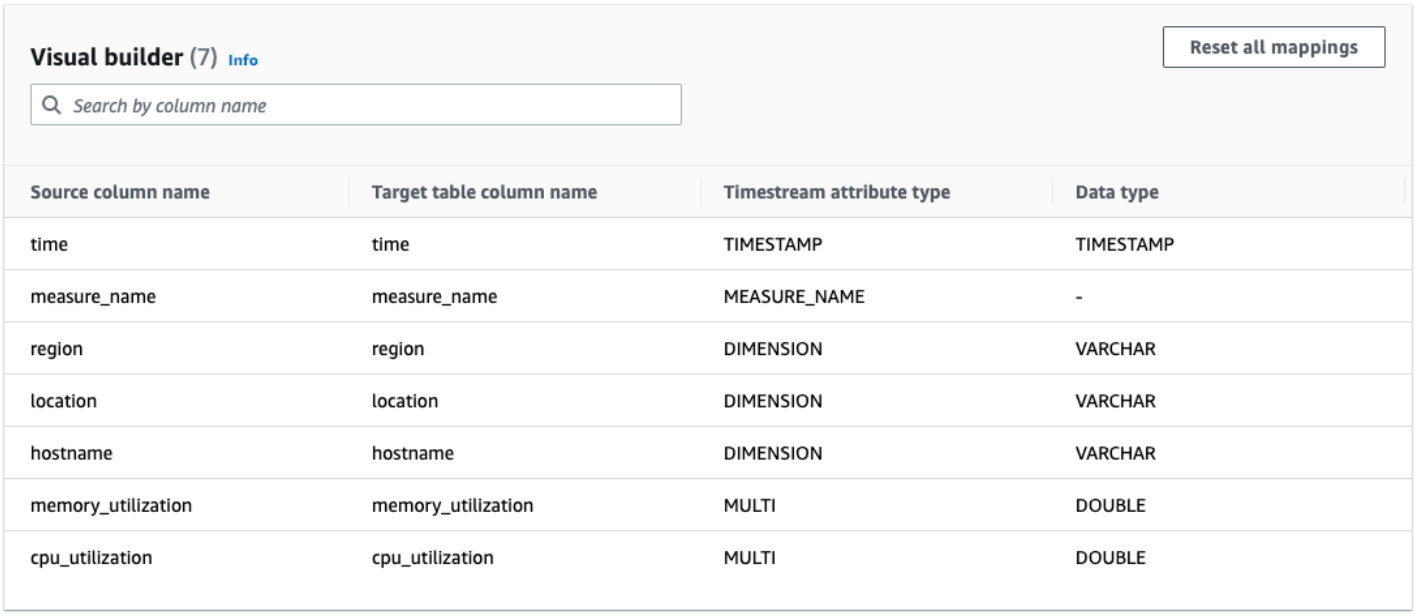
Al utilizar elAPI, se especifica `time` en el `TimeColumn` campo y un valor de unidad de tiempo compatible, como `MILLISECONDS` el del `TimeUnit` campo. Corresponden al nombre de la columna de origen y a la entrada de hora y hora en la consola. Puede agrupar o particionar los registros según `measure_name` se defina con la clave. `MeasureNameColumn`

En la muestra`region`,`location`, y `hostname` son dimensiones. Las dimensiones se mapean en una matriz de `DimensionMapping` objetos.

En el caso de las medidas, el valor se `TargetMultiMeasureAttributeName` convertirá en una columna de la tabla Timestream for. LiveAnalytics Puede conservar el mismo nombre, como en este ejemplo. O bien, puede especificar uno nuevo. `MeasureValueTypees` uno de los `DOUBLE` siguientes: `BIGINTBOOLEAN`,`VARCHAR`,, o`TIMESTAMP`.

```
{
  "TimeColumn": "time",
  "TimeUnit": "MILLISECONDS",
  "DimensionMappings": [
    {
      "SourceColumn": "region",
      "DestinationColumn": "region"
```

```
    },
    {
      "SourceColumn": "location",
      "DestinationColumn": "location"
    },
    {
      "SourceColumn": "hostname",
      "DestinationColumn": "hostname"
    }
  ],
  "MeasureNameColumn": "measure_name",
  "MultiMeasureMappings": {
    "MultiMeasureAttributeMappings": [
      {
        "SourceColumn": "memory_utilization",
        "TargetMultiMeasureAttributeName": "memory_utilization",
        "MeasureValueType": "DOUBLE"
      },
      {
        "SourceColumn": "cpu_utilization",
        "TargetMultiMeasureAttributeName": "cpu_utilization",
        "MeasureValueType": "DOUBLE"
      }
    ]
  }
}
```



Visual builder (7) Info

Reset all mappings

Search by column name

Source column name	Target table column name	Timestream attribute type	Data type
time	time	TIMESTAMP	TIMESTAMP
measure_name	measure_name	MEASURE_NAME	-
region	region	DIMENSION	VARCHAR
location	location	DIMENSION	VARCHAR
hostname	hostname	DIMENSION	VARCHAR
memory_utilization	memory_utilization	MULTI	DOUBLE
cpu_utilization	cpu_utilization	MULTI	DOUBLE

Mapeos de modelos de datos con un ejemplo `MixedMeasureMappings`

Le recomendamos que utilice este enfoque solo cuando necesite mapear registros de una sola medida en Timestream para. LiveAnalytics

Uso de la carga por lotes con la consola

Los siguientes son los pasos para usar la carga por lotes con el AWS Management Console. Puede descargar una muestra CSV en sample CSV.

Temas

- Acceda a la carga por lotes
- Cree una tarea de carga por lotes
- Reanude una tarea de carga por lotes
- Uso del generador visual

Acceda a la carga por lotes

Siga estos pasos para acceder a la carga por lotes mediante el AWS Management Console.

1. Abre la consola Amazon Timestream.
2. En el panel de navegación, elija Herramientas de administración y, a continuación, elija Tareas de carga por lotes.
3. Desde aquí, puede ver la lista de tareas de carga por lotes y profundizar en una tarea determinada para obtener más detalles. También puede crear y reanudar tareas.

Cree una tarea de carga por lotes

Siga estos pasos para crear una tarea de carga por lotes mediante AWS Management Console.

1. Abre la consola Amazon Timestream.
2. En el panel de navegación, elija Herramientas de administración y, a continuación, elija Tareas de carga por lotes.
3. Seleccione Crear tarea de carga por lotes.
4. En Destino de importación, elija lo siguiente.

- Base de datos de destino: seleccione el nombre de la base de datos creada enCreación de una base de datos de .
- Tabla de destino: seleccione el nombre de la tabla creada enCreación de una tabla.

Si es necesario, puede añadir una tabla desde este panel con el botón Crear tabla nueva.

5. Desde la ubicación de la fuente de datos S3 en Fuente de datos, seleccione el depósito S3 donde se almacenan los datos de la fuente. Utilice el botón Examinar S3 para ver los recursos de S3 a los que tiene acceso la AWS cuenta activa o introduzca la ubicación de S3URL. La fuente de datos debe estar ubicada en la misma región.
6. En la configuración de formato de archivo (sección ampliable), puede usar la configuración predeterminada para analizar los datos de entrada. También puede elegir la configuración avanzada. Desde allí, puede elegir los parámetros de CSV formato y seleccionar los parámetros para analizar los datos de entrada. Para obtener información sobre estos parámetros, consulteCSVparámetros de formato.
7. En Configurar el mapeo del modelo de datos, configure el modelo de datos. Para obtener orientación adicional sobre el modelo de datos, consulte Mapeos de modelos de datos para carga por lotes

   - En Mapeo de modelos de datos, elija la entrada de configuración de mapeo y elija una de las siguientes opciones.
     - Creador visual: para mapear datos de forma visual, elija TargetMultiMeasureNameo MeasureNameColumn. A continuación, desde Visual Builder, mapee las columnas.

       Visual Builder detecta y carga automáticamente los encabezados de las columnas de origen desde el archivo de fuente de datos cuando se selecciona un solo CSV archivo como fuente de datos. Elija el atributo y el tipo de datos para crear el mapeo.

       Para obtener información sobre el uso del generador visual, consulteUso del generador visual.
     - JSONeditor: un JSON editor de formato libre para configurar el modelo de datos. Elija esta opción si está familiarizado con Timestream for LiveAnalytics y desea crear mapeos avanzados de modelos de datos.
     - JSONarchivo de S3: seleccione un archivo de JSON modelo que haya almacenado en S3. Elija esta opción si ya ha configurado un modelo de datos y desea reutilizarlo para cargas de lotes adicionales.

8. En la ubicación de S3 de los registros de errores en el informe del registro de errores, seleccione la ubicación de S3 que se utilizará para informar de los errores. Para obtener información sobre cómo utilizar este informe, consulteUso de informes de errores de carga por lotes.
9. Para el tipo de clave de cifrado, elija una de las siguientes opciones.
   - Clave administrada por Amazon SSE S3 (-S3): clave de cifrado que Amazon S3 crea, administra y usa por usted.
   - AWS KMS key (SSE-KMS): clave de cifrado protegida por AWS Key Management Service ().AWS KMS
10. Elija Next (Siguiente).
11. En la página Revisar y crear, revise la configuración y edítela según sea necesario.

> **Note**
>
> No puede cambiar la configuración de las tareas de carga por lotes una vez creada la tarea. Los tiempos de finalización de las tareas variarán en función de la cantidad de datos que se importen.

12. Seleccione Crear tarea de carga por lotes.

Reanude una tarea de carga por lotes

Cuando selecciona una tarea de carga por lotes con el estado «Progreso detenido» y que aún se puede reanudar, se le solicitará que reanude la tarea. También hay un banner con el botón Reanudar la tarea para ver los detalles de esas tareas. Las tareas que se pueden reanudar tienen una fecha de caducidad. Una vez expirada esa fecha, las tareas no se pueden reanudar.

Uso del generador visual

Puede usar el generador visual para mapear las columnas de datos de origen, uno o más CSV archivos almacenados en un bucket de S3, a las columnas de destino de una tabla Timestream for LiveAnalytics .

> **Note**
>
> Su función necesitará el `SelectObjectContent` permiso para el archivo. De lo contrario, tendrá que añadir y eliminar columnas manualmente.

Modo de carga automática de columnas de origen

Timestream for LiveAnalytics puede escanear automáticamente el CSV archivo fuente en busca de nombres de columnas si solo se especifica un depósito. Cuando no haya ningún mapeo existente, puede elegir Importar columnas de origen.

1. Con la opción Visual Builder seleccionada en los ajustes de entrada de la configuración de mapeo, configure la entrada de hora y hora. `Milliseconds`es la configuración por defecto.
2. Haga clic en el botón Cargar columnas de origen para importar los encabezados de columna que se encuentran en el archivo de datos de origen. La tabla se rellenará con los nombres de los encabezados de las columnas de origen del archivo de origen de datos.
3. Elija el nombre de la columna de la tabla de destino, el tipo de atributo Timestream y el tipo de datos para cada columna de origen.

   Para obtener más información sobre estas columnas y sus posibles valores, consulte. Asignación de campos
4. Utilice la drag-to-fill función para establecer el valor de varias columnas a la vez.

Añada manualmente las columnas de origen

Si utilizas un segmento o un CSV prefijo y no uno soloCSV, puedes añadir y eliminar mapeos de columnas desde el editor visual con los botones Añadir mapeo de columnas y Eliminar mapeo de columnas. También hay un botón para restablecer las asignaciones.

Asignación de campos

- Nombre de la columna de origen: el nombre de una columna del archivo de origen que representa una medida que se va a importar. Timestream for LiveAnalytics puede rellenar este valor automáticamente cuando se utilizan las columnas de origen de importación.
- Nombre de la columna de la tabla de destino: entrada opcional que indica el nombre de la columna de la medida en la tabla de destino.
- Tipo de atributo de flujo temporal: el tipo de atributo de los datos de la columna de origen especificada, por ejemplo. `DIMENSION`
  - TIMESTAMP— Especifica cuándo se recopiló una medida.
  - MULTI— Se representan varias medidas.
  - DIMENSION— Metadatos de series temporales.
  - MEASURE_ NAME — Para los registros de una sola medida, este es el nombre de la medida.

- Tipo de datos: el tipo de columna Timestream, por ejemplo. `BOOLEAN`
  - BIGINT— Un entero de 64 bits.
  - BOOLEAN— Los dos valores de verdad de la lógica: verdadero y falso.
  - DOUBLE— Número de precisión variable de 64 bits.
  - TIMESTAMP— Una instancia temporal que utiliza un tiempo de precisión de nanosegundos y registra el tiempo UTC transcurrido desde la época de Unix.

Uso de la carga por lotes con el AWS CLI

Configuración

Para empezar a utilizar la carga por lotes, siga los siguientes pasos.

1. Instálelo AWS CLI siguiendo las instrucciones que se encuentran enAcceso a Amazon Timestream LiveAnalytics para usar el AWS CLI.
2. Ejecute el siguiente comando para comprobar que los CLI comandos de Timestream se han actualizado. Compruebe que create-batch-load-task esté en la lista.

   ```
   aws timestream-write help
   ```
3. Prepare una fuente de datos siguiendo las instrucciones dePreparación de un archivo de datos de carga por lotes.
4. Cree una base de datos y una tabla siguiendo las instrucciones deAcceso a Amazon Timestream LiveAnalytics para usar el AWS CLI.
5. Cree un depósito de S3 para la producción de informes. El depósito debe estar en la misma región. Para obtener más información sobre los buckets, consulte Crear, configurar y trabajar con buckets de Amazon S3.
6. Cree una tarea de carga por lotes. Para ver los pasos, consulte Cree una tarea de carga por lotes.
7. Confirme el estado de la tarea. Para ver los pasos, consulte Describa la tarea de carga por lotes.

Cree una tarea de carga por lotes

Puede crear una tarea de carga por lotes con el `create-batch-load-task` comando. Al crear una tarea de carga por lotes mediante elCLI, puede utilizar un JSON parámetro`cli-input-json`, que le permite agregar los parámetros en un solo JSON fragmento. También puede separar esos

detalles utilizando varios otros parámetros`data-model-configuration`, como `data-source-configuration``report-configuration`,`target-database-name`, y`target-table-name`.

Para ver un ejemplo, consulte Ejemplo de creación de una tarea de carga por lotes.

Describa la tarea de carga por lotes

Puede recuperar la descripción de una tarea de carga por lotes de la siguiente manera.

```
aws timestream-write describe-batch-load-task --task-id <value>
```

A continuación, se muestra un ejemplo de respuesta.

```
{
    "BatchLoadTaskDescription": {
        "TaskId": "<TaskId>",
        "DataSourceConfiguration": {
            "DataSourceS3Configuration": {
                "BucketName": "test-batch-load-west-2",
                "ObjectKeyPrefix": "sample.csv"
            },
            "CsvConfiguration": {},
            "DataFormat": "CSV"
        },
        "ProgressReport": {
            "RecordsProcessed": 2,
            "RecordsIngested": 0,
            "FileParseFailures": 0,
            "RecordIngestionFailures": 2,
            "FileFailures": 0,
            "BytesIngested": 119
        },
        "ReportConfiguration": {
            "ReportS3Configuration": {
                "BucketName": "test-batch-load-west-2",
                "ObjectKeyPrefix": "<ObjectKeyPrefix>",
                "EncryptionOption": "SSE_S3"
            }
        },
        "DataModelConfiguration": {
            "DataModel": {
                "TimeColumn": "timestamp",
                "TimeUnit": "SECONDS",
```

```
                "DimensionMappings": [
                    {
                        "SourceColumn": "vehicle",
                        "DestinationColumn": "vehicle"
                    },
                    {
                        "SourceColumn": "registration",
                        "DestinationColumn": "license"
                    }
                ],
                "MultiMeasureMappings": {
                    "TargetMultiMeasureName": "test",
                    "MultiMeasureAttributeMappings": [
                        {
                            "SourceColumn": "wgt",
                            "TargetMultiMeasureAttributeName": "weight",
                            "MeasureValueType": "DOUBLE"
                        },
                        {
                            "SourceColumn": "spd",
                            "TargetMultiMeasureAttributeName": "speed",
                            "MeasureValueType": "DOUBLE"
                        },
                        {
                            "SourceColumn": "fuel",
                            "TargetMultiMeasureAttributeName": "fuel",
                            "MeasureValueType": "DOUBLE"
                        },
                        {
                            "SourceColumn": "miles",
                            "TargetMultiMeasureAttributeName": "miles",
                            "MeasureValueType": "DOUBLE"
                        }
                    ]
                }
            }
        },
        "TargetDatabaseName": "BatchLoadExampleDatabase",
        "TargetTableName": "BatchLoadExampleTable",
        "TaskStatus": "FAILED",
        "RecordVersion": 1,
        "CreationTime": 1677167593.266,
        "LastUpdatedTime": 1677167602.38
    }
```

```
}
```

Enumere las tareas de carga por lotes

Puede enumerar las tareas de carga por lotes de la siguiente manera.

```
aws timestream-write list-batch-load-tasks
```

El resultado aparece de la siguiente manera.

```
{
    "BatchLoadTasks": [
        {
            "TaskId": "<TaskId>",
            "TaskStatus": "FAILED",
            "DatabaseName": "BatchLoadExampleDatabase",
            "TableName": "BatchLoadExampleTable",
            "CreationTime": 1677167593.266,
            "LastUpdatedTime": 1677167602.38
        }
    ]
}
```

Reanude la tarea de carga por lotes

Puede reanudar una tarea de carga por lotes de la siguiente manera.

```
aws timestream-write resume-batch-load-task --task-id <value>
```

Una respuesta puede indicar que se ha realizado correctamente o contener información sobre un error.

Ejemplo de creación de una tarea de carga por lotes

Example

1. Cree un flujo temporal para la LiveAnalytics base de datos denominada `BatchLoad` y una tabla denominada. `BatchLoadTest` Compruebe y, si es necesario, ajuste los valores de `MemoryStoreRetentionPeriodInHours` y. `MagneticStoreRetentionPeriodInDays`

```
aws timestream-write create-database --database-name BatchLoad \

aws timestream-write create-table --database-name BatchLoad \
--table-name BatchLoadTest \
--retention-properties "{\"MemoryStoreRetentionPeriodInHours\": 12,
 \"MagneticStoreRetentionPeriodInDays\": 100}"
```

2. Con la consola, cree un bucket de S3 y copie el `sample.csv` archivo en esa ubicación. Puede descargar una muestra CSV en sample CSV.
3. Con la consola, cree un depósito de S3 para que Timestream escriba un informe si la tarea de carga por lotes finaliza con errores. LiveAnalytics
4. Cree una tarea de carga por lotes. Asegúrese de reemplazar *$INPUT_BUCKET* y *$REPORT_BUCKET* con los depósitos que creó en los pasos anteriores.

```
aws timestream-write create-batch-load-task \
--data-model-configuration "{\
            \"DataModel\": {\
              \"TimeColumn\": \"timestamp\",\
              \"TimeUnit\": \"SECONDS\",\
              \"DimensionMappings\": [\
                {\
                  \"SourceColumn\": \"vehicle\"\
                },\
                {\
                  \"SourceColumn\": \"registration\",\
                  \"DestinationColumn\": \"license\"\
                }\
              ],
              \"MultiMeasureMappings\": {\
                \"TargetMultiMeasureName\": \"mva_measure_name\",\
                \"MultiMeasureAttributeMappings\": [\
                  {\
                    \"SourceColumn\": \"wgt\",\
                    \"TargetMultiMeasureAttributeName\": \"weight\",\
                    \"MeasureValueType\": \"DOUBLE\"\
                  },\
                  {\
                    \"SourceColumn\": \"spd\",\
                    \"TargetMultiMeasureAttributeName\": \"speed\",\
                    \"MeasureValueType\": \"DOUBLE\"\
                  },\
```

```
                  {\
                    \"SourceColumn\": \"fuel_consumption\",\
                    \"TargetMultiMeasureAttributeName\": \"fuel\",\
                    \"MeasureValueType\": \"DOUBLE\"\
                  },\
                  {\
                    \"SourceColumn\": \"miles\",\
                    \"MeasureValueType\": \"BIGINT\"\
                  }\
                ]\
              }\
            }\
          }" \
--data-source-configuration "{
            \"DataSourceS3Configuration\": {\
              \"BucketName\": \"$INPUT_BUCKET\",\
              \"ObjectKeyPrefix\": \"$INPUT_OBJECT_KEY_PREFIX\"
            },\
            \"DataFormat\": \"CSV\"\
          }" \
--report-configuration "{\
            \"ReportS3Configuration\": {\
              \"BucketName\": \"$REPORT_BUCKET\",\
              \"EncryptionOption\": \"SSE_S3\"\
            }\
          }" \
--target-database-name BatchLoad \
--target-table-name BatchLoadTest
```

El comando anterior devuelve el siguiente resultado.

```
{
    "TaskId": "TaskId "
}
```

5. Compruebe el progreso de la tarea. Asegúrese de reemplazar *$TASK_ID* con el identificador de tarea que se devolvió en el paso anterior.

```
aws timestream-write describe-batch-load-task --task-id $TASK_ID
```

Ejemplo de resultado

```
{
    "BatchLoadTaskDescription": {
        "ProgressReport": {
            "BytesIngested": 1024,
            "RecordsIngested": 2,
            "FileFailures": 0,
            "RecordIngestionFailures": 0,
            "RecordsProcessed": 2,
            "FileParseFailures": 0
        },
        "DataModelConfiguration": {
            "DataModel": {
                "DimensionMappings": [
                    {
                        "SourceColumn": "vehicle",
                        "DestinationColumn": "vehicle"
                    },
                    {
                        "SourceColumn": "registration",
                        "DestinationColumn": "license"
                    }
                ],
                "TimeUnit": "SECONDS",
                "TimeColumn": "timestamp",
                "MultiMeasureMappings": {
                    "MultiMeasureAttributeMappings": [
                        {
                            "TargetMultiMeasureAttributeName": "weight",
                            "SourceColumn": "wgt",
                            "MeasureValueType": "DOUBLE"
                        },
                        {
                            "TargetMultiMeasureAttributeName": "speed",
                            "SourceColumn": "spd",
                            "MeasureValueType": "DOUBLE"
                        },
                        {
                            "TargetMultiMeasureAttributeName": "fuel",
                            "SourceColumn": "fuel_consumption",
                            "MeasureValueType": "DOUBLE"
                        },
                        {
```

```
                            "TargetMultiMeasureAttributeName": "miles",
                            "SourceColumn": "miles",
                            "MeasureValueType": "DOUBLE"
                        }
                    ],
                    "TargetMultiMeasureName": "mva_measure_name"
                }
            }
        },
        "TargetDatabaseName": "BatchLoad",
        "CreationTime": 1672960381.735,
        "TaskStatus": "SUCCEEDED",
        "RecordVersion": 1,
        "TaskId": "TaskId ",
        "TargetTableName": "BatchLoadTest",
        "ReportConfiguration": {
            "ReportS3Configuration": {
                "EncryptionOption": "SSE_S3",
                "ObjectKeyPrefix": "ObjectKeyPrefix ",
                "BucketName": "test-report-bucket"
            }
        },
        "DataSourceConfiguration": {
            "DataSourceS3Configuration": {
                "ObjectKeyPrefix": "sample.csv",
                "BucketName": "test-input-bucket"
            },
            "DataFormat": "CSV",
            "CsvConfiguration": {}
        },
        "LastUpdatedTime": 1672960387.334
    }
}
```

Uso de la carga por lotes con el AWS SDKs

Para ver ejemplos de cómo crear, describir y enumerar las tareas de carga por lotes con las AWS SDKsCrear tarea de carga por lotes, consulteDescriba la tarea de carga por lotes,Listar tareas de carga por lotes, yReanudar tarea de carga por lotes.

Uso de informes de errores de carga por lotes

Las tareas de carga por lotes tienen uno de los siguientes valores de estado:

- `CREATED`(Creada): se crea la tarea.
- `IN_PROGRESS`(En curso): la tarea está en curso.
- `FAILED`(Fallo): la tarea se ha completado. Pero se detectaron uno o más errores.
- `SUCCEEDED`(Completada): la tarea se completó sin errores.
- `PROGRESS_STOPPED`(Progreso detenido): la tarea se detuvo pero no se completó. Puede intentar reanudar la tarea.
- `PENDING_RESUME`(Reanudación pendiente): la tarea está pendiente de reanudarse.

Cuando hay errores, se crea un informe de registro de errores en el bucket de S3 definido para ello. Los errores se clasifican como matrices independientes taskErrors o fileErrors en matrices independientes. A continuación se muestra un ejemplo de informe de errores.

```
{
    "taskId": "9367BE28418C5EF902676482220B631C",
    "taskErrors": [],
    "fileErrors": [
        {
            "fileName": "example.csv",
            "errors": [
                {
                    "reason": "The record timestamp is outside the time range of the
 data ingestion window.",
                    "lineRanges": [
                        [
                            2,
                            3
                        ]
                    ]
                }
            ]
        }
    ]
}
```

Uso de consultas programadas en Timestream para LiveAnalytics

La función de consulta programada de Amazon Timestream LiveAnalytics for es una solución totalmente gestionada, escalable y sin servidor para calcular y almacenar agregados, acumulaciones

y otros tipos de datos preprocesados que se suelen utilizar para paneles operativos, informes empresariales, análisis ad hoc y otras aplicaciones. Las consultas programadas permiten que los análisis en tiempo real sean más eficaces y rentables, lo que le permite obtener información adicional a partir de sus datos y seguir tomando mejores decisiones empresariales.

Con las consultas programadas, usted define las consultas de análisis en tiempo real que calculan los agregados, las acumulaciones y otras operaciones en los datos, y Amazon Timestream LiveAnalytics for ejecuta estas consultas de forma periódica y automática y escribe de forma fiable los resultados de las consultas en una tabla independiente. Por lo general, los datos se calculan y actualizan en estas tablas en cuestión de minutos.

A continuación, puede orientar sus paneles e informes para consultar las tablas que contienen datos agregados en lugar de consultar las tablas de origen, considerablemente más grandes. Esto se traduce en ganancias de rendimiento y costos que pueden superar varios órdenes de magnitud. Esto se debe a que las tablas con datos agregados contienen muchos menos datos que las tablas de origen, por lo que ofrecen consultas más rápidas y un almacenamiento de datos más económico.

Además, las tablas con consultas programadas ofrecen todas las funciones existentes de un Timestream for LiveAnalytics Table. Por ejemplo, puede consultar las tablas utilizando. SQL Puede visualizar los datos almacenados en las tablas con Grafana. También puede incorporar datos a la tabla mediante Amazon Kinesis, AmazonMSK, AWS IoT Core y Telegraf. Puede configurar las políticas de retención de datos en estas tablas para gestionar automáticamente el ciclo de vida de los datos.

Como la retención de datos de las tablas que contienen datos agregados está totalmente disociada de la de las tablas de origen, también puede optar por reducir la retención de datos de las tablas de origen y conservar los datos agregados durante mucho más tiempo, a una fracción del costo de almacenamiento de los datos. Las consultas programadas permiten que el análisis en tiempo real sea más rápido, económico y, por lo tanto, más accesible para muchos más clientes, de modo que puedan supervisar sus aplicaciones y tomar mejores decisiones empresariales basadas en los datos.

Temas

- Programe expresiones para consultas programadas
- Mapeos de modelos de datos para consultas programadas
- Mensajes de notificación de consultas programadas
- Informes de errores de consultas programadas
- Ejemplos y patrones de consultas programadas

Ventajas de las consultas programadas

Los beneficios de las consultas programadas son los siguientes:

- Facilidad operativa: las consultas programadas no requieren servidor y se gestionan por completo.
- Rendimiento y coste: dado que las consultas programadas calculan previamente los agregados, las acumulaciones u otras operaciones de análisis en tiempo real de los datos y almacenan los resultados en una tabla, las consultas que acceden a las tablas rellenadas por consultas programadas contienen menos datos que las tablas de origen. Por lo tanto, las consultas que se ejecutan en estas tablas son más rápidas y económicas. Las tablas rellenadas con cálculos programados contienen menos datos que sus tablas de origen y, por lo tanto, ayudan a reducir el costo de almacenamiento. También puede conservar estos datos durante más tiempo en el almacén de memoria por una fracción del coste de conservar los datos de origen en el almacén de memoria.
- Interoperabilidad: las tablas rellenadas con consultas programadas ofrecen todas las funciones existentes de Timestream para LiveAnalytics tablas y se pueden utilizar con todos los servicios y herramientas que funcionan con Timestream for. LiveAnalytics Consulte Trabajar con otros servicios para obtener más información.

Casos de uso de consultas programadas

Puede utilizar consultas programadas para elaborar informes empresariales que resuman la actividad de los usuarios finales en sus aplicaciones, de forma que pueda entrenar los modelos de aprendizaje automático para su personalización. También puede utilizar las consultas programadas para crear alarmas que detecten anomalías, intrusiones en la red o actividades fraudulentas, de forma que pueda tomar medidas correctivas de inmediato.

Además, puede utilizar consultas programadas para una gobernanza de datos más eficaz. Puede hacerlo concediendo acceso a la tabla de origen exclusivamente a las consultas programadas y

proporcionando a sus desarrolladores acceso únicamente a las tablas rellenadas por las consultas programadas. Esto minimiza el impacto de las consultas no intencionadas y de larga duración.

Ejemplo: usar análisis en tiempo real para detectar pagos fraudulentos y tomar mejores decisiones comerciales

Pensemos en un sistema de pago que procese las transacciones enviadas desde múltiples point-of-sale terminales distribuidas en las principales ciudades metropolitanas de los Estados Unidos. Desea utilizar Amazon Timestream LiveAnalytics para almacenar y analizar los datos de las transacciones, de modo que pueda detectar transacciones fraudulentas y ejecutar consultas de análisis en tiempo real. Estas consultas pueden ayudarle a responder a preguntas empresariales, como identificar las point-of-sale terminales más concurridas y menos utilizadas por hora, la hora más concurrida del día en cada ciudad y la ciudad con más transacciones por hora.

El sistema procesa aproximadamente 100 000 transacciones por minuto. Cada transacción almacenada en Amazon Timestream LiveAnalytics es de 100 bytes. Ha configurado 10 consultas que se ejecutan cada minuto para detectar varios tipos de pagos fraudulentos. También has creado 25 consultas que agrupan y dividen tus datos en varias dimensiones para ayudarte a responder a tus preguntas empresariales. Cada una de estas consultas procesa los datos de la última hora.

Ha creado un panel de control para mostrar los datos generados por estas consultas. El panel contiene 25 widgets, se actualiza cada hora y, por lo general, 10 usuarios acceden a él en un momento dado. Por último, el almacén de memoria está configurado con un período de retención de datos de 2 horas y el almacén magnético está configurado para tener un período de retención de datos de 6 meses.

En este caso, puede utilizar consultas de análisis en tiempo real para volver a calcular los datos cada vez que se accede al panel y se actualiza, o utilizar tablas derivadas para el panel. El coste de las consultas de los paneles basados en consultas de análisis en tiempo real será de 120,70$ al mes. Por el contrario, el coste de las consultas de panel impulsadas por tablas derivadas será de 12,27 USD al mes (consulte Amazon Timestream para conocer los precios). LiveAnalytics En este caso, el uso de tablas derivadas reduce el coste de la consulta unas 10 veces.

Conceptos de consultas programadas

Cadena de consulta: se trata de la consulta cuyo resultado está calculando previamente y almacenando en otra tabla Timestream for. LiveAnalytics Puede definir una consulta programada utilizando toda la SQL superficie de Timestream LiveAnalytics, lo que le proporciona la flexibilidad de

escribir consultas con expresiones de tabla comunes, consultas anidadas, funciones de ventana o cualquier tipo de funciones agregadas y escalares compatibles con Timestream para el lenguaje de consultas. LiveAnalytics

Expresión de programación: le permite especificar cuándo se ejecutarán las instancias de consulta programadas. Puede especificar las expresiones mediante una expresión cron (como ejecutar a las 8 de la mañana UTC todos los días) o una expresión de frecuencia (como ejecutar cada 10 minutos).

Configuración de destino: le permite especificar cómo asignar el resultado de una consulta programada a la tabla de destino, donde se almacenarán los resultados de esta consulta programada.

Configuración de notificaciones: Timestream for ejecuta LiveAnalytics automáticamente instancias de una consulta programada en función de la expresión de la programación. Recibirá una notificación por cada consulta de este tipo que se ejecute en un SNS tema que configure al crear una consulta programada. Esta notificación especifica si la instancia se ejecutó correctamente o si se detectó algún error. Además, proporciona información como los bytes medidos, los datos escritos en la tabla de destino, la hora de la próxima invocación, etc.

A continuación se muestra un ejemplo de este tipo de mensaje de notificación.

```
{
    "type":"AUTO_TRIGGER_SUCCESS",
    "arn":"arn:aws:timestream:us-east-1:123456789012:scheduled-query/
 PT1mPerMinutePerRegionMeasureCount-9376096f7309",
    "nextInvocationEpochSecond":1637302500,
    "scheduledQueryRunSummary":
    {
        "invocationEpochSecond":1637302440,
        "triggerTimeMillis":1637302445697,
        "runStatus":"AUTO_TRIGGER_SUCCESS",
        "executionStats":
        {
            "executionTimeInMillis":21669,
            "dataWrites":36864,
            "bytesMetered":13547036820,
            "recordsIngested":1200,
            "queryResultRows":1200
        }
    }
}
```

En este mensaje de notificación, `bytesMetered` están los bytes que la consulta escaneó en la tabla de origen y dataWrites son los bytes escritos en la tabla de destino.

> **Note**
>
> Si consume estas notificaciones mediante programación, tenga en cuenta que podrían añadirse nuevos campos al mensaje de notificación en el futuro.

Ubicación del informe de errores: las consultas programadas se ejecutan y almacenan datos de forma asíncrona en la tabla de destino. Si una instancia encuentra algún error (por ejemplo, datos no válidos que no se pudieron almacenar), los registros que detectaron errores se escriben en un informe de errores en la ubicación del informe de errores que especifiques al crear una consulta programada. Debe especificar el depósito y el prefijo de S3 para la ubicación. Timestream for LiveAnalytics añade el nombre de la consulta programada y la hora de invocación a este prefijo para ayudarle a identificar los errores asociados a una instancia específica de una consulta programada.

Etiquetado: si lo desea, puede especificar etiquetas que puede asociar a una consulta programada. Para obtener más información, consulte Etiquetar Timestream for Resources. LiveAnalytics

Ejemplo

En el siguiente ejemplo, se calcula un agregado simple mediante una consulta programada:

```
SELECT region, bin(time, 1m) as minute,
    SUM(CASE WHEN measure_name = 'metrics' THEN 20 ELSE 5 END) as numDataPoints
FROM raw_data.devops
WHERE time BETWEEN @scheduled_runtime - 10m AND @scheduled_runtime + 1m
GROUP BY bin(time, 1m), region
```

`@scheduled_runtime parameter`- En este ejemplo, observará que la consulta acepta un parámetro con un nombre especial`@scheduled_runtime`. Se trata de un parámetro especial (del tipo Timestamp) que el servicio establece al invocar una instancia específica de una consulta programada, de forma que se puede controlar de forma determinista el intervalo de tiempo durante el que una instancia específica de una consulta programada analiza los datos de la tabla de origen. Puede usarlo `@scheduled_runtime` en su consulta en cualquier ubicación en la que se espere un tipo de marca de tiempo.

Considera un ejemplo en el que estableces una expresión de programación: cron (0/5 * * *? *) donde la consulta programada se ejecutará en los minutos 0, 5, 10, 15, 20, 25, 30, 35, 40, 45,

50, 55 de cada hora. Para la instancia que se activa el 01/12/2021 a las 00:05:00, el parámetro @scheduled_runtime se inicializa con este valor, de forma que la instancia funciona en este momento con datos comprendidos en el intervalo del 30 de diciembre de 2021 a las 23:55:00 al 31 de diciembre de 2021 a las 00:06:00.

Instancias con intervalos de tiempo superpuestos: como verá en este ejemplo, dos instancias posteriores de una consulta programada pueden superponerse en sus intervalos de tiempo. Esto es algo que puede controlar en función de sus requisitos, los predicados de tiempo que especifique y la expresión del horario. En este caso, esta superposición permite que estos cálculos actualicen los agregados en función de cualquier dato cuya llegada se haya retrasado ligeramente (hasta 10 minutos en este ejemplo). La ejecución de la consulta iniciada el 01/12/2021 a las 00:00:00 cubrirá el intervalo de tiempo de las 23:50:00 del 2021-11-30 a las 00:01:00 y la ejecución de la consulta que se desencadena el 01/12/2021 a las 00:05:00 cubrirá el intervalo comprendido entre el 2021-11-30 23:55:00 y el 2021-12-01 00:06:00.

Para garantizar la exactitud y asegurarse de que los agregados almacenados en la tabla de destino coinciden con los agregados calculados a partir de la tabla de origen, Timestream for LiveAnalytics garantiza que el cálculo de 2021-12-01 00:05:00 se realice solo después de que se haya completado el cálculo de 2021-12-01 00:00:00. Los resultados de estos últimos cálculos pueden actualizar cualquier agregado previamente materializado si se genera un valor más nuevo. Internamente, Timestream LiveAnalytics utiliza versiones de registro, en las que a los registros generados por las últimas instancias de una consulta programada se les asigna un número de versión superior. Por lo tanto, los agregados calculados por la invocación el 01/12/2021 a las 00:05:00 pueden actualizar los agregados calculados por la invocación el 01/12/2021 a las 00:00:00, suponiendo que haya datos más recientes disponibles en la tabla de origen.

Activadores automáticos frente a activadores manuales: después de crear una consulta programada, Timestream for ejecutará automáticamente las instancias según la programación especificada. LiveAnalytics Estos activadores automáticos son gestionados en su totalidad por el servicio.

Sin embargo, puede haber situaciones en las que desee iniciar manualmente algunas instancias de una consulta programada. Por ejemplo, si una instancia específica falló en la ejecución de una consulta, si los datos llegaron tarde o si hubo actualizaciones en la tabla de origen tras la ejecución programada automática, o si desea actualizar la tabla de destino para intervalos de tiempo que no están cubiertos por las ejecuciones de consultas automatizadas (por ejemplo, para los intervalos de tiempo anteriores a la creación de una consulta programada).

Puede usar el ExecuteScheduledQuery API para iniciar manualmente una instancia específica de una consulta programada pasando el InvocationTime parámetro, que es un valor que se usa para el

parámetro @scheduled_runtime. A continuación se indican algunas consideraciones importantes a la hora de utilizar el ExecuteScheduledQuery API:

- Si va a activar varias de estas invocaciones, debe asegurarse de que estas invocaciones no generen resultados en intervalos de tiempo superpuestos. Si no puede garantizar que los intervalos de tiempo no se superpongan, asegúrese de que estas ejecuciones de consultas se inicien secuencialmente una tras otra. Si inicias simultáneamente varias ejecuciones de consultas que se superponen en sus intervalos de tiempo, puedes ver errores desencadenantes y, por lo tanto, conflictos de versiones en los informes de errores de estas ejecuciones de consultas.
- Puedes iniciar las invocaciones con cualquier valor de marca temporal para @scheduled_runtime. Por lo tanto, es su responsabilidad establecer los valores de manera adecuada para que los intervalos de tiempo adecuados se actualicen en la tabla de destino correspondiente a los rangos en los que se actualizaron los datos en la tabla de origen.

Programe expresiones para consultas programadas

Puede crear consultas programadas de forma automática mediante Amazon Timestream LiveAnalytics para las consultas programadas que utilizan expresiones cron o rate. Todas las consultas programadas utilizan la zona UTC horaria y la precisión mínima posible para las programaciones es de 1 minuto.

Hay dos formas de especificar las expresiones de programación: cron y rate. Las expresiones cron ofrecen un control de programación más detallado, mientras que las expresiones de velocidad son más sencillas de expresar pero carecen de un control detallado.

Por ejemplo, con una expresión cron, puede definir una consulta programada que se active a una hora determinada de un día determinado de cada semana o mes, o un minuto específico cada hora solo de lunes a viernes, y así sucesivamente. Por el contrario, las expresiones de velocidad inician una consulta programada a un ritmo normal, por ejemplo, una vez cada minuto, hora o día, a partir de la hora exacta en que se crea la consulta programada.

Expresión Cron

- Sintaxis

```
cron(fields)
```

Las expresiones Cron tienen seis campos obligatorios, que están separados por un espacio en blanco.

Campo	Valores	Caracteres comodín
Minutos	0-59	, - * /
Horas	0-23	, - * /
D ay-of-month	1-31	, - * ? / L W
Mes	1-12 o - JAN DEC	, - * /
D ay-of-week	1-7 o SUN - SAT	, - * ? L #
Año	1970-2199	, - * /

Caracteres comodín

- El comodín *, * (coma) incluye valores adicionales. En el campo MesJAN,FEB, MAR incluiría enero, febrero y marzo.
- El comodín *-* (guión) especifica los rangos. En el campo Day, 1-15 incluiría los días del 1 al 15 del mes especificado.
- El comodín *** (asterisco) incluye todos los valores del campo. En el campo Horas, *** incluiría todas las horas. No puede utilizar *** en los Day-of-week campos Day-of-month y. Si lo usa en uno, debe usar *? * en el otro.
- El comodín */* (barra inclinada) especifica los incrementos. En el campo Minutos, puede introducir 1/10 para especificar cada 10 minutos, empezando por el primer minuto de la hora (por ejemplo, los minutos 11, 21 y 31, etc.).
- ¿El *? El comodín * (signo de interrogación) especifica uno u otro. En el Day-of-month campo puede escribir *7* y si no le importa qué día de la semana es el 7, ¿puede escribir *? * en el campo. Day-of-week
- El comodín *L* de los Day-of-week campos Day-of-month o especifica el último día del mes o de la semana.
- El comodín W del Day-of-month campo especifica un día de la semana. En el Day-of-month campo, 3W especifica el día de la semana más cercano al tercer día del mes.

- El comodín *#* del Day-of-week campo especifica una instancia determinada del día de la semana especificado dentro de un mes. Por ejemplo, 3#2 sería el segundo martes del mes: el número 3 hace referencia al martes, ya que es el tercer día de la semana en el calendario anglosajón, mientras que 2 hace referencia al segundo día de ese tipo dentro de un mes.

> **Note**
>
> Si utiliza un carácter '#', solo puede definir una expresión en el campo. day-of-week Por ejemplo, “3#1,6#3” no es válido porque se interpreta como dos expresiones.

Limitaciones

- No puede especificar los Day-of-week campos Day-of-month y en la misma expresión cron. Si especificas un valor (o un *) en uno de los campos, ¿debes usar un *? * (signo de interrogación) en el otro.
- No se admiten las expresiones Cron que conducen a frecuencias superiores a 1 minuto.

Ejemplos

Minutos	Horas	Día del mes	Mes	Día de la semana	Año	Significado
0	10	*	*	?	*	Corre a las 10:00 a.m. (UTC) todos los días.
15	12	*	*	?	*	Corre a las 12:15 p.m. (UTC) todos los días.

Minutos	Horas	Día del mes	Mes	Día de la semana	Año	Significado
0	18	?	*	MON-FRI	*	Corre a las 6:00 p.m. (UTC) de lunes a viernes.
0	8	1	*	?	*	Corre a las 8:00 a.m. (UTC) todos los primeros días del mes.
0/15	*	*	*	?	*	Corre cada 15 minutos.
0/10	*	*	*	MON-FRI	*	Corre cada 10 minutos de lunes a viernes.
0/5	8-17	?	*	MON-FRI	*	Corre cada 5 minutos de lunes a viernes entre las 8:00 a.m. y las 5:55 p.m. (UTC).

Expresiones de frecuencia

- Una expresión de frecuencia comienza cuando se crea una regla de evento programado y, a continuación, se ejecuta en su programa definido. Las expresiones de frecuencia tienen dos campos obligatorios. Los campos están separados por un espacio en blanco.

  Sintaxis

  ```
  rate(value unit)
  ```

  - `value`: Un número positivo.
  - `unit`: La unidad de tiempo. Se requieren unidades diferentes para valores de 1 (por ejemplo, minuto) y valores superiores a 1 (por ejemplo, minutos). Valores válidos: minuto | minutos | hora | horas | día | días

Mapeos de modelos de datos para consultas programadas

Timestream for LiveAnalytics admite el modelado flexible de los datos de sus tablas y esta misma flexibilidad se aplica a los resultados de las consultas programadas que se materializan en otro Timestream for table. LiveAnalytics Con las consultas programadas, puede consultar cualquier tabla, ya sea que tenga datos en registros de múltiples medidas o registros de una sola medida, y escribir los resultados de la consulta utilizando registros de medidas múltiples o de una sola medida.

Se utiliza TargetConfiguration en la especificación de una consulta programada para asignar los resultados de la consulta a las columnas correspondientes de la tabla derivada de destino. En las siguientes secciones se describen las distintas formas de especificarlo TargetConfiguration para lograr distintos modelos de datos en la tabla derivada. En concreto, verá:

- Cómo escribir en registros de varias medidas cuando el resultado de la consulta no tiene un nombre de medida y se especifica el nombre de la medida objetivo en el TargetConfiguration.
- Cómo se usa el nombre de la medida en el resultado de la consulta para escribir registros de múltiples medidas.
- Cómo se puede definir un modelo para escribir varios registros con diferentes atributos de múltiples medidas.
- Cómo se puede definir un modelo para escribir en los registros de una sola medida de la tabla derivada.

- Cómo consultar registros de una sola medida o registros de varias medidas en una consulta programada y hacer que los resultados se materialicen en un registro de una sola medida o en un registro de varias medidas, lo que le permite elegir la flexibilidad de los modelos de datos.

Ejemplo: nombre de medida objetivo para registros de múltiples medidas

En este ejemplo, verá que la consulta lee datos de una tabla con datos de varias medidas y escribe los resultados en otra tabla mediante registros de varias medidas. El resultado de la consulta programada no tiene una columna de nombre de medida natural. Aquí, se especifica el nombre de la medida en la tabla derivada mediante la TargetMultiMeasureName propiedad de TargetConfiguration. TimestreamConfiguration.

```
{
    "Name" : "CustomMultiMeasureName",
    "QueryString" : "SELECT region, bin(time, 1h) as hour, AVG(memory_cached)
 as avg_mem_cached_1h, MIN(memory_free) as min_mem_free_1h, MAX(memory_used) as
 max_mem_used_1h, SUM(disk_io_writes) as sum_1h, AVG(disk_used) as avg_disk_used_1h,
 AVG(disk_free) as avg_disk_free_1h, MAX(cpu_user) as max_cpu_user_1h, MIN(cpu_idle) as
 min_cpu_idle_1h, MAX(cpu_system) as max_cpu_system_1h FROM raw_data.devops_multi WHERE
 time BETWEEN bin(@scheduled_runtime, 1h) - 14h AND bin(@scheduled_runtime, 1h) - 2h
 AND measure_name = 'metrics' GROUP BY region, bin(time, 1h)",
    "ScheduleConfiguration" : {
        "ScheduleExpression" : "cron(0 0/1 * * ? *)"
    },
    "NotificationConfiguration" : {
        "SnsConfiguration" : {
            "TopicArn" : "******"
        }
    },
    "ScheduledQueryExecutionRoleArn": "******",
    "TargetConfiguration": {
        "TimestreamConfiguration": {
            "DatabaseName" : "derived",
            "TableName" : "dashboard_metrics_1h_agg_1",
            "TimeColumn" : "hour",
            "DimensionMappings" : [
                {
                    "Name": "region",
                    "DimensionValueType" : "VARCHAR"
                }
            ],
            "MultiMeasureMappings" : {
```

```
                "TargetMultiMeasureName": "dashboard-metrics",
                "MultiMeasureAttributeMappings" : [
                    {
                        "SourceColumn" : "avg_mem_cached_1h",
                        "MeasureValueType" : "DOUBLE",
                        "TargetMultiMeasureAttributeName" : "avgMemCached"
                    },
                    {
                        "SourceColumn" : "min_mem_free_1h",
                        "MeasureValueType" : "DOUBLE"
                    },
                    {
                        "SourceColumn" : "max_mem_used_1h",
                        "MeasureValueType" : "DOUBLE"
                    },
                    {
                        "SourceColumn" : "sum_1h",
                        "MeasureValueType" : "DOUBLE",
                        "TargetMultiMeasureAttributeName" : "totalDiskWrites"
                    },
                    {
                        "SourceColumn" : "avg_disk_used_1h",
                        "MeasureValueType" : "DOUBLE"
                    },
                    {
                        "SourceColumn" : "avg_disk_free_1h",
                        "MeasureValueType" : "DOUBLE"
                    },
                    {
                        "SourceColumn" : "max_cpu_user_1h",
                        "MeasureValueType" : "DOUBLE",
                        "TargetMultiMeasureAttributeName" : "CpuUserP100"
                    },
                    {
                        "SourceColumn" : "min_cpu_idle_1h",
                        "MeasureValueType" : "DOUBLE"
                    },
                    {
                        "SourceColumn" : "max_cpu_system_1h",
                        "MeasureValueType" : "DOUBLE",
                        "TargetMultiMeasureAttributeName" : "CpuSystemP100"
                    }
                ]
            }
```

```
        }
    },
    "ErrorReportConfiguration": {
        "S3Configuration" : {
            "BucketName" : "******",
            "ObjectKeyPrefix": "errors",
            "EncryptionOption": "SSE_S3"
        }
    }
}
```

La asignación de este ejemplo crea un registro de varias medidas con el nombre de la medida dashboard-metrics y los nombres de los atributos: min_mem_free_1h, max_mem_used_1h, avg_disk_used_1h avgMemCached, avg_disk_free_1h, P100, min_cpu_idle_1h, P100. totalDiskWrites CpuUser CpuSystem Observe el uso opcional de TargetMultiMeasureAttributeName cambiar el nombre de las columnas de salida de la consulta por un nombre de atributo diferente utilizado para la materialización de los resultados.

El siguiente es el esquema de la tabla de destino una vez que se materialice la consulta programada. Como puede ver en el flujo temporal del tipo de LiveAnalytics atributo del siguiente resultado, los resultados se materializan en un registro de múltiples medidas con un nombre de medida únicodashboard-metrics, como se muestra en el esquema de medidas.

Columna	Tipo	Secuencia temporal del tipo de atributo LiveAnalytics
región	varchar	DIMENSION
measure_name	varchar	MEASURE_NAME
tiempo	Marca de tiempo	TIMESTAMP
CpuSystemP100	double	MULTI
avgMemCached	double	MULTI
min_cpu_idle_1h	double	MULTI
avg_disk_free_1h	double	MULTI
avg_disk_used_1h	double	MULTI

Columna	Tipo	Secuencia temporal del tipo de atributo LiveAnalytics
totalDiskWrites	double	MULTI
max_mem_used_1h	double	MULTI
min_mem_free_1h	double	MULTI
CpuUserP100	double	MULTI

Las siguientes son las medidas correspondientes obtenidas con una SHOW MEASURES consulta.

measure_name	data_type	Dimensiones
métricas del panel	múltiple	[{'dimension_name': 'región', 'data_type': 'varchar'}]

Ejemplo: usar el nombre de una medida de una consulta programada en registros de varias medidas

En este ejemplo, verá una consulta que lee una tabla con registros de una sola medida y materializa los resultados en registros de varias medidas. En este caso, el resultado de la consulta programada tiene una columna cuyos valores se pueden usar como nombres de medidas en la tabla de destino en la que se materializan los resultados de la consulta programada. A continuación, puede especificar el nombre de la medida para el registro de varias medidas de la tabla derivada mediante la MeasureNameColumn propiedad in. TargetConfiguration TimestreamConfiguration.

```
{
    "Name" : "UsingMeasureNameFromQueryResult",
    "QueryString" : "SELECT region, bin(time, 1h) as hour, measure_name, AVG(CASE WHEN
 measure_name IN ('memory_cached', 'disk_used', 'disk_free') THEN measure_value::double
 ELSE NULL END) as avg_1h, MIN(CASE WHEN measure_name IN ('memory_free', 'cpu_idle')
 THEN measure_value::double ELSE NULL END) as min_1h, SUM(CASE WHEN measure_name
 IN ('disk_io_writes') THEN measure_value::double ELSE NULL END) as sum_1h,
 MAX(CASE WHEN measure_name IN ('memory_used', 'cpu_user', 'cpu_system') THEN
 measure_value::double ELSE NULL END) as max_1h FROM raw_data.devops WHERE time
 BETWEEN bin(@scheduled_runtime, 1h) - 14h AND bin(@scheduled_runtime, 1h) - 2h AND
```

```
measure_name IN ('memory_free', 'memory_used', 'memory_cached', 'disk_io_writes',
'disk_used', 'disk_free', 'cpu_user', 'cpu_system', 'cpu_idle') GROUP BY region,
measure_name, bin(time, 1h)",
   "ScheduleConfiguration" : {
       "ScheduleExpression" : "cron(0 0/1 * * ? *)"
   },
   "NotificationConfiguration" : {
       "SnsConfiguration" : {
           "TopicArn" : "******"
       }
   },
   "ScheduledQueryExecutionRoleArn": "******",
   "TargetConfiguration": {
       "TimestreamConfiguration": {
           "DatabaseName" : "derived",
           "TableName" : "dashboard_metrics_1h_agg_2",
           "TimeColumn" : "hour",
           "DimensionMappings" : [
               {
                   "Name": "region",
                   "DimensionValueType" : "VARCHAR"
               }
           ],
           "MeasureNameColumn" : "measure_name",
           "MultiMeasureMappings" : {
               "MultiMeasureAttributeMappings" : [
                   {
                       "SourceColumn" : "avg_1h",
                       "MeasureValueType" : "DOUBLE"
                   },
                   {
                       "SourceColumn" : "min_1h",
                       "MeasureValueType" : "DOUBLE",
                       "TargetMultiMeasureAttributeName": "p0_1h"
                   },
                   {
                       "SourceColumn" : "sum_1h",
                       "MeasureValueType" : "DOUBLE"
                   },
                   {
                       "SourceColumn" : "max_1h",
                       "MeasureValueType" : "DOUBLE",
                       "TargetMultiMeasureAttributeName": "p100_1h"
                   }
```

```
                ]
            }
        }
    },
    "ErrorReportConfiguration": {
        "S3Configuration" : {
            "BucketName" : "******",
            "ObjectKeyPrefix": "errors",
            "EncryptionOption": "SSE_S3"
        }
    }
}
```

La asignación de este ejemplo creará registros de múltiples medidas con los atributos avg_1h, p0_1h, sum_1h, p100_1h y utilizará los valores de la columna measure_name del resultado de la consulta como nombre de medida para los registros de múltiples medidas de la tabla de destino. Además, tenga en cuenta que los ejemplos anteriores utilizan opcionalmente el subconjunto with de las asignaciones para cambiar el nombre de los atributos. TargetMultiMeasureAttributeName Por ejemplo, se cambió el nombre de min_1h a p0_1h y el de max_1h a p100_1h.

El siguiente es el esquema de la tabla de destino una vez que se materialice la consulta programada. Como puede ver en el flujo temporal del tipo de LiveAnalytics atributo del siguiente resultado, los resultados se materializan en un registro de múltiples medidas. Si observa el esquema de medidas, se ingirieron nueve nombres de medidas diferentes que corresponden a los valores que aparecen en los resultados de la consulta.

Columna	Tipo	Secuencia temporal del tipo de atributo LiveAnalytics
región	varchar	DIMENSION
measure_name	varchar	MEASURE_NAME
tiempo	Marca de tiempo	TIMESTAMP
sum_1h	double	MULTI
p100_1h	double	MULTI
p0_1h	double	MULTI

Columna	Tipo	Secuencia temporal del tipo de atributo LiveAnalytics
avg_1h	double	MULTI

Las siguientes son las medidas correspondientes obtenidas con una consulta. SHOW MEASURES

measure_name	data_type	Dimensiones
cpu_idle	múltiple	[{'dimension_name': 'región', 'data_type': 'varchar'}]
sistema_CPU	múltiple	[{'dimension_name': 'región', 'data_type': 'varchar'}]
cpu_usuario	múltiple	[{'dimension_name': 'región', 'data_type': 'varchar'}]
disk_free	múltiple	[{'dimension_name': 'región', 'data_type': 'varchar'}]
disk_io_writes	multi	[{'dimension_name': 'región', 'data_type': 'varchar'}]
disk_used	múltiple	[{'dimension_name': 'región', 'data_type': 'varchar'}]
memoria_caché	múltiple	[{'dimension_name': 'región', 'data_type': 'varchar'}]
libre de memoria	múltiple	[{'dimension_name': 'región', 'data_type': 'varchar'}]
libre de memoria	múltiple	[{'dimension_name': 'región', 'data_type': 'varchar'}]

Ejemplo: mapear los resultados a diferentes registros de múltiples medidas con diferentes atributos

El siguiente ejemplo muestra cómo puede asignar diferentes columnas del resultado de la consulta a diferentes registros de múltiples medidas con diferentes nombres de medidas. Si ve la siguiente definición de consulta programada, el resultado de la consulta tiene las siguientes columnas: region, hour, avg_mem_cached_1h, min_mem_free_1h, max_mem_used_1h, total_disk_io_writes_1h, avg_disk_used_1h, avg_disk_free_1h, max_cpu_user_1h h, max_cpu_system_1h, min_cpu_system_1h. `region`onse asigna a la dimensión y `hour` se asigna a la columna de tiempo.

La MixedMeasureMappings propiedad en. TargetConfiguration TimestreamConfigurationespecifica cómo asignar las medidas a los registros de múltiples medidas de la tabla derivada.

En este ejemplo específico, avg_mem_cached_1h, min_mem_free_1h, max_mem_used_1h se utilizan en un registro de múltiples medidas con el nombre de medida de mem_aggregates, total_disk_io_writes_1h, avg_disk_used_1h, avg_disk_free_1h se utilizan en otro registro de medidas múltiples con el nombre de medida de disquete k_aggregates y, finalmente, max_cpu_user_1h, max_cpu_system_1h, min_cpu_system_1h se utilizan en otro registro de medidas múltiples con el nombre de medida cpu_aggregates.

En estas asignaciones, también se puede utilizar opcionalmente TargetMultiMeasureAttributeName para cambiar el nombre de la columna de resultados de la consulta para que tenga un nombre de atributo diferente en la tabla de destino. Por ejemplo, la columna de resultados avg_mem_cached_1h pasa a llamarse, total_disk_io_writes_1h pasa a llamarse, etc. avgMemCached totalIOWrites

Al definir las asignaciones para los registros de varias medidas, Timestream for inspecciona todas las filas de los resultados de la consulta e ignora automáticamente los valores de las columnas que tienen valores. LiveAnalytics NULL Como resultado, en el caso de las asignaciones con varios nombres de medidas, si todos los valores de columna de ese grupo del mapeo son para una fila determinada, no se incorporará ningún valor NULL para el nombre de esa medida en esa fila.

Por ejemplo, en la siguiente asignación, avg_mem_cached_1h, min_mem_free_1h y max_mem_used_1h se mapean para medir el nombre mem_aggregates. Si para una fila determinada del resultado de la consulta, todos estos valores de columna son iguales, Timestream for no absorberá la medida mem_aggregates de esa fila. NULL LiveAnalytics Si las nueve columnas de una fila determinada lo sonNULL, aparecerá un error de usuario en el informe de errores.

```
{
```

```
   "Name" : "AggsInDifferentMultiMeasureRecords",
   "QueryString" : "SELECT region, bin(time, 1h) as hour, AVG(CASE WHEN measure_name
= 'memory_cached' THEN measure_value::double ELSE NULL END) as avg_mem_cached_1h,
MIN(CASE WHEN measure_name = 'memory_free' THEN measure_value::double ELSE
NULL END) as min_mem_free_1h, MAX(CASE WHEN measure_name = 'memory_used' THEN
measure_value::double ELSE NULL END) as max_mem_used_1h, SUM(CASE WHEN measure_name =
'disk_io_writes' THEN measure_value::double ELSE NULL END) as total_disk_io_writes_1h,
AVG(CASE WHEN measure_name = 'disk_used' THEN measure_value::double ELSE NULL END) as
avg_disk_used_1h, AVG(CASE WHEN measure_name = 'disk_free' THEN measure_value::double
ELSE NULL END) as avg_disk_free_1h, MAX(CASE WHEN measure_name = 'cpu_user' THEN
measure_value::double ELSE NULL END) as max_cpu_user_1h, MAX(CASE WHEN measure_name
= 'cpu_system' THEN measure_value::double ELSE NULL END) as max_cpu_system_1h,
MIN(CASE WHEN measure_name = 'cpu_idle' THEN measure_value::double ELSE NULL END)
as min_cpu_system_1h FROM raw_data.devops WHERE time BETWEEN bin(@scheduled_runtime,
1h) - 14h AND bin(@scheduled_runtime, 1h) - 2h AND measure_name IN ('memory_cached',
'memory_free', 'memory_used', 'disk_io_writes', 'disk_used', 'disk_free', 'cpu_user',
'cpu_system', 'cpu_idle') GROUP BY region, bin(time, 1h)",
   "ScheduleConfiguration" : {
       "ScheduleExpression" : "cron(0 0/1 * * ? *)"
   },
   "NotificationConfiguration" : {
       "SnsConfiguration" : {
           "TopicArn" : "******"
       }
   },
   "ScheduledQueryExecutionRoleArn": "******",
   "TargetConfiguration": {
       "TimestreamConfiguration": {
           "DatabaseName" : "derived",
           "TableName" : "dashboard_metrics_1h_agg_3",
           "TimeColumn" : "hour",
           "DimensionMappings" : [
               {
                   "Name": "region",
                   "DimensionValueType" : "VARCHAR"
               }
           ],
           "MixedMeasureMappings" : [
               {
                   "MeasureValueType" : "MULTI",
                   "TargetMeasureName" : "mem_aggregates",
                   "MultiMeasureAttributeMappings" : [
                       {
                           "SourceColumn" : "avg_mem_cached_1h",
```

```
                        "MeasureValueType" : "DOUBLE",
                        "TargetMultiMeasureAttributeName": "avgMemCached"
                    },
                    {
                        "SourceColumn" : "min_mem_free_1h",
                        "MeasureValueType" : "DOUBLE"
                    },
                    {
                        "SourceColumn" : "max_mem_used_1h",
                        "MeasureValueType" : "DOUBLE",
                        "TargetMultiMeasureAttributeName": "maxMemUsed"
                    }
                ]
            },
            {
                "MeasureValueType" : "MULTI",
                "TargetMeasureName" : "disk_aggregates",
                "MultiMeasureAttributeMappings" : [
                    {
                        "SourceColumn" : "total_disk_io_writes_1h",
                        "MeasureValueType" : "DOUBLE",
                        "TargetMultiMeasureAttributeName": "totalIOWrites"
                    },
                    {
                        "SourceColumn" : "avg_disk_used_1h",
                        "MeasureValueType" : "DOUBLE"
                    },
                    {
                        "SourceColumn" : "avg_disk_free_1h",
                        "MeasureValueType" : "DOUBLE"
                    }
                ]
            },
            {
                "MeasureValueType" : "MULTI",
                "TargetMeasureName" : "cpu_aggregates",
                "MultiMeasureAttributeMappings" : [
                    {
                        "SourceColumn" : "max_cpu_user_1h",
                        "MeasureValueType" : "DOUBLE"
                    },
                    {
                        "SourceColumn" : "max_cpu_system_1h",
                        "MeasureValueType" : "DOUBLE"
```

```
                        },
                        {
                            "SourceColumn" : "min_cpu_idle_1h",
                            "MeasureValueType" : "DOUBLE",
                            "TargetMultiMeasureAttributeName": "minCpuIdle"
                        }
                    ]
                }
            ]
        }
    },
    "ErrorReportConfiguration": {
        "S3Configuration" : {
            "BucketName" : "******",
            "ObjectKeyPrefix": "errors",
            "EncryptionOption": "SSE_S3"
        }
    }
}
```

El siguiente es el esquema de la tabla de destino una vez que se materialice la consulta programada.

Columna	Tipo	Secuencia temporal del tipo de atributo LiveAnalytics
región	varchar	DIMENSION
measure_name	varchar	MEASURE_NAME
tiempo	Marca de tiempo	TIMESTAMP
minCpuIdle	double	MULTI
max_cpu_system_1h	double	MULTI
max_cpu_user_1h	double	MULTI
avgMemCached	double	MULTI
maxMemUsed	double	MULTI
min_mem_free_1h	double	MULTI

Columna	Tipo	Secuencia temporal del tipo de atributo LiveAnalytics
avg_disk_free_1h	double	MULTI
avg_disk_used_1h	double	MULTI
totalIOWrites	double	MULTI

Las siguientes son las medidas correspondientes obtenidas con una consulta. SHOW MEASURES

measure_name	data_type	Dimensiones
cpu_aggregates	múltiple	[{'dimension_name': 'región', 'data_type': 'varchar'}]
disk_aggregates	múltiple	[{'dimension_name': 'región', 'data_type': 'varchar'}]
mem_aggregates	múltiple	[{'dimension_name': 'región', 'data_type': 'varchar'}]

Ejemplo: mapear los resultados a registros de una sola medida con el nombre de la medida a partir de los resultados de la consulta

El siguiente es un ejemplo de una consulta programada cuyos resultados se materializan en registros de una sola medida. En este ejemplo, el resultado de la consulta tiene la columna measure_name cuyos valores se utilizarán como nombres de medidas en la tabla de destino. El MixedMeasureMappings atributo se utiliza en. TargetConfiguration TimestreamConfiguration para especificar la asignación de la columna de resultados de la consulta a la medida escalar de la tabla de destino.

En la siguiente definición de ejemplo, se espera que el resultado de la consulta tenga nueve valores de measure_name distintos. Enumera todos estos nombres de medida en el mapeo y especifica qué columna usar como valor de medida única para ese nombre de medida. Por ejemplo, en este mapeo, si se ve el nombre de medida memory_cached para una fila de resultados determinada, el valor de la columna avg_1h se usa como valor de la medida cuando los datos se escriben en la tabla de destino.

Si lo desea, puede usarlo TargetMeasureName para proporcionar un nuevo nombre de medida para este valor.

```
{
    "Name" : "UsingMeasureNameColumnForSingleMeasureMapping",
    "QueryString" : "SELECT region, bin(time, 1h) as hour, measure_name, AVG(CASE WHEN
 measure_name IN ('memory_cached', 'disk_used', 'disk_free') THEN measure_value::double
 ELSE NULL END) as avg_1h, MIN(CASE WHEN measure_name IN ('memory_free', 'cpu_idle')
 THEN measure_value::double ELSE NULL END) as min_1h, SUM(CASE WHEN measure_name
 IN ('disk_io_writes') THEN measure_value::double ELSE NULL END) as sum_1h,
 MAX(CASE WHEN measure_name IN ('memory_used', 'cpu_user', 'cpu_system') THEN
 measure_value::double ELSE NULL END) as max_1h FROM raw_data.devops WHERE time
 BETWEEN bin(@scheduled_runtime, 1h) - 14h AND bin(@scheduled_runtime, 1h) - 2h AND
 measure_name IN ('memory_free', 'memory_used', 'memory_cached', 'disk_io_writes',
 'disk_used', 'disk_free', 'cpu_user', 'cpu_system', 'cpu_idle') GROUP BY region,
 bin(time, 1h), measure_name",
    "ScheduleConfiguration" : {
        "ScheduleExpression" : "cron(0 0/1 * * ? *)"
    },
    "NotificationConfiguration" : {
        "SnsConfiguration" : {
            "TopicArn" : "******"
        }
    },
    "ScheduledQueryExecutionRoleArn": "******",
    "TargetConfiguration": {
        "TimestreamConfiguration": {
            "DatabaseName" : "derived",
            "TableName" : "dashboard_metrics_1h_agg_4",
            "TimeColumn" : "hour",
            "DimensionMappings" : [
                {
                    "Name": "region",
                    "DimensionValueType" : "VARCHAR"
                }
            ],
            "MeasureNameColumn" : "measure_name",
            "MixedMeasureMappings" : [
                {
                    "MeasureName" : "memory_cached",
                    "MeasureValueType" : "DOUBLE",
                    "SourceColumn" : "avg_1h",
                    "TargetMeasureName" : "AvgMemCached"
```

```
                },
                {
                    "MeasureName" : "disk_used",
                    "MeasureValueType" : "DOUBLE",
                    "SourceColumn" : "avg_1h"
                },
                {
                    "MeasureName" : "disk_free",
                    "MeasureValueType" : "DOUBLE",
                    "SourceColumn" : "avg_1h"
                },
                {
                    "MeasureName" : "memory_free",
                    "MeasureValueType" : "DOUBLE",
                    "SourceColumn" : "min_1h",
                    "TargetMeasureName" : "MinMemFree"
                },
                {
                    "MeasureName" : "cpu_idle",
                    "MeasureValueType" : "DOUBLE",
                    "SourceColumn" : "min_1h"
                },
                {
                    "MeasureName" : "disk_io_writes",
                    "MeasureValueType" : "DOUBLE",
                    "SourceColumn" : "sum_1h",
                    "TargetMeasureName" : "total-disk-io-writes"
                },
                {
                    "MeasureName" : "memory_used",
                    "MeasureValueType" : "DOUBLE",
                    "SourceColumn" : "max_1h",
                    "TargetMeasureName" : "maxMemUsed"
                },
                {
                    "MeasureName" : "cpu_user",
                    "MeasureValueType" : "DOUBLE",
                    "SourceColumn" : "max_1h"
                },
                {
                    "MeasureName" : "cpu_system",
                    "MeasureValueType" : "DOUBLE",
                    "SourceColumn" : "max_1h"
                }
```

```
            ]
        }
    },
    "ErrorReportConfiguration": {
        "S3Configuration" : {
            "BucketName" : "******",
            "ObjectKeyPrefix": "errors",
            "EncryptionOption": "SSE_S3"
        }
    }
}
```

El siguiente es el esquema de la tabla de destino una vez que se materialice la consulta programada. Como puede ver en el esquema, la tabla utiliza registros de una sola medida. Si incluye el esquema de medidas de la tabla, verá las nueve medidas escritas en función del mapeo proporcionado en la especificación.

Columna	Tipo	Secuencia temporal del tipo de atributo LiveAnalytics
región	varchar	DIMENSION
measure_name	varchar	MEASURE_NAME
tiempo	Marca de tiempo	TIMESTAMP
measure_value::double	double	MEASURE_VALUE

Las siguientes son las medidas correspondientes obtenidas con una SHOW MEASURES consulta.

measure_name	data_type	Dimensiones
AvgMemCached	double	[{'dimension_name': 'region', 'data_type': 'varchar'}]
MinMemFree	double	[{'dimension_name': 'región', 'data_type': 'varchar'}]

measure_name	data_type	Dimensiones
cpu_idle	double	[{'dimension_name': 'región', 'data_type': 'varchar'}]
sistema_CPU	double	[{'dimension_name': 'región', 'data_type': 'varchar'}]
cpu_usuario	double	[{'dimension_name': 'región', 'data_type': 'varchar'}]
disk_free	double	[{'dimension_name': 'región', 'data_type': 'varchar'}]
disk_used	double	[{'dimension_name': 'región', 'data_type': 'varchar'}]
maxMemUsed	double	[{'dimension_name': 'región', 'data_type': 'varchar'}]
total-disk-io-writes	double	[{'dimension_name': 'región', 'data_type': 'varchar'}]

Ejemplo: mapear los resultados a registros de una sola medida con columnas de resultados de consultas como nombres de medidas

En este ejemplo, tiene una consulta cuyos resultados no tienen una columna con el nombre de la medida. En su lugar, querrá que el nombre de la columna de resultados de la consulta sea el nombre de la medida al mapear la salida a registros de una sola medida. Anteriormente, había un ejemplo en el que se escribía un resultado similar en un registro de múltiples medidas. En este ejemplo, verá cómo asignarlo a registros de una sola medida si se ajusta al escenario de su aplicación.

De nuevo, este mapeo se especifica mediante la MixedMeasureMappings propiedad in TargetConfiguration. TimestreamConfiguration. En el siguiente ejemplo, verá que el resultado de la consulta tiene nueve columnas. Las columnas de resultados se utilizan como nombres de medida y los valores como valores de medida única.

Por ejemplo, para una fila determinada del resultado de la consulta, el nombre de columna avg_mem_cached_1h se usa como nombre y valor de la columna asociados a la columna, y

avg_mem_cached_1h se usa como valor de medida para el registro de medida única. También puede utilizar TargetMeasureName un nombre de medida diferente en la tabla de destino. Por ejemplo, para los valores de la columna sum_1h, el mapeo especifica usar total_disk_io_writes_1h como nombre de medida en la tabla de destino. Si el valor de alguna columna es, se ignora la medida correspondiente. NULL

```
{
    "Name" : "SingleMeasureMappingWithoutMeasureNameColumnInQueryResult",
    "QueryString" : "SELECT region, bin(time, 1h) as hour, AVG(CASE WHEN measure_name
 = 'memory_cached' THEN measure_value::double ELSE NULL END) as avg_mem_cached_1h,
 AVG(CASE WHEN measure_name = 'disk_used' THEN measure_value::double ELSE NULL END) as
 avg_disk_used_1h, AVG(CASE WHEN measure_name = 'disk_free' THEN measure_value::double
 ELSE NULL END) as avg_disk_free_1h, MIN(CASE WHEN measure_name = 'memory_free' THEN
 measure_value::double ELSE NULL END) as min_mem_free_1h, MIN(CASE WHEN measure_name =
 'cpu_idle' THEN measure_value::double ELSE NULL END) as min_cpu_idle_1h, SUM(CASE WHEN
 measure_name = 'disk_io_writes' THEN measure_value::double ELSE NULL END) as sum_1h,
 MAX(CASE WHEN measure_name = 'memory_used' THEN measure_value::double ELSE NULL END)
 as max_mem_used_1h, MAX(CASE WHEN measure_name = 'cpu_user' THEN measure_value::double
 ELSE NULL END) as max_cpu_user_1h, MAX(CASE WHEN measure_name = 'cpu_system' THEN
 measure_value::double ELSE NULL END) as max_cpu_system_1h FROM raw_data.devops WHERE
 time BETWEEN bin(@scheduled_runtime, 1h) - 14h AND bin(@scheduled_runtime, 1h) - 2h
 AND measure_name IN ('memory_free', 'memory_used', 'memory_cached', 'disk_io_writes',
 'disk_used', 'disk_free', 'cpu_user', 'cpu_system', 'cpu_idle') GROUP BY region,
 bin(time, 1h)",
    "ScheduleConfiguration" : {
        "ScheduleExpression" : "cron(0 0/1 * * ? *)"
    },
    "NotificationConfiguration" : {
        "SnsConfiguration" : {
            "TopicArn" : "******"
        }
    },
    "ScheduledQueryExecutionRoleArn": "******",
    "TargetConfiguration": {
        "TimestreamConfiguration": {
            "DatabaseName" : "derived",
            "TableName" : "dashboard_metrics_1h_agg_5",
            "TimeColumn" : "hour",
            "DimensionMappings" : [
                {
                    "Name": "region",
                    "DimensionValueType" : "VARCHAR"
                }
```

```
            ],
            "MixedMeasureMappings" : [
                {
                    "MeasureValueType" : "DOUBLE",
                    "SourceColumn" : "avg_mem_cached_1h"
                },
                {
                    "MeasureValueType" : "DOUBLE",
                    "SourceColumn" : "avg_disk_used_1h"
                },
                {
                    "MeasureValueType" : "DOUBLE",
                    "SourceColumn" : "avg_disk_free_1h"
                },
                {
                    "MeasureValueType" : "DOUBLE",
                    "SourceColumn" : "min_mem_free_1h"
                },
                {
                    "MeasureValueType" : "DOUBLE",
                    "SourceColumn" : "min_cpu_idle_1h"
                },
                {
                    "MeasureValueType" : "DOUBLE",
                    "SourceColumn" : "sum_1h",
                    "TargetMeasureName" : "total_disk_io_writes_1h"
                },
                {
                    "MeasureValueType" : "DOUBLE",
                    "SourceColumn" : "max_mem_used_1h"
                },
                {
                    "MeasureValueType" : "DOUBLE",
                    "SourceColumn" : "max_cpu_user_1h"
                },
                {
                    "MeasureValueType" : "DOUBLE",
                    "SourceColumn" : "max_cpu_system_1h"
                }
            ]
        }
    },
    "ErrorReportConfiguration": {
        "S3Configuration" : {
```

```
            "BucketName" : "******",
            "ObjectKeyPrefix": "errors",
            "EncryptionOption": "SSE_S3"
        }
    }
}
```

El siguiente es el esquema de la tabla de destino una vez que se materialice la consulta programada. Como puede ver, la tabla de destino almacena registros con valores de medida única de tipo double. Del mismo modo, el esquema de medidas de la tabla muestra los nombres de las nueve medidas. Observe también que el nombre de medida total_disk_io_writes_1h está presente ya que la asignación cambió el nombre de sum_1h a total_disk_io_writes_1h.

Columna	Tipo	LiveAnalytics Secuencia temporal del tipo de atributo
región	varchar	DIMENSION
measure_name	varchar	MEASURE_NAME
tiempo	Marca de tiempo	TIMESTAMP
measure_value::double	double	MEASURE_VALUE

Las siguientes son las medidas correspondientes obtenidas con una SHOW MEASURES consulta.

measure_name	data_type	Dimensiones
avg_disk_free_1h	double	[{'dimension_name': 'región', 'data_type': 'varchar'}]
avg_disk_used_1h	double	[{'dimension_name': 'región', 'data_type': 'varchar'}]
avg_mem_cached_1h	double	[{'dimension_name': 'región', 'data_type': 'varchar'}]
max_cpu_system_1h	double	[{'dimension_name': 'región', 'data_type': 'varchar'}]

measure_name	data_type	Dimensiones
max_cpu_user_1h	double	[{'dimension_name': 'región', 'data_type': 'varchar'}]
max_mem_used_1h	double	[{'dimension_name': 'región', 'data_type': 'varchar'}]
min_cpu_idle_1h	double	[{'dimension_name': 'región', 'data_type': 'varchar'}]
min_mem_free_1h	double	[{'dimension_name': 'región', 'data_type': 'varchar'}]
total-disk-io-writes	double	[{'dimension_name': 'región', 'data_type': 'varchar'}]

Mensajes de notificación de consultas programadas

En esta sección se describen los mensajes que envía Timestream LiveAnalytics al crear, eliminar, ejecutar o actualizar el estado de una consulta programada.

Nombre del mensaje de notificación	Estructura	Descripción
CreatingNotificationMessage	`CreatingNotificationMessage { String arn; NotificationType type; }`	Este mensaje de notificación se envía antes de enviar la respuesta a `CreateScheduledQuery`. La consulta programada se habilita después de enviar esta notificación. arn: el ARN de la consulta programada que se está creando.

Nombre del mensaje de notificación	Estructura	Descripción
		tipo - SCHEDULED _ _ QUERY CREATING

Nombre del mensaje de notificación	Estructura	Descripción
UpdateNotificationMessage	`UpdateNotification Message { String arn; NotificationType type; QueryState state; }`	Este mensaje de notificación se envía cuando se actualiza una consulta programada. Timestream for LiveAnalytics puede deshabilitar la consulta programada automáticamente en caso de que se produzca un error irrecuperable, como: • AssumeRole error • Cualquier error 4xx que se produzca al comunicar se con el KMS momento en que se especifica una KMS clave gestionada por el cliente. • Cualquier error 4xx que se haya producido durante la ejecución de la consulta programada. • Cualquier error 4xx que se haya producido durante la ingesta de los resultados de la consulta arn: el ARN de la consulta programada que se está actualizando. tipo - SCHEDULED _ _ QUERY UPDATE

Nombre del mensaje de notificación	Estructura	Descripción
		estado - ENABLED o DISABLED
DeleteNotificationMessage	`DeletionNotificationMessage { String arn; NotificationType type; }`	Este mensaje de notificación se envía cuando se elimina una consulta programada. arn: el ARN de la consulta programada que se está creando. tipo - SCHEDULED _ _ QUERY DELETED

Nombre del mensaje de notificación	Estructura	Descripción

SuccessNotificationMessage	`SuccessNotificationMessage { NotificationType type; String arn; Date nextInvocationEpochSecond; ScheduledQueryRunSummary runSummary; }`
	
	`ScheduledQueryRunSummary { Date invocationTime; Date triggerTime; String runStatus; ExecutionStats executionstats; ErrorReportLocation errorReportLocation; String failureReason; }`
	
	`ExecutionStats { Long bytesMetered; Long dataWrites; Long queryResultRows; Long recordsIngested; Long executionTimeInMillis; }`
	
	`ErrorReportLocation {`
		Este mensaje de notificación se envía una vez ejecutada la consulta programada y los resultados se han ingerido correctamente.
		
		ARN- El ARN de la consulta programada que se va a eliminar.
		
		NotificationType- AUTO _ TRIGGER _ SUCCESS o MANUAL _ TRIGGER _SUCCESS.
		
		nextInvocationEpochSegundo: la próxima vez que Timestream for LiveAnalytics ejecute la consulta programada.
		
		runSummary- Información sobre la ejecución programada de la consulta.

Nombre del mensaje de notificación	Estructura	Descripción

	S3ReportLocation s3ReportLocation;
	}
	
	S3ReportLocation {
	String bucketName;
	String objectKey;
	}	

Nombre del mensaje de notificación	Estructura	Descripción

FailureNotificationMessage	`FailureNotificationMessage { NotificationType type; String arn; ScheduledQueryRunSummary runSummary; }`
	
	`ScheduledQueryRunSummary { Date invocationTime; Date triggerTime; String runStatus; ExecutionStats executionstats; ErrorReportLocation errorReportLocation; String failureReason; }`
	
	`ExecutionStats { Long bytesMetered; Long dataWrites; Long queryResultRows; Long recordsIngested; Long executionTimeInMillis; }`
	
	`ErrorReportLocation { S3ReportLocation s3ReportLocation;`
		Este mensaje de notificación se envía cuando se detecta un error durante la ejecución de una consulta programada o al ingerir los resultados de la consulta.
		
		arn: el ARN de la consulta programada que se está ejecutando.
		
		escriba - AUTO _ TRIGGER _ FAILURE o MANUAL _ TRIGGER _FAILURE.
		
		runSummary- Información sobre la ejecución de la consulta programada.

Nombre del mensaje de notificación	Estructura	Descripción

	`}`
	
	`S3ReportLocation {`
	`  String bucketName;`
	`  String objectKey;`
	`}`	

Informes de errores de consultas programadas

En esta sección se describen la ubicación, el formato y los motivos de los informes de errores que genera Timestream para LiveAnalytics cuando se detectan errores al ejecutar consultas programadas.

Temas

- Los motivos de los informes de errores de consultas programadas
- El error de consulta programado informa de la ubicación
- Formato de informes de errores de consulta programada
- Tipos de errores de consultas programadas
- Ejemplo de informes de errores de consultas programadas

Los motivos de los informes de errores de consultas programadas

Los informes de errores se generan para los errores recuperables. Los informes de errores no se generan para los errores no recuperables. Timestream for LiveAnalytics puede deshabilitar automáticamente las consultas programadas cuando se encuentran errores no recuperables. Entre ellos se incluyen:

- `AssumeRole`error
- ¿Se ha producido un error de cuatro veces al comunicarse con KMS cuando se especifica una clave gestionada por el cliente KMS
- Cualquier error 4xx que se haya producido al ejecutar una consulta programada
- Cualquier error 4xx que se haya producido durante la ingesta de los resultados de la consulta

En el caso de los errores no recuperables, Timestream for LiveAnalytics envía una notificación de error con un mensaje de error no recuperable. También se envía una notificación de actualización que indica que la consulta programada está deshabilitada.

El error de consulta programado informa de la ubicación

La ubicación de un informe de errores de consulta programada tiene la siguiente convención de nomenclatura:

```
s3://customer-bucket/customer-prefix/
```

A continuación se muestra un ejemplo de consulta programadaARN:

```
arn:aws:timestream:us-east-1:000000000000:scheduled-query/test-query-hd734tegrgfd
```

```
s3://customer-bucket/customer-prefix/test-query-hd734tegrgfd/<InvocationTime>/<Auto or
 Manual>/<Actual Trigger Time>
```

`Auto` indica las consultas programadas automáticamente por Timestream para y LiveAnalytics *`Manual`* indica las consultas programadas activadas manualmente por un usuario mediante una `ExecuteScheduledQuery` API acción en Amazon Timestream LiveAnalytics for Query. Para obtener más información al respecto`ExecuteScheduledQuery`, consulte. ExecuteScheduledQuery

Formato de informes de errores de consulta programada

Los informes de errores tienen el siguiente JSON formato:

```
{
    "reportId": <String>,            // A unique string ID for all error reports
 belonging to a particular scheduled query run
    "errors": [ <Error>, ... ],      // One or more errors
}
```

Tipos de errores de consultas programadas

El `Error` objeto puede ser de tres tipos:

- Registra los errores de ingestión

```
{
```

```
    "reason": <String>,              // The error message String
    "records": [ <Record>, ... ],    // One or more rejected records )
}
```

- Errores de análisis y validación de filas

```
{
    "reason": <String>,        // The error message String
    "rawLine": <String>,       // [Optional] The raw line String that is being parsed
 into record(s) to be ingested. This line has encountered the above-mentioned parse
 error.
}
```

- Errores generales

```
{
    "reason": <String>,        // The error message
}
```

Ejemplo de informes de errores de consultas programadas

El siguiente es un ejemplo de un informe de errores que se produjo debido a errores de ingesta.

```
{
    "reportId": "C9494AABE012D1FBC162A67EA2C18255",
    "errors": [
        {
            "reason": "The record timestamp is outside the time range
 [2021-11-12T14:18:13.354Z, 2021-11-12T16:58:13.354Z) of the memory store.",
            "records": [
                {
                    "dimensions": [
                        {
                            "name": "dim0",
                            "value": "d0_1",
                            "dimensionValueType": null
                        },
                        {
                            "name": "dim1",
                            "value": "d1_1",
                            "dimensionValueType": null
                        }
```

```
                    ],
                    "measureName": "random_measure_value",
                    "measureValue": "3.141592653589793",
                    "measureValues": null,
                    "measureValueType": "DOUBLE",
                    "time": "1637166175635000000",
                    "timeUnit": "NANOSECONDS",
                    "version": null
                },
                {
                    "dimensions": [
                        {
                            "name": "dim0",
                            "value": "d0_2",
                            "dimensionValueType": null
                        },
                        {
                            "name": "dim1",
                            "value": "d1_2",
                            "dimensionValueType": null
                        }
                    ],
                    "measureName": "random_measure_value",
                    "measureValue": "6.283185307179586",
                    "measureValues": null,
                    "measureValueType": "DOUBLE",
                    "time": "1637166175636000000",
                    "timeUnit": "NANOSECONDS",
                    "version": null
                },
                {
                    "dimensions": [
                        {
                            "name": "dim0",
                            "value": "d0_3",
                            "dimensionValueType": null
                        },
                        {
                            "name": "dim1",
                            "value": "d1_3",
                            "dimensionValueType": null
                        }
                    ],
                    "measureName": "random_measure_value",
```

```
                    "measureValue": "9.42477796076938",
                    "measureValues": null,
                    "measureValueType": "DOUBLE",
                    "time": "1637166175637000000",
                    "timeUnit": "NANOSECONDS",
                    "version": null
                },
                {
                    "dimensions": [
                        {
                            "name": "dim0",
                            "value": "d0_4",
                            "dimensionValueType": null
                        },
                        {
                            "name": "dim1",
                            "value": "d1_4",
                            "dimensionValueType": null
                        }
                    ],
                    "measureName": "random_measure_value",
                    "measureValue": "12.566370614359172",
                    "measureValues": null,
                    "measureValueType": "DOUBLE",
                    "time": "1637166175638000000",
                    "timeUnit": "NANOSECONDS",
                    "version": null
                }
            ]
        }
    ]
}
```


Ejemplos y patrones de consultas programadas

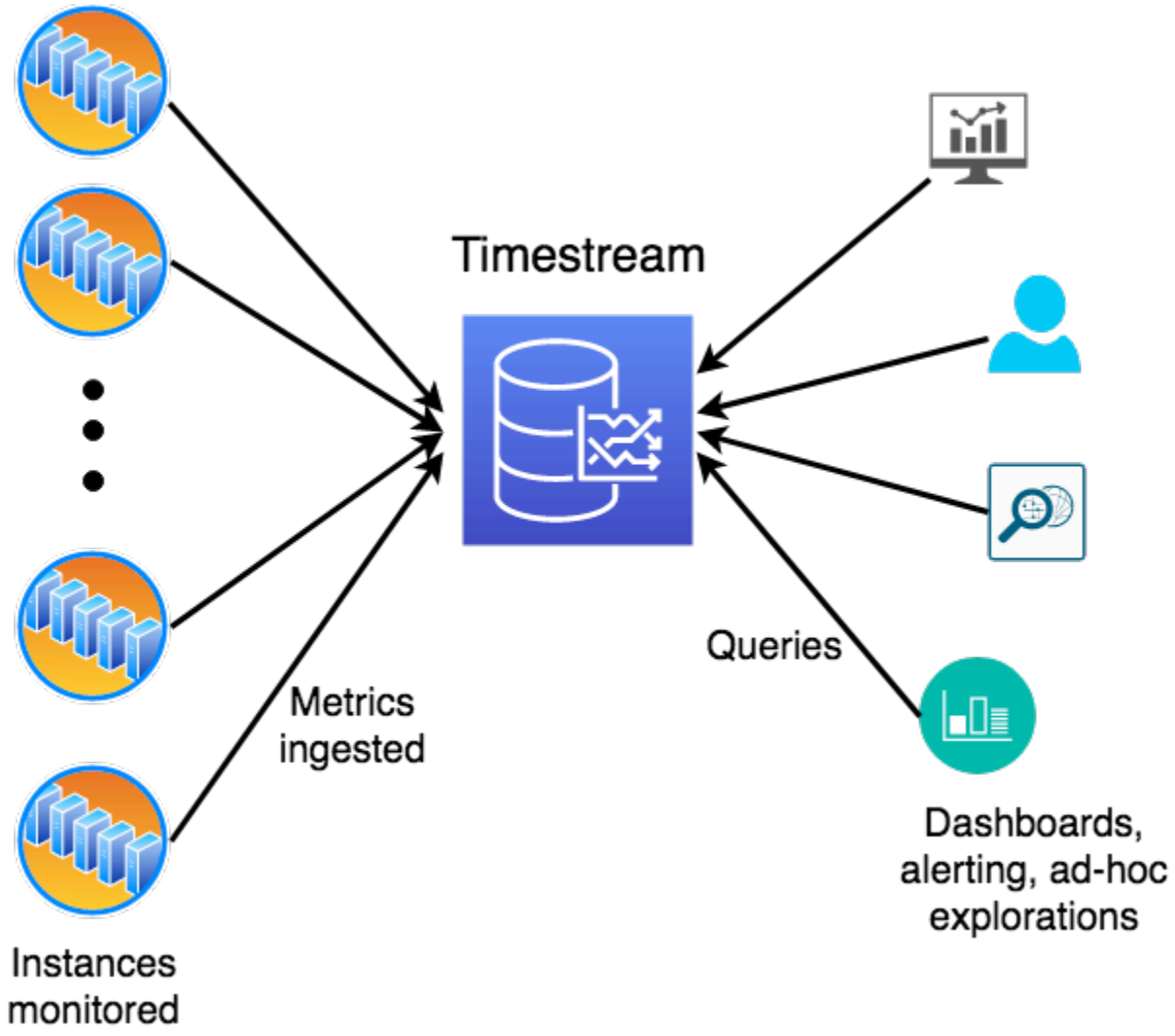
En esta sección se describen los patrones de uso de las consultas programadas, así como end-to-end algunos ejemplos.

Temas

- Ejemplo de esquema de consultas programadas
- Patrones de consulta programados
- Ejemplos de consultas programadas

Ejemplo de esquema de consultas programadas

En este ejemplo, utilizaremos una aplicación de ejemplo que imita las métricas de monitoreo de un DevOps escenario de una gran flota de servidores. Los usuarios desean alertar sobre un uso anómalo de los recursos, crear paneles de control sobre el comportamiento y la utilización agregados de la flota y realizar análisis sofisticados de los datos recientes e históricos para encontrar correlaciones. En el siguiente diagrama se ilustra la configuración en la que un conjunto de instancias supervisadas emite métricas a Timestream. LiveAnalytics Otro grupo de usuarios simultáneos emite consultas para alertas, paneles o análisis ad hoc, donde las consultas y la ingesta se ejecutan en paralelo.



La aplicación que se monitorea está modelada como un servicio altamente escalable que se implementa en varias regiones del mundo. Además, cada región se subdivide en varias unidades de escalado denominadas células que tienen un nivel de aislamiento en términos de infraestructura

dentro de la región. Además, cada celda se subdivide en silos, lo que representa un nivel de aislamiento del software. Cada silo tiene cinco microservicios que comprenden una instancia aislada del servicio. Cada microservicio tiene varios servidores con diferentes tipos de instancias y versiones de sistema operativo, que se implementan en tres zonas de disponibilidad. Estos atributos que identifican a los servidores que emiten las métricas se modelan como dimensiones en Timestream for. LiveAnalytics En esta arquitectura, tenemos una jerarquía de dimensiones (como región, celda, silo y microservice_name) y otras dimensiones transversales (como instance_type y availability_zone).

La aplicación emite una variedad de métricas (como cpu_user y memory_free) y eventos (como task_completed y gc_reclaimed). Cada métrica o evento está asociado a ocho dimensiones (como una región o una celda) que identifican de forma exclusiva al servidor que lo emite. Los datos se escriben con las 20 métricas almacenadas juntas en un registro de múltiples medidas con métricas de nombres de medidas y los 5 eventos se almacenan juntos en otro registro de múltiples medidas con eventos de nombres de medidas. El modelo de datos, el esquema y la generación de datos se encuentran en el generador de datos de código abierto. Además del esquema y las distribuciones de datos, el generador de datos proporciona un ejemplo del uso de varios escritores para ingerir datos en paralelo, utilizando la escala de ingesta de Timestream para LiveAnalytics ingerir millones de mediciones por segundo. A continuación, mostramos el esquema (tabla y esquema de medidas) y algunos ejemplos de datos del conjunto de datos.

Temas

- Registros de medidas múltiples
- Registros de medida única

Registros de medidas múltiples

Esquema de tabla

A continuación se muestra el esquema de la tabla una vez que los datos se ingieren mediante registros de múltiples medidas. Es el resultado de la consulta. DESCRIBE Suponiendo que los datos se ingieren en una base de datos raw_data y en la tabla devops, a continuación se muestra la consulta.

```
DESCRIBE "raw_data"."devops"
```

Columna	Tipo	Secuencia temporal del tipo de atributo LiveAnalytics
availability_zone	varchar	DIMENSION
microservice_name	varchar	DIMENSION
nombre_instancia	varchar	DIMENSION
nombre_proceso	varchar	DIMENSION
os_version	varchar	DIMENSION
jdk_version	varchar	DIMENSION
célula	varchar	DIMENSION
región	varchar	DIMENSION
silo	varchar	DIMENSION
instance_type	varchar	DIMENSION
measure_name	varchar	MEASURE_NAME
tiempo	Marca de tiempo	TIMESTAMP
libre de memoria	double	MULTI
cpu_steal	double	MULTI
cpu_iowait	double	MULTI
cpu_user	double	MULTI
memory_cached	double	MULTI
disk_io_reads	double	MULTI
cpu_hi	double	MULTI
latencia_por lectura	double	MULTI

Columna	Tipo	Secuencia temporal del tipo de atributo LiveAnalytics
net_bytes_out	double	MULTI
cpu_idle	double	MULTI
disk_free	double	MULTI
memoria_utilizada	double	MULTI
sistema_CPU	double	MULTI
descriptores de archivos en uso	double	MULTI
disk_used	double	MULTI
cpu_nice	double	MULTI
disk_io_writes	double	MULTI
cpu_si	double	MULTI
latencia_por escritura	double	MULTI
network_bytes_in	double	MULTI
estado final de la tarea	varchar	MULTI
gc_pause	double	MULTI
task_completada	bigint	MULTI
gc_reclaimed	double	MULTI

Esquema de medidas

A continuación se muestra el esquema de medidas devuelto por la SHOW MEASURES consulta.

```
SHOW MEASURES FROM "raw_data"."devops"
```

measure_name	data_type	Dimensiones
events	múltiple	[{"data_type» :"varchar», "dimension_name» :"availab ility_zone "}, {" data_type » :"varchar», "dimensio n_name» :"microservice_nam e "}, {" data_type» :"varchar», "dimension_name» :"instanc e_name "}, {" data_type » :"varchar», «dimensio n_name» :"nombre_proceso "}, {" data_type» :"varchar», "dimension_name» :"jdk_ver sion "}, {" data_type» :"varchar », "dimension_name» :"celda "}, {" data_type» :"varchar», "dimension_name» :"region "}, {" data_type» :« varchar», «nombre_dimensión» :"silo "}]
métricas	múltiple	[{"data_type» :"varchar», "dimension_name» :"availab ility_zone "}, {" data_type » :"varchar», "dimensio n_name» :"microservice_nam e "}, {" data_type» :"varchar», "dimension_name» :"instanc e_name "}, {" data_type » :"varchar», «dimensio n_name» :"os_version "}, {" data_type» :"varchar», "dimension_name» :"cell "}, {" data_type» :"varchar», "dimension_name» :"region "}, {" data_type» :"varchar

measure_name	data_type	Dimensiones
		», "dimension_name» :"silo "}, {" data_type» :"varchar», «nombre_dimensión» :"tipo_instancia "}]

Datos de ejemplo

re	C	S	a	n	n	in	o	n	jo	m	T	c	c	c	c	c	c	c	c	c	m	m	d	la	d	la	d	d	n	b	d	lib	ta	g	ta	g	s	s
			ity	ic	st	ty	n	o	o	a			m				t					til	ch	e	ra	ri	tu			y	ic	re	d	st		le	m	ed
us e	U Ec	us e- co sil	U E-1	a	i- -2 co - si 0 m co za at u - e	m e	A			m	1 1 1	6	0	3	0	0	0	0	0	5	8	5	9	3	2	6	2	8	4	3	5							
us e	U Ec	us e- co sil	U E-1	a	i- -2 co - si	m e	A			m	1 1 1	5	0	3	0	0	0	0	0	9	3	5	3	2	9	5	3	7	5	6	2							

re	C	S	a	n	n	in	o	n	jo	m	T	c	c	c	c	c	c	c	c	c	m	m	d	la	d	la	d	d	n	b	d	lib	ta	g	ta	g	se	la
			ity	ci	st	ty	on	o	oi	a			m				t					til	ch	e	pr	ri	pt			yt	ic	re	m				le	me
				ic	0 m c z a u s - e																				ra		tu				dr e d a e u		st					
u e	U E c	u e - c s il	U E -1	a	i- - -2 c c - si 0 m c z a u s - e	m e	A			m	1 1 1	4	0	4	0	0	0	0	0	9	4	5	3	8	3	7	6	8	5	4	8							

re	C	S	a	n	n	in	o	n	jo	m	T	c	c	c	c	c	c	c	c	c	m	m	d	la	d	la	d	d	n	b	d	lil	ta	g	ta	g	s	l
			ity	ci	st	ty	n	o	o	a			m				t					til	ch	e	pr	ri	pt			yt	ic	re	m	st		le		
				ic																					ra		tu					da							
																																ea							
																																u							
u	U	u	U	a	i-	m	A			m	1	3	0	5	0	0	0	0	0	4	7	2	4	4	3	8	6	2	1	2	1							
e	E	e	E		-2	e					1																													
	c	-	-1		c						1																													
		c			-																																			
		s			si																																			
		il			0																																			
					m																																			
					c																																			
					za																																			
					at																																			
					u																																			
					-																																			
					e																																			
u	U	u	U	a	i-	m	A			m	1	5	0	3	0	0	0	0	0	5	3	8	5	7	1	8	6	7	6	5	3							
e	E	e	E		-2	e					1																													
	c	-	-1		c						1																													
		c			-																																			
		s			si																																			
		il			0																																			
					m																																			
					c																																			
					za																																			
					at																																			
					u																																			
					-																																			
					e																																			

re	C	S	a	n	n	in	o	n	jo	m	T	c	c	c	c	c	c	c	c	c	m	m	d	la	d	la	d	d	n	b	d	li	ta	g	ta	g	s	c	l
			ity	ic	st	ty	n	o	o	a			m				t					til	ch	e	pr	ri	pt			y	ic	re	m	st		le	me	se	le
us	U	us	U	a	i-			a	J	e	1																					7	S	8	3	6	3	8
e	Co	el	E		-2			d			1																						_				4	4	
		-	-1		co			h			1																						W						
		co			-																												R						
		si			si																																		
		l			0																																		
					m																																		
					co																																		
					za																																		
					at																																		
					u																																		
					-																																		
					e																																		
us	U	us	U	a	i-			a	J	e	1																					7	S	2	4	7	3	4
e	Co	el	E		-2			d			1																						I				4	5	W
		-	-1		co			h			1																						T						JL
		co			-																																		
		si			si																																		
		l			0																																		
					m																																		
					co																																		
					za																																		
					at																																		
					u																																		
					-																																		
					e																																		

re	C	S	a	n	n	in	o	n	jo	m	T	c	c	c	c	c	c	c	c	c	m	m	d	la	d	la	d	d	n	b	d	li	ta	g	ta	g	s	s
			ity	ic	st	ty	n	o	o	a			m				t					til	ch	e	pr	ri	pt			y	ic	re	m	st		le	me	ed
us	U	us	U	a	i-			a	Jl	e	1																					6	S	6		2	72	2_	W
e	Ec	el	E		-2			d			1																						IT					3	
		-	-1		c			h			1																											JL	
		co			-						1																												
		si			si																																	
		lo			0																																	
					m																																	
					co																																	
					za																																	
					at																																	
					us																																	
					-																																	
					e																																	
us	U	us	U	a	i-			a	Jl	e	1																					2	S	5	1	99	9_	W
e	Ec	el	E		-2			d			1																						IT					JL	
		-	-1		c			h			1																												
		co			-																																	
		si			si																																	
		lo			0																																	
					m																																	
					co																																	
					za																																	
					at																																	
					us																																	
					-																																	
					e																																	

re	C	S	a	n	n	in	o	n	jo	m	T	c	c	c	c	c	c	c	c	c	m	m	d	la	d	la	d	d	n	b	d	lil	ta	g	ta	g	se	la
			ity	ic	st	ty	n	o	o	a			m				t					til	ch	e	pr	ri	pt			yt	ic	re	m	st		le	me	ed
us	US	us	US	av	i-			ac	JD	en	11																					3	SH	7	2	82	2,9

Registros de medida única

Timestream for LiveAnalytics también permite ingerir los datos con una medida por registro de serie temporal. A continuación se muestran los detalles del esquema cuando se ingieren mediante registros de una sola medida.

Esquema de tabla

A continuación se muestra el esquema de la tabla una vez que los datos se ingieren mediante registros de múltiples medidas. Es el resultado de la consulta. DESCRIBE Suponiendo que los datos se ingieren en una base de datos raw_data y en la tabla devops, a continuación se muestra la consulta.

```
DESCRIBE "raw_data"."devops_single"
```

Columna	Tipo	Secuencia temporal del tipo de atributo LiveAnalytics
availability_zone	varchar	DIMENSION
microservice_name	varchar	DIMENSION
nombre_instancia	varchar	DIMENSION
nombre_proceso	varchar	DIMENSION
os_version	varchar	DIMENSION
jdk_version	varchar	DIMENSION
célula	varchar	DIMENSION
región	varchar	DIMENSION
silo	varchar	DIMENSION
instance_type	varchar	DIMENSION
measure_name	varchar	MEASURE_NAME
tiempo	Marca de tiempo	TIMESTAMP
measure_value::double	double	MEASURE_VALUE
measure_value::bigint	bigint	MEASURE_VALUE
measure_value::varchar	varchar	MEASURE_VALUE

Esquema de medidas

A continuación se muestra el esquema de medidas devuelto por la SHOW MEASURES consulta.

```
SHOW MEASURES FROM "raw_data"."devops_single"
```

measure_name	data_type	Dimensiones
cpu_hi	double	[{'dimension_name': 'availabi lity_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_typ e': 'varchar'}, {'dimensi on_name': 'nombre_instancia' , 'data_type': 'varchar'}, {'dimension_name': 'os_versi on', 'data_type': 'varchar'}, {'dimension_name': 'cell', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}, {'dimension_name': instat tipo_de_de_de_de_datos ':' archivar '}]
cpu_inactivo	double	[{'dimension_name': 'availabi lity_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_typ e': 'varchar'}, {'dimensi on_name': 'nombre_instancia' , 'data_type': 'varchar'}, {'dimension_name': 'os_versi on', 'data_type': 'varchar'}, {'dimension_name': 'cell', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'},

measure_name	data_type	Dimensiones
		{'dimension_name': instat tipo_de_de_de_de_datos ':' archivar '}]
cpu_iowait	double	[{'dimension_name': 'availabi lity_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_typ e': 'varchar'}, {'dimensi on_name': 'nombre_instancia' , 'data_type': 'varchar'}, {'dimension_name': 'os_versi on', 'data_type': 'varchar'}, {'dimension_name': 'cell', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}, {'dimension_name': instat tipo_de_de_de_de_datos ':' archivar '}]

measure_name	data_type	Dimensiones
cpu_nice	double	[{'dimension_name': 'availabi lity_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_typ e': 'varchar'}, {'dimensi on_name': 'nombre_instancia' , 'data_type': 'varchar'}, {'dimension_name': 'os_versi on', 'data_type': 'varchar'}, {'dimension_name': 'cell', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}, {'dimension_name': instat tipo_de_de_de_de_datos ':' archivar '}]

measure_name	data_type	Dimensiones
cpu_si	double	[{'dimension_name': 'availabi lity_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_typ e': 'varchar'}, {'dimensi on_name': 'nombre_instancia' , 'data_type': 'varchar'}, {'dimension_name': 'os_versi on', 'data_type': 'varchar'}, {'dimension_name': 'cell', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}, {'dimension_name': instat tipo_de_de_de_de_datos ':' archivar '}]

measure_name	data_type	Dimensiones
cpu_steal	double	[{'dimension_name': 'availability_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_type': 'varchar'}, {'dimension_name': 'nombre_instancia', 'data_type': 'varchar'}, {'dimension_name': 'os_version', 'data_type': 'varchar'}, {'dimension_name': 'cell', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}, {'dimension_name': instat tipo_de_de_de_de_datos ':' archivar '}]

measure_name	data_type	Dimensiones
sistema_de_CPU	double	[{'dimension_name': 'availabi lity_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_typ e': 'varchar'}, {'dimensi on_name': 'nombre_instancia' , 'data_type': 'varchar'}, {'dimension_name': 'os_versi on', 'data_type': 'varchar'}, {'dimension_name': 'cell', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}, {'dimension_name': instat tipo_de_de_de_de_datos ':' archivar '}]

measure_name	data_type	Dimensiones
usuario_de_CPU	double	[{'dimension_name': 'availabi lity_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_typ e': 'varchar'}, {'dimensi on_name': 'nombre_instancia' , 'data_type': 'varchar'}, {'dimension_name': 'os_versi on', 'data_type': 'varchar'}, {'dimension_name': 'cell', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}, {'dimension_name': instat tipo_de_de_de_de_datos ':' archivar '}]

measure_name	data_type	Dimensiones
disk_free	double	[{'dimension_name': 'availabi lity_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_typ e': 'varchar'}, {'dimensi on_name': 'nombre_instancia' , 'data_type': 'varchar'}, {'dimension_name': 'os_versi on', 'data_type': 'varchar'}, {'dimension_name': 'cell', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}, {'dimension_name': instat tipo_de_de_de_de_datos ':' archivar '}]

measure_name	data_type	Dimensiones
disk_io_reads	double	[{'dimension_name': 'availabi lity_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_typ e': 'varchar'}, {'dimensi on_name': 'nombre_instancia' , 'data_type': 'varchar'}, {'dimension_name': 'os_versi on', 'data_type': 'varchar'}, {'dimension_name': 'cell', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}, {'dimension_name': instat tipo_de_de_de_de_datos ':' archivar '}]

measure_name	data_type	Dimensiones
disk_io_writes	double	[{'dimension_name': 'availabi lity_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_typ e': 'varchar'}, {'dimensi on_name': 'nombre_instancia' , 'data_type': 'varchar'}, {'dimension_name': 'os_versi on', 'data_type': 'varchar'}, {'dimension_name': 'cell', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}, {'dimension_name': instat tipo_de_de_de_de_datos ':' archivar '}]

measure_name	data_type	Dimensiones
disk_used	double	[{'dimension_name': 'availability_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_type': 'varchar'}, {'dimension_name': 'nombre_instancia', 'data_type': 'varchar'}, {'dimension_name': 'os_version', 'data_type': 'varchar'}, {'dimension_name': 'cell', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}, {'dimension_name': instat tipo_de_de_de_de_datos ':' archivar '}]

measure_name	data_type	Dimensiones
descriptores de archivos en uso	double	[{'dimension_name': 'availability_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_type': 'varchar'}, {'dimension_name': 'nombre_instancia', 'data_type': 'varchar'}, {'dimension_name': 'os_version', 'data_type': 'varchar'}, {'dimension_name': 'cell', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}, {'dimension_name': instat tipo_de_de_de_de_datos ':' archivar '}]

measure_name	data_type	Dimensiones
gc_pause	double	[{'dimension_name': 'availability_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_type': 'varchar'}, {'dimension_name': 'instance_name', 'data_type': 'varchar'}, {'dimension_name': 'process_name', 'data_type': 'varchar'}, {'dimension_name': 'jdk_version', 'data_type': 'varchar'}, {'dimension_name': 'celda', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}]

measure_name	data_type	Dimensiones
gc_reclaimed	double	[{'dimension_name': 'availability_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_type': 'varchar'}, {'dimension_name': 'instance_name', 'data_type': 'varchar'}, {'dimension_name': 'process_name', 'data_type': 'varchar'}, {'dimension_name': 'jdk_version', 'data_type': 'varchar'}, {'dimension_name': 'celda', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}]

measure_name	data_type	Dimensiones
latencia_por_lectura	double	[{'dimension_name': 'availability_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_type': 'varchar'}, {'dimension_name': 'nombre_instancia', 'data_type': 'varchar'}, {'dimension_name': 'os_version', 'data_type': 'varchar'}, {'dimension_name': 'cell', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}, {'dimension_name': instat tipo_de_de_de_de_datos ':' archivar '}]

measure_name	data_type	Dimensiones
latencia_por_escritura	double	[{'dimension_name': 'availability_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_type': 'varchar'}, {'dimension_name': 'nombre_instancia', 'data_type': 'varchar'}, {'dimension_name': 'os_version', 'data_type': 'varchar'}, {'dimension_name': 'cell', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}, {'dimension_name': instat tipo_de_de_de_de_datos ':' archivar '}]

measure_name	data_type	Dimensiones
memoria_cacheada	double	[{'dimension_name': 'availability_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_type': 'varchar'}, {'dimension_name': 'nombre_instancia', 'data_type': 'varchar'}, {'dimension_name': 'os_version', 'data_type': 'varchar'}, {'dimension_name': 'cell', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}, {'dimension_name': instat tipo_de_de_de_de_datos ':' archivar '}]

measure_name	data_type	Dimensiones
libre de memoria	double	[{'dimension_name': 'availability_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_type': 'varchar'}, {'dimension_name': 'instance_name', 'data_type': 'varchar'}, {'dimension_name': 'process_name', 'data_type': 'varchar'}, {'dimension_name': 'os_version', 'data_type': 'varchar'}, {'dimension_name': 'jdk_version', 'data_type': 'varchar'}, {'dimension_name': 'celda', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}, {'dimension_name': 'instance_type', 'datos_ tipo': 'varchar'}]

measure_name	data_type	Dimensiones
memoria_utilizada	double	[{'dimension_name': 'availability_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_type': 'varchar'}, {'dimension_name': 'nombre_instancia', 'data_type': 'varchar'}, {'dimension_name': 'os_version', 'data_type': 'varchar'}, {'dimension_name': 'cell', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}, {'dimension_name': instat tipo_de_de_de_de_datos ':' archivar '}]

measure_name	data_type	Dimensiones
net_bytes_in	double	[{'dimension_name': 'availabi lity_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_typ e': 'varchar'}, {'dimensi on_name': 'nombre_instancia' , 'data_type': 'varchar'}, {'dimension_name': 'os_versi on', 'data_type': 'varchar'}, {'dimension_name': 'cell', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}, {'dimension_name': instat tipo_de_de_de_de_datos ':' archivar '}]

measure_name	data_type	Dimensiones
salida de bytes de red	double	[{'dimension_name': 'availability_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_type': 'varchar'}, {'dimension_name': 'nombre_instancia', 'data_type': 'varchar'}, {'dimension_name': 'os_version', 'data_type': 'varchar'}, {'dimension_name': 'cell', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}, {'dimension_name': instat tipo_de_de_de_de_datos ':' archivar '}]

measure_name	data_type	Dimensiones
tarea_completada	bigint	[{'dimension_name': 'availability_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_type': 'varchar'}, {'dimension_name': 'instance_name', 'data_type': 'varchar'}, {'dimension_name': 'process_name', 'data_type': 'varchar'}, {'dimension_name': 'jdk_version', 'data_type': 'varchar'}, {'dimension_name': 'celda', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}]

measure_name	data_type	Dimensiones
estado final de la tarea	varchar	[{'dimension_name': 'availability_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_type': 'varchar'}, {'dimension_name': 'instance_name', 'data_type': 'varchar'}, {'dimension_name': 'process_name', 'data_type': 'varchar'}, {'dimension_name': 'jdk_version', 'data_type': 'varchar'}, {'dimension_name': 'celda', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}]

Datos de ejemplo

availa ity_zo	micro ice_n	nomb stanc	nomb ocesc	os_ve n	jdk_v on	Celda	regió	Silo	instar type	meas ame	Tiem	meas alue:: ble	meas alue:: int	measure_v alue::var char
eu-west--1	hércu	i- -zaZsv -1-celda 9-silo-2 0000(amaz .com hercu		AL20		eu-west- - celda	eu-west-	eu-west- - celd-S s ilo-2	r5.4xl e	cpu_l	34:57	0,871		

availa ity_zo	micro ice_n	nomb stanc	nomb ocesc	os_ve n	jdk_v on	Celda	regió	Silo	instar type	meas ame	Tiem	meas alue:: ble	meas alue:: int	measure_v alue::var char
		eu- west												
eu- west- -1	hércu	i- - zaZsv -1- celda 9- silo-2 0000(amaz .com hercu eu- west		AL20		eu- west- - celda	eu- west-	eu- west- - celd-9 s ilo-2	r5.4xl e	cpu_i	34:57	3,462		
eu- west- -1	hércu	i- - zaZsv -1- celda 9- silo-2 0000(amaz .com hercu eu- west		AL20		eu- west- - celda	eu- west-	eu- west- - celd-9 s ilo-2	r5.4xl e	cpu_i t	34:57	0,102		

availa ity_zo	micro ice_n	nomb stanc	nomb ocesc	os_ve n	jdk_v on	Celda	regió	Silo	instar type	meas ame	Tiem	meas alue:: ble	meas alue:: int	measure_v alue::var char
eu-west--1	hércu	i- -zaZsv -1-celda 9-silo-2 0000(amaz .com hercu eu-west		AL20		eu-west- -celda	eu-west-	eu-west- -celd-9 s ilo-2	r5.4xl e	cpu_r	34:57	0,601		
eu-west--1	hércu	i- -zaZsv -1-celda 9-silo-2 0000(amaz .com hercu eu-west		AL20		eu-west- -celda	eu-west-	eu-west- -celd-9 s ilo-2	r5.4xl e	cpu_s	34:57	0,164		

availa ity_zo	micro ice_n	nomb stanc	nomb ocesc	os_ve n	jdk_v on	Celda	regió	Silo	instar type	meas ame	Tiem	meas alue:: ble	meas alue:: int	measure_v alue::var char
eu-west--1	hércu	i- -zaZsv -1-celda 9-silo-2 0000(amaz .com hercu eu-west		AL20		eu-west- -celda	eu-west-	eu-west- -celd-9 s ilo-2	r5.4xl e	cpu_s	34:57	0,107		
eu-west--1	hércu	i- -zaZsv -1-celda 9-silo-2 0000(amaz .com hercu eu-west		AL20		eu-west- -celda	eu-west-	eu-west- -celd-9 s ilo-2	r5.4xl e	sister PU	34:57	0,457		

availa ity_zo	micro ice_n	nomb stanc	nomb ocesc	os_ve n	jdk_v on	Celda	regió	Silo	instar type	meas ame	Tiem	meas alue:: ble	meas alue:: int	measure_v alue::var char
eu-west- -1	hércu	i- - zaZsv -1- celda 9- silo-2 0000(amaz .com hercu eu- west		AL20		eu- west- - celda	eu- west-	eu- west- - celd-9 s ilo-2	r5.4xl e	cpu_u io	34:57	94,20		
eu-west- -1	hércu	i- - zaZsv -1- celda 9- silo-2 0000(amaz .com hercu eu- west		AL20		eu- west- - celda	eu- west-	eu- west- - celd-9 s ilo-2	r5.4xl e	disk_	34:57	72.51		

availa ity_zo	micro ice_n	nomb stanc	nomb ocesc	os_ve n	jdk_v on	Celda	regió	Silo	instar type	meas ame	Tiem	meas alue:: ble	meas alue:: int	measure_v alue::var char
eu-west--1	hércu	i- -zaZsv -1-celda 9-silo-2 0000(amaz .com hercu eu-west		AL20		eu-west- -celda	eu-west-	eu-west- -celd-9 s ilo-2	r5.4xl e	disk_i eads	34:57	81.73		
eu-west--1	hércu	i- -zaZsv -1-celda 9-silo-2 0000(amaz .com hercu eu-west		AL20		eu-west- -celda	eu-west-	eu-west- -celd-9 s ilo-2	r5.4xl e	disk_i rites	34:57	77,11		

availa ity_zo	micro ice_n	nomb stanc	nomb ocesc	os_ve n	jdk_v on	Celda	regió	Silo	instar type	meas ame	Tiem	meas alue:: ble	meas alue:: int	measure_v alue::var char
eu-west- -1	hércu	i- - zaZsv -1- celda 9- silo-2 0000(amaz .com hercu eu- west		AL20		eu- west- - celda	eu- west-	eu- west- - celd-9 s ilo-2	r5.4xl e	disk_	34:57	89,42		
eu-west- -1	hércu	i- - zaZsv -1- celda 9- silo-2 0000(amaz .com hercu eu- west		AL20		eu- west- - celda	eu- west-	eu- west- - celd-9 s ilo-2	r5.4xl e	descr res de archiv en uso	34:57	30,08		

availa ity_zo	micro ice_n	nomb stanc	nomb ocesc	os_ve n	jdk_v on	Celda	regió	Silo	instar type	meas ame	Tiem	meas alue:: ble	meas alue:: int	measure_v alue::var char
eu- west- -1	hércu	i- - zaZs -1- celda 9- silo-2 0000 amaz .com hercu eu- west	serve		JDK_	eu- west- - celda	eu- west-	eu- west- - celd- s ilo-2		gc_pa	34:57	60,28		
eu- west- -1	hércu	i- - zaZs -1- celda 9- silo-2 0000 amaz .com hercu eu- west	serve		JDK_	eu- west- - celda	eu- west-	eu- west- - celd- s ilo-2		gc_re med	34:57	75,28		

availa ity_zo	micro ice_n	nomb stanc	nomb ocesc	os_ve n	jdk_v on	Celda	regió	Silo	instar type	meas ame	Tiem	meas alue:: ble	meas alue:: int	measure_v alue::var char
eu-west- -1	hércu	i- - zaZsv -1- celda 9- silo-2 0000(amaz .com hercu eu- west		AL20		eu- west- - celda	eu- west-	eu- west- - celd-9 s ilo-2	r5.4xl e	laten(por_le ra	34:57	8,076		
eu-west- -1	hércu	i- - zaZsv -1- celda 9- silo-2 0000(amaz .com hercu eu- west		AL20		eu- west- - celda	eu- west-	eu- west- - celd-9 s ilo-2	r5.4xl e	laten(por_e tura	34:57	58,11		

availa ity_zo	micro ice_n	nomb stanc	nomb ocesc	os_ve n	jdk_v on	Celda	regió	Silo	instar type	meas ame	Tiem	meas alue:: ble	meas alue:: int	measure_v alue::var char
eu-west- -1	hércu	i- - zaZsv -1- celda 9- silo-2 0000(amaz .com hercu eu- west		AL20		eu- west- - celda	eu- west-	eu- west- - celd-9 s ilo-2	r5.4xl e	mem aché	34:57	87,56		
eu- west- -1	hércu	i- - zaZsv -1- celda 9- silo-2 0000(amaz .com hercu eu- west	serve		JDK_	eu- west- - celda	eu- west-	eu- west- - celd-9 s ilo-2		libre de mem	34:57	18,95		

availa ity_zo	micro ice_n	nomb stanc	nomb ocesc	os_ve n	jdk_v on	Celda	regió	Silo	instar type	meas ame	Tiem	meas alue:: ble	meas alue:: int	measure_v alue::var char
eu-west--1	hércu	i- -zaZsv -1-celda 9-silo-2 0000(amaz .com hercu eu-west		AL20		eu-west- -celda	eu-west-	eu-west- -celd-9 s ilo-2	r5.4xl e	libre de mem(	34:57	97,20		
eu-west--1	hércu	i- -zaZsv -1-celda 9-silo-2 0000(amaz .com hercu eu-west		AL20		eu-west- -celda	eu-west-	eu-west- -celd-9 s ilo-2	r5.4xl e	mem(tilizac	34:57	12,37		

availa ity_zo	micro ice_n	nomb stanc	nomb ocesc	os_ve n	jdk_v on	Celda	regió	Silo	instar type	meas ame	Tiem	meas alue:: ble	meas alue:: int	measure_v alue::var char
eu-west--1	hércu	i- -zaZsv -1-celda 9-silo-2 0000(amaz .com hercu eu-west		AL20		eu-west- -celda	eu-west-	eu-west- -celd-9 s ilo-2	r5.4xl e	netwo ytes_	34:57	31.02		
eu-west--1	hércu	i- -zaZsv -1-celda 9-silo-2 0000(amaz .com hercu eu-west		AL20		eu-west- -celda	eu-west-	eu-west- -celd-9 s ilo-2	r5.4xl e	bytes de red	34:57	0,542		

availa ity_zo	micro ice_n	nomb stanc	nomb ocesc	os_ve n	jdk_v on	Celda	regió	Silo	instar type	meas ame	Tiem	meas alue:: ble	meas alue:: int	measure_v alue::var char
eu-west--1	hércu	i- -zaZsv -1-celda 9-silo-2 0000(amaz .com hercu eu-west	serve		JDK_	eu-west- - celda	eu-west-	eu-west- - celd-9 s ilo-2		tarea_ pleta(	34:57		69	
eu-west--1	hércu	i- -zaZsv -1-celda 9-silo-2 0000(amaz .com hercu eu-west	serve		JDK_	eu-west- - celda	eu-west-	eu-west- - celd-9 s ilo-2		estad final de la tarea	34:57			SUCCESS_ ITH_RESUL T

Patrones de consulta programados

En esta sección encontrará algunos patrones comunes sobre cómo puede utilizar Amazon Timestream LiveAnalytics for Scheduled Queries para optimizar sus cuadros de mando a fin de que se carguen más rápido y con costes reducidos. Los ejemplos siguientes utilizan un escenario de

DevOps aplicación para ilustrar los conceptos clave que se aplican a las consultas programadas en general, independientemente del escenario de aplicación.

Las consultas programadas en Timestream LiveAnalytics le permiten expresar sus consultas utilizando toda la SQL superficie de Timestream for. LiveAnalytics La consulta puede incluir una o más tablas de origen, realizar agregaciones o cualquier otra consulta permitida por Timestream para el SQL idioma de Timestream y, a continuación, materializar los resultados LiveAnalytics de la consulta en otra tabla de destino de Timestream for. LiveAnalytics Para facilitar la exposición, en esta sección se hace referencia a la tabla de destino de una consulta programada como tabla derivada.

Los puntos clave que se tratan en esta sección son los siguientes.

- Utilice un agregado simple a nivel de flota para explicar cómo se puede definir una consulta programada y comprender algunos conceptos básicos.
- Cómo puede combinar los resultados del objetivo de una consulta programada (la tabla derivada) con los resultados de la tabla de origen para obtener las ventajas de coste y rendimiento de la consulta programada.
- ¿Cuáles son las ventajas y desventajas a la hora de configurar el período de actualización de las consultas programadas?
- Uso de consultas programadas para algunos escenarios comunes.
  - Realizar un seguimiento del último punto de datos de cada instancia antes de una fecha específica.
  - Valores distintos para una dimensión que se utilizarán para rellenar las variables en un panel de control.
- Cómo se gestionan los datos que llegan tarde en el contexto de las consultas programadas.
- Cómo utilizar las ejecuciones manuales puntuales para gestionar una variedad de situaciones que los activadores automáticos de las consultas programadas no contemplan directamente.

Temas

- Rellenar los cálculos previos históricos

Escenario

Los siguientes ejemplos utilizan un escenario DevOps de supervisión que se describe enEjemplo de esquema de consultas programadas.

Los ejemplos proporcionan la definición de consulta programada, en la que se pueden introducir las configuraciones adecuadas para saber dónde recibir las notificaciones del estado de ejecución de las consultas programadas, dónde recibir los informes de los errores encontrados durante la ejecución de una consulta programada y IAM qué función utiliza la consulta programada para realizar sus operaciones.

Puede crear estas consultas programadas tras rellenar las opciones anteriores, crear la tabla de destino (o derivada) y ejecutarla mediante la AWSCLI. Por ejemplo, supongamos que una definición de consulta programada está almacenada en un archivo,`scheduled_query_example.json`. Puede crear la consulta mediante el CLI comando.

```
aws timestream-query create-scheduled-query --cli-input-json  file://
scheduled_query_example.json --profile aws_profile --region us-east-1
```

En el comando anterior, el perfil transferido mediante la opción --profile debe tener los permisos adecuados para crear consultas programadas. Consulte Políticas basadas en la identidad para consultas programadas para obtener instrucciones detalladas sobre las políticas y los permisos.

Agregados simples a nivel de flota

En este primer ejemplo, se explican algunos de los conceptos básicos al trabajar con consultas programadas. Para ello, se utiliza un ejemplo sencillo de cálculo de agregados a nivel de flota. Con este ejemplo, aprenderá lo siguiente.

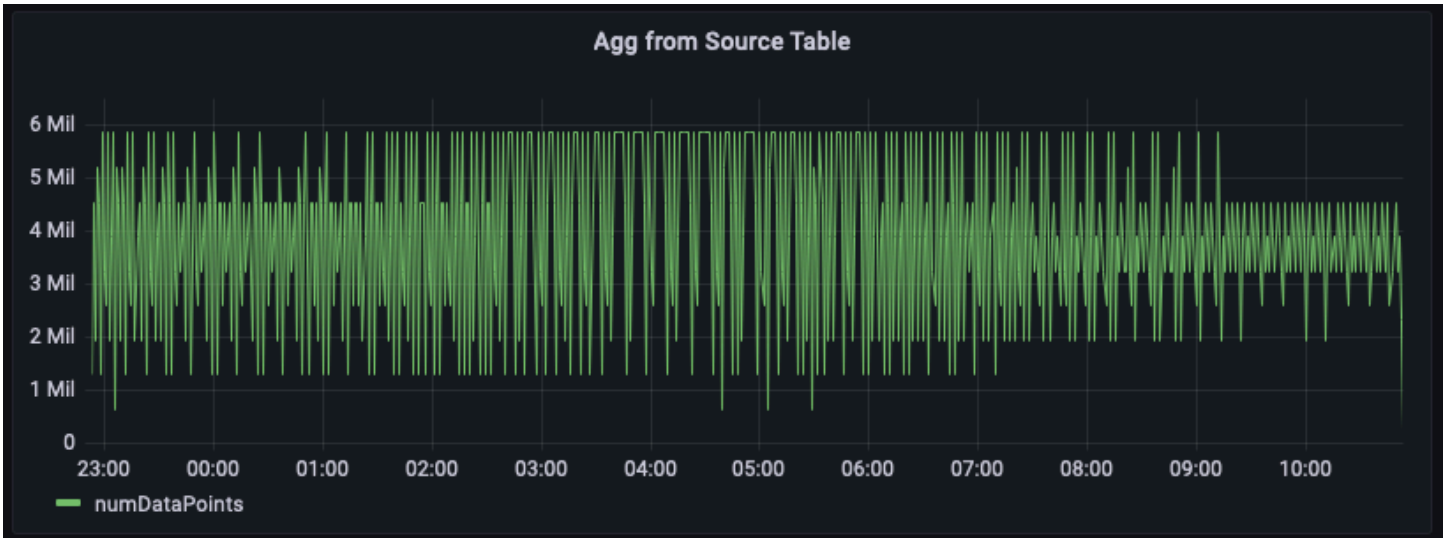
- Cómo tomar la consulta del panel de control que se utiliza para obtener estadísticas agregadas y asignarla a una consulta programada.
- Cómo LiveAnalytics gestiona Timestream for la ejecución de las diferentes instancias de su consulta programada.
- Cómo hacer que diferentes instancias de consultas programadas se superpongan en intervalos de tiempo y cómo mantener la exactitud de los datos en la tabla de destino para garantizar que el panel de control que utiliza los resultados de la consulta programada le proporcione resultados que coincidan con el mismo agregado calculado a partir de los datos sin procesar.

- Cómo configurar el intervalo de tiempo y la cadencia de actualización de la consulta programada.
- Cómo puede realizar un seguimiento automático de los resultados de las consultas programadas para ajustarlos de modo que la latencia de ejecución de las instancias de consulta se mantenga dentro de los plazos aceptables necesarios para actualizar los paneles.

Temas

Agregue datos de las tablas de origen

En este ejemplo, realiza un seguimiento del número de métricas emitidas por los servidores de una región determinada en cada minuto. El siguiente gráfico es un ejemplo que representa esta serie temporal para la región us-east-1.



A continuación se muestra un ejemplo de consulta para calcular este agregado a partir de los datos sin procesar. Filtra las filas de la región us-east-1 y, a continuación, calcula la suma por minuto teniendo en cuenta las 20 métricas (si measure_name es métrica) o 5 eventos (si measure_name es eventos). En este ejemplo, la ilustración gráfica muestra que el número de métricas emitidas varía entre 1,5 y 6 millones por minuto. Al trazar esta serie temporal durante varias horas (las últimas 12 horas en esta figura), esta consulta sobre los datos sin procesar analiza cientos de millones de filas.

```
WITH grouped_data AS (
    SELECT region, bin(time, 1m) as minute, SUM(CASE WHEN measure_name = 'metrics' THEN
 20 ELSE 5 END) as numDataPoints
    FROM "raw_data"."devops"
    WHERE time BETWEEN from_milliseconds(1636699996445) AND
 from_milliseconds(1636743196445)
        AND region = 'us-east-1'
    GROUP BY region, measure_name, bin(time, 1m)
)
SELECT minute, SUM(numDataPoints) AS numDataPoints
FROM grouped_data
GROUP BY minute
ORDER BY 1 desc, 2 desc
```

Consulta programada para calcular previamente los agregados

Si desea optimizar sus paneles para que se carguen más rápido y reducir los costes escaneando menos datos, puede utilizar una consulta programada para calcular previamente estos agregados. Las consultas programadas en Timestream for LiveAnalytics le permiten materializar estos cálculos previos en otra LiveAnalytics tabla Timestream for, que puede utilizar posteriormente para sus paneles.

El primer paso para crear una consulta programada consiste en identificar la consulta que desea precalcular. Tenga en cuenta que el cuadro de mando anterior se dibujó para la región us-east-1. Sin embargo, un usuario diferente puede querer el mismo agregado para una región diferente, por ejemplo, us-west-2 o eu-west-1. Para evitar crear una consulta programada para cada una de estas consultas, puede calcular previamente el agregado de cada región y materializar los agregados por región en otra tabla Timestream for. LiveAnalytics

La siguiente consulta proporciona un ejemplo del cálculo previo correspondiente. Como puede ver, es similar a la expresión de tabla habitual grouped_data que se utiliza en la consulta de los datos sin procesar, con la salvedad de dos diferencias: 1) no utiliza un predicado de región, por lo que podemos utilizar una consulta para realizar el cálculo previo de todas las regiones; y 2) utiliza un predicado temporal parametrizado con un parámetro especial @scheduled_runtime, que se explica en detalle a continuación.

```
SELECT region, bin(time, 1m) as minute,
    SUM(CASE WHEN measure_name = 'metrics' THEN 20 ELSE 5 END) as numDataPoints
FROM raw_data.devops
WHERE time BETWEEN @scheduled_runtime - 10m AND @scheduled_runtime + 1m
```

```
GROUP BY bin(time, 1m), region
```

La consulta anterior se puede convertir en una consulta programada mediante la siguiente especificación. A la consulta programada se le asigna un nombre, un mnemotécnico fácil de usar. Luego incluye la QueryString, a ScheduleConfiguration, que es una expresión cron. Especifica para TargetConfiguration qué se asignan los resultados de la consulta a la tabla de destino en Timestream. LiveAnalytics Por último, especifica otras configuraciones, como la siguiente: en la NotificationConfiguration que se envían notificaciones sobre las ejecuciones individuales de la consulta, ErrorReportConfiguration en la que se redacta un informe en caso de que la consulta detecte algún error y ScheduledQueryExecutionRoleArn, que es la función que se utiliza para realizar las operaciones de la consulta programada.

```
{
    "Name": "MultiPT5mPerMinutePerRegionMeasureCount",
    "QueryString": "SELECT region, bin(time, 1m) as minute, SUM(CASE WHEN measure_name
 = 'metrics' THEN 20 ELSE 5 END) as numDataPoints FROM raw_data.devops WHERE time
 BETWEEN @scheduled_runtime - 10m AND @scheduled_runtime + 1m GROUP BY bin(time, 1m),
 region",
    "ScheduleConfiguration": {
        "ScheduleExpression": "cron(0/5 * * * ? *)"
    },
    "NotificationConfiguration": {
        "SnsConfiguration": {
            "TopicArn": "******"
        }
    },
    "TargetConfiguration": {
        "TimestreamConfiguration": {
            "DatabaseName": "derived",
            "TableName": "per_minute_aggs_pt5m",
            "TimeColumn": "minute",
            "DimensionMappings": [
                {
                    "Name": "region",
                    "DimensionValueType": "VARCHAR"
                }
            ],
            "MultiMeasureMappings": {
                "TargetMultiMeasureName": "numDataPoints",
                "MultiMeasureAttributeMappings": [
                    {
                        "SourceColumn": "numDataPoints",
```

```
                        "MeasureValueType": "BIGINT"
                    }
                ]
            }
        }
    },
    "ErrorReportConfiguration": {
        "S3Configuration" : {
            "BucketName" : "******",
            "ObjectKeyPrefix": "errors",
            "EncryptionOption": "SSE_S3"
        }
    },
    "ScheduledQueryExecutionRoleArn": "******"
}
```

En el ejemplo, el ScheduleExpression cron (0/5 * * *? *) implica que la consulta se ejecuta una vez cada 5 minutos, los días 5, 10, 15,... minutos de cada hora de cada día. Estas marcas de tiempo en las que se activa una instancia específica de esta consulta son las que se traducen en el parámetro @scheduled_runtime utilizado en la consulta. Por ejemplo, consideremos la instancia en la que esta consulta programada se ejecuta el 1 de diciembre de 2021 a las 00:00:00. En este caso, el parámetro @scheduled_runtime se inicializa con la marca de tiempo 2021-12-01 00:00:00 al invocar la consulta. Por lo tanto, esta instancia específica se ejecutará en la marca de tiempo 2021-12-01 00:00:00 y calculará los agregados por minuto desde el intervalo de tiempo 2021-11-30 23:50:00 hasta 2021-12-01 00:01:00. Del mismo modo, la siguiente instancia de esta consulta se activa en la marca de tiempo 2021-12-01 00:05:00 y, en ese caso, la consulta calculará los agregados por minuto desde el intervalo de tiempo 2021-11-30 23:55:00 hasta 2021-12-01 00:06:00. Por lo tanto, el parámetro @scheduled_runtime proporciona una consulta programada para calcular previamente los agregados para los intervalos de tiempo configurados utilizando el tiempo de invocación de las consultas.

Tenga en cuenta que dos instancias posteriores de la consulta se superponen en sus intervalos de tiempo. Esto es algo que puede controlar en función de sus necesidades. En este caso, esta superposición permite que estas consultas actualicen los agregados en función de cualquier dato cuya llegada se haya retrasado ligeramente (hasta 5 minutos en este ejemplo). Para garantizar la exactitud de las consultas materializadas, Timestream for LiveAnalytics garantiza que la consulta del 2021-12-01 00:05:00 se realice solo después de que se haya completado la consulta del 2021-12-01 00:00:00 y que los resultados de estas últimas consultas puedan actualizar cualquier agregado previamente materializado si se genera un valor más nuevo. Por ejemplo, si algunos

datos de la marca de tiempo 2021-11-30 23:59:00 llegaron después de ejecutarse la consulta de 2021-12-01 00:00:00 pero antes de la consulta de 2021-12-01 00:05:00, la ejecución de 2021-12-01 00:05:00 volverá a calcular los agregados del minuto 2021-11-30 23:59:00 y esto hará que el agregado anterior se actualice con el valor recién calculado. Puede confiar en esta semántica de las consultas programadas para llegar a un equilibrio entre la rapidez con la que actualiza sus cálculos previos y la forma en que puede gestionar correctamente algunos datos con retraso en la llegada. A continuación, se analizan consideraciones adicionales sobre cómo se compensa esta cadencia de actualización con la actualización de los datos y cómo se aborda la actualización de los agregados para los datos que llegan con más retraso o si la fuente del cálculo programado tiene valores actualizados que requerirían volver a calcular los agregados.

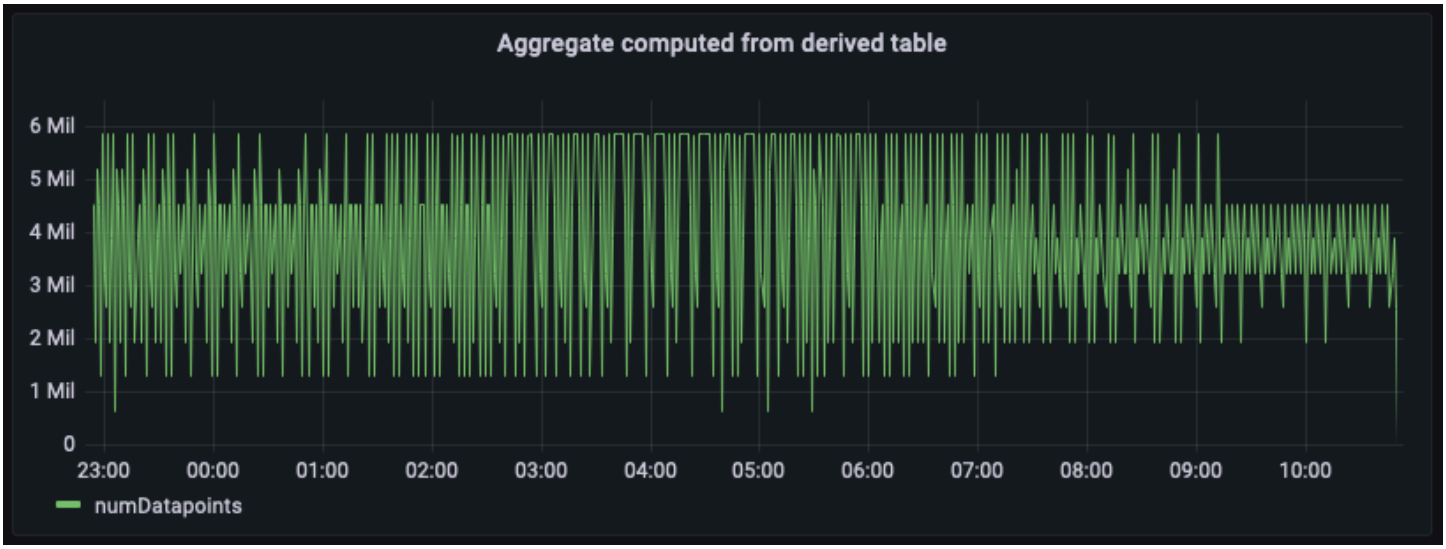
Cada cálculo programado tiene una configuración de notificaciones en la que Timestream for LiveAnalytics envía una notificación cada vez que se ejecuta una configuración programada. Puede configurar un SNS tema para recibir las notificaciones de cada invocación. Además del estado de éxito o error de una instancia específica, también tiene varias estadísticas, como el tiempo que tardó en ejecutarse el cálculo, el número de bytes que escaneó el cálculo y el número de bytes que el cálculo escribió en su tabla de destino. Puede usar estas estadísticas para ajustar aún más la consulta, programar la configuración o realizar un seguimiento del gasto de las consultas programadas. Un aspecto que vale la pena destacar es el tiempo de ejecución de una instancia. En este ejemplo, el cálculo programado está configurado para ejecutarse cada 5 minutos. El tiempo de ejecución determinará el retraso con el que estará disponible el cálculo previo, que también definirá el retraso en el panel cuando utilice los datos precalculados en los paneles. Además, si este retraso es constantemente superior al intervalo de actualización, por ejemplo, si el tiempo de ejecución es superior a 5 minutos para un cálculo configurado para que se actualice cada 5 minutos, es importante ajustar el cálculo para que se ejecute más rápido a fin de evitar más retrasos en los paneles.

Agregar a partir de una tabla derivada

Ahora que ha configurado las consultas programadas y los agregados están precalculados y materializados en otra LiveAnalytics tabla de flujo temporal especificada en la configuración de destino del cálculo programado, puede usar los datos de esa tabla para escribir SQL consultas que potencien sus paneles de control. A continuación se muestra un equivalente de la consulta que utiliza los preagregados materializados para generar el agregado del recuento de puntos de datos por minuto para us-east-1.

```
SELECT bin(time, 1m) as minute, SUM(numDataPoints) as numDatapoints
FROM "derived"."per_minute_aggs_pt5m"
```

```
WHERE time BETWEEN from_milliseconds(1636699996445) AND
 from_milliseconds(1636743196445)
    AND region = 'us-east-1'
GROUP BY bin(time, 1m)
ORDER BY 1 desc
```



La figura anterior representa el agregado calculado a partir de la tabla de agregados. Al comparar este panel con el panel calculado a partir de los datos fuente sin procesar, observará que coinciden exactamente, aunque estos agregados se retrasan unos minutos debido al intervalo de actualización que configuró para el cálculo programado más el tiempo de ejecución.

Esta consulta sobre los datos precalculados escanea datos varios órdenes de magnitud inferiores a los agregados calculados a partir de los datos fuente sin procesar. En función de la granularidad de las agregaciones, esta reducción puede reducir fácilmente hasta 100 veces el coste y la latencia de las consultas. La ejecución de este cálculo programado conlleva un coste. Sin embargo, en función de la frecuencia con la que se actualicen estos paneles y del número de usuarios simultáneos que los carguen, se acabará reduciendo considerablemente los costes totales si se utilizan estos cálculos previos. Además, los tiempos de carga de los paneles son entre 10 y 100 veces más rápidos.

Agregue la combinación de tablas fuente y derivadas

Los paneles creados con las tablas derivadas pueden tener un retraso. Si el escenario de su aplicación requiere que los paneles tengan los datos más recientes, puede utilizar la potencia y la flexibilidad del SQL soporte de Timestream for para LiveAnalytics combinar los datos más recientes de la tabla de origen con los agregados históricos de la tabla derivada para formar una vista combinada. Esta vista combinada utiliza la semántica de unión y los intervalos de tiempo que no

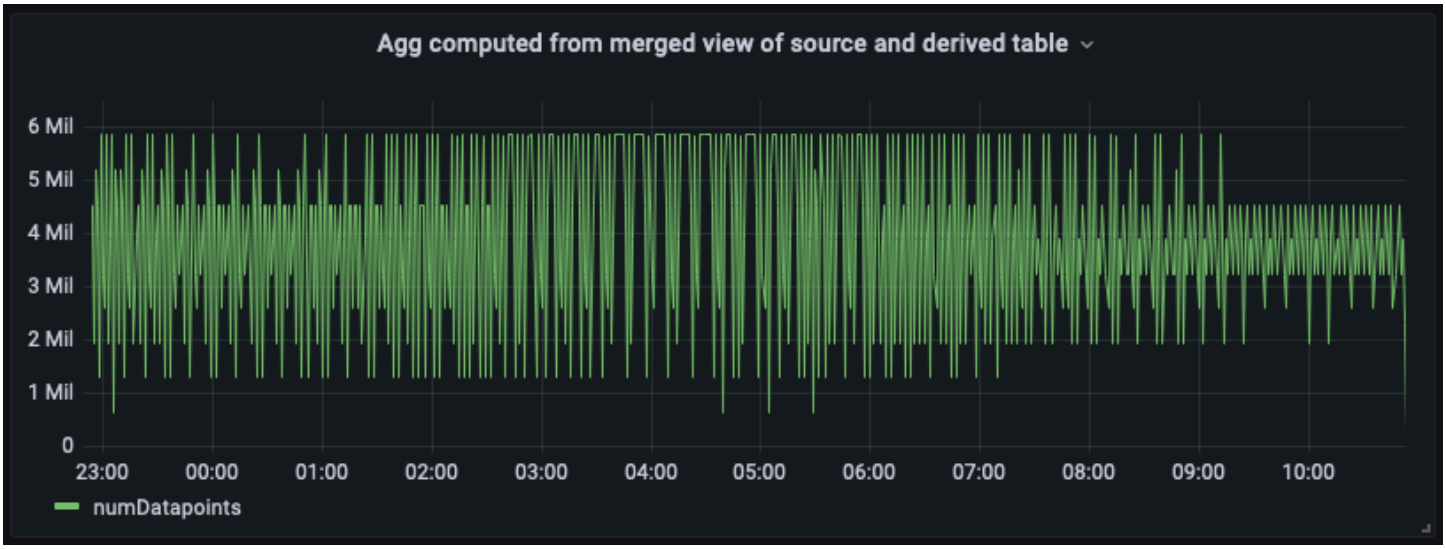
se superponen entre la tabla de origen SQL y la tabla derivada. En el siguiente ejemplo, utilizamos la palabra «derivada».» tabla derivada «per_minute_aggs_pt5m». Dado que el cálculo programado para esa tabla derivada se actualiza una vez cada 5 minutos (según la especificación de la expresión de programación), la siguiente consulta utiliza los 15 minutos de datos más recientes de la tabla de origen y cualquier dato de más de 15 minutos de la tabla derivada y, a continuación, une los resultados para crear la vista combinada que tiene lo mejor de ambos mundos: la economía y la baja latencia mediante la lectura de los agregados precalculados más antiguos de la tabla derivada y la frescura de los agregados de la tabla de origen. para potenciar sus casos de uso de análisis en tiempo real.

Tenga en cuenta que este enfoque de unión tendrá una latencia de consulta ligeramente mayor en comparación con la consulta únicamente de la tabla derivada y también escaneará datos ligeramente más altos, ya que agrega los datos sin procesar en tiempo real para completar el intervalo de tiempo más reciente. Sin embargo, esta vista combinada seguirá siendo considerablemente más rápida y económica en comparación con la agregación sobre la marcha desde la tabla de origen, especialmente en el caso de los cuadros de mando que representan días o semanas de datos. Puede ajustar los intervalos de tiempo de este ejemplo para adaptarlos a las necesidades de actualización y a la tolerancia al retraso de su aplicación.

```
WITH aggregated_source_data AS (
    SELECT bin(time, 1m) as minute, SUM(CASE WHEN measure_name = 'metrics' THEN 20 ELSE
 5 END) as numDatapoints
    FROM "raw_data"."devops"
    WHERE time BETWEEN bin(from_milliseconds(1636743196439), 1m) - 15m AND
 from_milliseconds(1636743196439)
        AND region = 'us-east-1'
    GROUP BY bin(time, 1m)
), aggregated_derived_data AS (
    SELECT bin(time, 1m) as minute, SUM(numDataPoints) as numDatapoints
    FROM "derived"."per_minute_aggs_pt5m"
    WHERE time BETWEEN from_milliseconds(1636699996439) AND
 bin(from_milliseconds(1636743196439), 1m) - 15m
        AND region = 'us-east-1'
    GROUP BY bin(time, 1m)
)
SELECT minute, numDatapoints
FROM (
    (
    SELECT *
    FROM aggregated_derived_data
    )
```

```
    UNION
    (
    SELECT *
    FROM aggregated_source_data
    )
)
ORDER BY 1 desc
```

A continuación se muestra el panel del panel de control con esta vista unificada y combinada. Como puede ver, el cuadro de mando tiene un aspecto casi idéntico al de la vista calculada a partir de la tabla derivada, con la salvedad de que tendrá la mayor up-to-date cantidad de agregados en el extremo derecho.



Sumado a partir de cálculos programados que se actualizan con frecuencia

Según la frecuencia con la que se carguen los paneles y la latencia que desee para su panel, existe otro método para obtener resultados más actualizados en el panel: hacer que el cálculo programado actualice los agregados con más frecuencia. Por ejemplo, a continuación se muestra la configuración del mismo cálculo programado, excepto que se actualiza una vez cada minuto (observe el horario express cron (0/1 * * * *? *)). Con esta configuración, la tabla derivada per_minute_aggs_pt1m tendrá agregados mucho más recientes en comparación con el escenario en el que el cálculo especificaba un programa de actualización de una vez cada 5 minutos.

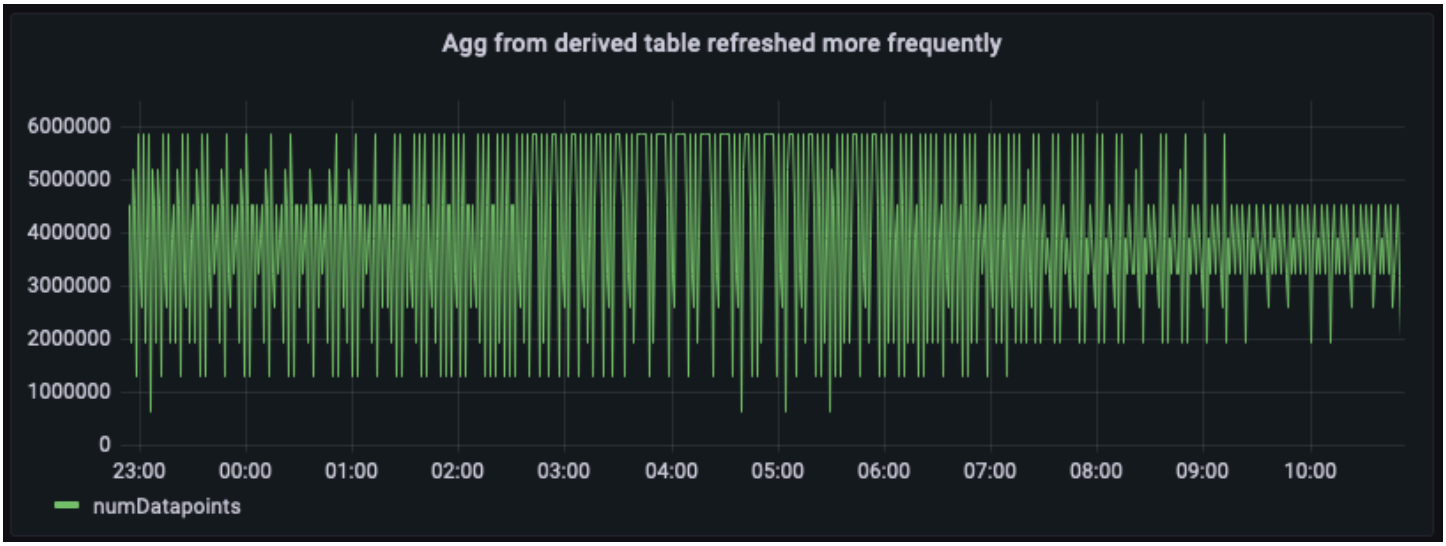
```
{
    "Name": "MultiPT1mPerMinutePerRegionMeasureCount",
    "QueryString": "SELECT region, bin(time, 1m) as minute, SUM(CASE WHEN measure_name
 = 'metrics' THEN 20 ELSE 5 END) as numDataPoints FROM raw_data.devops WHERE time
```

```
 BETWEEN @scheduled_runtime - 10m AND @scheduled_runtime + 1m GROUP BY bin(time, 1m),
 region",
    "ScheduleConfiguration": {
        "ScheduleExpression": "cron(0/1 * * * ? *)"
    },
    "NotificationConfiguration": {
        "SnsConfiguration": {
            "TopicArn": "******"
        }
    },
    "TargetConfiguration": {
        "TimestreamConfiguration": {
            "DatabaseName": "derived",
            "TableName": "per_minute_aggs_pt1m",
            "TimeColumn": "minute",
            "DimensionMappings": [
                {
                    "Name": "region",
                    "DimensionValueType": "VARCHAR"
                }
            ],
            "MultiMeasureMappings": {
                "TargetMultiMeasureName": "numDataPoints",
                "MultiMeasureAttributeMappings": [
                    {
                        "SourceColumn": "numDataPoints",
                        "MeasureValueType": "BIGINT"
                    }
                ]
            }
        }
    },
    "ErrorReportConfiguration": {
        "S3Configuration" : {
            "BucketName" : "******",
            "ObjectKeyPrefix": "errors",
            "EncryptionOption": "SSE_S3"
        }
    },
    "ScheduledQueryExecutionRoleArn": "******"
}
```

```
SELECT bin(time, 1m) as minute, SUM(numDataPoints) as numDatapoints
```

```
FROM "derived"."per_minute_aggs_pt1m"
WHERE time BETWEEN from_milliseconds(1636699996446) AND
 from_milliseconds(1636743196446)
    AND region = 'us-east-1'
GROUP BY bin(time, 1m), region
ORDER BY 1 desc
```

Como la tabla derivada tiene agregados más recientes, ahora puede consultarla directamente per_minute_aggs_pt1m para obtener agregados más actualizados, como puede verse en la consulta anterior y en la instantánea del panel de control que aparece a continuación.



Tenga en cuenta que actualizar el cálculo programado con un horario más rápido (por ejemplo, 1 minuto en lugar de 5 minutos) aumentará los costos de mantenimiento del cálculo programado. El mensaje de notificación de cada ejecución de un cálculo proporciona estadísticas sobre la cantidad de datos que se escanearon y la cantidad de datos que se escribieron en la tabla derivada. Del mismo modo, si usa la vista combinada para unir la tabla derivada, consulta los costos en la vista combinada y la latencia de carga del panel será mayor en comparación con solo consultar la tabla derivada. Por lo tanto, el enfoque que elija dependerá de la frecuencia con la que se actualicen sus paneles y de los costes de mantenimiento de las consultas programadas. Si tiene decenas de usuarios que actualizan los paneles aproximadamente una vez cada minuto, actualizar la tabla derivada con más frecuencia probablemente se traduzca en una reducción general de los costes.

El último punto de cada dispositivo

Es posible que su aplicación requiera que lea la última medición emitida por un dispositivo. Puede haber casos de uso más generales para obtener la última medición de un dispositivo antes de una determinadadate/time or the first measurement for a device after a given date/time. Si tiene

millones de dispositivos y años de datos, es posible que esta búsqueda requiera escanear grandes cantidades de datos.

A continuación, verás un ejemplo de cómo puedes usar las consultas programadas para optimizar la búsqueda del último punto emitido por un dispositivo. También puedes usar el mismo patrón para optimizar la consulta del primer punto si tu aplicación lo necesita.

Temas

Calculado a partir de la tabla fuente

A continuación, se muestra un ejemplo de consulta para encontrar la última medición emitida por los servicios en una implementación específica (por ejemplo, los servidores de un microservicio determinado dentro de una región, celda, silo y zona de disponibilidad determinados). En la aplicación de ejemplo, esta consulta devolverá la última medición de cientos de servidores. Tenga en cuenta también que esta consulta tiene un predicado de tiempo ilimitado y busca cualquier dato anterior a una marca de tiempo determinada.

> **Note**
>
> Para obtener información sobre las funciones y, consulte`max`. `max_by` Funciones de agregación

```
SELECT instance_name, MAX(time) AS time, MAX_BY(gc_pause, time) AS last_measure
FROM "raw_data"."devops"
WHERE time < from_milliseconds(1636685271872)
    AND measure_name = 'events'
    AND region = 'us-east-1'
    AND cell = 'us-east-1-cell-10'
    AND silo = 'us-east-1-cell-10-silo-3'
    AND availability_zone = 'us-east-1-1'
    AND microservice_name = 'hercules'
GROUP BY region, cell, silo, availability_zone, microservice_name,
    instance_name, process_name, jdk_version
```

```
ORDER BY instance_name, time DESC
```

Tabla derivada para precalcular con una granularidad diaria

Puede convertir el caso de uso anterior en un cálculo programado. Si los requisitos de su aplicación son tales que tal vez necesite obtener estos valores para toda su flota en varias regiones, celdas, silos, zonas de disponibilidad y microservicios, puede utilizar un cálculo programado para calcular previamente los valores de toda su flota. Esa es la potencia de las consultas programadas sin servidor de Timestream, que permite que estas consultas se escalen según los requisitos de escalado de su aplicación. LiveAnalytics

A continuación, se muestra una consulta para calcular previamente el último punto en todos los servidores de un día determinado. Tenga en cuenta que la consulta solo tiene un predicado de tiempo y no un predicado sobre las dimensiones. El predicado de tiempo limita la consulta al día anterior a la hora en que se activa el cálculo en función de la expresión de programación especificada.

```
SELECT region, cell, silo, availability_zone, microservice_name,
    instance_name, process_name, jdk_version,
    MAX(time) AS time, MAX_BY(gc_pause, time) AS last_measure
FROM raw_data.devops
WHERE time BETWEEN bin(@scheduled_runtime, 1d) - 1d AND bin(@scheduled_runtime, 1d)
    AND measure_name = 'events'
GROUP BY region, cell, silo, availability_zone, microservice_name,
    instance_name, process_name, jdk_version
```

A continuación, se muestra una configuración para el cálculo programado que utiliza la consulta anterior, que ejecuta esa consulta UTC todos los días a la 01:00 horas para calcular la suma del día anterior. La expresión de programación cron (0 1 * *? *) controla este comportamiento y se ejecuta una hora después del final del día para tener en cuenta que los datos llegan con un día de retraso.

```
{
    "Name": "PT1DPerInstanceLastpoint",
    "QueryString": "SELECT region, cell, silo, availability_zone, microservice_name,
 instance_name, process_name, jdk_version, MAX(time) AS time, MAX_BY(gc_pause, time)
 AS last_measure FROM raw_data.devops WHERE time BETWEEN bin(@scheduled_runtime, 1d) -
 1d AND bin(@scheduled_runtime, 1d) AND measure_name = 'events' GROUP BY region, cell,
 silo, availability_zone, microservice_name, instance_name, process_name, jdk_version",
    "ScheduleConfiguration": {
        "ScheduleExpression": "cron(0 1 * * ? *)"
    },
```

```
    "NotificationConfiguration": {
        "SnsConfiguration": {
            "TopicArn": "******"
        }
    },
    "TargetConfiguration": {
        "TimestreamConfiguration": {
            "DatabaseName": "derived",
            "TableName": "per_timeseries_lastpoint_pt1d",
            "TimeColumn": "time",
            "DimensionMappings": [
                {
                    "Name": "region",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "cell",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "silo",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "availability_zone",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "microservice_name",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "instance_name",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "process_name",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "jdk_version",
                    "DimensionValueType": "VARCHAR"
                }
            ],
```

```
            "MultiMeasureMappings": {
                "TargetMultiMeasureName": "last_measure",
                "MultiMeasureAttributeMappings": [
                    {
                        "SourceColumn": "last_measure",
                        "MeasureValueType": "DOUBLE"
                    }
                ]
            }
        }
    },
    "ErrorReportConfiguration": {
        "S3Configuration" : {
            "BucketName" : "******",
            "ObjectKeyPrefix": "errors",
            "EncryptionOption": "SSE_S3"
        }
    },
    "ScheduledQueryExecutionRoleArn": "******"
}
```

Calculado a partir de una tabla derivada

Una vez que haya definido la tabla derivada mediante la configuración anterior y al menos una instancia de la consulta programada haya materializado los datos en la tabla derivada, ahora puede consultar la tabla derivada para obtener la última medición. A continuación, se muestra un ejemplo de consulta en la tabla derivada.

```
SELECT instance_name, MAX(time) AS time, MAX_BY(last_measure, time) AS last_measure
FROM "derived"."per_timeseries_lastpoint_pt1d"
WHERE time < from_milliseconds(1636746715649)
    AND measure_name = 'last_measure'
    AND region = 'us-east-1'
    AND cell = 'us-east-1-cell-10'
    AND silo = 'us-east-1-cell-10-silo-3'
    AND availability_zone = 'us-east-1-1'
    AND microservice_name = 'hercules'
GROUP BY region, cell, silo, availability_zone, microservice_name,
    instance_name, process_name, jdk_version
ORDER BY instance_name, time DESC
```

Combinación de la fuente y la tabla derivada

Al igual que en el ejemplo anterior, los datos de la tabla derivada no tendrán las escrituras más recientes. Por lo tanto, puede volver a utilizar un patrón similar al anterior para combinar los datos de la tabla derivada con los datos más antiguos y utilizar los datos de origen para la punta restante. A continuación se muestra un ejemplo de una consulta de este tipo que utiliza un UNION enfoque similar. Como el requisito de la aplicación es encontrar la última medición antes de un período de tiempo, y esta hora de inicio puede ser pasada, la forma de escribir esta consulta consiste en utilizar la hora proporcionada, utilizar los datos de origen con una antigüedad máxima de un día a partir de la hora especificada y, a continuación, utilizar la tabla derivada en los datos más antiguos. Como puede ver en el ejemplo de consulta que aparece a continuación, el predicado temporal de los datos de origen está limitado. Esto garantiza un procesamiento eficiente en la tabla de origen, que tiene un volumen de datos significativamente mayor, y luego el predicado de tiempo ilimitado se encuentra en la tabla derivada.

```
WITH last_point_derived AS (
    SELECT instance_name, MAX(time) AS time, MAX_BY(last_measure, time) AS last_measure
    FROM "derived"."per_timeseries_lastpoint_pt1d"
    WHERE time < from_milliseconds(1636746715649)
        AND measure_name = 'last_measure'
        AND region = 'us-east-1'
        AND cell = 'us-east-1-cell-10'
        AND silo = 'us-east-1-cell-10-silo-3'
        AND availability_zone = 'us-east-1-1'
        AND microservice_name = 'hercules'
    GROUP BY region, cell, silo, availability_zone, microservice_name,
        instance_name, process_name, jdk_version
), last_point_source AS (
    SELECT instance_name, MAX(time) AS time, MAX_BY(gc_pause, time) AS last_measure
    FROM "raw_data"."devops"
    WHERE time < from_milliseconds(1636746715649) AND time >
 from_milliseconds(1636746715649) - 26h
        AND measure_name = 'events'
        AND region = 'us-east-1'
        AND cell = 'us-east-1-cell-10'
        AND silo = 'us-east-1-cell-10-silo-3'
        AND availability_zone = 'us-east-1-1'
        AND microservice_name = 'hercules'
    GROUP BY region, cell, silo, availability_zone, microservice_name,
        instance_name, process_name, jdk_version
)
SELECT instance_name, MAX(time) AS time, MAX_BY(last_measure, time) AS last_measure
```

```
FROM (
    SELECT * FROM last_point_derived
    UNION
    SELECT * FROM last_point_source
)
GROUP BY instance_name
ORDER BY instance_name, time DESC
```

La anterior es solo una ilustración de cómo se pueden estructurar las tablas derivadas. Si tiene años de datos, puede usar más niveles de agregaciones. Por ejemplo, puede tener agregados mensuales además de agregados diarios y puede tener agregados horarios antes que los diarios. De este modo, puedes combinar los datos más recientes para rellenar la última hora, los horarios para rellenar el último día, los diarios para rellenar el último mes y los mensuales para rellenar los datos anteriores. El número de niveles que configure, en comparación con el programa de actualización, dependerá de sus requisitos en cuanto a la frecuencia con la que se produzcan estas consultas y del número de usuarios que las emitan simultáneamente.

Valores de dimensión únicos

Es posible que tenga un caso de uso en el que tenga paneles en los que desee utilizar los valores únicos de las dimensiones como variables para desglosar las métricas correspondientes a un segmento de datos específico. La siguiente instantánea es un ejemplo en el que el panel rellena previamente los valores únicos de varias dimensiones, como la región, la celda, el silo, el microservicio y la zona de disponibilidad. A continuación, mostramos un ejemplo de cómo puede utilizar las consultas programadas para acelerar considerablemente el cálculo de estos valores distintos de estas variables a partir de las métricas que está rastreando.

Temas

- Sobre datos sin procesar
- Calcule previamente los valores de dimensión únicos
- Calcular las variables de la tabla derivada

Sobre datos sin procesar

Se puede utilizar SELECT DISTINCT para calcular los distintos valores que se ven en los datos. Por ejemplo, si desea obtener los distintos valores de la región, puede utilizar la consulta de este formulario.

```
SELECT DISTINCT region
```

```
FROM "raw_data"."devops"
WHERE time > ago(1h)
ORDER BY 1
```

Es posible que esté rastreando millones de dispositivos y miles de millones de series temporales. Sin embargo, en la mayoría de los casos, estas variables interesantes son para las dimensiones de cardinalidad inferior, en las que hay de unos pocos a decenas de valores. La computación DISTINCT a partir de datos sin procesar puede requerir el escaneo de grandes volúmenes de datos.

Calcule previamente los valores de dimensión únicos

Desea que estas variables se carguen rápidamente para que sus paneles sean interactivos. Además, estas variables suelen calcularse cada vez que se carga el panel, por lo que también es recomendable que sean rentables. Puede optimizar la búsqueda de estas variables mediante consultas programadas y su materialización en una tabla derivada.

En primer lugar, debe identificar las dimensiones para las que debe calcular los DISTINCT valores o columnas que utilizará en los predicados al calcular el DISTINCT valor.

En este ejemplo, puede ver que el panel rellena valores distintos para las dimensiones region, cell, silo, availability_zone y microservice. Por lo tanto, puede utilizar la siguiente consulta para calcular previamente estos valores únicos.

```
SELECT region, cell, silo, availability_zone, microservice_name,
    min(@scheduled_runtime) AS time, COUNT(*) as numDataPoints
FROM raw_data.devops
WHERE time BETWEEN @scheduled_runtime - 15m AND @scheduled_runtime
GROUP BY region, cell, silo, availability_zone, microservice_name
```

Hay algunas cosas importantes que hay que tener en cuenta a este respecto.

- Puede usar un cálculo programado para calcular previamente los valores de muchas consultas diferentes. Por ejemplo, está utilizando la consulta anterior para calcular previamente los valores de cinco variables diferentes. Por lo tanto, no necesita uno para cada variable. Puede usar este mismo patrón para identificar el cómputo compartido en varios paneles a fin de optimizar la cantidad de consultas programadas que debe mantener.
- Los valores únicos de las dimensiones no son intrínsecamente datos de series temporales. Así que conviertes esto en series temporales usando @scheduled_runtime. Al asociar estos datos al parámetro @scheduled_runtime, también puede rastrear qué valores únicos aparecieron en un momento dado, creando así datos de series temporales a partir de ellos.

- En el ejemplo anterior, verá que se está realizando un seguimiento de un valor métrico. En este ejemplo se usa COUNT (*). Puede calcular otros agregados significativos si quiere hacer un seguimiento de ellos para sus paneles.

A continuación, se muestra una configuración para un cálculo programado mediante la consulta anterior. En este ejemplo, se configura para que se actualice una vez cada 15 minutos mediante la expresión de programación cron (0/15 * * *? *).

```
{
    "Name": "PT15mHighCardPerUniqueDimensions",
    "QueryString": "SELECT region, cell, silo, availability_zone, microservice_name,
 min(@scheduled_runtime) AS time, COUNT(*) as numDataPoints FROM raw_data.devops WHERE
 time BETWEEN @scheduled_runtime - 15m AND @scheduled_runtime GROUP BY region, cell,
 silo, availability_zone, microservice_name",
    "ScheduleConfiguration": {
        "ScheduleExpression": "cron(0/15 * * * ? *)"
    },
    "NotificationConfiguration": {
        "SnsConfiguration": {
            "TopicArn": "******"
        }
    },
    "TargetConfiguration": {
        "TimestreamConfiguration": {
            "DatabaseName": "derived",
            "TableName": "hc_unique_dimensions_pt15m",
            "TimeColumn": "time",
            "DimensionMappings": [
                {
                    "Name": "region",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "cell",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "silo",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "availability_zone",
```

```
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "microservice_name",
                    "DimensionValueType": "VARCHAR"
                }
            ],
            "MultiMeasureMappings": {
                "TargetMultiMeasureName": "count_multi",
                "MultiMeasureAttributeMappings": [
                    {
                        "SourceColumn": "numDataPoints",
                        "MeasureValueType": "BIGINT"
                    }
                ]
            }
        }
    },
    "ErrorReportConfiguration": {
        "S3Configuration" : {
            "BucketName" : "******",
            "ObjectKeyPrefix": "errors",
            "EncryptionOption": "SSE_S3"
        }
    },
    "ScheduledQueryExecutionRoleArn": "******"
}
```

Calcular las variables de la tabla derivada

Una vez que el cálculo programado prematerialice los valores únicos de la tabla derivada hc_unique_dimensions_pt15m, puede utilizar la tabla derivada para calcular de forma eficiente los valores únicos de las dimensiones. A continuación, se muestran ejemplos de consultas sobre cómo calcular los valores únicos y cómo utilizar otras variables como predicados en estas consultas de valores únicos.

Region

```
SELECT DISTINCT region
FROM "derived"."hc_unique_dimensions_pt15m"
WHERE time > ago(1h)
ORDER BY 1
```

Celda

```
SELECT DISTINCT cell
FROM "derived"."hc_unique_dimensions_pt15m"
WHERE time > ago(1h)
  AND region = '${region}'
ORDER BY 1
```

Silo

```
SELECT DISTINCT silo
FROM "derived"."hc_unique_dimensions_pt15m"
WHERE time > ago(1h)
   AND region = '${region}' AND cell = '${cell}'
ORDER BY 1
```

Microservicio

```
SELECT DISTINCT microservice_name
FROM "derived"."hc_unique_dimensions_pt15m"
WHERE time > ago(1h)
   AND region = '${region}' AND cell = '${cell}'
ORDER BY 1
```

Zona de disponibilidad

```
SELECT DISTINCT availability_zone
FROM "derived"."hc_unique_dimensions_pt15m"
WHERE time > ago(1h)
   AND region = '${region}' AND cell = '${cell}' AND silo = '${silo}'
ORDER BY 1
```

Gestión de los datos que llegan tarde

Es posible que haya situaciones en las que los datos lleguen con bastante retraso; por ejemplo, el momento en que se ingresaron los datos en Timestream se retrasa considerablemente en comparación con la marca de tiempo asociada a las filas que se ingieren. LiveAnalytics En los ejemplos anteriores, ha visto cómo puede utilizar los intervalos de tiempo definidos por el parámetro @scheduled_runtime para tener en cuenta algunos datos que llegan tarde. Sin embargo, si tiene casos de uso en los que los datos pueden retrasarse horas o días, es posible que necesite un patrón diferente para asegurarse de que los cálculos previos de la tabla derivada se actualizan

adecuadamente para reflejar los datos que llegan tarde. Para obtener información general sobre los datos que llegan tarde, consulte. Escribir datos (inserciones y ajustes)

A continuación, verá dos formas diferentes de abordar estos datos que llegan tarde.

- Si tiene retrasos predecibles en la llegada de los datos, puede utilizar otro cálculo programado para ponerse al día para actualizar los agregados en función de los datos que lleguen tarde.
- Si tiene retrasos impredecibles o, en ocasiones, los datos llegan tarde, puede utilizar las ejecuciones manuales para actualizar las tablas derivadas.

En este análisis se describen los escenarios de llegada tardía de los datos. Sin embargo, los mismos principios se aplican a las correcciones de datos, en las que se modifican los datos de la tabla de origen y se desean actualizar los agregados de las tablas derivadas.

Temas

- Consultas de puesta al día programadas
- Ejecuciones manuales para datos impredecibles que llegan tarde

Consultas de puesta al día programadas

Consulta agregando datos que llegaron a tiempo

A continuación, se muestra un patrón en el que se muestra cómo se pueden utilizar de forma automática para actualizar los agregados en caso de que se produzcan retrasos predecibles en la llegada de los datos. Considera uno de los ejemplos anteriores de un cálculo programado con datos en tiempo real que se muestran a continuación. Este cálculo programado actualiza la tabla derivada una vez cada 30 minutos y ya tiene en cuenta los datos con un retraso de hasta una hora.

```
{
    "Name": "MultiPT30mPerHrPerTimeseriesDPCount",
    "QueryString": "SELECT region, cell, silo, availability_zone, microservice_name,
 instance_type, os_version, instance_name, process_name, jdk_version, bin(time,
 1h) as hour, SUM(CASE WHEN measure_name = 'metrics' THEN 20 ELSE 5 END) as
 numDataPoints FROM raw_data.devops WHERE time BETWEEN bin(@scheduled_runtime, 1h)
 - 1h AND @scheduled_runtime + 1h GROUP BY region, cell, silo, availability_zone,
 microservice_name, instance_type, os_version, instance_name, process_name,
 jdk_version, bin(time, 1h)",
    "ScheduleConfiguration": {
        "ScheduleExpression": "cron(0/30 * * * ? *)"
```

```
    },
    "NotificationConfiguration": {
        "SnsConfiguration": {
            "TopicArn": "******"
        }
    },
    "TargetConfiguration": {
        "TimestreamConfiguration": {
            "DatabaseName": "derived",
            "TableName": "dp_per_timeseries_per_hr",
            "TimeColumn": "hour",
            "DimensionMappings": [
                {
                    "Name": "region",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "cell",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "silo",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "availability_zone",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "microservice_name",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "instance_type",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "os_version",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "instance_name",
                    "DimensionValueType": "VARCHAR"
                },
```

```
                {
                    "Name": "process_name",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "jdk_version",
                    "DimensionValueType": "VARCHAR"
                }
            ],
            "MultiMeasureMappings": {
                "TargetMultiMeasureName": "numDataPoints",
                "MultiMeasureAttributeMappings": [
                    {
                        "SourceColumn": "numDataPoints",
                        "MeasureValueType": "BIGINT"
                    }
                ]
            }
        }
    },
    "ErrorReportConfiguration": {
        "S3Configuration" : {
            "BucketName" : "******",
            "ObjectKeyPrefix": "errors",
            "EncryptionOption": "SSE_S3"
        }
    },
    "ScheduledQueryExecutionRoleArn": "******"
}
```

Consulta de puesta al día que actualiza los agregados de los datos que llegan tarde

Ahora bien, si considera el caso de que sus datos pueden retrasarse unas 12 horas. A continuación se muestra una variante de la misma consulta. Sin embargo, la diferencia es que calcula los agregados a partir de los datos que se retrasan hasta 12 horas en comparación con el momento en que se activa el cálculo programado. Por ejemplo, si ve la consulta en el siguiente ejemplo, el intervalo de tiempo al que se dirige esta consulta es entre 2 y 14 horas antes de que se active la consulta. Además, si observa la expresión de programación cron (0, 0,12 * * *? *), activará el cálculo a las 00:00 UTC y a las 12:00 todos los días. UTC Por lo tanto, cuando la consulta se active el 01/12/2021 a las 00:00:00, la consulta actualizará los agregados en el intervalo de tiempo entre el 30 de noviembre de 2021 y las 10:00:00 del 30 de diciembre de 2021 hasta el 30 de noviembre de 2021 a las 22:00:00. Las consultas programadas utilizan una semántica ascendente similar a la

de Timestream para LiveAnalytics las escrituras, donde esta consulta de recuperación actualizará los valores agregados con valores más nuevos si hay datos que llegan tarde a la ventana o si se encuentran agregados más nuevos (por ejemplo, aparece una nueva agrupación en este agregado que no estaba presente cuando se activó el cálculo programado original), luego el nuevo agregado se insertará en la tabla derivada. Del mismo modo, cuando la siguiente instancia se active el 01/12/2021 a las 12:00:00, esa instancia actualizará los agregados en el rango de 2021-11-30 22:00:00 a 2021-12-01 10:00:00.

```
      {
   "Name": "MultiPT12HPerHrPerTimeseriesDPCountCatchUp",
   "QueryString": "SELECT region, cell, silo, availability_zone, microservice_name,
instance_type, os_version, instance_name, process_name, jdk_version, bin(time, 1h)
as hour, SUM(CASE WHEN measure_name = 'metrics' THEN 20 ELSE 5 END) as numDataPoints
FROM raw_data.devops WHERE time BETWEEN bin(@scheduled_runtime, 1h) - 14h AND
bin(@scheduled_runtime, 1h) - 2h GROUP BY region, cell, silo, availability_zone,
microservice_name, instance_type, os_version, instance_name, process_name,
jdk_version, bin(time, 1h)",
   "ScheduleConfiguration": {
       "ScheduleExpression": "cron(0 0,12 * * ? *)"
   },
   "NotificationConfiguration": {
       "SnsConfiguration": {
           "TopicArn": "******"
       }
   },
   "TargetConfiguration": {
       "TimestreamConfiguration": {
           "DatabaseName": "derived",
           "TableName": "dp_per_timeseries_per_hr",
           "TimeColumn": "hour",
           "DimensionMappings": [
               {
                   "Name": "region",
                   "DimensionValueType": "VARCHAR"
               },
               {
                   "Name": "cell",
                   "DimensionValueType": "VARCHAR"
               },
               {
                   "Name": "silo",
```

```
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "availability_zone",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "microservice_name",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "instance_type",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "os_version",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "instance_name",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "process_name",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "jdk_version",
                    "DimensionValueType": "VARCHAR"
                }
            ],
            "MultiMeasureMappings": {
                "TargetMultiMeasureName": "numDataPoints",
                "MultiMeasureAttributeMappings": [
                    {
                        "SourceColumn": "numDataPoints",
                        "MeasureValueType": "BIGINT"
                    }
                ]
            }
        }
    },
    "ErrorReportConfiguration": {
        "S3Configuration" : {
```

```
            "BucketName" : "******",
            "ObjectKeyPrefix": "errors",
            "EncryptionOption": "SSE_S3"
        }
    },
    "ScheduledQueryExecutionRoleArn": "******"
}
```

El ejemplo anterior es un ejemplo en el que se supone que su llegada tardía está limitada a 12 horas y que está bien actualizar la tabla derivada una vez cada 12 horas para que los datos lleguen más tarde del intervalo de tiempo real. Puede adaptar este patrón para actualizar la tabla derivada una vez cada hora, de modo que la tabla derivada refleje antes los datos que llegan tarde. Del mismo modo, puede adaptar el intervalo de tiempo para que tenga más de 12 horas, por ejemplo, un día o incluso una semana o más, para gestionar datos predecibles que lleguen tarde.

Ejecuciones manuales para datos impredecibles que llegan tarde

Puede haber casos en los que tenga datos impredecibles que lleguen tarde o en los que haya realizado cambios en los datos de origen y haya actualizado algunos valores posteriormente. En todos estos casos, puede activar manualmente consultas programadas para actualizar la tabla derivada. A continuación se muestra un ejemplo de cómo puede lograrlo.

Suponga que tiene el caso de uso en el que el cálculo está escrito en la tabla derivada dp_per_timeseries_per_hr. Los datos base de la tabla devops se actualizaron en el intervalo de tiempo 2021-11-30 23:00:00 - 2021-12-01 00:00:00. Hay dos consultas programadas diferentes que se pueden utilizar para actualizar esta tabla derivada: Multi 0 y Multi. PT3 mPerHr PerTimeseries DPCount PT12HPerHrPerTimeseriesDPCountCatchUp Cada cálculo programado para el que se crea en Timestream LiveAnalytics tiene un valor único ARN que se obtiene al crear el cálculo o al realizar una operación de lista. Para realizar esta operación, puede utilizar el valor ARN para el cálculo y un valor para el parámetro @scheduled_runtime utilizado por la consulta.

Suponga que el cálculo de Multi PT3 0 mPerHr PerTimeseries DPCount tiene un ARN arn_1 y desea utilizar este cálculo para actualizar la tabla derivada. Como el cálculo programado anterior actualiza los agregados 1 hora antes y 1 hora después del valor @scheduled_runtime, puede cubrir el intervalo de tiempo de la actualización (2021-11-30 23:00:00 - 2021-12-01 00:00:00) utilizando el valor 2021-12-01 00:00:00 para el parámetro @scheduled_runtime. Para ello, puede utilizar el para pasar este cálculo y el valor del ExecuteScheduledQuery API parámetro de tiempo en segundos ARN de época (pulgadas). UTC A continuación se muestra un ejemplo en el que se utiliza el patrón AWS CLI y se puede seguir el mismo patrón utilizando cualquiera de los SDKs compatibles con Timestream. LiveAnalytics

```
aws timestream-query execute-scheduled-query --scheduled-query-arn arn_1 --invocation-
time 1638316800 --profile profile --region us-east-1
```

En el ejemplo anterior, el perfil es el AWS perfil que tiene los privilegios adecuados para realizar esta API llamada y 1638316800 corresponde a la segunda época del 01/12/2021 00:00:00. Este activador manual se comporta casi igual que el disparador automático, suponiendo que el sistema haya activado esta invocación en el período de tiempo deseado.

Si realizó una actualización durante un período de tiempo más largo, supongamos que los datos base se actualizaron entre el 30 de noviembre de 2021 a las 23:00:00 y el 1 de diciembre de 2021 a las 11:00:00, puede activar las consultas anteriores varias veces para cubrir todo este intervalo de tiempo. Por ejemplo, puede realizar seis ejecuciones diferentes de la siguiente manera.

```
aws timestream-query execute-scheduled-query --scheduled-query-arn arn_1 --invocation-
time 1638316800 --profile profile --region us-east-1

aws timestream-query execute-scheduled-query --scheduled-query-arn arn_1 --invocation-
time 1638324000 --profile profile --region us-east-1

aws timestream-query execute-scheduled-query --scheduled-query-arn arn_1 --invocation-
time 1638331200 --profile profile --region us-east-1

aws timestream-query execute-scheduled-query --scheduled-query-arn arn_1 --invocation-
time 1638338400 --profile profile --region us-east-1

aws timestream-query execute-scheduled-query --scheduled-query-arn arn_1 --invocation-
time 1638345600 --profile profile --region us-east-1

aws timestream-query execute-scheduled-query --scheduled-query-arn arn_1 --invocation-
time 1638352800 --profile profile --region us-east-1
```

Los seis comandos anteriores corresponden al cálculo programado que se ejecutó el 2021-12-01 00:00:00, 2021-12-01 02:00:00, 2021-12-01 04:0:00, 2021-12-01 06:00:00, 2021-12-01 08:00:00, y 2021-12-01 10:00:

Como alternativa, puede utilizar el cálculo Multi, que se activa el 1 de diciembre de 2021 a las 13:00:00 para una ejecución, a fin de actualizar los agregados para todo el intervalo de tiempo de 12 horas. PT12HPerHrPerTimeseriesDPCountCatchUp Por ejemplo, si arn_2 es el valor de ese cálculo, ARN puede ejecutar el siguiente comando desde. CLI

```
aws timestream-query execute-scheduled-query --scheduled-query-arn arn_2 --invocation-
time 1638363600 --profile profile --region us-east-1
```

Vale la pena señalar que, en el caso de un disparador manual, puede utilizar una marca de tiempo para el parámetro de tiempo de invocación que no necesite estar alineada con las marcas de tiempo de ese disparador automático. Por ejemplo, en el ejemplo anterior, activaste el cálculo el 1 de diciembre de 2021 a las 13:00:00 aunque la programación automática solo se activa en las marcas horarias 2021-12-01 10:00:00, 2021-12-01 12:00:00, y 2021-12-02 00:00:00. Timestream for le ofrece la flexibilidad de activarlo con los valores adecuados según sea necesario para sus operaciones manuales. LiveAnalytics

A continuación se presentan algunas consideraciones importantes a la hora de utilizar el. ExecuteScheduledQuery API

- Si va a activar varias de estas invocaciones, debe asegurarse de que estas invocaciones no generen resultados en intervalos de tiempo superpuestos. Por ejemplo, en los ejemplos anteriores, hubo seis invocaciones. Cada invocación abarca un intervalo de tiempo de 2 horas y, por lo tanto, las marcas de tiempo de la invocación se distribuyeron en dos horas cada una para evitar cualquier superposición en las actualizaciones. Esto garantiza que los datos de la tabla derivada terminen en un estado en el que las coincidencias sean agregados de la tabla de origen. Si no puede garantizar que los intervalos de tiempo no se superpongan, asegúrese de que las ejecuciones se activen secuencialmente una tras otra. Si desencadena varias ejecuciones de forma simultánea y se superponen en sus intervalos de tiempo, en los informes de errores correspondientes a estas ejecuciones puede haber errores de activación, por lo que es posible que aparezcan conflictos de versiones. A los resultados generados por una invocación de consulta programada se les asigna una versión en función del momento en que se activó la invocación. Por lo tanto, las filas generadas por las invocaciones más recientes tienen versiones superiores. Un registro de versión superior puede sobrescribir un registro de versión inferior. En el caso de las consultas programadas que se activan automáticamente, Timestream for administra LiveAnalytics automáticamente las programaciones para que no se produzcan estos problemas incluso si las siguientes invocaciones tienen intervalos de tiempo superpuestos.
- Como mencionamos anteriormente, puedes activar las invocaciones con cualquier valor de marca de tiempo para @scheduled_runtime. Por lo tanto, es su responsabilidad establecer los valores de manera adecuada para que los intervalos de tiempo adecuados se actualicen en la tabla derivada correspondiente a los rangos en los que se actualizaron los datos en la tabla de origen.
- También puede utilizar estos activadores manuales para consultas programadas que estén en ese DISABLED estado. Esto le permite definir consultas especiales que no se ejecutan de forma

automática, ya que se encuentran en ese DISABLED estado. Por el contrario, puede utilizar sus activadores manuales para gestionar las correcciones de datos o los casos prácticos de llegadas tardías.

Rellenar los cálculos previos históricos

Al crear un cálculo programado, Timestream for LiveAnalytics gestiona las ejecuciones de las consultas en el futuro, mientras que la actualización se rige por la expresión de programación que proporcione. En función de la cantidad de datos históricos de la tabla de origen, es posible que desee actualizar la tabla derivada con los agregados correspondientes a los datos históricos. Puede utilizar la lógica anterior para los activadores manuales a fin de rellenar los agregados históricos.

Por ejemplo, si consideramos la tabla derivada per_timeseries_lastpoint_pt1d, el cálculo programado se actualiza una vez al día para el día anterior. Si la tabla de origen contiene datos de un año, puede utilizarla ARN para este cálculo programado y activarla manualmente para todos los días de hasta un año de antigüedad, de modo que la tabla derivada tenga rellenadas todas las consultas históricas. Observa que aquí se aplican todas las advertencias sobre los activadores manuales. Además, si la tabla derivada está configurada de manera que la ingesta histórica se grabe en el almacén magnético de la tabla derivada, conozca las mejores prácticas y los límites para escribir en el almacén magnético.

Ejemplos de consultas programadas

Esta sección contiene ejemplos de cómo puede utilizar las consultas programadas de Timestream for para optimizar los costes y los tiempos de carga LiveAnalytics del panel de control a la hora de visualizar las estadísticas de toda la flota y supervisar eficazmente su flota de dispositivos. Las consultas programadas en Timestream LiveAnalytics le permiten expresar sus consultas utilizando toda la superficie de Timestream. SQL LiveAnalytics La consulta puede incluir una o más tablas de origen, realizar agregaciones o cualquier otra consulta que permita Timestream para el SQL idioma de destino y, a continuación, almacenar los resultados LiveAnalytics de la consulta en otra tabla de destino de Timestream for. LiveAnalytics

En esta sección, se hace referencia a la tabla de destino de una consulta programada como tabla derivada.

Por ejemplo, utilizaremos una DevOps aplicación en la que supervise una gran flota de servidores que se desplieguen en varios despliegues (como regiones, celdas y silos) y varios microservicios, y usted haga un seguimiento de las estadísticas de toda la flota mediante Timestream. LiveAnalytics

El esquema de ejemplo que utilizaremos se describe en el Esquema de muestra de consultas programadas.

Se describirán los siguientes escenarios.

- Cómo convertir un cuadro de mando, graficando las estadísticas agregadas a partir de los datos sin procesar que se introducen en Timestream en una consulta programada y, a continuación, cómo utilizar los agregados precalculados para crear un nuevo cuadro de mando que muestre las estadísticas agregadas. LiveAnalytics
- Cómo combinar las consultas programadas para obtener una vista agregada y los datos granulares sin procesar para profundizar en los detalles. Esto le permite almacenar y analizar los datos sin procesar y, al mismo tiempo, optimizar las operaciones habituales de toda la flota mediante consultas programadas.
- Cómo optimizar los costes mediante consultas programadas buscando qué agregados se utilizan en varios paneles y hacer que la misma consulta programada rellene varios paneles del mismo panel o de varios paneles.

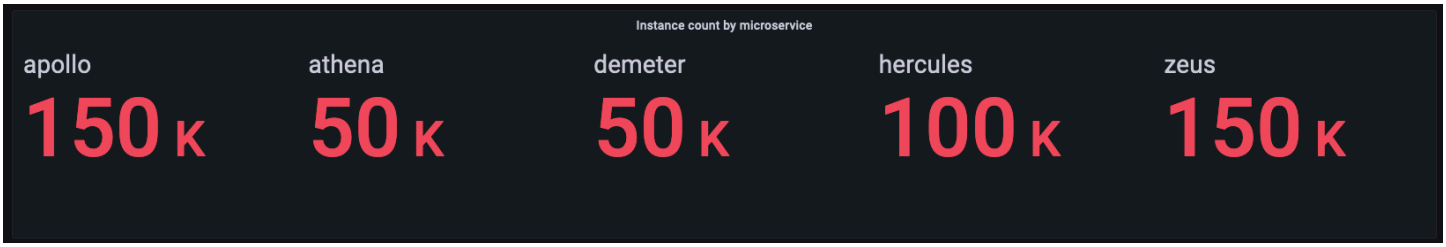
Temas

- Convertir un panel agregado en una consulta programada
- Uso de consultas programadas y datos sin procesar para los desgloses
- Optimizar los costos al compartir las consultas programadas en todos los paneles
- Comparación de una consulta de una tabla base con una consulta de los resultados de una consulta programada

Convertir un panel agregado en una consulta programada

Suponga que está calculando las estadísticas de toda la flota, como el número de anfitriones de la flota en función de los cinco microservicios y las seis regiones en las que está desplegado su servicio. En la siguiente instantánea, puedes ver que hay 500 000 servidores que emiten métricas y que algunas de las regiones más grandes (por ejemplo, us-east-1) tienen más de 200 000 servidores.

Al calcular estos agregados, en los que se calculan nombres de instancias distintos en cientos de gigabytes de datos, se puede producir una latencia de consulta de decenas de segundos, además del coste de digitalización de los datos.



Consulta original del panel

El agregado que se muestra en el panel del tablero se calcula a partir de datos sin procesar mediante la consulta siguiente. La consulta utiliza varios SQL constructos, como recuentos distintos y múltiples funciones de agregación.

```
SELECT CASE WHEN microservice_name = 'apollo' THEN num_instances ELSE NULL END AS
 apollo,
    CASE WHEN microservice_name = 'athena' THEN num_instances ELSE NULL END AS athena,
    CASE WHEN microservice_name = 'demeter' THEN num_instances ELSE NULL END AS
 demeter,
    CASE WHEN microservice_name = 'hercules' THEN num_instances ELSE NULL END AS
 hercules,
    CASE WHEN microservice_name = 'zeus' THEN num_instances ELSE NULL END AS zeus
FROM (
    SELECT microservice_name, SUM(num_instances) AS num_instances
    FROM (
        SELECT microservice_name, COUNT(DISTINCT instance_name) as num_instances
        FROM "raw_data"."devops"
        WHERE time BETWEEN from_milliseconds(1636526171043) AND
 from_milliseconds(1636612571043)
            AND measure_name = 'metrics'
        GROUP BY region, cell, silo, availability_zone, microservice_name
    )
    GROUP BY microservice_name
)
```

Se convierte en una consulta programada

La consulta anterior se puede convertir en una consulta programada de la siguiente manera. Primero debe calcular los distintos nombres de host de una implementación determinada en una región, celda, silo, zona de disponibilidad y microservicio. A continuación, se suman los hosts para calcular un recuento por hora y por microservicio. Al utilizar el `@scheduled_runtime` parámetro compatible con las consultas programadas, puede volver a calcularlo para la última hora en que se invoque la consulta. La `bin(@scheduled_runtime, 1h)` `WHERE` cláusula de la consulta interna garantiza

que, incluso si la consulta está programada a media hora, se seguirán obteniendo los datos de toda la hora.

Aunque la consulta calcula los agregados por hora, como se verá en la configuración de cálculo programado, está configurada para que se actualice cada media hora, de modo que pueda recibir las actualizaciones en la tabla derivada con mayor rapidez. Puede ajustarlo en función de sus requisitos de actualización, por ejemplo, volver a calcular los agregados cada 15 minutos o volver a calcularlos según los límites horarios.

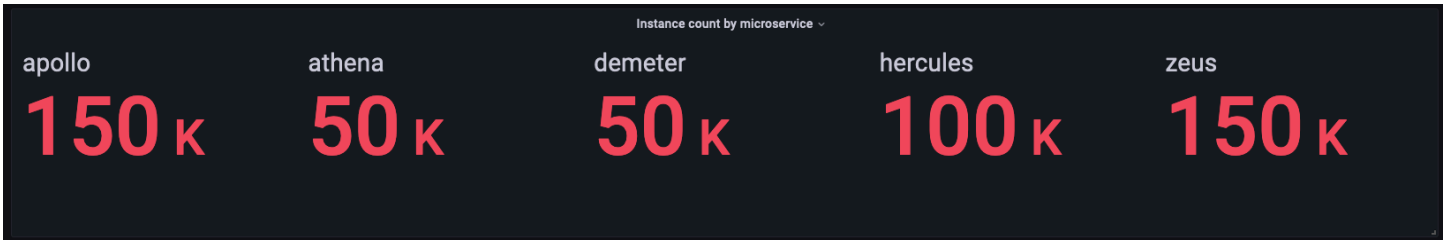
```
SELECT microservice_name, hour, SUM(num_instances) AS num_instances
FROM (
        SELECT microservice_name, bin(time, 1h) AS hour,
            COUNT(DISTINCT instance_name) as num_instances
       FROM raw_data.devops
       WHERE time BETWEEN bin(@scheduled_runtime, 1h) - 1h AND @scheduled_runtime

           AND measure_name = 'metrics'
       GROUP BY region, cell, silo, availability_zone, microservice_name, bin(time, 1h)

     )
GROUP BY microservice_name, hour
```

```
{
    "Name": "MultiPT30mHostCountMicroservicePerHr",
    "QueryString": "SELECT microservice_name, hour, SUM(num_instances) AS num_instances
    FROM (        SELECT microservice_name, bin(time, 1h) AS hour, COUNT(DISTINCT
 instance_name) as num_instances        FROM raw_data.devops        WHERE time BETWEEN
 bin(@scheduled_runtime, 1h) - 1h AND @scheduled_runtime            AND measure_name
 = 'metrics'        GROUP BY region, cell, silo, availability_zone, microservice_name,
 bin(time, 1h)    )    GROUP BY microservice_name, hour",
    "ScheduleConfiguration": {
        "ScheduleExpression": "cron(0/30 * * * ? *)"
    },
    "NotificationConfiguration": {
        "SnsConfiguration": {
            "TopicArn": "******"
        }
    },
    "TargetConfiguration": {
        "TimestreamConfiguration": {
            "DatabaseName": "derived",
            "TableName": "host_count_pt1h",
```

```
            "TimeColumn": "hour",
            "DimensionMappings": [
                {
                    "Name": "microservice_name",
                    "DimensionValueType": "VARCHAR"
                }
            ],
            "MultiMeasureMappings": {
                "TargetMultiMeasureName": "num_instances",
                "MultiMeasureAttributeMappings": [
                    {
                        "SourceColumn": "num_instances",
                        "MeasureValueType": "BIGINT"
                    }
                ]
            }
        }
    },
    "ErrorReportConfiguration": {
        "S3Configuration" : {
            "BucketName" : "******",
            "ObjectKeyPrefix": "errors",
            "EncryptionOption": "SSE_S3"
        }
    },
    "ScheduledQueryExecutionRoleArn": "******"
}
```

Uso de los resultados precalculados en un nuevo panel

Ahora verá cómo crear su panel de vista agregada utilizando la tabla derivada de la consulta programada que creó. A partir de la instantánea del panel, también podrá validar que los agregados calculados a partir de la tabla derivada y la tabla base también coinciden. Una vez que haya creado los paneles con las tablas derivadas, observará que el tiempo de carga es considerablemente más rápido y los costes que supone utilizar las tablas derivadas en comparación con el cálculo de estos agregados a partir de los datos sin procesar. A continuación, se muestra una instantánea del panel con datos precalculados y la consulta utilizada para representar este panel con datos precalculados almacenados en la tabla como «derivados». host_count_pt1h». Tenga en cuenta que la estructura de la consulta es muy similar a la consulta que se utilizó en el panel de control sobre datos sin procesar, excepto que utiliza la tabla derivada, que ya calcula los distintos recuentos que esta consulta agrega.




```
SELECT CASE WHEN microservice_name = 'apollo' THEN num_instances ELSE NULL END AS
 apollo,
    CASE WHEN microservice_name = 'athena' THEN num_instances ELSE NULL END AS athena,
    CASE WHEN microservice_name = 'demeter' THEN num_instances ELSE NULL END AS
 demeter,
    CASE WHEN microservice_name = 'hercules' THEN num_instances ELSE NULL END AS
 hercules,
    CASE WHEN microservice_name = 'zeus' THEN num_instances ELSE NULL END AS zeus
FROM (
    SELECT microservice_name, AVG(num_instances) AS num_instances
    FROM (
        SELECT microservice_name, bin(time, 1h), SUM(num_instances) as num_instances
        FROM "derived"."host_count_pt1h"
        WHERE time BETWEEN from_milliseconds(1636567785421) AND
 from_milliseconds(1636654185421)
            AND measure_name = 'num_instances'
        GROUP BY microservice_name, bin(time, 1h)
    )
    GROUP BY microservice_name
)
```

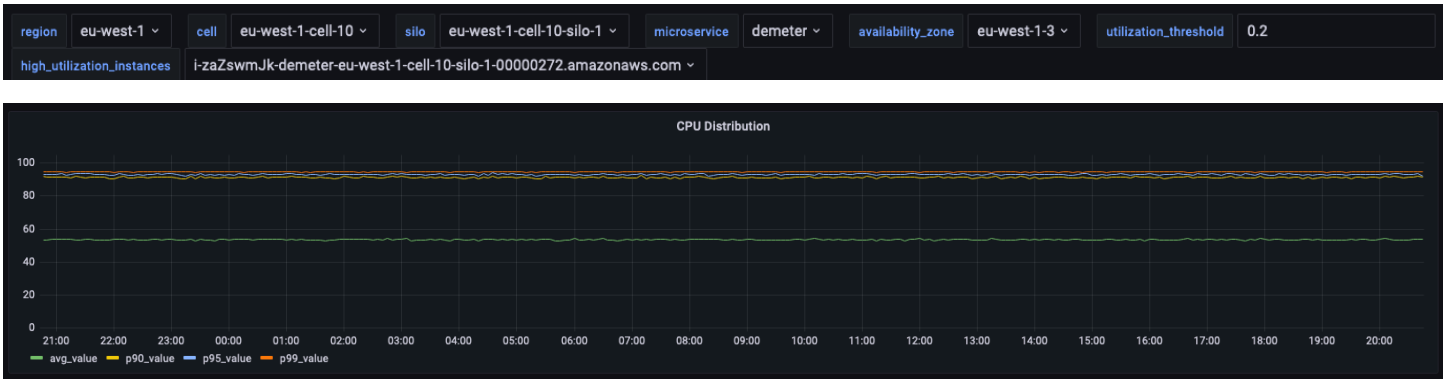
Uso de consultas programadas y datos sin procesar para los desgloses

Puede usar las estadísticas agregadas de su flota para identificar las áreas que necesitan desglosarse y, a continuación, usar los datos sin procesar para desglosar datos granulares y obtener información más detallada.

En este ejemplo, verá cómo puede utilizar el panel agregado para identificar cualquier implementación (una implementación es para un microservicio determinado dentro de una región, celda, silo y zona de disponibilidad determinados) que parezca tener una mayor CPU utilización en comparación con otras implementaciones. A continuación, puede profundizar para obtener una mejor comprensión utilizando los datos sin procesar. Dado que estos desgloses pueden ser poco frecuentes y solo acceden a los datos relevantes para la implementación, puede utilizar los datos sin procesar para este análisis y no es necesario utilizar consultas programadas.

Desglose por despliegue

El siguiente panel proporciona información detallada sobre estadísticas más detalladas y a nivel de servidor dentro de una implementación determinada. Para ayudarlo a desglosar las diferentes partes de su flota, este panel utiliza variables como la región, la celda, el silo, el microservicio y la zona de disponibilidad. A continuación, muestra algunas estadísticas agregadas para esa implementación.



En la consulta siguiente, puede ver que los valores elegidos en el menú desplegable de las variables se utilizan como predicados en la `WHERE` cláusula de la consulta, lo que le permite centrarse únicamente en los datos de la implementación. A continuación, el panel traza las CPU métricas agregadas de las instancias de esa implementación. Puede utilizar los datos sin procesar para realizar este desglose con la latencia de consultas interactivas para obtener información más detallada.

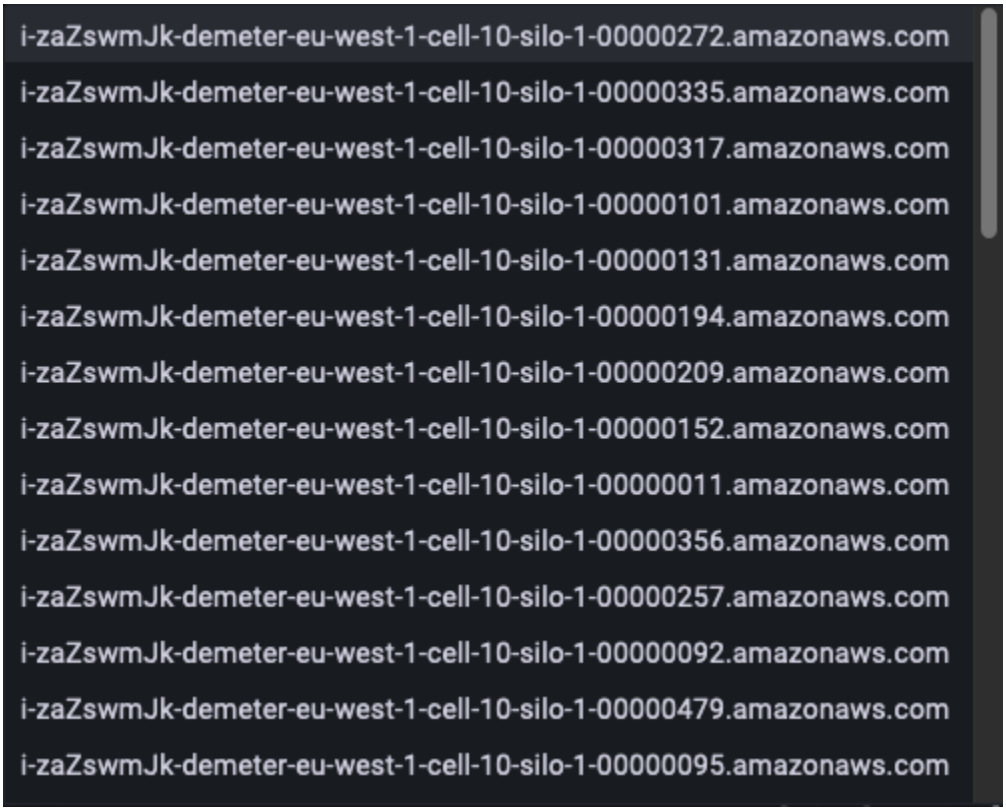
```
SELECT bin(time, 5m) as minute,
    ROUND(AVG(cpu_user), 2) AS avg_value,
    ROUND(APPROX_PERCENTILE(cpu_user, 0.9), 2) AS p90_value,
    ROUND(APPROX_PERCENTILE(cpu_user, 0.95), 2) AS p95_value,
    ROUND(APPROX_PERCENTILE(cpu_user, 0.99), 2) AS p99_value
FROM "raw_data"."devops"
WHERE time BETWEEN from_milliseconds(1636527099476) AND
 from_milliseconds(1636613499476)
    AND region = 'eu-west-1'
    AND cell = 'eu-west-1-cell-10'
    AND silo = 'eu-west-1-cell-10-silo-1'
    AND microservice_name = 'demeter'
    AND availability_zone = 'eu-west-1-3'
    AND measure_name = 'metrics'
GROUP BY bin(time, 5m)
ORDER BY 1
```

Estadísticas a nivel de instancia

Este panel calcula además otra variable que también enumera los servidores/instancias con un alto nivel de CPU utilización, ordenados en orden descendente de utilización. La consulta utilizada para calcular esta variable se muestra a continuación.

```
WITH microservice_cell_avg AS (
    SELECT AVG(cpu_user) AS microservice_avg_metric
    FROM "raw_data"."devops"
    WHERE $__timeFilter
        AND measure_name = 'metrics'
        AND region = '${region}'
        AND cell = '${cell}'
        AND silo = '${silo}'
        AND availability_zone = '${availability_zone}'
        AND microservice_name = '${microservice}'
), instance_avg AS (
    SELECT instance_name,
        AVG(cpu_user) AS instance_avg_metric
    FROM "raw_data"."devops"
    WHERE $__timeFilter
        AND measure_name = 'metrics'
        AND region = '${region}'
        AND cell = '${cell}'
        AND silo = '${silo}'
        AND microservice_name = '${microservice}'
        AND availability_zone = '${availability_zone}'
    GROUP BY availability_zone, instance_name
)
SELECT i.instance_name
FROM instance_avg i CROSS JOIN microservice_cell_avg m
WHERE i.instance_avg_metric > (1 + ${utilization_threshold}) *
 m.microservice_avg_metric
ORDER BY i.instance_avg_metric DESC
```

En la consulta anterior, la variable se recalcula dinámicamente en función de los valores elegidos para las demás variables. Una vez que se haya rellenado la variable para una implementación, puede seleccionar instancias individuales de la lista para visualizar mejor las métricas de esa instancia. Puede seleccionar las distintas instancias en el menú desplegable de los nombres de las instancias, tal y como se muestra en la siguiente instantánea.



Los paneles anteriores muestran las estadísticas de la instancia seleccionada y, a continuación, se muestran las consultas que se utilizan para obtener estas estadísticas.

```
SELECT BIN(time, 30m) AS time_bin,
    AVG(cpu_user) AS avg_cpu,
    ROUND(APPROX_PERCENTILE(cpu_user, 0.99), 2) as p99_cpu
```

```
FROM "raw_data"."devops"
WHERE time BETWEEN from_milliseconds(1636527099477) AND
 from_milliseconds(1636613499477)
    AND measure_name = 'metrics'
    AND region = 'eu-west-1' AND cell = 'eu-west-1-cell-10' AND silo = 'eu-west-1-
cell-10-silo-1'
    AND availability_zone = 'eu-west-1-3' AND microservice_name = 'demeter'
    AND instance_name = 'i-zaZswmJk-demeter-eu-west-1-cell-10-
silo-1-00000272.amazonaws.com'
GROUP BY BIN(time, 30m)
ORDER BY time_bin desc
```

```
SELECT BIN(time, 30m) AS time_bin,
    AVG(memory_used) AS avg_memory,
    ROUND(APPROX_PERCENTILE(memory_used, 0.99), 2) as p99_memory
FROM "raw_data"."devops"
WHERE time BETWEEN from_milliseconds(1636527099477) AND
 from_milliseconds(1636613499477)
    AND measure_name = 'metrics'
    AND region = 'eu-west-1' AND cell = 'eu-west-1-cell-10' AND silo = 'eu-west-1-
cell-10-silo-1'
    AND availability_zone = 'eu-west-1-3' AND microservice_name = 'demeter'
    AND instance_name = 'i-zaZswmJk-demeter-eu-west-1-cell-10-
silo-1-00000272.amazonaws.com'
GROUP BY BIN(time, 30m)
ORDER BY time_bin desc
```

```
SELECT COUNT(gc_pause)
FROM "raw_data"."devops"
WHERE time BETWEEN from_milliseconds(1636527099477) AND
 from_milliseconds(1636613499478)
    AND measure_name = 'events'
    AND region = 'eu-west-1' AND cell = 'eu-west-1-cell-10' AND silo = 'eu-west-1-
cell-10-silo-1'
    AND availability_zone = 'eu-west-1-3' AND microservice_name = 'demeter'
    AND instance_name = 'i-zaZswmJk-demeter-eu-west-1-cell-10-
silo-1-00000272.amazonaws.com'
```

```
SELECT avg(gc_pause) as avg, round(approx_percentile(gc_pause, 0.99), 2) as p99
FROM "raw_data"."devops"
WHERE time BETWEEN from_milliseconds(1636527099478) AND
 from_milliseconds(1636613499478)
```

```
    AND measure_name = 'events'
    AND region = 'eu-west-1' AND cell = 'eu-west-1-cell-10' AND silo = 'eu-west-1-
cell-10-silo-1'
    AND availability_zone = 'eu-west-1-3' AND microservice_name = 'demeter'
    AND instance_name = 'i-zaZswmJk-demeter-eu-west-1-cell-10-
silo-1-00000272.amazonaws.com'
```

```
SELECT BIN(time, 30m) AS time_bin,
    AVG(disk_io_reads) AS avg,
    ROUND(APPROX_PERCENTILE(disk_io_reads, 0.99), 2) as p99
FROM "raw_data"."devops"
WHERE time BETWEEN from_milliseconds(1636527099478) AND
 from_milliseconds(1636613499478)
    AND measure_name = 'metrics'
    AND region = 'eu-west-1' AND cell = 'eu-west-1-cell-10' AND silo = 'eu-west-1-
cell-10-silo-1'
    AND availability_zone = 'eu-west-1-3' AND microservice_name = 'demeter'
    AND instance_name = 'i-zaZswmJk-demeter-eu-west-1-cell-10-
silo-1-00000272.amazonaws.com'
GROUP BY BIN(time, 30m)
ORDER BY time_bin desc
```

Optimizar los costos al compartir las consultas programadas en todos los paneles

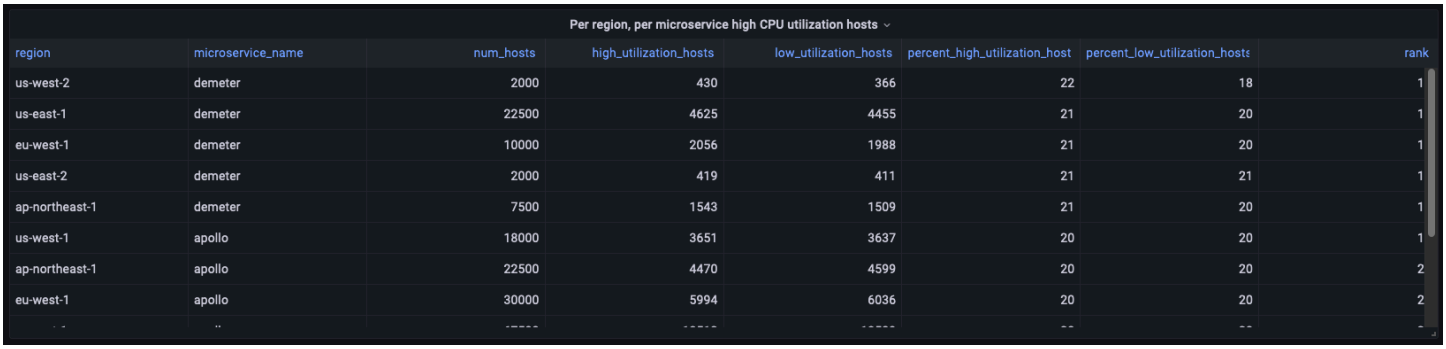
En este ejemplo, veremos un escenario en el que varios paneles del panel de control muestran variaciones de información similar (se encuentran muchos CPU anfitriones y una fracción de la flota con un alto nivel de CPU utilización) y cómo se puede utilizar la misma consulta programada para calcular previamente los resultados que luego se utilizan para rellenar varios paneles. Esta reutilización optimiza aún más los costes, ya que en lugar de utilizar diferentes consultas programadas, una para cada panel, se utiliza únicamente el propietario.

Paneles de panel de control con datos sin procesar

CPUutilización por región y por microservicio

El primer panel calcula las instancias cuya CPU utilización media es un umbral inferior o superior a la CPU utilización anterior para un despliegue determinado en una región, célula, silo, zona de disponibilidad y microservicio. A continuación, clasifica la región y el microservicio que tienen el porcentaje más alto de hosts con un alto nivel de utilización. Ayuda a identificar la temperatura de funcionamiento de los servidores de una implementación específica y, posteriormente, a profundizar en los problemas para comprender mejor los problemas.

La consulta del panel demuestra la flexibilidad de Timestream para LiveAnalytics SQL ayudar a realizar tareas analíticas complejas con expresiones de tablas, funciones de ventana, uniones, etc. habituales.



Per region, per microservice high CPU utilization hosts

region	microservice_name	num_hosts	high_utilization_hosts	low_utilization_hosts	percent_high_utilization_host	percent_low_utilization_hosts	rank
us-west-2	demeter	2000	430	366	22	18	1
us-east-1	demeter	22500	4625	4455	21	20	1
eu-west-1	demeter	10000	2056	1988	21	20	1
us-east-2	demeter	2000	419	411	21	21	1
ap-northeast-1	demeter	7500	1543	1509	21	20	1
us-west-1	apollo	18000	3651	3637	20	20	1
ap-northeast-1	apollo	22500	4470	4599	20	20	2
eu-west-1	apollo	30000	5994	6036	20	20	2

Consulta:

```
WITH microservice_cell_avg AS (
    SELECT region, cell, silo, availability_zone, microservice_name, AVG(cpu_user) AS
 microservice_avg_metric
    FROM "raw_data"."devops"
    WHERE time BETWEEN from_milliseconds(1636526593876) AND
 from_milliseconds(1636612993876)
        AND measure_name = 'metrics'
    GROUP BY region, cell, silo, availability_zone, microservice_name
), instance_avg AS (
    SELECT region, cell, silo, availability_zone, microservice_name, instance_name,
        AVG(cpu_user) AS instance_avg_metric
    FROM "raw_data"."devops"
    WHERE time BETWEEN from_milliseconds(1636526593876) AND
 from_milliseconds(1636612993876)
        AND measure_name = 'metrics'
    GROUP BY region, cell, silo, availability_zone, microservice_name, instance_name
), instances_above_threshold AS (
  SELECT i.*,
    CASE WHEN i.instance_avg_metric > (1 + 0.2) * m.microservice_avg_metric THEN 1 ELSE
 0 END AS high_utilization,
    CASE WHEN i.instance_avg_metric < (1 - 0.2) * m.microservice_avg_metric THEN 1 ELSE
 0 END AS low_utilization
  FROM instance_avg i INNER JOIN microservice_cell_avg m
    ON i.region = m.region AND i.cell = m.cell AND i.silo = m.silo AND
 i.availability_zone = m.availability_zone
      AND m.microservice_name = i.microservice_name
), per_deployment_high AS (
```

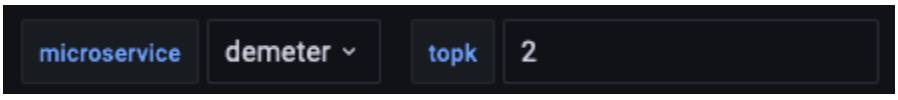
```
SELECT region, microservice_name, COUNT(*) AS num_hosts, SUM(high_utilization) AS
 high_utilization_hosts, SUM(low_utilization) AS low_utilization_hosts,
    ROUND(SUM(high_utilization) * 100.0 / COUNT(*), 0) AS
 percent_high_utilization_hosts,
    ROUND(SUM(low_utilization) * 100.0 / COUNT(*), 0) AS percent_low_utilization_hosts
FROM instances_above_threshold
GROUP BY region, microservice_name
), per_region_ranked AS (
    SELECT *,
        DENSE_RANK() OVER (PARTITION BY region ORDER BY percent_high_utilization_hosts
 DESC, high_utilization_hosts DESC) AS rank
    FROM per_deployment_high
)
SELECT *
FROM per_region_ranked
WHERE rank <= 2
ORDER BY percent_high_utilization_hosts desc, rank asc
```

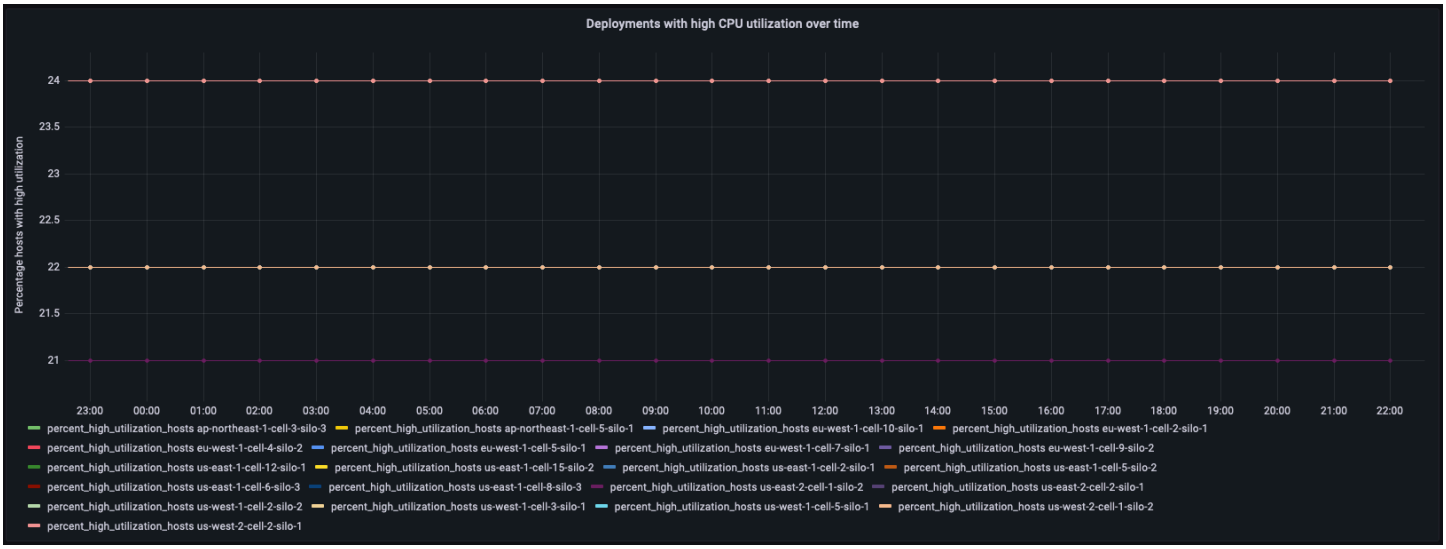
Profundice en un microservicio para encontrar puntos críticos

El siguiente panel le permite profundizar en uno de los microservicios para averiguar en qué región, célula y silo específicos del microservicio está funcionando, qué fracción o fracción de su flota tiene un mayor uso. CPU Por ejemplo, en el panel de control de toda la flota, viste que el demeter de microservicios aparecía entre los primeros puestos de la clasificación, por lo que en este panel te interesaría profundizar en ese microservicio.

Este panel utiliza una variable para seleccionar el microservicio y analizarlo en detalle, y los valores de la variable se rellenan con valores únicos de la dimensión. Una vez que haya elegido el microservicio, el resto del panel se actualizará.

Como se muestra a continuación, el primer panel muestra el porcentaje de hosts de una implementación (una región, una celda y un silo en el caso de un microservicio) a lo largo del tiempo y la consulta correspondiente, que se utiliza para trazar el panel. Este gráfico en sí mismo identifica una implementación específica que tiene un porcentaje más alto de hosts con un porcentaje alto. CPU





Consulta:

```
WITH microservice_cell_avg AS (
    SELECT region, cell, silo, availability_zone, microservice_name, bin(time, 1h) as
 hour, AVG(cpu_user) AS microservice_avg_metric
    FROM "raw_data"."devops"
    WHERE time BETWEEN from_milliseconds(1636526898831) AND
 from_milliseconds(1636613298831)
        AND measure_name = 'metrics'
        AND microservice_name = 'demeter'
    GROUP BY region, cell, silo, availability_zone, microservice_name, bin(time, 1h)
), instance_avg AS (
    SELECT region, cell, silo, availability_zone, microservice_name, instance_name,
 bin(time, 1h) as hour,
        AVG(cpu_user) AS instance_avg_metric
    FROM "raw_data"."devops"
    WHERE time BETWEEN from_milliseconds(1636526898831) AND
 from_milliseconds(1636613298831)
        AND measure_name = 'metrics'
        AND microservice_name = 'demeter'
    GROUP BY region, cell, silo, availability_zone, microservice_name, instance_name,
 bin(time, 1h)
), instances_above_threshold AS (
  SELECT i.*,
    CASE WHEN i.instance_avg_metric > (1 + 0.2) * m.microservice_avg_metric THEN 1 ELSE
 0 END AS high_utilization
  FROM instance_avg i INNER JOIN microservice_cell_avg m
    ON i.region = m.region AND i.cell = m.cell AND i.silo = m.silo AND
 i.availability_zone = m.availability_zone
```

```
      AND m.microservice_name = i.microservice_name AND m.hour = i.hour
), high_utilization_percent AS (
    SELECT region, cell, silo, microservice_name, hour, COUNT(*) AS num_hosts,
 SUM(high_utilization) AS high_utilization_hosts,
        ROUND(SUM(high_utilization) * 100.0 / COUNT(*), 0) AS
 percent_high_utilization_hosts
    FROM instances_above_threshold
    GROUP BY region, cell, silo, microservice_name, hour
), high_utilization_ranked AS (
    SELECT region, cell, silo, microservice_name,
        DENSE_RANK() OVER (PARTITION BY region ORDER BY
 AVG(percent_high_utilization_hosts) desc, AVG(high_utilization_hosts) desc) AS rank
    FROM high_utilization_percent
    GROUP BY region, cell, silo, microservice_name
)
SELECT hup.silo, CREATE_TIME_SERIES(hour, hup.percent_high_utilization_hosts) AS
 percent_high_utilization_hosts
FROM high_utilization_percent hup INNER JOIN high_utilization_ranked hur
    ON hup.region = hur.region AND hup.cell = hur.cell AND hup.silo = hur.silo AND
 hup.microservice_name = hur.microservice_name
WHERE rank <= 2
GROUP BY hup.region, hup.cell, hup.silo
ORDER BY hup.silo
```

Convertir en una única consulta programada, lo que permite su reutilización

Es importante tener en cuenta que se realiza un cálculo similar en los diferentes paneles de los dos paneles. Puede definir una consulta programada independiente para cada panel. Aquí verá cómo puede optimizar aún más sus costes definiendo una consulta programada cuyos resultados se pueden utilizar para representar los tres paneles.

La siguiente es la consulta que captura los agregados que se calculan y utilizan para los diferentes paneles. Observará varios aspectos importantes en la definición de esta consulta programada.

- La flexibilidad y la potencia del área de SQL superficie son compatibles con las consultas programadas, en las que se pueden utilizar expresiones de tablas comunes, uniones, enunciados de casos, etc.
- Puede usar una consulta programada para calcular las estadísticas con una granularidad más precisa de la que podría necesitar un panel específico y para todos los valores que un panel pueda usar para diferentes variables. Por ejemplo, verá que los agregados se calculan en una región, una célula, un silo y un microservicio. Por lo tanto, puede combinarlos para crear agregados a

nivel regional o regional y a nivel de microservicio. Del mismo modo, la misma consulta calcula los agregados de todas las regiones, celdas, silos y microservicios. Permite aplicar filtros en estas columnas para obtener los agregados de un subconjunto de valores. Por ejemplo, puede calcular los agregados para cualquier región, por ejemplo, us-east-1, o cualquier microservicio, por ejemplo, demeter, o profundizar en un despliegue específico dentro de una región, celda, silo y microservicio. Este enfoque optimiza aún más los costos de mantenimiento de los agregados precalculados.

```
WITH microservice_cell_avg AS (
    SELECT region, cell, silo, availability_zone, microservice_name, bin(time, 1h) as
 hour, AVG(cpu_user) AS microservice_avg_metric
    FROM raw_data.devops
    WHERE time BETWEEN bin(@scheduled_runtime, 1h) - 1h AND bin(@scheduled_runtime, 1h)
 + 1h
        AND measure_name = 'metrics'
    GROUP BY region, cell, silo, availability_zone, microservice_name, bin(time, 1h)

), instance_avg AS (
    SELECT region, cell, silo, availability_zone, microservice_name, instance_name,
 bin(time, 1h) as hour,
        AVG(cpu_user) AS instance_avg_metric
   FROM raw_data.devops
   WHERE time BETWEEN bin(@scheduled_runtime, 1h) - 1h AND bin(@scheduled_runtime, 1h)
 + 1h
       AND measure_name = 'metrics'
   GROUP BY region, cell, silo, availability_zone, microservice_name, instance_name,
 bin(time, 1h)
), instances_above_threshold AS (
    SELECT i.*,
        CASE WHEN i.instance_avg_metric > (1 + 0.2) * m.microservice_avg_metric THEN 1
 ELSE 0 END AS high_utilization,
        CASE WHEN i.instance_avg_metric < (1 - 0.2) * m.microservice_avg_metric THEN 1
 ELSE 0 END AS low_utilization
   FROM instance_avg i INNER JOIN microservice_cell_avg m
       ON i.region = m.region AND i.cell = m.cell AND i.silo = m.silo AND
 i.availability_zone = m.availability_zone
           AND m.microservice_name = i.microservice_name AND m.hour = i.hour
)
SELECT region, cell, silo, microservice_name, hour,
     COUNT(*) AS num_hosts, SUM(high_utilization) AS high_utilization_hosts,
 SUM(low_utilization) AS low_utilization_hosts
```

```
FROM instances_above_threshold GROUP BY region, cell, silo, microservice_name, hour
```

La siguiente es una definición de consulta programada para la consulta anterior. La expresión de programación está configurada para actualizarse cada 30 minutos y actualiza los datos hasta hace una hora. De nuevo, utiliza la construcción bin (@scheduled_runtime, 1h) para obtener los eventos de la hora completa. En función de los requisitos de actualización de la aplicación, puede configurarla para que se actualice con mayor o menor frecuencia. Al utilizar el BETWEEN intervalo de WHERE tiempo (@scheduled_runtime, 1h) - intervalo AND de 1 hora (@scheduled_runtime, 1h) + 1 hora, podemos asegurarnos de que, incluso si actualiza una vez cada 15 minutos, obtendrá los datos completos de la hora actual y de la hora anterior.

Más adelante, verá cómo los tres paneles utilizan estos agregados escritos en la tabla deployment_cpu_stats_per_hr para visualizar las métricas que son relevantes para el panel.

```
{
    "Name": "MultiPT30mHighCpuDeploymentsPerHr",
    "QueryString": "WITH microservice_cell_avg AS (    SELECT region, cell,
 silo, availability_zone, microservice_name, bin(time, 1h) as hour, AVG(cpu_user)
 AS microservice_avg_metric    FROM raw_data.devops    WHERE time BETWEEN
 bin(@scheduled_runtime, 1h) - 1h AND bin(@scheduled_runtime, 1h) + 1h    AND
 measure_name = 'metrics'    GROUP BY region, cell, silo, availability_zone,
 microservice_name, bin(time, 1h)    ), instance_avg AS (    SELECT region,
 cell, silo, availability_zone, microservice_name, instance_name, bin(time, 1h)
 as hour,    AVG(cpu_user) AS instance_avg_metric    FROM raw_data.devops
 WHERE time BETWEEN bin(@scheduled_runtime, 1h) - 1h AND bin(@scheduled_runtime,
 1h) + 1h    AND measure_name = 'metrics'    GROUP BY region, cell, silo,
 availability_zone, microservice_name, instance_name, bin(time, 1h)    ),
 instances_above_threshold AS (    SELECT i.*,    CASE WHEN i.instance_avg_metric >
 (1 + 0.2) * m.microservice_avg_metric THEN 1 ELSE 0 END AS high_utilization,    CASE
 WHEN i.instance_avg_metric < (1 - 0.2) * m.microservice_avg_metric THEN 1 ELSE 0 END
 AS low_utilization    FROM instance_avg i INNER JOIN microservice_cell_avg m    ON
 i.region = m.region AND i.cell = m.cell AND i.silo = m.silo AND i.availability_zone
 = m.availability_zone    AND m.microservice_name = i.microservice_name AND m.hour =
 i.hour    )     SELECT region, cell, silo, microservice_name, hour,         COUNT(*)
 AS num_hosts, SUM(high_utilization) AS high_utilization_hosts, SUM(low_utilization) AS
 low_utilization_hosts    FROM instances_above_threshold GROUP BY region, cell, silo,
 microservice_name, hour",
    "ScheduleConfiguration": {
        "ScheduleExpression": "cron(0/30 * * * ? *)"
    },
    "NotificationConfiguration": {
        "SnsConfiguration": {
```

```
            "TopicArn": "******"
        }
    },
    "TargetConfiguration": {
        "TimestreamConfiguration": {
            "DatabaseName": "derived",
            "TableName": "deployment_cpu_stats_per_hr",
            "TimeColumn": "hour",
            "DimensionMappings": [
                {
                    "Name": "region",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "cell",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "silo",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "microservice_name",
                    "DimensionValueType": "VARCHAR"
                }
            ],
            "MultiMeasureMappings": {
                "TargetMultiMeasureName": "cpu_user",
                "MultiMeasureAttributeMappings": [
                    {
                        "SourceColumn": "num_hosts",
                        "MeasureValueType": "BIGINT"
                    },
                    {
                        "SourceColumn": "high_utilization_hosts",
                        "MeasureValueType": "BIGINT"
                    },
                    {
                        "SourceColumn": "low_utilization_hosts",
                        "MeasureValueType": "BIGINT"
                    }
                ]
            }
        }
```

```
    },
    "ErrorReportConfiguration": {
        "S3Configuration" : {
            "BucketName" : "******",
            "ObjectKeyPrefix": "errors",
            "EncryptionOption": "SSE_S3"
        }
    },
    "ScheduledQueryExecutionRoleArn": "******"
}
```

Cuadro de mando a partir de resultados precalculados

Hosts de alta CPU utilización

En el caso de los hosts de alta utilización, verá cómo los distintos paneles utilizan los datos de deployment_cpu_stats_per_hr para calcular los distintos agregados necesarios para los paneles. Por ejemplo, este panel proporciona información a nivel regional, por lo que presenta los agregados agrupados por región y microservicio, sin filtrar ninguna región o microservicio.

Per region, per microservice high utilization hosts

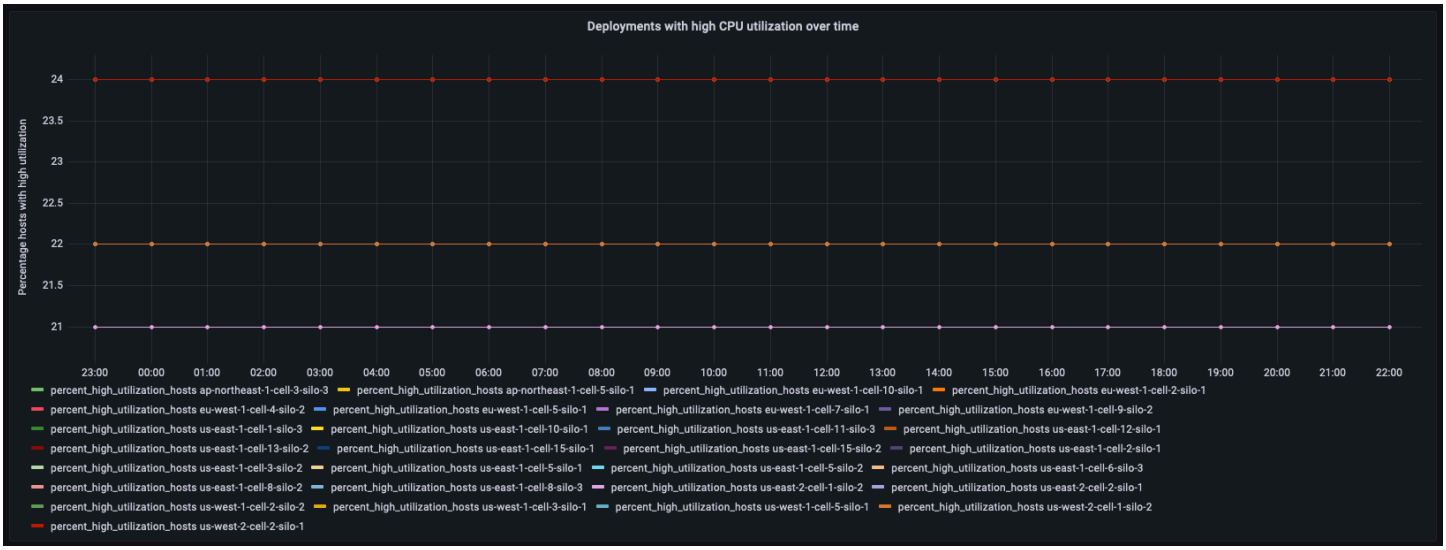
region	microservice_name	num_hosts	high_utilization_hosts	low_utilization_hosts	percent_high_utilization_host	percent_low_utilization_hosts	rank
us-west-2	demeter	1962	423	359	22	18	1
us-east-2	demeter	2000	419	411	21	21	1
us-east-1	demeter	22500	4628	4455	21	20	1
ap-northeast-1	demeter	7500	1544	1509	21	20	1
eu-west-1	demeter	9983	2056	1984	21	20	1
us-west-1	apollo	18000	3657	3643	20	20	1
ap-northeast-1	apollo	22500	4470	4599	20	20	2
us-east-2	hercules	4000	813	752	20	19	2

```
WITH per_deployment_hosts AS (
    SELECT region, cell, silo, microservice_name,
        AVG(num_hosts) AS num_hosts,
        AVG(high_utilization_hosts) AS high_utilization_hosts,
        AVG(low_utilization_hosts) AS low_utilization_hosts
    FROM "derived"."deployment_cpu_stats_per_hr"
    WHERE time BETWEEN from_milliseconds(1636567785437) AND
 from_milliseconds(1636654185437)
        AND measure_name = 'cpu_user'
    GROUP BY region, cell, silo, microservice_name
), per_deployment_high AS (
    SELECT region, microservice_name,
        SUM(num_hosts) AS num_hosts,
        ROUND(SUM(high_utilization_hosts), 0) AS high_utilization_hosts,
```

```
        ROUND(SUM(low_utilization_hosts),0) AS low_utilization_hosts,
        ROUND(SUM(high_utilization_hosts) * 100.0 / SUM(num_hosts)) AS
 percent_high_utilization_hosts,
        ROUND(SUM(low_utilization_hosts) * 100.0 / SUM(num_hosts)) AS
 percent_low_utilization_hosts
    FROM per_deployment_hosts
    GROUP BY region, microservice_name
),
per_region_ranked AS (
    SELECT *,
        DENSE_RANK() OVER (PARTITION BY region ORDER BY percent_high_utilization_hosts
 DESC, high_utilization_hosts DESC) AS rank
    FROM per_deployment_high
)
SELECT *
FROM per_region_ranked
WHERE rank <= 2
ORDER BY percent_high_utilization_hosts desc, rank asc
```

Profundice en un microservicio para encontrar implementaciones de alto uso CPU

En el siguiente ejemplo, se vuelve a utilizar la tabla derivada deployment_cpu_stats_per_hr, pero ahora se aplica un filtro para un microservicio específico (demeter en este ejemplo, ya que en el panel agregado se muestran los hosts de alta utilización). Este panel hace un seguimiento del porcentaje de hosts de alta utilización a lo largo del tiempo. CPU




```
WITH high_utilization_percent AS (
    SELECT region, cell, silo, microservice_name, bin(time, 1h) AS hour, MAX(num_hosts)
 AS num_hosts,
```

```
        MAX(high_utilization_hosts) AS high_utilization_hosts,
        ROUND(MAX(high_utilization_hosts) * 100.0 / MAX(num_hosts)) AS
 percent_high_utilization_hosts
    FROM "derived"."deployment_cpu_stats_per_hr"
    WHERE time BETWEEN from_milliseconds(1636525800000) AND
 from_milliseconds(1636612200000)
        AND measure_name = 'cpu_user'
        AND microservice_name = 'demeter'
    GROUP BY region, cell, silo, microservice_name, bin(time, 1h)
), high_utilization_ranked AS (
    SELECT region, cell, silo, microservice_name,
        DENSE_RANK() OVER (PARTITION BY region ORDER BY
 AVG(percent_high_utilization_hosts) desc, AVG(high_utilization_hosts) desc) AS rank
    FROM high_utilization_percent
    GROUP BY region, cell, silo, microservice_name
)
SELECT hup.silo, CREATE_TIME_SERIES(hour, hup.percent_high_utilization_hosts) AS
 percent_high_utilization_hosts
FROM high_utilization_percent hup INNER JOIN high_utilization_ranked hur
    ON hup.region = hur.region AND hup.cell = hur.cell AND hup.silo = hur.silo AND
 hup.microservice_name = hur.microservice_name
WHERE rank <= 2
GROUP BY hup.region, hup.cell, hup.silo
ORDER BY hup.silo
```

Comparación de una consulta de una tabla base con una consulta de los resultados de una consulta programada

En este ejemplo de consulta Timestream, utilizamos el siguiente esquema, consultas de ejemplo y resultados para comparar una consulta de una tabla base con una consulta de una tabla derivada de resultados de consultas programadas. Con una consulta planificada y bien planificada, puede obtener una tabla derivada con menos filas y otras características que permiten realizar consultas más rápidas de lo que sería posible en la tabla base original.

Para ver un vídeo que describe este escenario, consulte Mejorar el rendimiento de las consultas y reducir los costes mediante consultas programadas en Amazon LiveAnalytics Timestream for.

Para este ejemplo, utilizamos el siguiente escenario:

- Región — us-east-1
- Tabla base — `"clickstream"."shopping"`
- Tabla derivada — `"clickstream"."aggregate"`

Tabla base

A continuación se describe el esquema de la tabla base.

Columna	Tipo	Secuencia temporal del tipo de atributo LiveAnalytics
channel	varchar	MULTI
description	varchar	MULTI
evento	varchar	DIMENSION
ip_address	varchar	DIMENSION
measure_name	varchar	MEASURE_NAME
producto	varchar	MULTI
product_id	varchar	MULTI
quantity	double	MULTI
consulta	varchar	MULTI
session_id	varchar	DIMENSION
user_group	varchar	DIMENSION
user_id	varchar	DIMENSION

A continuación se describen las medidas de la tabla base. Una tabla base hace referencia a una tabla de Timestream en la que se ejecuta la consulta programada.

- measure_name — `metrics`
- datos — varios
- dimensiones:

```
[ ( user_group, varchar ),( user_id, varchar ),( session_id, varchar ),( ip_address,
 varchar ),( event, varchar ) ]
```

Consulta en una tabla base

La siguiente es una consulta ad hoc que recopila los recuentos por un total de 5 minutos en un intervalo de tiempo determinado.

```
SELECT BIN(time, 5m) as time,
channel,
product_id,
SUM(quantity) as product_quantity
FROM "clickstream"."shopping"
WHERE BIN(time, 5m) BETWEEN '2023-05-11 10:10:00.000000000' AND '2023-05-11
 10:30:00.000000000'
AND channel = 'Social media'
and product_id = '431412'
GROUP BY BIN(time, 5m),channel,product_id
```

Salida:

```
duration:1.745 sec
Bytes scanned: 29.89 MB
Query Id: AEBQEANMHG7MHHBHCKJ3BSOE3QUGIDBGWCCP5I6J6YUW5CVJZ2M3JCJ27QRMM7A
Row count:5
```

Consulta programada

La siguiente es una consulta programada que se ejecuta cada 5 minutos.

```
SELECT BIN(time, 5m) as time, channel as measure_name, product_id, product,
SUM(quantity) as product_quantity
FROM "clickstream"."shopping"
WHERE time BETWEEN BIN(@scheduled_runtime, 5m) - 10m AND BIN(@scheduled_runtime, 5m) -
 5m
AND channel = 'Social media'
GROUP BY BIN(time, 5m), channel, product_id, product
```

Consulta en una tabla derivada

La siguiente es una consulta ad hoc en una tabla derivada. Una tabla derivada hace referencia a una tabla de flujo temporal que contiene los resultados de una consulta programada.

```
SELECT time, measure_name, product_id,product_quantity
```

```
FROM "clickstream"."aggregate"
WHERE time BETWEEN '2023-05-11 10:10:00.000000000' AND '2023-05-11 10:30:00.000000000'
AND measure_name = 'Social media'
and product_id = '431412'
```

Salida:

```
duration: 0.2960 sec
Bytes scanned: 235.00 B
QueryID: AEBQEANMHHAAQU4FFTT6CFM6UYXTL4SMLZV22MFP4KV2Z7IRVOPLOMLDD6BR33Q
Row count: 5
```

Comparación

A continuación, se comparan los resultados de una consulta de una tabla base con los de una consulta de una tabla derivada. La misma consulta de una tabla derivada que ha agregado los resultados mediante una consulta programada se completa más rápido y con menos bytes escaneados.

Estos resultados muestran el valor de usar consultas programadas para agregar datos y así realizar consultas más rápidas.

	Consulta en la tabla base	Consulta en la tabla derivada
Duración	1.745 segundos	0.2960 segundos
Bytes escaneados	29.89 MB	235 bytes
Recuento de filas	5	5

Se utiliza UNLOAD para exportar los resultados de las consultas a S3 desde Timestream para LiveAnalytics

LiveAnalytics Por ahora, Amazon Timestream le permite exportar los resultados de sus consultas a Amazon S3 de forma rentable y segura utilizando la declaración. `UNLOAD` Gracias a `UNLOAD` esta declaración, ahora puede exportar datos de series temporales a buckets S3 seleccionados en formato Apache Parquet o con valores separados por comas (CSV), lo que proporciona flexibilidad para almacenar, combinar y analizar sus datos de series temporales con otros servicios. La `UNLOAD`

declaración le permite exportar los datos de forma comprimida, lo que reduce los datos transferidos y el espacio de almacenamiento necesario. UNLOADtambién admite la partición en función de los atributos seleccionados al exportar los datos, lo que mejora el rendimiento y reduce el tiempo de procesamiento de los servicios intermedios que acceden a los datos. Además, puede usar las claves administradas de Amazon S3 (SSE-S3) o las claves administradas por AWS Key Management Service (AWS KMS) (SSE-KMS) para cifrar los datos exportados.

Ventajas de UNLOAD Timestream para LiveAnalytics

Los beneficios clave de usar la UNLOAD declaración son los siguientes.

- Facilidad operativa: con UNLOAD esta declaración, puede exportar gigabytes de datos en una sola solicitud de consulta en Apache Parquet o en el CSV formato Apache Parquet, lo que proporciona flexibilidad para seleccionar el formato que mejor se adapte a sus necesidades de procesamiento posterior y facilita la creación de lagos de datos.
- Seguro y rentable: la UNLOAD declaración ofrece la capacidad de exportar sus datos a un depósito de S3 de forma comprimida y de cifrar (SSE- KMS o SSE _S3) sus datos mediante claves administradas por el cliente, lo que reduce los costos de almacenamiento de datos y protege contra el acceso no autorizado.
- Rendimiento: con UNLOAD esta declaración, puede particionar los datos al exportarlos a un bucket de S3. La partición de los datos permite a los servicios posteriores procesar los datos en paralelo, lo que reduce su tiempo de procesamiento. Además, los servicios intermedios pueden procesar solo los datos que necesitan, lo que reduce los recursos de procesamiento necesarios y, por lo tanto, los costes asociados.

Casos de uso UNLOAD de Timestream para LiveAnalytics

Puede usar la UNLOAD instrucción para escribir datos en su bucket de S3 de la siguiente manera.

- Cree un almacén de datos: puede exportar gigabytes de resultados de consultas a un depósito de S3 y añadir más fácilmente datos de series temporales a su lago de datos. Puede utilizar servicios como Amazon Athena y Amazon Redshift para combinar sus datos de series temporales con otros datos relevantes para obtener información empresarial compleja.
- Cree canalizaciones de datos de IA y aprendizaje automático: la UNLOAD declaración le permite crear fácilmente canalizaciones de datos para sus modelos de aprendizaje automático que acceden a datos de series temporales, lo que facilita el uso de datos de series temporales con servicios como Amazon y SageMaker Amazon. EMR

- Simplifique el ETL procesamiento: la exportación de datos a depósitos de S3 puede simplificar el proceso de realizar operaciones de extracción, transformación y carga (ETL) en los datos, lo que le permite utilizar herramientas o AWS servicios de terceros, como AWS Glue, para procesar y transformar los datos sin problemas.

UNLOADConceptos

Sintaxis

```
UNLOAD (SELECT statement)
 TO 's3://bucket-name/folder'
 WITH ( option = expression [, ...] )
```

¿dónde `option` está

```
{ partitioned_by = ARRAY[ col_name[,…] ]
 | format = [ '{ CSV | PARQUET }' ]
 | compression = [ '{ GZIP | NONE }' ]
 | encryption = [ '{ SSE_KMS | SSE_S3 }' ]
 | kms_key = '<string>'
 | field_delimiter ='<character>'
 | escaped_by = '<character>'
 | include_header = ['{true, false}']
 | max_file_size = '<value>'
 | }
```

Parámetros

SELECTdeclaración

La sentencia de consulta utilizada para seleccionar y recuperar datos de uno o más Timestream para LiveAnalytics tablas.

```
(SELECT column 1, column 2, column 3 from database.table
      where measure_name = "ABC" and timestamp between ago (1d) and now() )
```

Cláusula TO

```
TO 's3://bucket-name/folder'
```

o

```
TO 's3://access-point-alias/folder'
```

La TO cláusula de la UNLOAD declaración especifica el destino de la salida de los resultados de la consulta. Debe proporcionar la ruta completa, incluido el nombre del bucket de Amazon S3 o Amazon S3 access-point-alias con la ubicación de la carpeta en Amazon S3 donde Timestream for LiveAnalytics escribe los objetos del archivo de salida. El bucket de S3 debe ser propiedad de la misma cuenta y estar en la misma región. Además del conjunto de resultados de la consulta, Timestream for LiveAnalytics escribe los archivos de manifiesto y metadatos en la carpeta de destino especificada.

PARTITIONEDCláusula _BY

```
partitioned_by = ARRAY [col_name[,…] , (default: none)
```

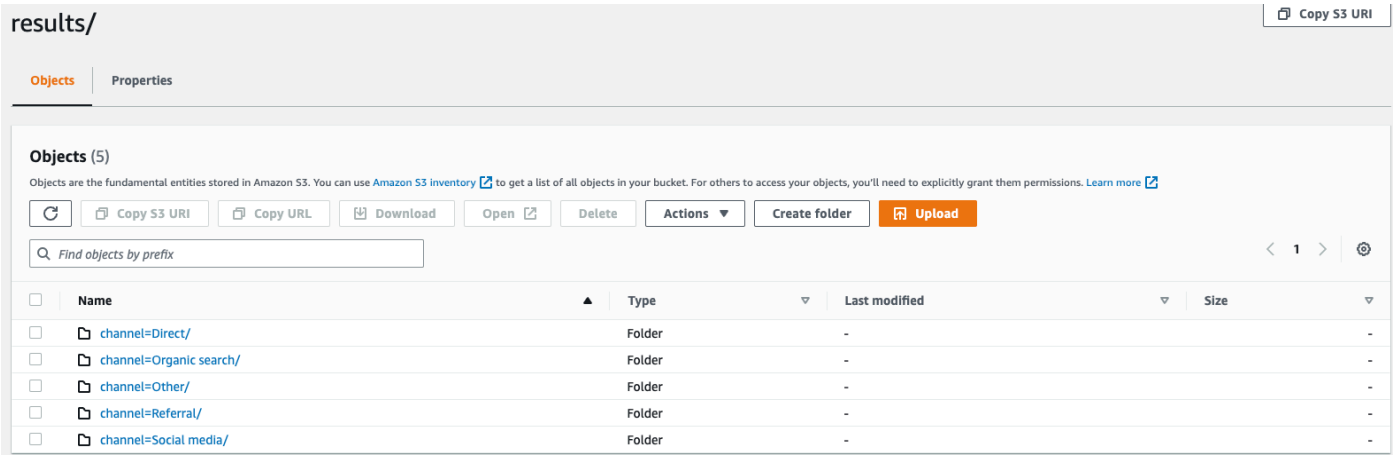
La `partitioned_by` cláusula se utiliza en las consultas para agrupar y analizar los datos de forma granular. Al exportar los resultados de la consulta al bucket de S3, puede elegir dividir los datos en función de una o más columnas de la consulta seleccionada. Al particionar los datos, los datos exportados se dividen en subconjuntos según la columna de partición y cada subconjunto se almacena en una carpeta independiente. Dentro de la carpeta de resultados que contiene los datos exportados, se crea automáticamente una subcarpeta`folder/results/partition column = partition value/`. Sin embargo, ten en cuenta que las columnas particionadas no se incluyen en el archivo de salida.

`partitioned_by`no es una cláusula obligatoria en la sintaxis. Si decide exportar los datos sin ninguna partición, puede excluir la cláusula de la sintaxis.

Example

Supongamos que está supervisando los datos del flujo de clics de su sitio web y tiene 5 canales de tráfico`direct`, a saber,`Social Media`, `Organic Search``Other`, y. `Referral` Al exportar los datos, puede optar por particionarlos mediante la columna`Channel`. Dentro de tu carpeta de datos`s3://bucketname/results`, tendrás cinco carpetas, cada una con su nombre de canal respectivo. Por ejemplo, `s3://bucketname/results/channel=Social Media/.` dentro de esta carpeta encontrarás los datos de todos los clientes que llegaron a tu sitio web a través del `Social Media` canal. Del mismo modo, dispondrá de otras carpetas para el resto de los canales.

Datos exportados particionados por columna de canales



FORMAT

```
format = [ '{ CSV | PARQUET }' , default: CSV
```

Las palabras clave para especificar el formato de los resultados de la consulta que se escriben en su bucket de S3. Puede exportar los datos como valores separados por comas (CSV) utilizando una coma (,) como delimitador predeterminado o en el formato Apache Parquet, un formato eficiente de almacenamiento en columnas abiertas para el análisis.

COMPRESSION

```
compression = [ '{ GZIP | NONE }' ], default: GZIP
```

Puede comprimir los datos exportados mediante un algoritmo de compresión GZIP o descomprimirlos especificando la opción. NONE

ENCRYPTION

```
encryption = [ '{ SSE_KMS | SSE_S3 }' ], default: SSE_S3
```

Los archivos de salida de Amazon S3 se cifran con la opción de cifrado que haya seleccionado. Además de sus datos, los archivos de manifiesto y metadatos también se cifran en función de la opción de cifrado que haya seleccionado. Actualmente, admitimos el KMS cifrado SSE _S3 y SSE _. SSE_S3 es un cifrado del lado del servidor en el que Amazon S3 cifra los datos mediante el cifrado estándar de cifrado avanzado () de 256 bits. AES SSE_ KMS es un cifrado del lado del servidor para cifrar los datos mediante claves administradas por el cliente.

KMS_KEY

```
kms_key = '<string>'
```

KMSLa clave es una clave definida por el cliente para cifrar los resultados de las consultas exportadas. KMSKey Management Service (AWS KMS) gestiona la AWS clave de forma segura y se utiliza para cifrar archivos de datos en Amazon S3.

FIELD_DELIMITER

```
field_delimiter ='<character>' , default: (,)
```

Al exportar los datos en CSV formato, este campo especifica un único ASCII carácter que se utiliza para separar los campos del archivo de salida, como un carácter vertical (|), una coma (,) o una tabulación (/t). El delimitador predeterminado de los CSV archivos es un carácter de coma. Si un valor de los datos contiene el delimitador elegido, el delimitador aparecerá entre comillas. Por ejemplo, si el valor de sus datos lo contiene`Time,stream`, este valor se indicará como `"Time,stream"` en los datos exportados. El carácter de comilla utilizado por Timestream LiveAnalytics son comillas dobles («).

Evite especificar el carácter de retorno (ASCII13, hexadecimal`0D`, texto '\ r') o el carácter de salto de línea (ASCII10, hexadecimal 0A, texto'\n') `FIELD_DELIMITER` si desea incluir encabezados en elCSV, ya que eso impedirá que muchos analizadores puedan analizar los encabezados correctamente en el resultado resultante. CSV

ESCAPED_BY

```
escaped_by = '<character>', default: (\)
```

Al exportar los datos en CSV formato, este campo especifica el carácter que debe tratarse como un carácter de escape en el archivo de datos escrito en el bucket de S3. El escape se produce en los siguientes escenarios:

1. Si el valor en sí contiene el carácter entre comillas («), se escapará utilizando un carácter de escape. Por ejemplo, si el valor es`Time"stream`, donde (\) es el carácter de escape configurado, se escapa como`Time\"stream`.
2. Si el valor contiene el carácter de escape configurado, se escapará. Por ejemplo, si el valor es`Time\stream`, se escapará como`Time\\stream`.

> **Note**
>
> Si la salida exportada contiene tipos de datos complejos, como matrices, filas o series temporales, se serializará como una cadena. JSON A continuación se muestra un ejemplo.

Tipo de datos	Valor real	Cómo se escapa el valor en el CSV formato [JSONcade na serializada]
Matriz	`[ 23,24,25 ]`	`"[23,24,25]"`
Fila	`( x=23.0, y=hello )`	`"{\"x\":23.0,\"y\":\"hello\"}"`
Serie temporal	`[ ( time=1970-01-01 00:00:00.000000010 , value=100.0 ), ( time=1970-01-01 00:00:00.000000012, value=120.0 ) ]`	`"[{\"time\":\"1970-01-01 00:00:00.000000010Z\",\"value\":100.0},{\"time\":\"1970-01-01 00:00:00.000000012Z\",\"value\":120.0}]"`

INCLUDE_HEADER

```
include_header = 'true' , default: 'false'
```

Al exportar los datos en CSV formato, este campo permite incluir los nombres de las columnas en la primera fila de los archivos de CSV datos exportados.

Los valores aceptados son «verdadero» y «falso» y el valor predeterminado es «falso». Las opciones de transformación de texto, por ejemplo, `escaped_by` `field_delimiter` se aplican también a los encabezados.

> **Note**
>
> Al incluir encabezados, es importante que no seleccione un carácter de retorno vertical (ASCII13, hexadecimal 0D, texto '\ r') o un carácter de salto de línea (ASCII10, hexadecimal 0A, texto'\n') como talesFIELD_DELIMITER, ya que esto impedirá que muchos analizadores puedan analizar correctamente los encabezados en el resultado resultante. CSV

MAX_FILE_SIZE

```
max_file_size = 'X[MB|GB]' , default: '78GB'
```

Este campo especifica el tamaño máximo de los archivos que la UNLOAD declaración crea en Amazon S3. La UNLOAD declaración puede crear varios archivos, pero el tamaño máximo de cada archivo escrito en Amazon S3 será aproximadamente el especificado en este campo.

El valor del campo debe estar comprendido entre 16 MB y 78 GB, ambos inclusive. Puede especificarlo en números enteros, por ejemplo12GB, o en decimales, como 0.5GB o24.7MB. El valor predeterminado es 78 GB.

El tamaño real del archivo es aproximado cuando se escribe el archivo, por lo que es posible que el tamaño máximo real no sea exactamente igual al número que especifique.

¿Qué está escrito en mi bucket de S3?

Por cada UNLOAD consulta que se ejecute correctamente, Timestream for LiveAnalytics escribe los resultados de la consulta, el archivo de metadatos y el archivo de manifiesto en el bucket de S3. Si ha particionado los datos, tiene todas las carpetas de particiones en la carpeta de resultados. El archivo de manifiesto contiene una lista de los archivos que escribió el UNLOAD comando. El archivo de metadatos contiene información que describe las características, propiedades y atributos de los datos escritos.

¿Cuál es el nombre del archivo exportado?

El nombre del archivo exportado contiene dos componentes, el primer componente es el QueryID y el segundo es un identificador único.

CSVarchivos

```
S3://bucket_name/results/<queryid>_<UUID>.csv
S3://bucket_name/results/<partitioncolumn>=<partitionvalue>/<queryid>_<UUID>.csv
```

CSVFichero comprimido

```
S3://bucket_name/results/<partitioncolumn>=<partitionvalue>/<queryid>_<UUID>.gz
```

Archivo de parquet

```
S3://bucket_name/results/<partitioncolumn>=<partitionvalue>/<queryid>_<UUID>.parquet
```

Archivos de metadatos y manifiestos

```
S3://bucket_name/<queryid>_<UUID>_manifest.json
S3://bucket_name/<queryid>_<UUID>_metadata.json
```

Como los datos en CSV formato se almacenan a nivel de archivo, al comprimir los datos al exportarlos a S3, el archivo tendrá la extensión «.gz». Sin embargo, los datos de Parquet se comprimen a nivel de columna, por lo que incluso si comprime los datos durante la exportación, el archivo seguirá teniendo la extensión.parquet.

¿Qué información contiene cada archivo?

Archivo de manifiesto

El archivo de manifiesto proporciona información sobre la lista de archivos que se exportan con la UNLOAD ejecución. El archivo de manifiesto está disponible en el bucket de S3 proporcionado con un nombre de archivo:`s3://<bucket_name>/<queryid>_<UUID>_manifest.json`. El archivo de manifiesto contendrá la URL de los archivos de la carpeta de resultados, el número de registros y el tamaño de los archivos respectivos, y los metadatos de la consulta (que son el total de bytes y filas exportados a S3 para la consulta).

```
{
  "result_files": [
    {
        "url":"s3://my_timestream_unloads/ec2_metrics/
AEDAGANLHLBH4OLISD3CVOZZRWPX5GV2XCXRBKCVD554N6GWPWWXBP7LSG74V2Q_1448466917_szCL4YgVYzGXj2lS.gz"
        "file_metadata":
            {
                "content_length_in_bytes": 32295,
```

```
                "row_count": 10
            }
    },
    {
        "url":"s3://my_timestream_unloads/ec2_metrics/
AEDAGANLHLBH4OLISD3CVOZZRWPX5GV2XCXRBKCVD554N6GWPWWXBP7LSG74V2Q_1448466917_szCL4YgVYzGXj2lS.gz"
        "file_metadata":
            {
                "content_length_in_bytes": 62295,
                "row_count": 20
            }
    },
  ],
  "query_metadata":
    {
      "content_length_in_bytes": 94590,
      "total_row_count": 30,
      "result_format": "CSV",
      "result_version": "Amazon Timestream version 1.0.0"
    },
  "author": {
        "name": "Amazon Timestream",
        "manifest_file_version": "1.0"
  }
}
```

Metadatos

El archivo de metadatos proporciona información adicional sobre el conjunto de datos, como el nombre de la columna, el tipo de columna y el esquema. <queryid>El archivo de metadatos está disponible en el bucket de S3 proporcionado con un nombre de archivo: S3: //bucket_name/ _< >_metadata.json UUID

A continuación se muestra un ejemplo de un archivo de metadatos.

```
{
    "ColumnInfo": [
        {
            "Name": "hostname",
            "Type": {
                "ScalarType": "VARCHAR"
            }
        },
```

```
        {
            "Name": "region",
            "Type": {
                "ScalarType": "VARCHAR"
            }
        },
        {
            "Name": "measure_name",
            "Type": {
                "ScalarType": "VARCHAR"
            }
        },
        {
            "Name": "cpu_utilization",
            "Type": {
                "TimeSeriesMeasureValueColumnInfo": {
                    "Type": {
                        "ScalarType": "DOUBLE"
                    }
                }
            }
        }
  ],
  "Author": {
        "Name": "Amazon Timestream",
        "MetadataFileVersion": "1.0"
  }
}
```

La información de las columnas que se comparte en el archivo de metadatos tiene la misma estructura que la que `ColumnInfo` se envía en la API respuesta a las `SELECT` consultas.

Resultados

La carpeta de resultados contiene los datos exportados en Apache Parquet o en CSV formato.

Ejemplo

Al enviar una `UNLOAD` consulta como la que se muestra a continuación a través de QueryAPI,

```
UNLOAD(SELECT user_id, ip_address, event, session_id, measure_name, time, query,
 quantity, product_id, channel
                    FROM sample_clickstream.sample_shopping WHERE time BETWEEN ago(2d)
 AND now())
```

```
                TO 's3://my_timestream_unloads/withoutpartition/' WITH ( format='CSV',
 compression='GZIP')
```

UNL0ADla respuesta a la consulta tendrá 1 fila x 3 columnas. Esas 3 columnas son:

- filas de tipoBIGINT: indica el número de filas exportadas
- metadataFile de tipoVARCHAR, que es el S3 URI del archivo de metadatos exportado
- manifestFile de tipoVARCHAR, que es el S3 URI del archivo de manifiesto exportado

Obtendrá la siguiente respuesta de QueryAPI:

```
{
    "Rows": [
        {
            "Data": [
                {
                    "ScalarValue": "20" # No of rows in output across all files
                },
                {
                    "ScalarValue": "s3://my_timestream_unloads/withoutpartition/
AEDAAANGH3D7FYHOBQGQQMEAISCJ45B42OWWJMOT4N6RRJICZUA7R25VYVOHJIY_<UUID>_metadata.json"
 #Metadata file
                },
                {
                    "ScalarValue": "s3://my_timestream_unloads/withoutpartition/
AEDAAANGH3D7FYHOBQGQQMEAISCJ45B42OWWJMOT4N6RRJICZUA7R25VYVOHJIY_<UUID>_manifest.json"
 #Manifest file
                }
            ]
        }
    ],
    "ColumnInfo": [
        {
            "Name": "rows",
            "Type": {
                "ScalarType": "BIGINT"
            }
        },
        {
            "Name": "metadataFile",
            "Type": {
                "ScalarType": "VARCHAR"
```

```
            }
        },
        {
            "Name": "manifestFile",
            "Type": {
                "ScalarType": "VARCHAR"
            }
        }
    ],
    "QueryId": "AEDAAANGH3D7FYHOBQGQQMEAISCJ45B42OWWJMOT4N6RRJICZUA7R25VYVOHJIY",
    "QueryStatus": {
        "ProgressPercentage": 100.0,
        "CumulativeBytesScanned": 1000,
        "CumulativeBytesMetered": 10000000
    }
}
```

Tipos de datos

La UNLOAD instrucción admite todos los tipos de datos de Timestream para el lenguaje LiveAnalytics de consulta descrito en, Tipos de datos compatibles excepto `time` y. `unknown`

Requisitos previos para: de Timestream para UNLOAD LiveAnalytics

Los siguientes son los requisitos previos para escribir datos en S3 utilizando UNLOAD From Timestream for. LiveAnalytics

- Debe tener permiso para leer los datos del Timestream para poder LiveAnalytics utilizarlos en un comando. UNLOAD
- Debe tener un bucket de Amazon S3 en la misma AWS región que su Timestream para obtener LiveAnalytics recursos.
- Para el bucket de S3 seleccionado, asegúrese de que la política de bucket de S3 también tenga permisos que permitan a Timestream exportar los datos LiveAnalytics .
- Las credenciales utilizadas para ejecutar la UNLOAD consulta deben tener los permisos de AWS Identity and Access Management (IAM) necesarios para que Timestream pueda LiveAnalytics escribir los datos en S3. Un ejemplo de política sería el siguiente:

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [{
            "Effect": "Allow",
            "Action": [
                "timestream:Select",
                "timestream:ListMeasures",
                "timestream:WriteRecords",
                "timestream:Unload"
            ],
            "Resource": "arn:aws:timestream:<region>:<account_id>:database/
<database_name>/table/<table_name>"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketAcl",
                "s3:PutObject",
                "s3:GetObjectMetadata",
                "s3:AbortMultipartUpload"
            ],
            "Resource": [
                "arn:aws:s3:::<S3_Bucket_Created>",
                "arn:aws:s3:::<S3_Bucket_Created>/*"
            ]
        }
    ]
}
```

Para obtener más información sobre estos permisos de escritura de S3, consulte la guía de Amazon Simple Storage Service. Si utiliza una KMS clave para cifrar los datos exportados, consulte lo siguiente para conocer IAM las políticas adicionales requeridas.

```
{
    "Version": "2012-10-17",
    "Statement": [
      {
        "Effect": "Allow",
        "Action": [
            "kms:DescribeKey",
            "kms:Decrypt",
            "kms:GenerateDataKey*"
        ],
        "Resource": "<account_id>-arn:aws:kms:<region>:<account_id>:key/*",
        "Condition": {
```

```
            "ForAnyValue:StringLike": {
                "kms:ResourceAliases": "alias/<Alias_For_Generated_Key>"
            }
        }
    }, {
        "Effect": "Allow",
        "Action": [
            "kms:CreateGrant"
        ],
        "Resource": "<account_id>-arn:aws:kms:<region>:<account_id>:key/*",
        "Condition": {
            "ForAnyValue:StringEquals": {
                "kms:EncryptionContextKeys": "aws:timestream:<database_name>"
            },
            "Bool": {
                "kms:GrantIsForAWSResource": true
            },
            "StringLike": {
                "kms:ViaService": "timestream.<region>.amazonaws.com"
            },
            "ForAnyValue:StringLike": {
                "kms:ResourceAliases": "alias/<Alias_For_Generated_Key>"
            }
        }
    }
]
}
```

Prácticas recomendadas de UNLOAD Timestream para LiveAnalytics

Las siguientes son las mejores prácticas relacionadas con el UNLOAD comando.

- La cantidad de datos que se pueden exportar al bucket de S3 mediante el UNLOAD comando no está limitada. Sin embargo, la consulta agota el tiempo de espera en 60 minutos y recomendamos no exportar más de 60 GB de datos en una sola consulta. Si necesita exportar más de 60 GB de datos, divida el trabajo en varias consultas.
- Si bien puede enviar miles de solicitudes a S3 para cargar los datos, se recomienda paralelizar las operaciones de escritura con varios prefijos de S3. Consulte la documentación aquí. La velocidad de API llamadas de S3 podría reducirse cuando varios lectores o grabadores acceden a la misma carpeta.

- Dado el límite de longitud de la clave S3 para definir un prefijo, recomendamos que los nombres de los cubos y carpetas tengan entre 10 y 15 caracteres, especialmente cuando se utilice una cláusula. `partitioned_by`
- Si recibes un 4XX o 5XX para las consultas que contienen la `UNLOAD` sentencia, es posible que los resultados parciales se escriban en el bucket de S3. Timestream for LiveAnalytics no elimina ningún dato del depósito. Antes de ejecutar otra `UNLOAD` consulta con el mismo destino de S3, se recomienda eliminar manualmente los archivos creados por la consulta fallida. Puede identificar los archivos escritos por una consulta fallida con la información correspondiente`QueryExecutionId`. En el caso de las consultas fallidas, Timestream for LiveAnalytics no exporta un archivo de manifiesto al bucket de S3.
- Timestream for LiveAnalytics utiliza la carga en varias partes para exportar los resultados de las consultas a S3. Cuando recibes un 4XX o 5XX de Timestream para consultas que contienen una UNLOAD declaración, Timestream LiveAnalytics for LiveAnalytics hace todo lo posible por evitar la carga de varias partes, pero es posible que algunas partes incompletas se queden atrás. Por lo tanto, te recomendamos configurar una limpieza automática de las cargas incompletas de varias partes en tu bucket de S3 siguiendo las instrucciones que se indican aquí.

Recomendaciones para acceder a los datos en CSV formato mediante un analizador CSV

- CSVlos analizadores no permiten tener el mismo carácter en el carácter delimitador, de escape y entre comillas.
- Algunos CSV analizadores no pueden interpretar tipos de datos complejos, como las matrices, por lo que recomendamos interpretarlos mediante un deserializador. JSON

Recomendaciones para acceder a los datos en formato Parquet

1. Si su caso de uso requiere que el esquema admita de UTF 8 a 8 caracteres, también conocido como nombre de columna, le recomendamos que utilice la biblioteca Parquet-MR.
2. La marca de tiempo de los resultados se representa como un entero de 12 bytes () INT96
3. Las series temporales se representarán de la siguiente manera`array<row<time, value>>`: otras estructuras anidadas utilizarán los tipos de datos correspondientes compatibles con el formato Parquet

Uso de la cláusula partition_by

- La columna utilizada en el `partitioned_by` campo debe ser la última columna de la consulta de selección. Si se usa más de una columna en el `partitioned_by` campo, las columnas deben ser las últimas columnas de la consulta de selección y estar en el mismo orden en que se usaron en el `partition_by` campo.
- Los valores de las columnas utilizados para dividir los datos (`partitioned_by`campo) solo pueden contener ASCII caracteres. Mientras que Timestream for LiveAnalytics permite de UTF 8 a 8 caracteres en los valores, S3 solo admite ASCII caracteres como claves de objeto.

Ejemplo de caso de uso de Timestream para UNLOAD LiveAnalytics

Suponga que está monitoreando las métricas de sesión de los usuarios, las fuentes de tráfico y las compras de productos de su sitio web de comercio electrónico. Utiliza Timestream LiveAnalytics para obtener información en tiempo real sobre el comportamiento de los usuarios y las ventas de productos y realizar análisis de marketing sobre los canales de tráfico (búsquedas orgánicas, redes sociales, tráfico directo, campañas de pago, etc.) que atraen a los clientes al sitio web.

Temas

- Exportación de los datos sin particiones
- Particionar los datos por canal
- Particionar los datos por evento
- Particionar los datos por canal y evento
- Archivos de manifiestos y metadatos
- Uso de rastreadores de Glue para crear el catálogo de datos de Glue

Exportación de los datos sin particiones

Desea exportar los dos últimos días de sus datos en CSV formato.

```
UNLOAD(SELECT user_id, ip_address, event, session_id, measure_name, time,
query, quantity, product_id, channel
FROM sample_clickstream.sample_shopping
WHERE time BETWEEN ago(2d) AND now())
TO 's3://<bucket_name>/withoutpartition'
WITH (  format='CSV',
```

```
compression='GZIP')
```

Particionar los datos por canal

Desea exportar los datos de los dos últimos días en CSV formato, pero le gustaría tener los datos de cada canal de tráfico en una carpeta independiente. Para ello, debe particionar los datos mediante la `channel` columna que se muestra a continuación.

```
UNLOAD(SELECT user_id, ip_address, event, session_id, measure_name, time,
query, quantity, product_id, channel
FROM sample_clickstream.sample_shopping
WHERE time BETWEEN ago(2d) AND now())
TO 's3://<bucket_name>/partitionbychannel/'
WITH (
partitioned_by = ARRAY ['channel'],
format='CSV',
compression='GZIP')
```

Particionar los datos por evento

Desea exportar los datos de los dos últimos días en CSV formato, pero le gustaría tener los datos de cada evento en una carpeta independiente. Para ello, debe particionar los datos mediante la `event` columna, tal y como se muestra a continuación.

```
UNLOAD(SELECT user_id, ip_address, channel, session_id, measure_name, time,
query, quantity, product_id, event
FROM sample_clickstream.sample_shopping
WHERE time BETWEEN ago(2d) AND now())
TO 's3://<bucket_name>/partitionbyevent/'
WITH (
partitioned_by = ARRAY ['event'],
format='CSV',
compression='GZIP')
```

Particionar los datos por canal y evento

Desea exportar los datos de los dos últimos días en CSV formato, pero le gustaría que los datos de cada canal y, dentro del canal, almacenaran cada evento en una carpeta independiente. Para ello, debe particionar los datos utilizando ambas `event` columnas, tal `channel` y como se muestra a continuación.

```
UNLOAD(SELECT user_id, ip_address, session_id, measure_name, time,
query, quantity, product_id, channel,event
FROM sample_clickstream.sample_shopping
WHERE time BETWEEN ago(2d) AND now())
TO 's3://<bucket_name>/partitionbychannelevent/'
WITH (
partitioned_by = ARRAY ['channel','event'],
format='CSV',
compression='GZIP')
```

Archivos de manifiestos y metadatos

Archivo de manifiesto

El archivo de manifiesto proporciona información sobre la lista de archivos que se exportan con la UNLOAD ejecución. El archivo de manifiesto está disponible en el bucket de S3 proporcionado con un nombre de archivo:`S3://bucket_name/<queryid>_<UUID>_manifest.json`. El archivo de manifiesto contendrá la URL de los archivos de la carpeta de resultados, el número de registros y el tamaño de los archivos respectivos, y los metadatos de la consulta (que son el total de bytes y filas exportados a S3 para la consulta).

```
{
  "result_files": [
    {
        "url":"s3://my_timestream_unloads/ec2_metrics/
AEDAGANLHLBH4OLISD3CVOZZRWPX5GV2XCXRBKCVD554N6GWPWWXBP7LSG74V2Q_1448466917_szCL4YgVYzGXj2lS.gz"
        "file_metadata":
            {
                "content_length_in_bytes": 32295,
                "row_count": 10
            }
    },
    {
        "url":"s3://my_timestream_unloads/ec2_metrics/
AEDAGANLHLBH4OLISD3CVOZZRWPX5GV2XCXRBKCVD554N6GWPWWXBP7LSG74V2Q_1448466917_szCL4YgVYzGXj2lS.gz"
        "file_metadata":
            {
                "content_length_in_bytes": 62295,
                "row_count": 20
            }
    },
  ],
```

```
  "query_metadata":
    {
      "content_length_in_bytes": 94590,
      "total_row_count": 30,
      "result_format": "CSV",
      "result_version": "Amazon Timestream version 1.0.0"
    },
  "author": {
        "name": "Amazon Timestream",
        "manifest_file_version": "1.0"
  }
}
```

Metadatos

El archivo de metadatos proporciona información adicional sobre el conjunto de datos, como el nombre de la columna, el tipo de columna y el esquema. <queryid>El archivo de metadatos está disponible en el bucket de S3 proporcionado con un nombre de archivo: S3: //bucket_name/ _< >_metadata.json UUID

A continuación se muestra un ejemplo de un archivo de metadatos.

```
{
    "ColumnInfo": [
        {
            "Name": "hostname",
            "Type": {
                "ScalarType": "VARCHAR"
            }
        },
        {
            "Name": "region",
            "Type": {
                "ScalarType": "VARCHAR"
            }
        },
        {
            "Name": "measure_name",
            "Type": {
                "ScalarType": "VARCHAR"
            }
        },
        {
```

```
            "Name": "cpu_utilization",
            "Type": {
                "TimeSeriesMeasureValueColumnInfo": {
                    "Type": {
                        "ScalarType": "DOUBLE"
                    }
                }
            }
        }
    ],
    "Author": {
        "Name": "Amazon Timestream",
        "MetadataFileVersion": "1.0"
    }
}
```

La información de las columnas que se comparte en el archivo de metadatos tiene la misma estructura que la que `ColumnInfo` se envía en la API respuesta a las `SELECT` consultas.

Uso de rastreadores de Glue para crear el catálogo de datos de Glue

1. Inicie sesión en su cuenta con las credenciales de administrador para la siguiente validación.
2. Cree una base de datos de Crawler for Glue siguiendo las instrucciones que se proporcionan aquí. Tenga en cuenta que la carpeta S3 que se debe proporcionar en la fuente de datos debe ser la carpeta de `UNLOAD` resultados, por ejemplo. `s3://my_timestream_unloads/results`
3. Ejecute el rastreador siguiendo las instrucciones que se indican aquí.
4. Vea la tabla de Glue.
   - Ve a AWS Glue → Tables.
   - Verás una nueva tabla creada con el prefijo de tabla proporcionado al crear el rastreador.
   - Puede ver la información del esquema y la partición haciendo clic en la vista de detalles de la tabla.

Los siguientes son otros AWS servicios y proyectos de código abierto que utilizan el catálogo de datos de AWS Glue.

- Amazon Athena: para obtener más información, consulte Descripción de las tablas, bases de datos y catálogos de datos en la Guía del usuario de Amazon Athena.

- Amazon Redshift Spectrum: para obtener más información, consulte Consulta de datos externos con Amazon Redshift Spectrum en la Guía para desarrolladores de bases de datos de Amazon Redshift.
- Amazon EMR: para obtener más información, consulta Cómo usar políticas basadas en recursos para el EMR acceso de Amazon al catálogo de datos de AWS Glue en la Guía de EMR administración de Amazon.
- AWS Cliente de Glue Data Catalog para Apache Hive Metastore: para obtener más información sobre este GitHub proyecto, consulte Cliente de AWS Glue Data Catalog para Apache Hive Metastore.

Límites de From Timestream para UNLOAD LiveAnalytics

Los siguientes son los límites relacionados con el `UNLOAD` comando.

- La simultaneidad de las consultas que utilizan la `UNLOAD` sentencia es de 1 consulta por segundo (QPS). Superar la tasa de consultas puede provocar una limitación.
- Las consultas que contienen `UNLOAD` una sentencia pueden exportar como máximo 100 particiones por consulta. Se recomienda comprobar el recuento específico de la columna seleccionada antes de utilizarla para particionar los datos exportados.
- Las consultas que contienen `UNLOAD` sentencias caducan después de 60 minutos.
- El tamaño máximo de los archivos que crea la `UNLOAD` declaración en Amazon S3 es de 78 GB.

Para conocer otros límites de Timestream, consulte LiveAnalytics Cuotas

Uso de la información sobre consultas para optimizar las consultas en Amazon Timestream

Query Insights es una función de ajuste del rendimiento que le ayuda a optimizar sus consultas, mejorar su rendimiento y reducir los costes. Con la información sobre las consultas, puede evaluar la eficacia de las consultas al reducir las consultas en función de las claves de partición temporal, temporal y espacial. Con la información de las consultas, también puede identificar áreas de mejora para mejorar el rendimiento de las consultas. Además, con la información sobre las consultas, puede evaluar la eficacia con la que sus consultas utilizan la indexación basada en el tiempo y en la clave de partición para optimizar la recuperación de datos. Para optimizar el rendimiento de las consultas, es esencial ajustar los parámetros temporales y espaciales que rigen la ejecución de las consultas.

Temas

Ventajas de la información sobre consultas

Los principales beneficios del uso de la información sobre consultas son los siguientes:

- Identificación de consultas ineficientes: Query Insights proporciona información sobre cómo depurar las tablas a las que accede la consulta en función del tiempo y de los atributos. Esta información le ayuda a identificar las tablas a las que no se accede de forma óptima.
- Optimización del modelo de datos y el particionamiento: puede utilizar la información de las consultas para acceder a su modelo de datos y su estrategia de particionamiento y ajustarlos.
- Ajustar las consultas: la información sobre las consultas destaca las oportunidades de utilizar los índices de forma más eficaz.

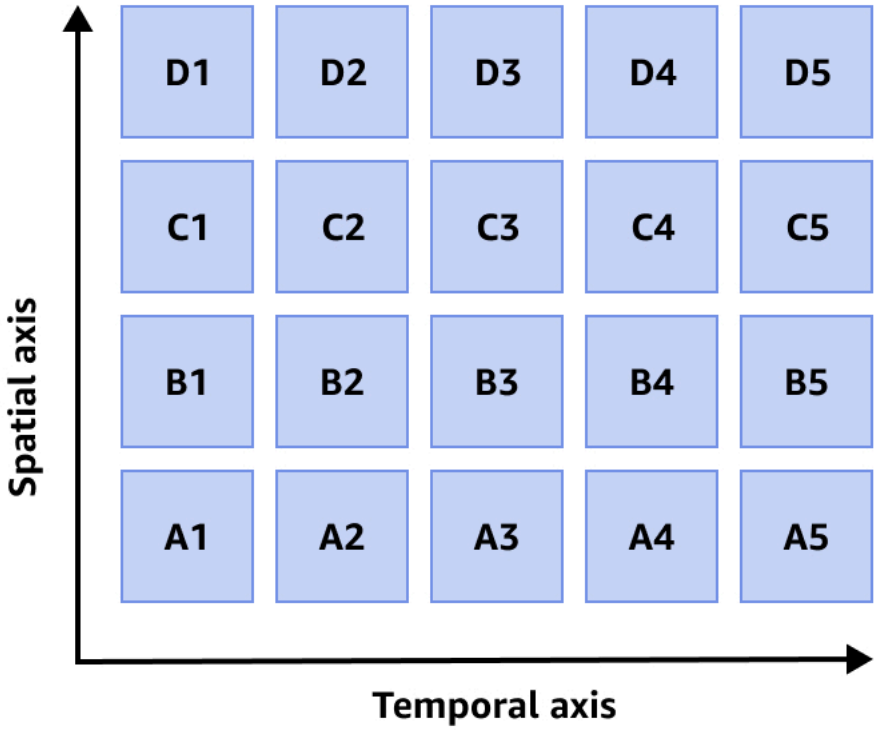
Optimización del acceso a los datos en Amazon Timestream

Puede optimizar los patrones de acceso a los datos en Amazon Timestream mediante el esquema de particionamiento de Timestream o técnicas de organización de datos.

Temas

Esquema de particionamiento de Timestream

Amazon Timestream utiliza un esquema de particionamiento altamente escalable en el que cada tabla de Timestream puede tener cientos, miles o incluso millones de particiones independientes. Un servicio de indexación y seguimiento de particiones de alta disponibilidad gestiona la partición, lo que minimiza el impacto de los errores y hace que el sistema sea más resistente.



Organización de datos

Timestream almacena cada punto de datos que ingiere en una sola partición. Al introducir datos en una tabla Timestream, Timestream crea particiones automáticamente en función de las marcas de tiempo, la clave de partición y otros atributos de contexto de los datos. Además de particionar los datos a tiempo (partición temporal), Timestream también divide los datos en función de la clave de partición seleccionada y otras dimensiones (partición espacial). Este enfoque está diseñado para distribuir el tráfico de escritura y permitir una depuración eficaz de los datos para las consultas.

La función de información sobre consultas proporciona información valiosa sobre la eficiencia de la consulta, que incluye la cobertura espacial y la cobertura temporal de las consultas.

Temas

- QuerySpatialCoverage
- QueryTemporalCoverage

QuerySpatialCoverage

La QuerySpatialCoveragemétrica proporciona información sobre la cobertura espacial de la consulta ejecutada y de la tabla con el ajuste espacial más ineficiente. Esta información puede ayudarlo a identificar las áreas de mejora en la estrategia de partición para mejorar la reducción espacial. El valor de la `QuerySpatialCoverage` métrica oscila entre 0 y 1. Cuanto más bajo sea el valor de la métrica, más óptima será la depuración de las consultas en el eje espacial. Por ejemplo, un valor de 0,1 indica que la consulta explora el 10% del eje espacial. Un valor de 1 indica que la consulta escanea el 100% del eje espacial.

Example Uso de la información de las consultas para analizar la cobertura espacial de una consulta

Supongamos que tiene una base de datos de Timestream que almacena datos meteorológicos. Suponga que la temperatura se registra cada hora desde estaciones meteorológicas ubicadas en diferentes estados de los Estados Unidos. Imagine que elige `State` la clave de partición definida por el cliente (CDPK) para particionar los datos por estado.

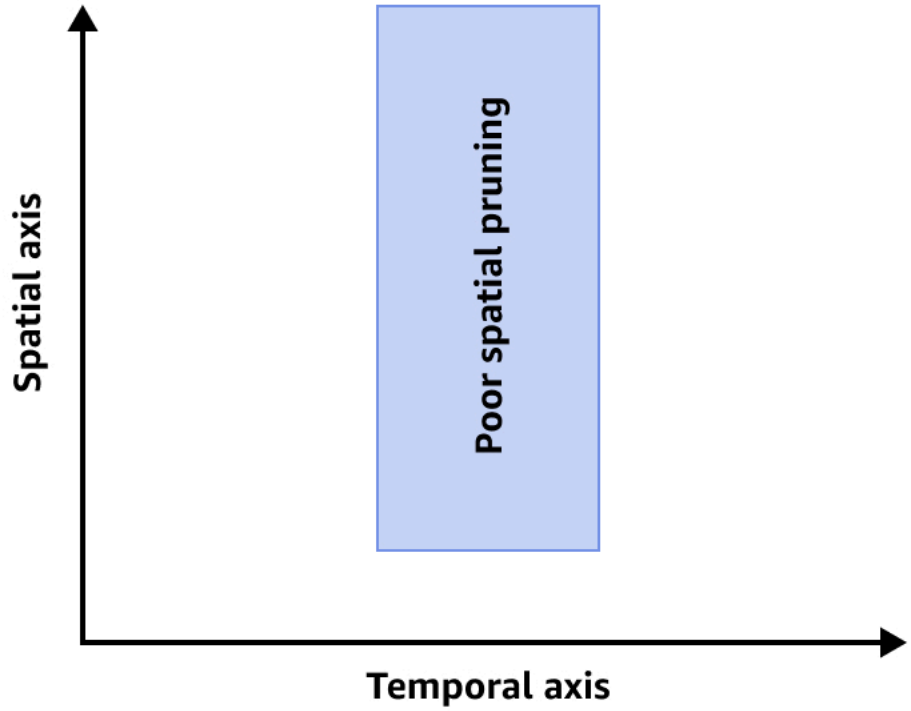
Supongamos que ejecuta una consulta para recuperar la temperatura media de todas las estaciones meteorológicas de California entre las 14 y las 16 horas de un día específico. En el siguiente ejemplo, se muestra la consulta para este escenario.

```
SELECT AVG(temperature)
FROM "weather_data"."hourly_weather"
WHERE time >= '2024-10-01 14:00:00' AND time < '2024-10-01 16:00:00'
  AND state = 'CA';
```

Con la función de información sobre consultas, puede analizar la cobertura espacial de la consulta. Imagine que la `QuerySpatialCoverage` métrica devuelve un valor de 0,02. Esto significa que la consulta solo escaneó el 2% del eje espacial, lo cual es eficiente. En este caso, la consulta pudo reducir el rango espacial de manera efectiva, recuperando únicamente datos de California e ignorando los datos de otros estados.

Por el contrario, si la `QuerySpatialCoverage` métrica devolviera un valor de 0,8, indicaría que la consulta escaneó el 80% del eje espacial, lo que es menos eficiente. Esto podría sugerir que es necesario afinar la estrategia de partición para mejorar la depuración espacial. Por ejemplo, puede seleccionar la clave de partición como ciudad o región en lugar de como estado. Al analizar la `QuerySpatialCoverage` métrica, puede identificar oportunidades para optimizar su estrategia de particionamiento y mejorar el rendimiento de sus consultas.

La siguiente imagen muestra una mala depuración espacial.



Para mejorar la eficiencia de la poda espacial, puede realizar una de las siguientes acciones o ambas:

- `measure_name`Añada la clave de partición predeterminada o utilice los CDPK predicados en la consulta.
- Si ya ha agregado los atributos mencionados en el punto anterior, elimine las funciones relacionadas con estos atributos o cláusulas, como. `LIKE`

QueryTemporalCoverage

La `QueryTemporalCoverage` métrica proporciona información sobre el rango temporal explorado por la consulta ejecutada, incluida la tabla con el rango de tiempo más grande escaneado. El valor de la `QueryTemporalCoverage` métrica es el intervalo de tiempo representado en nanosegundos. Cuanto más bajo sea el valor de esta métrica, más óptima será la reducción de la consulta en el rango temporal. Por ejemplo, una consulta que escanea los últimos minutos de datos es más eficaz que una consulta que escanea todo el intervalo de tiempo de la tabla.

Example

Supongamos que tiene una base de datos de Timestream que almacena datos de sensores de IoT, con mediciones tomadas cada minuto desde dispositivos ubicados en una planta de fabricación. Suponga que ha dividido sus datos por. `device_ID`

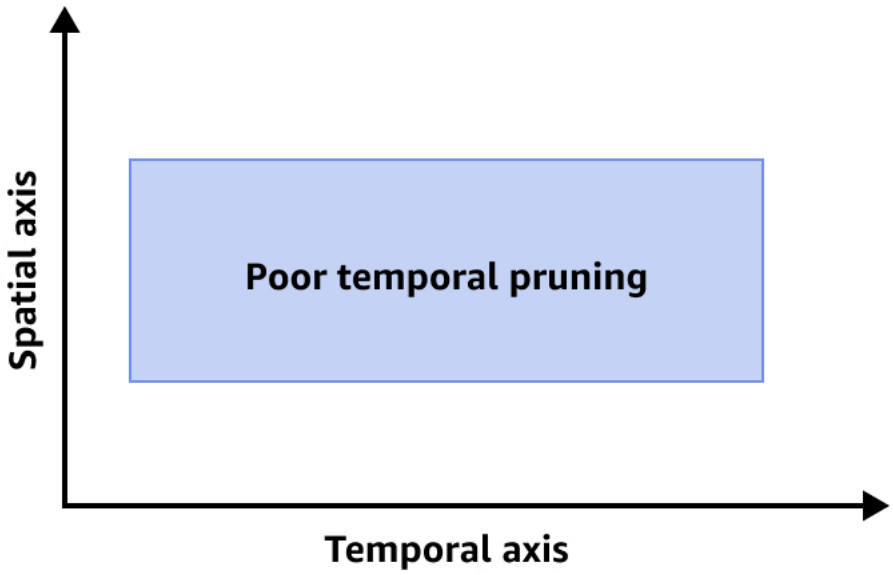
Supongamos que ejecuta una consulta para recuperar la lectura media del sensor de un dispositivo específico durante los últimos 30 minutos. En el siguiente ejemplo, se muestra la consulta para este escenario.

```
SELECT AVG(sensor_reading)
FROM "sensor_data"."factory_1"
WHERE device_id = 'DEV_123'
  AND time >= NOW() - INTERVAL 30 MINUTE and time < NOW();
```

Con la función de información sobre consultas, puede analizar el rango temporal escaneado por la consulta. Imagine que la `QueryTemporalCoverage` métrica devuelve un valor de 1800000000000 nanosegundos (30 minutos). Esto significa que la consulta solo escaneó los datos de los últimos 30 minutos, lo que constituye un intervalo temporal relativamente estrecho. Esto es una buena señal porque indica que la consulta pudo reducir eficazmente la partición temporal y solo recuperó los datos solicitados.

Por el contrario, si la `QueryTemporalCoverage` métrica arrojó un valor de 1 año en nanosegundos, indica que la consulta analizó un intervalo de tiempo de un año en la tabla, lo que resulta menos eficiente. Esto podría sugerir que la consulta no está optimizada para la reducción temporal, por lo que podría mejorarla añadiendo filtros de tiempo.

La siguiente imagen muestra una poda temporal deficiente.



Para mejorar la poda temporal, le recomendamos que realice una de las siguientes acciones o todas ellas:

- Agregue los predicados de tiempo que faltan en la consulta y asegúrese de que los predicados de tiempo estén ajustando el intervalo de tiempo deseado.
- Elimine funciones, como las relacionadas con `MAX()` los predicados de tiempo.
- Agregue predicados de tiempo a todas las subconsultas. Esto es importante si las subconsultas unen tablas grandes o realizan operaciones complejas.

Habilitación de la información sobre consultas en Amazon Timestream

Puede habilitar la información de las consultas para sus consultas con la información proporcionada directamente a través de la respuesta a la consulta. Habilitar la información de las consultas no requiere infraestructura adicional ni implica costes adicionales. Al habilitar la información sobre las consultas, devuelve campos de metadatos relacionados con el rendimiento de la consulta, además de los resultados de la consulta, como parte de la respuesta a la consulta. Puedes usar esta información para ajustar tus consultas a fin de mejorar el rendimiento de las consultas y reducir el costo de las consultas.

Para obtener información sobre cómo habilitar la información sobre las consultas, consulteEjecutar una consulta.

Para ver ejemplos de las respuestas devueltas al habilitar la información sobre las consultas, consulte Ejemplos de consultas programadas.

> **Note**
>
> - Cuando habilitas la información sobre las consultas, la frecuencia limita la consulta a 1 consulta por segundo (QPS). Para evitar que el rendimiento se vea afectado, le recomendamos encarecidamente que habilite la información sobre las consultas solo durante la fase de evaluación de las consultas, antes de implementarlas en producción.
> - La información proporcionada en las estadísticas de las consultas acaba siendo coherente, lo que significa que podría cambiar a medida que se vayan incorporando nuevos datos a las tablas.

Optimizar las consultas mediante la información y la respuesta a las consultas

Supongamos que utiliza Amazon Timestream LiveAnalytics para monitorizar el consumo de energía en varios lugares. Imagine que tiene dos tablas en su base de datos denominadas `raw-metrics` y. `aggregate-metrics`

La `raw-metrics` tabla almacena datos de energía detallados a nivel de dispositivo y contiene las siguientes columnas:

- Timestamp
- Estado, por ejemplo, Washington
- Device ID (ID de dispositivo)
- Consumo de energía

Los datos de esta tabla se recopilan y almacenan de forma minute-by-minute granulada. La tabla se utiliza `State` como. CDPK

La `aggregate-metrics` tabla almacena el resultado de una consulta programada para agregar los datos de consumo de energía de todos los dispositivos cada hora. Esta tabla contiene las siguientes columnas:

- Timestamp

- Estado, por ejemplo, Washington
- Consumo total de energía

La `aggregate-metrics` tabla almacena estos datos con una granularidad horaria. La tabla se utiliza `State` como. CDPK

Temas

- Consultando el consumo de energía de las últimas 24 horas
- Optimizar la consulta para el rango temporal
- Optimización de la consulta de cobertura espacial
- Rendimiento de consultas mejorado

Consultando el consumo de energía de las últimas 24 horas

Supongamos que desea extraer la energía total consumida en Washington en las últimas 24 horas. Para encontrar estos datos, puede aprovechar los puntos fuertes de ambas tablas: `raw-metrics` y`aggregate-metrics`. La `aggregate-metrics` tabla proporciona datos sobre el consumo de energía por hora de las últimas 23 horas, mientras que la `raw-metrics` tabla ofrece datos minuciosos de la última hora. Al consultar ambas tablas, puede obtener una imagen completa y precisa del consumo de energía en Washington durante las últimas 24 horas.

```
SELECT am.time, am.state, am.total_energy_consumption,
rm.time, rm.state, rm.device_id, rm.energy_consumption
FROM
 "metrics"."aggregate-metrics" am
 LEFT JOIN "metrics"."raw-metrics" rm ON am.state = rm.state
WHERE rm.time >= ago(1h) and rm.time < now()
```

Esta consulta de ejemplo se proporciona únicamente con fines ilustrativos y es posible que no funcione tal cual. Su objetivo es demostrar el concepto, pero es posible que tengas que modificarlo para adaptarlo a tu caso de uso o entorno específicos.

Tras ejecutar esta consulta, es posible que notes que el tiempo de respuesta de la consulta es más lento de lo esperado. Para identificar la causa principal de este problema de rendimiento, puede utilizar la función de información sobre las consultas para analizar el rendimiento de la consulta y optimizar su ejecución.

En el siguiente ejemplo, se muestra la respuesta de Query Insights.

```
queryInsightsResponse={
                QuerySpatialCoverage: {
                    Max: {
                        Value: 1.0,
                        TableArn: arn:aws:timestream:us-east-1:123456789012:database/
metrics/table/raw-metrics,
                        PartitionKey: [State]
                    }
                },
                QueryTemporalRange: {
                    Max: {
                        Value:31540000000000000 //365 days,
                        TableArn: arn:aws:timestream:us-east-1:123456789012:database/
metrics/table/aggregate-metrics
                    }
                },
                QueryTableCount: 2,
                OutputRows: 83,
                OutputBytes: 590
```

La respuesta de Query Insights proporciona la siguiente información:

- Rango temporal: la consulta analizó un rango temporal excesivo de 365 días para la `aggregate-metrics` tabla. Esto indica un uso ineficiente del filtrado temporal.
- Cobertura espacial: la consulta escaneó todo el rango espacial (100%) de la `raw-metrics` tabla. Esto sugiere que el filtrado espacial no se está utilizando de forma eficaz.

Si la consulta accede a más de una tabla, Query Insights proporciona las métricas de la tabla con el patrón de acceso menos óptimo.

Optimizar la consulta para el rango temporal

En función de la respuesta de la información sobre la consulta, puede optimizar la consulta para el rango temporal, como se muestra en el siguiente ejemplo.

```
SELECT am.time, am.state, am.total_energy_consumption,
rm.time, rm.state, rm.device_id, rm.energy_consumption
FROM
  "metrics"."aggregate-metrics" am
```

```
  LEFT JOIN "metrics"."raw-metrics" rm ON am.state = rm.state
WHERE
  am.time >=  ago(23h) and am.time < now()
  AND rm.time >=  ago(1h) and rm.time < now()
  AND rm.state = 'Washington'
```

Si vuelve a ejecutar el `QueryInsights` comando, devolverá la siguiente respuesta.

```
queryInsightsResponse={
                QuerySpatialCoverage: {
                    Max: {
                        Value: 1.0,
                        TableArn: arn:aws:timestream:us-east-1:123456789012:database/
metrics/table/aggregate-metrics,
                        PartitionKey: [State]
                    }
                },
                QueryTemporalRange: {
                    Max: {
                        Value: 82800000000000 //23 hours,
                        TableArn: arn:aws:timestream:us-east-1:123456789012:database/
metrics/table/aggregate-metrics
                    }
                },
                QueryTableCount: 2,
                OutputRows: 83,
                OutputBytes: 590
```

Esta respuesta muestra que la cobertura espacial de la `aggregate-metrics` tabla sigue siendo del 100%, lo que resulta ineficiente. En la siguiente sección, se muestra cómo optimizar la consulta de cobertura espacial.

Optimización de la consulta de cobertura espacial

En función de la respuesta de información sobre la consulta, puede optimizar la consulta para determinar la cobertura espacial, como se muestra en el siguiente ejemplo.

```
SELECT am.time, am.state, am.total_energy_consumption,
rm.time, rm.state, rm.device_id, rm.energy_consumption
FROM
  "metrics"."aggregate-metrics" am
  LEFT JOIN "metrics"."raw-metrics" rm ON am.state = rm.state
```

```
WHERE
  am.time >=  ago(23h) and am.time < now()
  AND am.state ='Washington'
  AND rm.time >=  ago(1h) and rm.time < now()
  AND rm.state = 'Washington'
```

Si vuelve a ejecutar el `QueryInsights` comando, devolverá la siguiente respuesta.

```
queryInsightsResponse={
                QuerySpatialCoverage: {
                    Max: {
                        Value: 0.02,
                        TableArn: arn:aws:timestream:us-east-1:123456789012:database/
metrics/table/aggregate-metrics,
                        PartitionKey: [State]
                    }
                },
                QueryTemporalRange: {
                    Max: {
                        Value: 82800000000000 //23 hours,
                        TableArn: arn:aws:timestream:us-east-1:123456789012:database/
metrics/table/aggregate-metrics
                    }
                },
                QueryTableCount: 2,
                OutputRows: 83,
                OutputBytes: 590
```

Rendimiento de consultas mejorado

Tras optimizar la consulta, Query Insights proporciona la siguiente información:

- La poda temporal de la `aggregate-metrics` mesa es de 23 horas. Esto indica que solo se escanean 23 horas del rango temporal.
- La reducción espacial de la `aggregate-metrics` mesa es de 0.02. Esto indica que solo se escanea el 2% de los datos del rango espacial de la tabla. La consulta escanea una parte muy pequeña de las tablas, lo que permite obtener un rendimiento rápido y reducir la utilización de los recursos. La mejora de la eficiencia de depuración indica que la consulta ahora está optimizada para mejorar el rendimiento.

Trabajando con AWS Backup

La funcionalidad de protección de datos de Amazon Timestream LiveAnalytics for es una solución totalmente gestionada que le ayuda a cumplir sus requisitos de conformidad normativa y continuidad empresarial. La funcionalidad se habilita mediante la integración nativa con AWS Backup un servicio de respaldo unificado diseñado para simplificar la creación, migración, restauración y eliminación de copias de seguridad, al tiempo que proporciona mejores informes y auditorías. Mediante la integración con AWS Backup, puede utilizar una solución de protección de datos centralizada, totalmente gestionada y basada en políticas para crear copias de seguridad inmutables y gestionar de forma centralizada la protección de los datos de su aplicación, abarcando Timestream y otros servicios compatibles. AWS AWS Backup

Para utilizar la funcionalidad, debe habilitar esta opción para proteger sus recursos de Timestream. AWS Backup Las opciones de suscripción se aplican a la cuenta y AWS región específicas, por lo que es posible que tengas que registrarte en varias regiones con la misma cuenta. Para obtener más información sobre AWS Backup, consulte la Guía para AWS Backup desarrolladores.

La funcionalidad de protección de datos disponible AWS Backup incluye lo siguiente.

Copias de seguridad programadas: puede configurar copias de seguridad programadas periódicamente de su Timestream para LiveAnalytics tablas mediante planes de copia de seguridad.

Copia entre cuentas y regiones: puede copiar automáticamente sus copias de seguridad a otra bóveda de copias de seguridad de una AWS región o cuenta diferente, lo que le permite cumplir con sus requisitos de protección de datos.

Almacenamiento en frío por niveles: puede configurar sus copias de seguridad para implementar reglas de ciclo de vida a fin de eliminarlas o hacer la transición de las copias de seguridad a un almacenamiento en frío. Esto puede ayudarle a optimizar los costos de las copias de seguridad.

Etiquetas: puede etiquetar automáticamente sus copias de seguridad con fines de facturación y asignación de costos.

Cifrado: los datos de la copia de seguridad se almacenan en la AWS Backup bóveda. Esto le permite cifrar y proteger sus copias de seguridad mediante una AWS KMS clave que es independiente de la clave de cifrado de Timestream for LiveAnalytics Table.

Proteja las copias de seguridad mediante el WORM modelo: puede utilizar AWS Backup Vault Lock para habilitar la configuración write-once-read-many (WORM) para las copias de seguridad. Con AWS Backup Vault Lock, puede añadir una capa de defensa adicional que proteja las copias de

seguridad contra operaciones de eliminación involuntarias o malintencionadas, los cambios en los períodos de retención de las copias de seguridad y las actualizaciones de la configuración del ciclo de vida. Para obtener más información, consulte Bloqueo de almacenes de AWS Backup.

La función de protección de datos está disponible en todas las regiones. Para obtener más información sobre la funcionalidad, consulte la Guía para AWS Backup desarrolladores.

Hacer copias de seguridad y restaurar tablas de Timestream: cómo funciona

Puede crear copias de seguridad de sus tablas de Amazon Timestream. En esta sección se proporciona información general acerca de qué ocurre durante los procesos de copia de seguridad y restauración.

Temas

Copias de seguridad

Puede utilizar la función de copia de seguridad bajo demanda para crear copias de seguridad completas de su Amazon Timestream LiveAnalytics para tablas. En esta sección se proporciona información general acerca de qué ocurre durante los procesos de copia de seguridad y restauración.

Puede crear una copia de seguridad de sus datos de Timestream con una granularidad de tabla. Puede iniciar una copia de seguridad de la tabla seleccionada mediante la consola Timestream, la consola o AWS Backup . SDK CLI La copia de seguridad se crea de forma asíncrona y todos los datos de la tabla hasta la hora de inicio de la copia de seguridad se incluyen en la copia de seguridad. Sin embargo, existe la posibilidad de que algunos de los datos ingresados en la tabla mientras se está realizando la copia de seguridad también se incluyan en la copia de seguridad. Para proteger sus datos, puede crear una copia de seguridad a pedido de una sola vez o programar una copia de seguridad periódica de la tabla.

Mientras se está realizando una copia de seguridad, no puede hacer lo siguiente.

- Pausar o cancelar la operación de backup.
- Eliminar la tabla de origen del backup.

- Deshabilitar los backup de una tabla si uno de ellos está en curso.

Una vez configurada, AWS Backup proporciona programas de respaldo automatizados, administración de la retención y administración del ciclo de vida, lo que elimina la necesidad de scripts personalizados y procesos manuales. Para obtener más información, consulte la Guía para AWS Backup desarrolladores

Todas las secuencias de tiempo de las LiveAnalytics copias de seguridad son de naturaleza incremental, lo que implica que la primera copia de seguridad de una tabla es una copia de seguridad completa y cada copia de seguridad posterior de la misma tabla es una copia de seguridad incremental, en la que solo se copian los cambios en los datos desde la última copia de seguridad. Como los datos de Timestream for LiveAnalytics se almacenan en un conjunto de particiones, todas las particiones que hayan cambiado debido a la ingesta de nuevos datos o a las actualizaciones de los datos existentes desde la última copia de seguridad se copian en las copias de seguridad posteriores.

Si utiliza Timestream para LiveAnalytics consola, las copias de seguridad creadas para todos los recursos de la cuenta aparecen en la pestaña Copias de seguridad. Además, las copias de seguridad también se muestran en los detalles de la tabla.

Restauraciones

Puede restaurar una tabla desde el Timestream para LiveAnalytics consolaSDK, AWS Backup consola o. AWS CLI Puede restaurar todos los datos de la copia de seguridad o configurar los ajustes de retención de la tabla para restaurar los datos seleccionados. Al iniciar una restauración, puede configurar los siguientes ajustes de la tabla.

- Database Name (Nombre de base de datos)
- Nombre de la tabla
- Retención del almacén de memoria
- Retención magnética de almacenamiento
- Habilite las escrituras con almacenamiento magnético
- Ubicación de los registros de errores de S3 (opcional)
- IAMfunción que AWS Backup asumirá al restaurar la copia de seguridad

Las configuraciones anteriores son independientes de la tabla de origen. Para restaurar todos los datos de la copia de seguridad, le recomendamos que configure los ajustes de la nueva tabla de

manera que la suma del período de retención del almacén de memoria y el período de retención del almacenamiento magnético sea mayor que la diferencia entre la marca de tiempo más antigua y la actual. Si selecciona una copia de seguridad incremental para restaurarla, se restauran todos los datos (la incremental más los datos completos subyacentes). Tras una restauración correcta, la tabla estará activa y podrá realizar operaciones de ingestión o consulta en la tabla restaurada. Sin embargo, no puede realizar estas operaciones mientras la restauración esté en curso. Una vez restaurada, la tabla es similar a cualquier otra tabla de tu cuenta.

Example Restaure todos los datos de una copia de seguridad

Este ejemplo tiene las siguientes suposiciones.

Marca de tiempo más antigua: `August 1, 2021 0:00:00`

- Ahora — `November 9, 2022 0:00:00`

Para restaurar todos los datos de una copia de seguridad, introduzca y compare los valores de la siguiente manera.

1. Introduzca Memory Store Retention y Magnetic Store Retention. Por ejemplo, suponga estos valores.
   - Retención del almacén de memoria: 12 horas
   - Retención de almacenamiento magnético: 500 días
2. Calcula la suma de la retención del almacén de memoria y la retención del almacén magnético.

```
12 hours + (500 * 24 hours) =
12 hours + 12,000 hours =
12,012 hours
```

3. Encuentra la diferencia entre la marca de tiempo más antigua y la actual.

```
November 9, 2022 0:00:00 - August 1, 2021 0:00:00 =
465 days =
465 * 24 hours =
11,160 hours
```

4. Asegúrese de que la suma de los valores de retención en el segundo paso sea mayor que la diferencia de tiempos en el tercer paso. Ajuste los tiempos de retención si es necesario.

```
12,012 > 11,160
```

```
true
```

Example Restaure los datos seleccionados de una copia de seguridad

En este ejemplo se parte de la siguiente suposición.

- Ahora — `November 9, 2022 0:00:00`

Para restaurar solo los datos seleccionados de una copia de seguridad, introduzca y compare los valores de la siguiente manera.

1. Determine la marca de tiempo más temprana requerida. Por ejemplo, supongamos. `December 4, 2021 0:00:00`
2. Encuentra la diferencia entre la primera marca de tiempo requerida y la actual.

```
November 9, 2022 0:00:00 - December 4, 2021 0:00:00 =
340 days =
340 * 24 hours =
8,160 hours
```

3. Introduzca el valor deseado para la retención del almacén de memoria. Por ejemplo, introduzca 12 horas.
4. Reste el valor de la diferencia en el segundo paso.

```
8,160 hours - 12 hours =
8148 hours
```

5. Introduzca ese valor para la retención de Magnetic Store.

Puede copiar una copia de seguridad de su Timestream para los datos de LiveAnalytics la tabla en una AWS región diferente y, a continuación, restaurarla en esa nueva región. Puedes copiar y restaurar copias de seguridad entre regiones AWS comerciales y regiones AWS GovCloud (de EE. UU.). Solo pagará por los datos que copie de la región de origen y por los que restaure en una nueva tabla de la región de destino.

Una vez restaurada la tabla, debe configurar manualmente lo siguiente en la tabla restaurada.

- AWS Políticas de Identity and Access Management (IAM)

- Etiquetas
- Consultas programadas

Los tiempos de restauración están directamente relacionados con la configuración de las tablas. Estos incluyen el tamaño de las tablas, el número de particiones subyacentes, la cantidad de datos restaurados en la memoria almacenada y otras variables. Una práctica recomendada a la hora de planificar la recuperación ante desastres es documentar periódicamente los tiempos medios de finalización de la restauración y establecer cómo estos tiempos afectan al objetivo general de tiempo de recuperación (RTO).

Todas las API acciones y la consola de backup y restauración se capturan y registran AWS CloudTrail para su registro, supervisión continua y auditoría.

Creación de copias de seguridad de las tablas de Amazon Timestream

En esta sección se describe cómo habilitar AWS Backup y crear copias de seguridad programadas y bajo demanda para Amazon Timestream.

Temas

Habilitar la protección AWS Backup de Timestream para los datos LiveAnalytics

Debe habilitar su uso AWS Backup con Timestream para. LiveAnalytics

Para habilitarlo AWS Backup en la LiveAnalytics consola Timestream, lleve a cabo los siguientes pasos.

1. Inicie sesión en la Consola de AWS License Manager.
2. Aparece un banner emergente en la parte superior de la página del LiveAnalytics panel de control de Timestream for para permitir la compatibilidad con Timestream for AWS Backup data. LiveAnalytics De lo contrario, en el panel de navegación, selecciona Copias de seguridad.
3. En la ventana Backup, verá el banner para activarlo AWS Backup. Seleccione Habilitar.

Data Protection Through ya AWS Backup está disponible para su Timestream for LiveAnalytics tables.

Para habilitarlo AWS Backup, consulte la AWS Backup documentación para habilitarlo mediante consola y mediante programación.

Si decides desactivar la protección AWS Backup de LiveAnalytics datos en tu Timestream después de haberla activado, inicia sesión a través de la AWS Backup consola y mueve el botón hacia la izquierda.

Si no puedes activar o desactivar las AWS Backup funciones, es posible que tu AWS administrador deba realizar esas acciones.

Creación de copias de seguridad bajo demanda

Para crear una copia de seguridad bajo demanda de un Timestream for LiveAnalytics Table, sigue estos pasos.

1. Inicie sesión en la Consola de AWS License Manager.
2. En el panel de navegación del lado izquierdo de la consola, elija Backups.
3. Seleccione Create on-demand backup (Crear copia de seguridad bajo demanda).
4. Continúe seleccionando la configuración en la ventana de copia de seguridad.
5. Puede crear una copia de seguridad ahora, iniciar una copia de seguridad inmediatamente o seleccionar una ventana de copia de seguridad para iniciar la copia de seguridad.
6. Seleccione la política de administración del ciclo de vida de su copia de seguridad. Puede transferir los datos de la copia de seguridad a un almacenamiento en frío, donde debe conservar la copia de seguridad durante un mínimo de 90 días. Puede establecer el período de retención necesario para la copia de seguridad. Puede seleccionar una bóveda existente o bien seleccionar crear una nueva bóveda de copia de seguridad para ir a la AWS Backup consola y crear una nueva bóveda de copia de seguridad <documentation link on creating a new backup vault here>
7. Seleccione la IAM función adecuada.
8. Si desea asignar una o varias etiquetas a su copia de seguridad bajo demanda, introduzca una key (clave) y un value (valor) opcional y elija Add tag (Añadir etiqueta).
9. Elija crear una copia de seguridad bajo demanda. Esto lo llevará a la página Backup, donde verá una lista de trabajos.

10 Elija el ID de trabajo de copia de seguridad que corresponda al recurso del que desea realizar la copia de seguridad para ver los detalles de ese trabajo.

Copias de seguridad programadas

Para programar una copia de seguridad, consulte Crear una copia de seguridad programada.

Restauración de una copia de seguridad de una tabla de Amazon Timestream

En esta sección se describe cómo restaurar una copia de seguridad de una tabla de Amazon Timestream.

Temas

- Restauración de un Timestream para una tabla desde LiveAnalytics AWS Backup
- Restaurar una secuencia temporal de una LiveAnalytics tabla en otra región o cuenta

Restauración de un Timestream para una tabla desde LiveAnalytics AWS Backup

Para restaurar tu Timestream for LiveAnalytics table AWS Backup usando Timestream para LiveAnalytics consola, sigue estos pasos.

1. Inicie sesión en la Consola de AWS License Manager.
2. En el panel de navegación del lado izquierdo de la consola, elija Backups.
3. Para restaurar un recurso, pulsa el botón de radio situado junto al ID del punto de recuperación del recurso. En la esquina superior derecha del panel, elija Restaurar.
4. Introduzca los ajustes de configuración de la tabla, es decir, el nombre de la base de datos y el nombre de la tabla. Tenga en cuenta que el nombre de la tabla restaurada debe ser diferente del nombre de la tabla de origen original.
5. Configure los ajustes de retención de memoria y almacenamiento magnético.
6. En Rol de restauración, elija el IAM rol que AWS Backup asumirá para esta restauración.
7. Seleccione Restaurar copia de seguridad. En la parte superior de la página, aparecerá un mensaje con información sobre el trabajo de restauración.

> **Note**
>
> Se le cobrará por la restauración de toda la copia de seguridad, independientemente de la memoria configurada y de los períodos de retención del almacenamiento magnético. Sin embargo, una vez completada la restauración, la tabla restaurada solo contendrá los datos dentro de los períodos de retención configurados.

Restaurar una secuencia temporal de una LiveAnalytics tabla en otra región o cuenta

Para restaurar una secuencia temporal de una LiveAnalytics tabla en otra región o cuenta, primero tendrás que copiar la copia de seguridad en esa nueva región o cuenta. Para copiar en otra cuenta, esa cuenta primero debe otorgarle permiso. Una vez que hayas copiado tu Timestream como LiveAnalytics copia de seguridad en la nueva región o cuenta, podrás restaurarla siguiendo el proceso descrito en la sección anterior.

Copiar una copia de seguridad de una tabla de Amazon Timestream

Puede realizar una copia de una copia de seguridad actual. Puede copiar las copias de seguridad a varias AWS cuentas o AWS regiones a pedido o automáticamente como parte de un plan de copias de seguridad programado. La replicación entre regiones resulta especialmente útil si hay requisitos de continuidad del negocio o de conformidad que exigen que las copias de seguridad deben almacenarse a una distancia mínima de los datos de producción.

Las copias de seguridad entre cuentas son útiles para copiar de forma segura sus copias de seguridad en una o más cuentas AWS de su organización por motivos operativos o de seguridad. Si la copia de seguridad original se elimina accidentalmente, puede copiar la copia de seguridad de su cuenta de destino a su cuenta de origen y, a continuación, iniciar la restauración. Para poder hacerlo, debe tener dos cuentas que pertenezcan a la misma organización en el servicio Organizations y los permisos necesarios para las cuentas. Al copiar una copia de seguridad incremental en otra cuenta o región, también se copia la copia de seguridad completa asociada.

Las copias heredan la configuración de la copia de seguridad de origen, a menos que especifique lo contrario. Existe una excepción. Si especificas que tu nueva copia sea «Nunca» caducará. Con esta configuración, la nueva copia sigue heredando su fecha de vencimiento de origen. Si desea que la nueva copia de seguridad sea permanente, configure las copias de seguridad de origen para que nunca venzan o especifique que la nueva copia venza 100 años después de su creación.

Para copiar una copia de seguridad de la consola Timestream, sigue estos pasos.

1. Inicie sesión en la Consola de AWS License Manager.
2. En el panel de navegación del lado izquierdo de la consola, elija Backups.
3. Seleccione el botón de radio situado junto al ID del punto de recuperación del recurso. En la esquina superior derecha del panel, selecciona Acciones y elige Copiar.
4. Selecciona Continuar con la AWS copia de seguridad y sigue los pasos para la copia de seguridad entre cuentas.

Actualmente, Timestream para LiveAnalytics consola no admite de forma nativa la copia de copias de seguridad programadas y bajo demanda entre cuentas y regiones, por lo que hay que navegar hasta allí AWS Backup para realizar la operación.

Eliminación de copias de seguridad

En esta sección se describe cómo eliminar una copia de seguridad de un Timestream for Table. LiveAnalytics

Para eliminar una copia de seguridad de la consola Timestream, sigue estos pasos.

1. Inicie sesión en la Consola de AWS License Manager.
2. En el panel de navegación del lado izquierdo de la consola, elija Backups.
3. Seleccione el botón de radio situado junto al ID del punto de recuperación del recurso. En la esquina superior derecha del panel, selecciona Acciones y elige Eliminar.
4. Selecciona Continuar con la AWS copia de seguridad y sigue los pasos para eliminar copias de seguridad que se indican en Eliminar copias de seguridad.

> **Note**
>
> Al eliminar una copia de seguridad incremental, solo se elimina la copia de seguridad incremental y no se elimina la copia de seguridad completa subyacente.

Cuota y límites

AWS Backup limita las copias de seguridad a una copia de seguridad simultánea por recurso. Por lo tanto, las solicitudes adicionales de copia de seguridad programadas o bajo demanda para el recurso se ponen en cola y se iniciarán solo después de que se complete el trabajo de copia de seguridad

existente. Si el trabajo de copia de seguridad no se inicia o no se completa dentro de la ventana de copia de seguridad, se produce un error en la solicitud. Para obtener más información sobre AWS Backup los límites, consulte AWS Backup Limits en la Guía para desarrolladores AWS de Backup.

Al crear una copia de seguridad, puede ejecutar hasta cuatro copias de seguridad simultáneas por cuenta. Del mismo modo, puede ejecutar una restauración simultánea por cuenta. Si inicia más de cuatro trabajos de copia de seguridad simultáneamente, solo se inician cuatro trabajos de copia de seguridad y los trabajos restantes se reintentarán periódicamente. Una vez iniciado, si el trabajo de copia de seguridad no se completa dentro del período de tiempo de copia de seguridad configurado, se produce un error en la tarea de copia de seguridad. Si el trabajo de copia de seguridad fallido es una copia de seguridad bajo demanda, puede volver a intentar la copia de seguridad y, en el caso de las copias de seguridad programadas, se intenta realizar la tarea siguiendo el siguiente programa.

Claves de partición definidas por el cliente

Amazon Timestream LiveAnalytics para claves de partición definidas por el cliente es una función de Timestream que permite a los clientes definir sus propias claves de partición LiveAnalytics para sus tablas. El particionamiento es una técnica que se utiliza para distribuir los datos entre varias unidades de almacenamiento físicas, lo que permite una recuperación de datos más rápida y eficiente. Con las claves de partición definidas por el cliente, los clientes pueden crear un esquema de partición que se adapte mejor a sus patrones de consulta y casos de uso.

Con Timestream para las claves de partición LiveAnalytics definidas por el cliente, los clientes pueden elegir nombres de una dimensión como clave de partición para sus tablas. Esto permite una mayor flexibilidad a la hora de definir el esquema de partición de sus datos. Al seleccionar la clave de partición correcta, los clientes pueden optimizar su modelo de datos, mejorar el rendimiento de sus consultas y reducir la latencia de las consultas.

Temas

- Crear claves de partición para las tablas existentes
- Secuencia temporal para la validación LiveAnalytics del esquema con claves de partición compuestas personalizadas

Uso de claves de partición definidas por el cliente

Si tiene un patrón de consulta bien definido con dimensiones de cardinalidad altas y requiere una latencia de consulta baja, un Timestream para una clave de partición LiveAnalytics definida por el cliente puede ser una herramienta útil para mejorar su modelo de datos. Por ejemplo, si eres una empresa minorista que realiza un seguimiento de las interacciones con los clientes en tu sitio web, es probable que los patrones de acceso principales se basen en el identificador del cliente y la marca de tiempo. Al definir el identificador del cliente como la clave de partición, los datos se pueden distribuir de manera uniforme, lo que reduce la latencia y, en última instancia, mejora la experiencia del usuario.

Otro ejemplo es el sector de la salud, donde los dispositivos portátiles recopilan datos de los sensores para rastrear los signos vitales de los pacientes. El patrón de acceso principal sería mediante el identificador del dispositivo y la marca de tiempo, con una alta cardinalidad en ambas dimensiones. Al definir el ID del dispositivo como clave de partición, puede optimizar la ejecución de las consultas y garantizar un rendimiento sostenido a largo plazo.

En resumen, Timestream para las claves de partición LiveAnalytics definidas por el cliente resulta más útil cuando se tiene un patrón de consulta claro, dimensiones de cardinalidad altas y se necesita una latencia baja para las consultas. Al definir una clave de partición que se ajuste al patrón de consulta, puede optimizar la ejecución de la consulta y garantizar un rendimiento sostenido a largo plazo.

Cómo empezar con las claves de partición definidas por el cliente

En la consola, selecciona Tablas y crea una tabla nueva. También puede utilizar una SDK para acceder a la `CreateTable` acción y crear nuevas tablas que puedan incluir una clave de partición definida por el cliente.

Cree una tabla con una clave de partición de tipo dimensión

Puede usar los siguientes fragmentos de código para crear una tabla con una clave de partición de tipo de dimensión.

Java

```
public void createTableWithDimensionTypePartitionKeyExample() {
        System.out.println("Creating table");
        CreateTableRequest createTableRequest = new CreateTableRequest();
        createTableRequest.setDatabaseName(DATABASE_NAME);
        createTableRequest.setTableName(TABLE_NAME);
        final RetentionProperties retentionProperties = new RetentionProperties()
                .withMemoryStoreRetentionPeriodInHours(HT_TTL_HOURS)
                .withMagneticStoreRetentionPeriodInDays(CT_TTL_DAYS);
        createTableRequest.setRetentionProperties(retentionProperties);

        // Can specify enforcement level with OPTIONAL or REQUIRED
        final List<PartitionKey> partitionKeyWithDimensionAndOptionalEnforcement =
 Collections.singletonList(new PartitionKey()
            .withName(COMPOSITE_PARTITION_KEY_DIM_NAME)
            .withType(PartitionKeyType.DIMENSION)
            .withEnforcementInRecord(PartitionKeyEnforcementLevel.OPTIONAL));
        Schema schema = new Schema();

 schema.setCompositePartitionKey(partitionKeyWithDimensionAndOptionalEnforcement);
        createTableRequest.setSchema(schema);

        try {
            writeClient.createTable(createTableRequest);
            System.out.println("Table [" + TABLE_NAME + "] successfully created.");
        } catch (ConflictException e) {
            System.out.println("Table [" + TABLE_NAME + "] exists on database [" +
 DATABASE_NAME + "] . Skipping database creation");
        }
    }
```

Java v2

```
public void createTableWithDimensionTypePartitionKeyExample() {
        System.out.println("Creating table");
        final RetentionProperties retentionProperties =
 RetentionProperties.builder()
                .memoryStoreRetentionPeriodInHours(HT_TTL_HOURS)
                .magneticStoreRetentionPeriodInDays(CT_TTL_DAYS)
                .build();
        // Can specify enforcement level with OPTIONAL or REQUIRED
```

```
        final List<PartitionKey> partitionKeyWithDimensionAndOptionalEnforcement =
 Collections.singletonList(PartitionKey
                .builder()
                .name(COMPOSITE_PARTITION_KEY_DIM_NAME)
                .type(PartitionKeyType.DIMENSION)
                .enforcementInRecord(PartitionKeyEnforcementLevel.OPTIONAL)
                .build());
        final Schema schema = Schema.builder()

 .compositePartitionKey(partitionKeyWithDimensionAndOptionalEnforcement).build();
        final CreateTableRequest createTableRequest = CreateTableRequest.builder()
                .databaseName(DATABASE_NAME)
                .tableName(TABLE_NAME)
                .retentionProperties(retentionProperties)
                .schema(schema)
                .build();

        try {
            writeClient.createTable(createTableRequest);
            System.out.println("Table [" + TABLE_NAME + "] successfully created.");
        } catch (ConflictException e) {
            System.out.println("Table [" + TABLE_NAME + "] exists on database [" +
 DATABASE_NAME + "] . Skipping database creation");
        }
    }
```

Go v1

```
func createTableWithDimensionTypePartitionKeyExample(){
        // Can specify enforcement level with OPTIONAL or REQUIRED
        partitionKeyWithDimensionAndOptionalEnforcement :=
 []*timestreamwrite.PartitionKey{
                {
                    Name:                aws.String(CompositePartitionKeyDimName),
                    EnforcementInRecord: aws.String("OPTIONAL"),
                    Type:                aws.String("DIMENSION"),
                },
        }
        createTableInput := &timestreamwrite.CreateTableInput{
             DatabaseName: aws.String(*databaseName),
             TableName:    aws.String(*tableName),
             // Enable MagneticStoreWrite for Table
```

```
            MagneticStoreWriteProperties:
&timestreamwrite.MagneticStoreWriteProperties{
                EnableMagneticStoreWrites: aws.Bool(true),
                // Persist MagneticStoreWrite rejected records in S3
                MagneticStoreRejectedDataLocation:
&timestreamwrite.MagneticStoreRejectedDataLocation{
                    S3Configuration: &timestreamwrite.S3Configuration{
                        BucketName:       aws.String("timestream-sample-bucket"),
                        ObjectKeyPrefix:  aws.String("TimeStreamCustomerSampleGo"),
                        EncryptionOption: aws.String("SSE_S3"),
                    },
                },
            },
            Schema: &timestreamwrite.Schema{
                CompositePartitionKey:
partitionKeyWithDimensionAndOptionalEnforcement,
            }
        }
        _, err := writeSvc.CreateTable(createTableInput)
   }
```

Go v2

```
 func (timestreamBuilder TimestreamBuilder)
CreateTableWithDimensionTypePartitionKeyExample() error {
        partitionKeyWithDimensionAndOptionalEnforcement := []types.PartitionKey{
            {
                Name:               aws.String(CompositePartitionKeyDimName),
                EnforcementInRecord: types.PartitionKeyEnforcementLevelOptional,
                Type:               types.PartitionKeyTypeDimension,
            },
        }
        _, err := timestreamBuilder.WriteSvc.CreateTable(context.TODO(),
&timestreamwrite.CreateTableInput{
            DatabaseName: aws.String(databaseName),
            TableName:    aws.String(tableName),
            MagneticStoreWriteProperties: &types.MagneticStoreWriteProperties{
                EnableMagneticStoreWrites: aws.Bool(true),
                // Persist MagneticStoreWrite rejected records in S3
                MagneticStoreRejectedDataLocation:
&types.MagneticStoreRejectedDataLocation{
                    S3Configuration: &types.S3Configuration{
                        BucketName:       aws.String(s3BucketName),
```

```
                        EncryptionOption: "SSE_S3",
                    },
                },
            },
            Schema: &types.Schema{
                CompositePartitionKey:
 partitionKeyWithDimensionAndOptionalEnforcement,
            },
        })

        if err != nil {
            fmt.Println("Error:")
            fmt.Println(err)
        } else {
            fmt.Println("Create table is successful")
        }
        return err
    }
```

Python

```
def create_table_with_measure_name_type_partition_key(self):
        print("Creating table")
        retention_properties = {
            'MemoryStoreRetentionPeriodInHours': HT_TTL_HOURS,
            'MagneticStoreRetentionPeriodInDays': CT_TTL_DAYS
        }
        partitionKey_with_measure_name = [
            {'Type': 'MEASURE'}
        ]
        schema = {
            'CompositePartitionKey': partitionKey_with_measure_name
        }
        try:
            self.client.create_table(DatabaseName=DATABASE_NAME,
 TableName=TABLE_NAME,
                                     RetentionProperties=retention_properties,
 Schema=schema)
            print("Table [%s] successfully created." % TABLE_NAME)
        except self.client.exceptions.ConflictException:
            print("Table [%s] exists on database [%s]. Skipping table creation" % (
                TABLE_NAME, DATABASE_NAME))
        except Exception as err:
```

```
            print("Create table failed:", err)
```

Comprobar la configuración del esquema de particionamiento

Puedes comprobar cómo se configura una tabla para el esquema de particionamiento de dos maneras. En la consola, selecciona Bases de datos y selecciona la tabla que deseas consultar. También puedes usar una SDK para acceder a la `DescribeTable` acción.

Describe una tabla con una clave de partición

Puede usar los siguientes fragmentos de código para describir una tabla con una clave de partición.

Java

```
   public void describeTable() {
       System.out.println("Describing table");
       final DescribeTableRequest describeTableRequest = new
DescribeTableRequest();
       describeTableRequest.setDatabaseName(DATABASE_NAME);
       describeTableRequest.setTableName(TABLE_NAME);
       try {
           DescribeTableResult result =
amazonTimestreamWrite.describeTable(describeTableRequest);
           String tableId = result.getTable().getArn();
           System.out.println("Table " + TABLE_NAME + " has id " + tableId);
           // If table is created with composite partition key, it can be described
with
           //
System.out.println(result.getTable().getSchema().getCompositePartitionKey());
       } catch (final Exception e) {
           System.out.println("Table " + TABLE_NAME + " doesn't exist = " + e);
           throw e;
       }
   }
```

El siguiente es un ejemplo de salida.

1. La tabla tiene una clave de partición de tipo de dimensión

```
[{Type: DIMENSION,Name: hostId,EnforcementInRecord: OPTIONAL}]
```

2. La tabla tiene el nombre de la medida, el tipo de clave de partición

```
[{Type: MEASURE,}]
```

3. Obtener la clave de partición compuesta de una tabla creada sin especificar la clave de partición compuesta

```
[{Type: MEASURE,}]
```

Java v2

```
 public void describeTable() {
       System.out.println("Describing table");
       final DescribeTableRequest describeTableRequest =
DescribeTableRequest.builder()
               .databaseName(DATABASE_NAME).tableName(TABLE_NAME).build();
       try {
           DescribeTableResponse response =
writeClient.describeTable(describeTableRequest);
           String tableId = response.table().arn();
           System.out.println("Table " + TABLE_NAME + " has id " + tableId);
           // If table is created with composite partition key, it can be described
with
           //
System.out.println(response.table().schema().compositePartitionKey());
       } catch (final Exception e) {
           System.out.println("Table " + TABLE_NAME + " doesn't exist = " + e);
           throw e;
       }
   }
```

El siguiente es un ejemplo de salida.

1. La tabla tiene una clave de partición de tipo de dimensión

```
[PartitionKey(Type=DIMENSION, Name=hostId, EnforcementInRecord=OPTIONAL)]
```

2. La tabla tiene el nombre de la medida, el tipo de clave de partición

```
[PartitionKey(Type=MEASURE)]
```

3. Al obtener la clave de partición compuesta de una tabla creada sin especificar la clave de partición compuesta, se devolverá

```
[PartitionKey(Type=MEASURE)]
```

Go v1

```
    <tablistentry>
     <tabname> Go </tabname>
     <tabcontent>
      <programlisting language="go"></programlisting>
     </tabcontent>
    </tablistentry>
```

El siguiente es un ejemplo de salida.

```
{
  Table: {
    Arn: "arn:aws:timestream:us-west-2:533139590831:database/devops/table/
host_metrics_dim_pk_1",
    CreationTime: 2023-05-31 01:52:00.511 +0000 UTC,
    DatabaseName: "devops",
    LastUpdatedTime: 2023-05-31 01:52:00.511 +0000 UTC,
    MagneticStoreWriteProperties: {
      EnableMagneticStoreWrites: true,
      MagneticStoreRejectedDataLocation: {
        S3Configuration: {
          BucketName: "timestream-sample-bucket-west",
          EncryptionOption: "SSE_S3",
          ObjectKeyPrefix: "TimeStreamCustomerSampleGo"
        }
      }
    },
    RetentionProperties: {
      MagneticStoreRetentionPeriodInDays: 73000,
      MemoryStoreRetentionPeriodInHours: 6
    },
    Schema: {
      CompositePartitionKey: [{
          EnforcementInRecord: "OPTIONAL",
          Name: "hostId",
```

```
          Type: "DIMENSION"
        }]
    },
    TableName: "host_metrics_dim_pk_1",
    TableStatus: "ACTIVE"
  }
}
```

Go v2

```
func (timestreamBuilder TimestreamBuilder) DescribeTable()
(*timestreamwrite.DescribeTableOutput, error) {
        describeTableInput := &timestreamwrite.DescribeTableInput{
            DatabaseName: aws.String(databaseName),
            TableName:    aws.String(tableName),
        }
       describeTableOutput, err :=
timestreamBuilder.WriteSvc.DescribeTable(context.TODO(), describeTableInput)

       if err != nil {
           fmt.Printf("Failed to describe table with Error: %s", err.Error())
       } else {
           fmt.Printf("Describe table is successful : %s\n",
JsonMarshalIgnoreError(*describeTableOutput))
           // If table is created with composite partition key, it will be included
in the output
       }

       return describeTableOutput, err
   }
```

El siguiente es un ejemplo de salida.

```
{
  "Table": {
    "Arn":"arn:aws:timestream:us-east-1:351861611069:database/cdpk-wr-db/table/
host_metrics_dim_pk",
    "CreationTime":"2023-05-31T22:36:10.66Z",
    "DatabaseName":"cdpk-wr-db",
    "LastUpdatedTime":"2023-05-31T22:36:10.66Z",
    "MagneticStoreWriteProperties":{
      "EnableMagneticStoreWrites":true,
      "MagneticStoreRejectedDataLocation":{
```

```
        "S3Configuration":{
          "BucketName":"error-configuration-sample-s3-bucket-cq8my",
          "EncryptionOption":"SSE_S3",
          "KmsKeyId":null,"ObjectKeyPrefix":null
        }
      }
    },
    "RetentionProperties":{
      "MagneticStoreRetentionPeriodInDays":73000,
      "MemoryStoreRetentionPeriodInHours":6
    },
    "Schema":{
      "CompositePartitionKey":[{
        "Type":"DIMENSION",
        "EnforcementInRecord":"OPTIONAL",
        "Name":"hostId"
      }]
    },
    "TableName":"host_metrics_dim_pk",
    "TableStatus":"ACTIVE"
  },
  "ResultMetadata":{}
}
```

Python

```
 def describe_table(self):
       print('Describing table')
       try:
           result = self.client.describe_table(DatabaseName=DATABASE_NAME,
TableName=TABLE_NAME)
           print("Table [%s] has id [%s]" % (TABLE_NAME, result['Table']['Arn']))
           # If table is created with composite partition key, it can be described
with
           # print(result['Table']['Schema'])
       except self.client.exceptions.ResourceNotFoundException:
           print("Table doesn't exist")
       except Exception as err:
           print("Describe table failed:", err)
```

El siguiente es un ejemplo de salida.

1. La tabla tiene una clave de partición de tipo de dimensión

```
[{'CompositePartitionKey': [{'Type': 'DIMENSION', 'Name': 'hostId',
 'EnforcementInRecord': 'OPTIONAL'}]}]
```

2. La tabla tiene el nombre de la medida, el tipo de clave de partición

```
[{'CompositePartitionKey': [{'Type': 'MEASURE'}]}]
```

3. Obtener la clave de partición compuesta de una tabla creada sin especificar la clave de partición compuesta

```
[{'CompositePartitionKey': [{'Type': 'MEASURE'}]}]
```

Actualización de la configuración del esquema de particionamiento

Puede actualizar la configuración de la tabla para el esquema de particionamiento con una acción SDK con acceso a la `UpdateTable` acción.

Actualiza una tabla con una clave de partición

Puede usar los siguientes fragmentos de código para actualizar una tabla con una clave de partición.

Java

```
   public void updateTableCompositePartitionKeyEnforcement() {
       System.out.println("Updating table");

       UpdateTableRequest updateTableRequest = new UpdateTableRequest();
       updateTableRequest.setDatabaseName(DATABASE_NAME);
       updateTableRequest.setTableName(TABLE_NAME);

       // Can update enforcement level for dimension type partition key with
OPTIONAL or REQUIRED enforcement
       final List<PartitionKey> partitionKeyWithDimensionAndRequiredEnforcement =
Collections.singletonList(new PartitionKey()
           .withName(COMPOSITE_PARTITION_KEY_DIM_NAME)
           .withType(PartitionKeyType.DIMENSION)
           .withEnforcementInRecord(PartitionKeyEnforcementLevel.REQUIRED));
       Schema schema = new Schema();

schema.setCompositePartitionKey(partitionKeyWithDimensionAndRequiredEnforcement);
```

```
        updateTableRequest.withSchema(schema);

        writeClient.updateTable(updateTableRequest);
        System.out.println("Table updated");
```

Java v2

```
   public void updateTableCompositePartitionKeyEnforcement() {
        System.out.println("Updating table");
        // Can update enforcement level for dimension type partition key with
OPTIONAL or REQUIRED enforcement
        final List<PartitionKey> partitionKeyWithDimensionAndRequiredEnforcement =
Collections.singletonList(PartitionKey
            .builder()
            .name(COMPOSITE_PARTITION_KEY_DIM_NAME)
            .type(PartitionKeyType.DIMENSION)
            .enforcementInRecord(PartitionKeyEnforcementLevel.REQUIRED)
            .build());
        final Schema schema = Schema.builder()

.compositePartitionKey(partitionKeyWithDimensionAndRequiredEnforcement).build();
        final UpdateTableRequest updateTableRequest = UpdateTableRequest.builder()

.databaseName(DATABASE_NAME).tableName(TABLE_NAME).schema(schema).build();

        writeClient.updateTable(updateTableRequest);
        System.out.println("Table updated");
```

Go v1

```
// Update table partition key enforcement attribute
   updateTableInput := &timestreamwrite.UpdateTableInput{
        DatabaseName: aws.String(*databaseName),
        TableName:    aws.String(*tableName),
        // Can update enforcement level for dimension type partition key with
OPTIONAL or REQUIRED enforcement
        Schema: &timestreamwrite.Schema{
            CompositePartitionKey: []*timestreamwrite.PartitionKey{
                {
                        Name:
aws.String(CompositePartitionKeyDimName),
                        EnforcementInRecord: aws.String("REQUIRED"),
                        Type:               aws.String("DIMENSION"),
```

```
                },
            }},
    }
    updateTableOutput, err := writeSvc.UpdateTable(updateTableInput)
        if err != nil {
            fmt.Println("Error:")
            fmt.Println(err)
        } else {
            fmt.Println("Update table is successful, below is the output:")
            fmt.Println(updateTableOutput)
        }
```

Go v2

```
// Update table partition key enforcement attribute
        updateTableInput := &timestreamwrite.UpdateTableInput{
            DatabaseName: aws.String(*databaseName),
            TableName:    aws.String(*tableName),
            // Can update enforcement level for dimension type partition key with
OPTIONAL or REQUIRED enforcement
            Schema: &types.Schema{
                CompositePartitionKey: []types.PartitionKey{
                    {
                        Name:
aws.String(CompositePartitionKeyDimName),
                        EnforcementInRecord:
types.PartitionKeyEnforcementLevelRequired,
                        Type:               types.PartitionKeyTypeDimension,
                    },
                }},
        }
        updateTableOutput, err :=
timestreamBuilder.WriteSvc.UpdateTable(context.TODO(), updateTableInput)
        if err != nil {
            fmt.Println("Error:")
            fmt.Println(err)
        } else {
            fmt.Println("Update table is successful, below is the output:")
            fmt.Println(updateTableOutput)
        }
```

Python

```
   def update_table(self):
       print('Updating table')
       try:
           # Can update enforcement level for dimension type partition key with
OPTIONAL or REQUIRED enforcement
           partition_key_with_dimension_and_required_enforcement = [
               {
                   'Type': 'DIMENSION',
                   'Name': COMPOSITE_PARTITION_KEY_DIM_NAME,
                   'EnforcementInRecord': 'REQUIRED'
               }
           ]
           schema = {
               'CompositePartitionKey':
partition_key_with_dimension_and_required_enforcement
           }
           self.client.update_table(DatabaseName=DATABASE_NAME,
TableName=TABLE_NAME,
                                    Schema=schema)
           print('Table updated.')
       except Exception as err:
           print('Update table failed:', err)
```

Ventajas de las claves de partición definidas por el cliente

Rendimiento de consultas mejorado: las claves de partición definidas por el cliente le permiten optimizar la ejecución de las consultas y mejorar el rendimiento general de las consultas. Al definir claves de partición que se ajusten a los patrones de consulta, puede minimizar el escaneo de datos y optimizar la depuración de datos, lo que reduce la latencia de las consultas.

Mejor previsibilidad del rendimiento a largo plazo: las claves de partición definidas por el cliente permiten a los clientes distribuir los datos de manera uniforme entre las particiones, lo que mejora la eficiencia de la administración de datos. Esto garantizará que el rendimiento de las consultas se mantenga estable a medida que los datos almacenados se amplíen con el tiempo.

Limitaciones de las claves de partición definidas por el cliente

Como flujo temporal para el LiveAnalytics usuario, es importante tener en cuenta las limitaciones de una clave de partición del cliente. En primer lugar, requiere una buena comprensión de la carga

de trabajo y de los patrones de consulta. Esto significa que debe tener una idea clara de qué dimensiones se utilizan con más frecuencia como principales condiciones de filtrado en las consultas y cuáles tienen una alta cardinalidad para utilizar las claves de partición de la forma más eficaz.

En segundo lugar, las claves de partición deben definirse en el momento de la creación de la tabla y no se pueden añadir a las tablas existentes. Esto significa que debe considerar detenidamente su estrategia de particionamiento antes de crear una tabla para asegurarse de que se ajuste a las necesidades de su empresa.

Por último, es importante tener en cuenta que, una vez creada la tabla, no podrá cambiar la clave de partición posteriormente. Esto significa que debes probar y evaluar minuciosamente tu estrategia de particionamiento antes de comprometerte con ella. Teniendo en cuenta estas limitaciones, la clave de partición definida por el cliente de Timestream puede mejorar considerablemente el rendimiento de las consultas y la satisfacción a largo plazo.

Claves de partición definidas por el cliente y dimensiones de baja cardinalidad

Si decide utilizar una clave de partición con una cardinalidad muy baja, como una región o un estado específicos, es importante tener en cuenta que los datos de otras entidades, por ejemplo, y otras `customerIDProductCategory`, podrían terminar repartidos en demasiadas particiones, a veces con pocos o ningún dato presente. Esto puede provocar una ejecución ineficiente de las consultas y una disminución del rendimiento.

Para evitarlo, le recomendamos que elija dimensiones que no solo formen parte de su condición de filtrado clave, sino que tengan una mayor cardinalidad. Esto ayudará a garantizar que los datos se distribuyan uniformemente entre las particiones y mejorará el rendimiento de las consultas.

Crear claves de partición para las tablas existentes

Si ya tiene tablas en Timestream LiveAnalytics y desea utilizar claves de partición definidas por el cliente, tendrá que migrar los datos a una nueva tabla con la definición de esquema de partición deseada. Esto se puede hacer mediante la exportación a S3 y la carga por lotes juntos, lo que implica exportar los datos de la tabla existente a S3, modificar los datos para incluir la clave de partición (si es necesario) y agregar encabezados a los CSV archivos y, a continuación, importar los datos a una nueva tabla con el esquema de partición deseado definido. Tenga en cuenta que este método puede llevar mucho tiempo y resultar costoso, especialmente en el caso de tablas grandes.

Como alternativa, puede utilizar consultas programadas para migrar los datos a una nueva tabla con el esquema de particiones deseado. Este método implica crear una consulta programada que lea de la tabla existente y escriba en la nueva tabla. La consulta programada se puede configurar para que se ejecute de forma regular hasta que se hayan migrado todos los datos. Tenga en cuenta que se le cobrará por leer y escribir los datos durante el proceso de migración.

Secuencia temporal para la validación LiveAnalytics del esquema con claves de partición compuestas personalizadas

La validación de esquemas en Timestream for LiveAnalytics ayuda a garantizar que los datos ingresados en la base de datos cumplan con el esquema especificado, lo que minimiza los errores de ingesta y mejora la calidad de los datos. En particular, la validación de esquemas es especialmente útil cuando se adopta una clave de partición definida por el cliente con el objetivo de optimizar el rendimiento de las consultas.

¿Qué es Timestream para la validación de LiveAnalytics esquemas con claves de partición definidas por el cliente?

El Timestream para la validación de LiveAnalytics esquemas es una función que valida los datos que se ingieren en una tabla Timestream for en función de un esquema predefinido. LiveAnalytics Este esquema define el modelo de datos, incluida la clave de partición, los tipos de datos y las restricciones de los registros que se insertan.

Cuando se utiliza una clave de partición definida por el cliente, la validación del esquema se vuelve aún más crucial. Las claves de partición le permiten especificar una clave de partición, que determina el modo en que se almacenan sus datos en Timestream. LiveAnalytics Al validar los datos entrantes con respecto al esquema con una clave de partición personalizada, puede garantizar la coherencia de los datos, detectar los errores a tiempo y mejorar la calidad general de los datos almacenados en Timestream for. LiveAnalytics

Cómo utilizar Timestream para la validación de LiveAnalytics esquemas con claves de partición compuestas personalizadas

Para usar Timestream para la validación de LiveAnalytics esquemas con claves de partición compuestas personalizadas, siga estos pasos:

Piensa en el aspecto que tendrán tus patrones de consulta: para elegir y definir correctamente el esquema de tu LiveAnalytics tabla Timestream for, debes empezar por los requisitos de consulta.

Especifique claves de partición compuestas personalizadas: al crear la tabla, especifique una clave de partición personalizada. Esta clave determina el atributo que se utilizará para particionar los datos de la tabla. Puede elegir entre claves de dimensiones y teclas de medida para la partición. Una clave de dimensión divide los datos en función del nombre de una dimensión, mientras que una clave de medida divide los datos en función del nombre de la medida.

Establezca niveles de cumplimiento: para garantizar un particionamiento de datos adecuado y las ventajas que ello conlleva, Amazon LiveAnalytics Timestream for le permite establecer niveles de cumplimiento para cada clave de partición de su esquema. El nivel de cumplimiento determina si la dimensión de clave de partición es obligatoria u opcional al ingerir registros. Puede elegir entre dos opciones: `REQUIRED` la clave de partición debe estar presente en el registro ingerido y `OPTIONAL` la clave de partición no tiene por qué estar presente. Se recomienda utilizar el nivel de exigencia al `REQUIRED` utilizar una partición definida por el cliente para garantizar que los datos estén correctamente particionados y aprovechar todas las ventajas de esta función. Además, puede cambiar la configuración del nivel de cumplimiento en cualquier momento después de la creación del esquema para ajustarla a sus requisitos de ingesta de datos.

Ingesta de datos: al incorporar datos a la LiveAnalytics tabla Timestream for, el proceso de validación del esquema comparará los registros con el esquema definido con claves de partición compuestas personalizadas. Si los registros no se ajustan al esquema, Timestream for LiveAnalytics devolverá un error de validación.

Gestionar los errores de validación: en caso de errores de validación, Timestream for LiveAnalytics devolverá a `ValidationException` o a`RejectedRecordsException`, según el tipo de error. Asegúrese de gestionar estas excepciones en su aplicación y de tomar las medidas adecuadas, como corregir los registros incorrectos y volver a intentar la ingestión.

Actualizar los niveles de cumplimiento: si es necesario, puede actualizar el nivel de cumplimiento de las claves de partición después de crear la tabla mediante la `UpdateTable` acción. Sin embargo, es importante tener en cuenta que algunos aspectos de la configuración de la clave de partición, como el nombre y el tipo, no se pueden cambiar después de crear la tabla. Si cambia el nivel de cumplimiento de `REQUIRED` a`OPTIONAL`, se aceptarán todos los registros independientemente de la presencia del atributo seleccionado como clave de partición definida por el cliente. Por el contrario, si cambias el nivel de cumplimiento de `OPTIONAL` a`REQUIRED`, es posible que empiecen a aparecer errores de escritura de hasta cuatro veces en los registros que no cumplan esta condición. Por lo tanto, es esencial elegir el nivel de cumplimiento adecuado para su caso de uso al crear la tabla, en función de los requisitos de particionamiento de los datos.

Cuándo usar Timestream para la validación de LiveAnalytics esquemas con claves de partición compuestas personalizadas

Se debe utilizar Timestream para la validación de LiveAnalytics esquemas con claves de partición compuestas personalizadas en situaciones en las que la coherencia de los datos, la calidad y la optimización de las particiones sean cruciales. Al aplicar un esquema durante la ingesta de datos, puede evitar errores e inconsistencias que podrían provocar un análisis incorrecto o la pérdida de información valiosa.

Interacción con los trabajos de carga por lotes

Al configurar un trabajo de carga por lotes para importar datos a una tabla con una clave de partición definida por el cliente, existen algunos escenarios que podrían afectar al proceso:

1. Si el nivel de cumplimiento está establecido en`OPTIONAL`, se mostrará una alerta en la consola durante el flujo de creación si la clave de partición no se mapea durante la configuración del trabajo. Esta alerta no aparecerá cuando se utilice la tecla API oCLI.
2. Si el nivel de cumplimiento está establecido en`REQUIRED`, se rechazará la creación de puestos de trabajo a menos que la clave de partición se asigne a una columna de datos de origen.
3. Si el nivel de cumplimiento se cambia a `REQUIRED` uno posterior a la creación del trabajo, el trabajo seguirá ejecutándose, pero los registros que no tengan la asignación adecuada para la clave de partición se rechazarán con un error de 4xx.

Interacción con una consulta programada

Al configurar un trabajo de consulta programado para calcular y almacenar agregados, resúmenes y otros tipos de datos preprocesados en una tabla con una clave de partición definida por el cliente, existen algunos escenarios que podrían afectar al proceso:

1. Si el nivel de cumplimiento está establecido en`OPTIONAL`, se mostrará una alerta si la clave de partición no está mapeada durante la configuración del trabajo. Esta alerta no aparecerá cuando se utilice la tecla API oCLI.
2. Si el nivel de cumplimiento está establecido en`REQUIRED`, se rechazará la creación de puestos de trabajo a menos que la clave de partición se asigne a una columna de datos de origen.
3. Si el nivel de cumplimiento se cambia a uno `REQUIRED` posterior a la creación del trabajo y los resultados de la consulta programada no contienen la dimensión de la clave de partición, se producirá un error en las siguientes iteraciones del trabajo.

Agregar etiquetas a los recursos

Puede etiquetar Amazon Timestream LiveAnalytics para los recursos mediante etiquetas. Las etiquetas le permiten categorizar sus recursos de diferentes maneras, por ejemplo, por finalidad, propietario, entorno u otros criterios. Las etiquetas pueden ayudar a hacer lo siguiente:

- Identificar rápidamente un recurso según las etiquetas que le haya asignado.
- Consulte AWS las facturas desglosadas por etiquetas.

El etiquetado es compatible con AWS servicios como Amazon Elastic Compute Cloud (AmazonEC2), Amazon Simple Storage Service (Amazon S3), Timestream LiveAnalytics for y más. Un etiquetado eficiente puede ofrecerle información detallada sobre los costos permitiendo crear informes sobre los servicios que llevan una etiqueta determinada.

Para empezar a usar etiquetas, haga lo siguiente:

1. Comprenda las restricciones de etiquetado.
2. Cree etiquetas mediante operaciones de etiquetado.

Por último, es conveniente seguir estrategias de etiquetado óptimas. Para obtener más información, consulte Estrategias de etiquetado de AWS.

Restricciones de etiquetado

Cada etiqueta consta de una clave y un valor, ambos definidos por el usuario. Se aplican las siguientes restricciones:

- Cada flujo temporal de una LiveAnalytics tabla solo puede tener una etiqueta con la misma clave. Si intenta añadir una etiqueta existente, el valor de la etiqueta existente se actualiza al nuevo valor.
- Un valor actúa como descriptor dentro de una categoría de etiquetas. En Timestream, LiveAnalytics el valor no puede estar vacío ni ser nulo.
- Las claves y los valores de las etiquetas distinguen entre mayúsculas y minúsculas.
- La longitud máxima de la clave es de 128 caracteres Unicode.
- La longitud máxima del valor es de 256 caracteres Unicode.
- Los caracteres permitidos son letras, espacios en blanco y números, además de los caracteres especiales siguientes: `+ - = . _ : /`

- El número máximo de etiquetas por recurso es 50.
- AWSA los nombres y valores de las etiquetas asignadas se les asigna automáticamente el `aws:` prefijo, que usted no puede asignar. AWS-los nombres de etiqueta asignados no cuentan para el límite de 50 etiquetas. Los nombres de etiquetas asignados por el usuario presentan el prefijo `user:` en el informe de asignación de costos.
- No es posible antedatar la aplicación de una etiqueta.

Operaciones de etiquetado

Puede añadir, enumerar, editar o eliminar etiquetas de bases de datos y tablas mediante Amazon Timestream LiveAnalytics para consola, lenguaje de consulta o AWS Command Line Interface ().AWS CLI

Temas

- Agregar etiquetas a bases de datos y tablas nuevas o existentes mediante la consola

Agregar etiquetas a bases de datos y tablas nuevas o existentes mediante la consola

Puede usar la LiveAnalytics consola Timestream para agregar etiquetas a nuevas bases de datos, tablas y consultas programadas al crearlas. También puede añadir, editar o eliminar etiquetas de tablas existentes.

Para etiquetar bases de datos al crearlas (consola)

1. Abra la consola Timestream en /timestream. https://console.aws.amazon.com
2. En el panel de navegación, seleccione Bases de datos y, a continuación, Crear base de datos.
3. En la página Crear base de datos, introduzca un nombre para la base de datos. Introduzca una clave y un valor para la etiqueta y, a continuación, elija Añadir nueva etiqueta.
4. Elija Crear base de datos.

Para etiquetar tablas al crearlas (consola)

1. Abra la consola Timestream en /timestream. https://console.aws.amazon.com
2. En el panel de navegación, elija Tablas y, a continuación, seleccione Crear tabla.
3. En la página Crear flujo temporal para una tabla, introduzca un LiveAnalytics nombre para la tabla. Introduzca una clave y un valor para la etiqueta y elija Añadir nueva etiqueta.

4. Elija Crear tabla.

Para etiquetar las consultas programadas al crearlas (consola)

1. Abre la consola de Timestream en /timestream. https://console.aws.amazon.com
2. En el panel de navegación, elija Consultas programadas y, a continuación, elija Crear consulta programada.
3. En el paso 3. En la página de configuración de la consulta, seleccione Añadir nueva etiqueta. Escriba una clave y un valor para la etiqueta. Para agregar otras etiquetas, elija Agregar nueva etiqueta.
4. Elija Next (Siguiente).

Para etiquetar recursos existentes (consola)

1. Abra la consola Timestream en /timestream. https://console.aws.amazon.com
2. En el panel de navegación, seleccione Bases de datos, Tablas o Consultas programadas.
3. Elija una base de datos o una tabla de la lista. A continuación, elija Administrar etiquetas para añadir, editar o eliminar sus etiquetas.

Para obtener más información sobre la estructura de las etiquetas, consulte Restricciones de etiquetado.

Seguridad en Timestream para LiveAnalytics

La seguridad en la nube AWS es la máxima prioridad. Como AWS cliente, usted se beneficia de una arquitectura de centro de datos y red diseñada para cumplir con los requisitos de las organizaciones más sensibles a la seguridad.

La seguridad es una responsabilidad compartida entre usted AWS y usted. El modelo de responsabilidad compartida la describe como seguridad de la nube y seguridad en la nube:

- Seguridad de la nube: AWS es responsable de proteger la infraestructura que ejecuta AWS los servicios en la AWS nube. AWS también le proporciona servicios que puede utilizar de forma segura. Auditores externos prueban y verifican periódicamente la eficacia de nuestra seguridad en el marco de los programas de conformidad de AWS. Para obtener más información sobre

los programas de cumplimiento que se aplican a Timestream LiveAnalytics, consulte los AWS servicios incluidos en el ámbito de aplicación por programa de cumplimiento.

- Seguridad en la nube: su responsabilidad viene determinada por el AWS servicio que utilice. Usted también es responsable de otros factores incluida la confidencialidad de los datos, los requisitos de la empresa y la legislación y los reglamentos aplicables.

Esta documentación le ayudará a entender cómo aplicar el modelo de responsabilidad compartida cuando utilice Timestream para. LiveAnalytics Los siguientes temas le muestran cómo configurar Timestream LiveAnalytics para cumplir sus objetivos de seguridad y conformidad. También aprenderá a utilizar otros AWS servicios que pueden ayudarle a supervisar y proteger sus recursos de Timestream. LiveAnalytics

Temas

- Protección de datos en Timestream para LiveAnalytics
- Administración de identidades y accesos para Amazon Timestream para LiveAnalytics
- Registro y supervisión en Timestream para LiveAnalytics
- Resiliencia en Amazon Timestream Live Analytics
- Seguridad de la infraestructura en Amazon Timestream Live Analytics
- Análisis de configuración y vulnerabilidad en Timestream
- Respuesta a incidentes en Timestream para LiveAnalytics
- VPCpuntos finales ()AWS PrivateLink
- Prácticas recomendadas de seguridad para Amazon Timestream para LiveAnalytics

Protección de datos en Timestream para LiveAnalytics

El modelo de se aplica a protección de datos en Amazon Timestream Live Analytics. Como se describe en este modelo, AWS es responsable de proteger la infraestructura global en la que se ejecutan todos los. Nube de AWS Usted es responsable de mantener el control sobre el contenido alojado en esta infraestructura. Usted también es responsable de las tareas de administración y configuración de seguridad para los Servicios de AWS que utiliza. Para obtener más información sobre la privacidad de los datos, consulte la sección Privacidad de datos FAQ. Para obtener información sobre la protección de datos en Europa, consulte el modelo de responsabilidad AWS compartida y la entrada del GDPR blog sobre AWS seguridad.

Con fines de protección de datos, le recomendamos que proteja Cuenta de AWS las credenciales y configure los usuarios individuales con AWS IAM Identity Center o AWS Identity and Access Management (IAM). De esta manera, solo se otorgan a cada usuario los permisos necesarios para cumplir sus obligaciones laborales. También recomendamos proteger sus datos de la siguiente manera:

- Utilice la autenticación multifactorial (MFA) con cada cuenta.
- UseSSL/TLSpara comunicarse con AWS los recursos. Necesitamos TLS 1.2 y recomendamos TLS 1.3.
- Configure API y registre la actividad del usuario con AWS CloudTrail. Para obtener información sobre el uso de CloudTrail senderos para capturar AWS actividades, consulte Cómo trabajar con CloudTrail senderos en la Guía del AWS CloudTrail usuario.
- Utilice soluciones de AWS cifrado, junto con todos los controles de seguridad predeterminados Servicios de AWS.
- Utilice servicios de seguridad administrados avanzados, como Amazon Macie, que lo ayuden a detectar y proteger los datos confidenciales almacenados en Amazon S3.
- Si necesita entre FIPS 140 y 3 módulos criptográficos validados para acceder a AWS través de una interfaz de línea de comandos o unaAPI, utilice un FIPS terminal. Para obtener más información sobre los FIPS puntos finales disponibles, consulte la Norma federal de procesamiento de información () FIPS 140-3.

Se recomienda encarecidamente no introducir nunca información confidencial o sensible, como, por ejemplo, direcciones de correo electrónico de clientes, en etiquetas o campos de formato libre, tales como el campo Nombre. Esto incluye cuando trabaja con Timestream Live Analytics u otro dispositivo Servicios de AWS mediante la consola,, API o. AWS CLI AWS SDKs Cualquier dato que ingrese en etiquetas o campos de formato libre utilizados para nombres se puede emplear para los registros de facturación o diagnóstico. Si proporciona una URL a un servidor externo, le recomendamos encarecidamente que no incluya información sobre las credenciales URL para validar su solicitud a ese servidor.

Para obtener información más detallada sobre Timestream para temas de protección de LiveAnalytics datos como el cifrado en reposo y la administración de claves, selecciona cualquiera de los temas disponibles a continuación.

Temas

- Cifrado en reposo

- Cifrado en tránsito
- Administración de claves

Cifrado en reposo

Timestream para el LiveAnalytics cifrado en reposo proporciona una seguridad mejorada al cifrar todos los datos en reposo mediante claves de cifrado almacenadas en ().AWS Key Management ServiceAWS KMS Esta funcionalidad ayuda a reducir la carga y la complejidad operativas que conlleva la protección de información confidencial. Con el cifrado en reposo, puede crear aplicaciones sensibles a la seguridad que necesitan cumplimiento estricto de cifrado y requisitos normativos.

- El cifrado está activado de forma predeterminada en la LiveAnalytics base de datos Timestream for y no se puede desactivar. El algoritmo de cifrado AES -256 estándar del sector es el algoritmo de cifrado predeterminado que se utiliza.
- AWS KMS es necesario para el cifrado en reposo en Timestream para. LiveAnalytics
- No puede cifrar sólo un subconjunto de elementos de una tabla.
- No es necesario modificar las aplicaciones cliente de base de datos para utilizar el cifrado.

Si no proporciona una clave, Timestream for LiveAnalytics crea y utiliza una AWS KMS clave con el nombre `alias/aws/timestream` de su cuenta.

Puede utilizar su propia clave gestionada por el cliente KMS para cifrar los datos de Timestream. LiveAnalytics Para obtener más información sobre las claves de Timestream for, consulte. LiveAnalytics Administración de claves

Timestream for LiveAnalytics almacena sus datos en dos niveles de almacenamiento, almacenamiento de memoria y almacenamiento magnético. Los datos del almacén de memoria se cifran mediante una clave de servicio Timestream. LiveAnalytics Los datos del almacén magnético se cifran con su clave. AWS KMS

El servicio Timestream Query requiere credenciales para acceder a sus datos. Estas credenciales se cifran con su clave. KMS

> **Note**
>
> Timestream for LiveAnalytics no requiere todas las operaciones AWS KMS de descifrado. En cambio, mantiene una caché local de claves durante 5 minutos con tráfico activo. Todos los

cambios en los permisos se propagan a través del Timestream LiveAnalytics del sistema de forma uniforme en un plazo máximo de 5 minutos.

Cifrado en tránsito

Todos sus datos de Timestream Live Analytics se cifran en tránsito. De forma predeterminada, todas las comunicaciones con y desde Timestream for LiveAnalytics están protegidas mediante el cifrado Transport Layer Security (). TLS

Administración de claves

Puede administrar las claves de Amazon Timestream Live Analytics mediante AWS el Servicio de administración de claves ().AWS KMS Timestream Live Analytics requiere el uso de KMS para cifrar sus datos. Dispone de las siguientes opciones para la administración de claves, según el grado de control que necesite sobre ellas:

Recursos de bases de datos y tablas

- Clave gestionada por Timestream Live Analytics: si no proporciona una clave, Timestream Live Analytics creará una clave utilizando. `alias/aws/timestream` KMS
- Clave gestionada por el cliente: se admiten las claves gestionadas por el cliente. KMS Elija esta opción si necesita tener más control sobre los permisos y el ciclo de vida de sus claves, incluida la posibilidad de rotarlas automáticamente cada año.

Recurso de consulta programada

- Clave propiedad de Timestream Live Analytics: si no proporciona una clave, Timestream Live Analytics utilizará su propia KMS clave para cifrar el recurso de consulta, que está presente en la cuenta de timestream. Consulte las claves propias en la guía para desarrolladores para obtener más información AWS . KMS
- Clave gestionada por el KMS cliente: se admiten las claves gestionadas por el cliente. Elija esta opción si necesita tener más control sobre los permisos y el ciclo de vida de sus claves, incluida la posibilidad de rotarlas automáticamente cada año.

KMSno se admiten las claves de un almacén de claves externo (XKS).

Administración de identidades y accesos para Amazon Timestream para LiveAnalytics

AWS Identity and Access Management (IAM) es una Servicio de AWS que ayuda al administrador a controlar de forma segura el acceso a los recursos. AWS IAMlos administradores controlan quién puede autenticarse (iniciar sesión) y quién está autorizado (tiene permisos) para usar Timestream como fuente de recursos. LiveAnalytics IAMes una Servicio de AWS que puede utilizar sin coste adicional.

Temas

Público

La forma de usar AWS Identity and Access Management (IAM) varía según el trabajo que realices en Timestream. LiveAnalytics

Usuario del servicio: si utilizas el LiveAnalytics servicio Timestream for para realizar tu trabajo, el administrador te proporcionará las credenciales y los permisos que necesitas. A medida que vaya utilizando más LiveAnalytics funciones de Timestream for para realizar su trabajo, es posible que necesite permisos adicionales. Entender cómo se administra el acceso puede ayudarlo a solicitar los permisos correctos al administrador. Si no puede acceder a una función de Timestream, consulte. LiveAnalytics Solución de problemas de identidad y acceso en Amazon Timestream LiveAnalytics

Administrador de servicios: si está a cargo de Timestream para LiveAnalytics los recursos de su empresa, probablemente tenga acceso completo a Timestream for. LiveAnalytics Su trabajo consiste en determinar a qué flujo temporal de LiveAnalytics funciones y recursos deben acceder los usuarios del servicio. A continuación, debe enviar solicitudes a su IAM administrador para cambiar los permisos de los usuarios del servicio. Revise la información de esta página para comprender los conceptos básicos delAM. Para obtener más información sobre cómo su empresa puede utilizar IAM

Timestream para LiveAnalytics, consulte. Cómo funciona Amazon Timestream for LiveAnalytics con IAM

IAMadministrador: si es IAM administrador, puede que desee obtener más información sobre cómo puede redactar políticas para administrar el acceso a Timestream. LiveAnalytics Para ver un ejemplo de Timestream para políticas LiveAnalytics basadas en la identidad que puede utilizar, consulte. IAM Amazon Timestream LiveAnalytics para ejemplos de políticas basadas en la identidad

Autenticación con identidades

La autenticación es la forma de iniciar sesión con sus AWS credenciales de identidad. Debe estar autenticado (con quien haya iniciado sesión AWS) como IAM usuario o asumiendo un IAM rol. Usuario raíz de la cuenta de AWS

Puede iniciar sesión AWS como una identidad federada mediante las credenciales proporcionadas a través de una fuente de identidad. AWS IAM Identity Center Los usuarios (IAMIdentity Center), la autenticación de inicio de sesión único de su empresa y sus credenciales de Google o Facebook son ejemplos de identidades federadas. Al iniciar sesión como una identidad federada, el administrador configuró previamente la federación de identidades mediante roles. IAM Cuando accede AWS mediante la federación, asume indirectamente un rol.

Según el tipo de usuario que sea, puede iniciar sesión en el portal AWS Management Console o en el de AWS acceso. Para obtener más información sobre cómo iniciar sesión AWS, consulte Cómo iniciar sesión Cuenta de AWS en su Guía del AWS Sign-In usuario.

Si accede AWS mediante programación, AWS incluye un kit de desarrollo de software (SDK) y una interfaz de línea de comandos (CLI) para firmar criptográficamente sus solicitudes con sus credenciales. Si no utilizas AWS herramientas, debes firmar las solicitudes tú mismo. Para obtener más información sobre cómo usar el método recomendado para firmar las solicitudes usted mismo, consulte la versión 4 de la AWS firma para ver API las solicitudes en la Guía del IAM usuario.

Independientemente del método de autenticación que use, es posible que deba proporcionar información de seguridad adicional. Por ejemplo, le AWS recomienda que utilice la autenticación multifactorial (MFA) para aumentar la seguridad de su cuenta. Para obtener más información, consulte Autenticación multifactorial en la Guía del AWS IAM Identity Center usuario y Autenticación AWS multifactorial IAM en la Guía del IAM usuario.

Usuarios y grupos de IAM

Un IAMusuario es una identidad propia Cuenta de AWS que tiene permisos específicos para una sola persona o aplicación. Siempre que sea posible, recomendamos utilizar credenciales

temporales en lugar de crear IAM usuarios con credenciales de larga duración, como contraseñas y claves de acceso. Sin embargo, si tiene casos de uso específicos que requieren credenciales a largo plazo con IAM los usuarios, le recomendamos que rote las claves de acceso. Para obtener más información, consulte Rotar las claves de acceso con regularidad para los casos de uso que requieran credenciales de larga duración en la Guía del IAM usuario.

Un IAMgrupo es una identidad que especifica un conjunto de IAM usuarios. No puede iniciar sesión como grupo. Puede usar los grupos para especificar permisos para varios usuarios a la vez. Los grupos facilitan la administración de los permisos para grandes conjuntos de usuarios. Por ejemplo, puede asignar un nombre a un grupo IAMAdminsy concederle permisos para administrar IAM los recursos.

Los usuarios son diferentes de los roles. Un usuario se asocia exclusivamente a una persona o aplicación, pero la intención es que cualquier usuario pueda asumir un rol que necesite. Los usuarios tienen credenciales de larga duración permanentes; no obstante, los roles proporcionan credenciales temporales. Para obtener más información, consulte Casos de uso para IAM usuarios en la Guía del IAM usuario.

IAMroles

Un IAMrol es una identidad dentro de ti Cuenta de AWS que tiene permisos específicos. Es similar a un IAM usuario, pero no está asociado a una persona específica. Para asumir temporalmente un IAM rol en la AWS Management Console, puede cambiar de un IAM rol de usuario a uno (consola). Puede asumir un rol llamando a una AWS API operación AWS CLI o o utilizando una operación personalizadaURL. Para obtener más información sobre los métodos de uso de los roles, consulte Métodos para asumir un rol en la Guía del IAM usuario.

IAMlos roles con credenciales temporales son útiles en las siguientes situaciones:

- Acceso de usuario federado: para asignar permisos a una identidad federada, puede crear un rol y definir sus permisos. Cuando se autentica una identidad federada, se asocia la identidad al rol y se le conceden los permisos define el rol. Para obtener información sobre los roles para la federación, consulte Creación de un rol para un proveedor de identidad externo en la Guía del IAM usuario. Si usa IAM Identity Center, configura un conjunto de permisos. Para controlar a qué pueden acceder sus identidades después de autenticarse, IAM Identity Center correlaciona el conjunto de permisos con un rol en. IAM Para obtener información acerca de los conjuntos de permisos, consulte Conjuntos de permisos en la Guía del usuario de AWS IAM Identity Center .
- Permisos IAM de usuario temporales: un IAM usuario o rol puede asumir un IAM rol para asumir temporalmente diferentes permisos para una tarea específica.

- Acceso multicuenta: puedes usar un IAM rol para permitir que alguien (un responsable de confianza) de una cuenta diferente acceda a los recursos de tu cuenta. Los roles son la forma principal de conceder acceso entre cuentas. Sin embargo, con algunos Servicios de AWS, puedes adjuntar una política directamente a un recurso (en lugar de usar un rol como proxy). Para conocer la diferencia entre las funciones y las políticas basadas en recursos para el acceso multicuenta, consulta el tema sobre el acceso a los recursos entre cuentas IAM en la Guía del IAM usuario.
- Acceso entre servicios: algunos Servicios de AWS utilizan funciones en otros. Servicios de AWS Por ejemplo, cuando realizas una llamada en un servicio, es habitual que ese servicio ejecute aplicaciones en Amazon EC2 o almacene objetos en Amazon S3. Es posible que un servicio haga esto usando los permisos de la entidad principal, usando un rol de servicio o usando un rol vinculado al servicio.
  - Sesiones de acceso directo (FAS): cuando utilizas un IAM usuario o un rol para realizar acciones en AWS ellas, se te considera director. Cuando utiliza algunos servicios, es posible que realice una acción que desencadene otra acción en un servicio diferente. FASutiliza los permisos del principal que llama a an Servicio de AWS, junto con los que solicitan, Servicio de AWS para realizar solicitudes a los servicios descendentes. FASlas solicitudes solo se realizan cuando un servicio recibe una solicitud que requiere interacciones con otros recursos Servicios de AWS o para completarse. En este caso, debe tener permisos para realizar ambas acciones. Para obtener detalles sobre la política a la hora de realizar FAS solicitudes, consulte Reenviar sesiones de acceso.
  - Función de servicio: una función de servicio es una IAMfunción que un servicio asume para realizar acciones en su nombre. Un IAM administrador puede crear, modificar y eliminar un rol de servicio desde dentroIAM. Para obtener más información, consulte Crear un rol para delegar permisos Servicio de AWS en un rol en el IAMManual del usuario.
  - Función vinculada a un servicio: una función vinculada a un servicio es un tipo de función de servicio que está vinculada a un. Servicio de AWS El servicio puede asumir el rol para realizar una acción en su nombre. Los roles vinculados al servicio aparecen en usted Cuenta de AWS y son propiedad del servicio. Un IAM administrador puede ver los permisos de los roles vinculados al servicio, pero no editarlos.
- Aplicaciones que se ejecutan en Amazon EC2: puedes usar un IAM rol para administrar las credenciales temporales de las aplicaciones que se ejecutan en una EC2 instancia y que realizan AWS CLI o AWS API solicitan. Esto es preferible a almacenar las claves de acceso dentro de la EC2 instancia. Para asignar un AWS rol a una EC2 instancia y ponerlo a disposición de todas sus aplicaciones, debe crear un perfil de instancia adjunto a la instancia. Un perfil de instancia contiene el rol y permite que los programas que se ejecutan en la EC2 instancia obtengan credenciales

temporales. Para obtener más información, consulte Uso de un IAM rol para conceder permisos a aplicaciones que se ejecutan en EC2 instancias de Amazon en la Guía del IAM usuario.

Administración de acceso mediante políticas

El acceso se controla AWS creando políticas y adjuntándolas a AWS identidades o recursos. Una política es un objeto AWS que, cuando se asocia a una identidad o un recurso, define sus permisos. AWS evalúa estas políticas cuando un director (usuario, usuario raíz o sesión de rol) realiza una solicitud. Los permisos en las políticas determinan si la solicitud se permite o se deniega. La mayoría de las políticas se almacenan AWS como JSON documentos. Para obtener más información sobre la estructura y el contenido de los documentos de JSON políticas, consulte Descripción general de JSON las políticas en la Guía del IAM usuario.

Los administradores pueden usar AWS JSON las políticas para especificar quién tiene acceso a qué. Es decir, qué entidad principal puede realizar acciones en qué recursos y en qué condiciones.

De forma predeterminada, los usuarios y los roles no tienen permisos. Para conceder a los usuarios permiso para realizar acciones en los recursos que necesitan, un IAM administrador puede crear IAM políticas. A continuación, el administrador puede añadir las IAM políticas a las funciones y los usuarios pueden asumir las funciones.

IAMlas políticas definen los permisos para una acción independientemente del método que se utilice para realizar la operación. Por ejemplo, suponga que dispone de una política que permite la acción `iam:GetRole`. Un usuario con esa política puede obtener información sobre el rol de AWS Management Console AWS CLI, el o el AWS API.

Políticas basadas en identidad

Las políticas basadas en la identidad son documentos de política de JSON permisos que se pueden adjuntar a una identidad, como un IAM usuario, un grupo de usuarios o un rol. Estas políticas controlan qué acciones pueden realizar los usuarios y los roles, en qué recursos y en qué condiciones. Para obtener información sobre cómo crear una política basada en la identidad, consulte Definir IAM permisos personalizados con políticas administradas por el cliente en la Guía del usuario. IAM

Las políticas basadas en identidades pueden clasificarse además como políticas insertadas o políticas administradas. Las políticas insertadas se integran directamente en un único usuario, grupo o rol. Las políticas administradas son políticas independientes que puede adjuntar a varios usuarios,

grupos y funciones de su empresa. Cuenta de AWS Las políticas administradas incluyen políticas AWS administradas y políticas administradas por el cliente. Para saber cómo elegir entre una política gestionada o una política integrada, consulte Elegir entre políticas gestionadas y políticas integradas en la Guía del IAM usuario.

Políticas basadas en recursos

Las políticas basadas en recursos son documentos de JSON política que se adjuntan a un recurso. Algunos ejemplos de políticas basadas en recursos son las políticas de confianza de IAM roles y las políticas de bucket de Amazon S3. En los servicios que admiten políticas basadas en recursos, los administradores de servicios pueden utilizarlos para controlar el acceso a un recurso específico. Para el recurso al que se asocia la política, la política define qué acciones puede realizar una entidad principal especificada en ese recurso y en qué condiciones. Debe especificar una entidad principal en una política en función de recursos. Los principales pueden incluir cuentas, usuarios, roles, usuarios federados o. Servicios de AWS

Las políticas basadas en recursos son políticas insertadas que se encuentran en ese servicio. No puede usar políticas AWS administradas desde una política IAM basada en recursos.

Listas de control de acceso () ACLs

Las listas de control de acceso (ACLs) controlan qué responsables (miembros de la cuenta, usuarios o roles) tienen permisos para acceder a un recurso. ACLsson similares a las políticas basadas en recursos, aunque no utilizan el formato de documento de JSON políticas.

Amazon S3 AWS WAF y Amazon VPC son ejemplos de servicios compatiblesACLs. Para obtener más informaciónACLs, consulte la descripción general de la lista de control de acceso (ACL) en la Guía para desarrolladores de Amazon Simple Storage Service.

Otros tipos de políticas

AWS admite tipos de políticas adicionales y menos comunes. Estos tipos de políticas pueden establecer el máximo de permisos que los tipos de políticas más frecuentes le conceden.

- Límites de permisos: un límite de permisos es una función avanzada en la que se establecen los permisos máximos que una política basada en la identidad puede conceder a una IAM entidad (IAMusuario o rol). Puede establecer un límite de permisos para una entidad. Los permisos resultantes son la intersección de las políticas basadas en la identidad de la entidad y los límites de permisos. Las políticas basadas en recursos que especifiquen el usuario o rol en el campo `Principal` no estarán restringidas por el límite de permisos. Una denegación explícita en

cualquiera de estas políticas anulará el permiso. Para obtener más información sobre los límites de los permisos, consulte los límites de los permisos para IAM las entidades en la Guía del IAMusuario.

- Políticas de control de servicios (SCPs): SCPs son JSON políticas que especifican los permisos máximos para una organización o unidad organizativa (OU) en AWS Organizations. AWS Organizations es un servicio para agrupar y administrar de forma centralizada varios de los Cuentas de AWS que son propiedad de su empresa. Si habilitas todas las funciones de una organización, puedes aplicar políticas de control de servicios (SCPs) a una o a todas tus cuentas. SCPLimita los permisos de las entidades en las cuentas de los miembros, incluidas las de cada una Usuario raíz de la cuenta de AWS. Para obtener más información sobre OrganizationsSCPs, consulte las políticas de control de servicios en la Guía del AWS Organizations usuario.
- Políticas de sesión: las políticas de sesión son políticas avanzadas que se pasan como parámetro cuando se crea una sesión temporal mediante programación para un rol o un usuario federado. Los permisos de la sesión resultantes son la intersección de las políticas basadas en identidades del rol y las políticas de la sesión. Los permisos también pueden proceder de una política en función de recursos. Una denegación explícita en cualquiera de estas políticas anulará el permiso. Para obtener más información, consulte las políticas de sesión en la Guía del IAM usuario.

Varios tipos de políticas

Cuando se aplican varios tipos de políticas a una solicitud, los permisos resultantes son más complicados de entender. Para saber cómo se AWS determina si se debe permitir una solicitud cuando se trata de varios tipos de políticas, consulte la lógica de evaluación de políticas en la Guía del IAM usuario.

Cómo funciona Amazon Timestream for LiveAnalytics con IAM

Antes de administrar el acceso IAM a Timestream LiveAnalytics, debe saber qué IAM funciones están disponibles para su uso con Timestream. LiveAnalytics Para obtener una visión general de cómo funcionan Timestream for LiveAnalytics y otros AWS serviciosIAM, consulte los servicios con los que funcionan en la AWS Guía del usuario. IAM IAM

Temas

- Cronología de las políticas basadas en la identidad LiveAnalytics
- Cronología de las políticas basadas en recursos LiveAnalytics
- Autorización basada en Timestream para las etiquetas LiveAnalytics

- Secuencia temporal de los roles LiveAnalytics IAM

Cronología de las políticas basadas en la identidad LiveAnalytics

Con las políticas IAM basadas en la identidad, puede especificar las acciones y los recursos permitidos o denegados, así como las condiciones en las que se permiten o deniegan las acciones. Timestream for LiveAnalytics admite acciones y recursos específicos y claves de condición. Para obtener más información sobre todos los elementos que se utilizan en una JSON política, consulte la Referencia sobre los elementos IAM JSON de la política en la Guía del IAMusuario.

Acciones

Los administradores pueden usar AWS JSON políticas para especificar quién tiene acceso a qué. Es decir, qué entidad principal puede realizar acciones en qué recursos y en qué condiciones.

El `Action` elemento de una JSON política describe las acciones que puede utilizar para permitir o denegar el acceso en una política. Las acciones de política suelen tener el mismo nombre que la AWS API operación asociada. Hay algunas excepciones, como las acciones que solo permiten permisos y que no tienen una operación coincidente. API También hay algunas operaciones que requieren varias acciones en una política. Estas acciones adicionales se denominan acciones dependientes.

Incluya acciones en una política para conceder permisos y así llevar a cabo la operación asociada.

Puede especificar las siguientes acciones en el elemento Acción de una IAM declaración de política. Utilice políticas para conceder permisos para realizar una operación enAWS. Cuando utilizas una acción en una política, normalmente permites o deniegas el acceso a la API operación, CLI comando o SQL comando con el mismo nombre.

En algunos casos, una sola acción controla el acceso a una API operación y a un SQL comando. Asimismo, algunas operaciones requieren varias acciones diferentes.

Para ver una lista de los tipos de Timestream compatibles, consulta la siguiente tabla: LiveAnalytics `Action`

> **Note**
>
> Para todas las bases de datos específicas`Actions`, puede especificar una base de datos ARN para limitar la acción a una base de datos concreta.

Acciones	Descripción	Nivel de acceso	Tipos de recursos (*necesarios)
DescribeEndpoints	Devuelve el punto final de Timestrea m al que se deben realizar las solicitudes posteriores.	Todos	*
Seleccionar	Ejecute consultas en Timestream que seleccionen datos de una o más tablas. Consulte esta nota para obtener una explicación detallada	Leer	tabla*
CancelQuery	Cancelar una consulta.	Leer	*
ListTables	Obtenga la lista de tablas.	Enumeración	base de datos*
ListDatabases	Obtenga la lista de bases de datos.	Enumeración	*
ListMeasures	Obtenga la lista de medidas.	Leer	tabla*
DescribeTable	Obtenga la descripci ón de la tabla.	Leer	tabla*
DescribeDatabase	Obtenga la descripci ón de la base de datos.	Leer	base de datos*
SelectValues	Ejecute consultas que no requieran la especificación de un	Leer	*

Acciones	Descripción	Nivel de acceso	Tipos de recursos (*necesarios)
	recurso en particula r. Consulte esta nota para obtener una explicación detallada.		
WriteRecords	Inserte datos en Timestream.	Escritura	tabla*
CreateTable	Crear una tabla de .	Escritura	base de datos*
CreateDatabase	Cree una base de datos.	Escritura	*
DeleteDatabase	Eliminar una base de datos.	Escritura	*
UpdateDatabase	Actualizar una base de datos.	Escritura	*
DeleteTable	Eliminar una tabla.	Escritura	base de datos*
UpdateTable	Actualizar una tabla.	Escritura	base de datos*

SelectValues frente a seleccionar:

`SelectValues`es una `Action` que se utiliza para consultas que no requieren un recurso. Un ejemplo de consulta que no requiere un recurso es el siguiente:

```
SELECT 1
```

Tenga en cuenta que esta consulta no hace referencia a un flujo temporal específico para LiveAnalytics un recurso. Considera otro ejemplo:

```
SELECT now()
```

Esta consulta devuelve la marca de tiempo actual mediante la `now()` función, pero no requiere que se especifique ningún recurso. `SelectValuesse` utiliza a menudo para realizar pruebas, por lo que Timestream for LiveAnalytics puede ejecutar consultas sin recursos. Ahora, consideremos una `Select` consulta:

```
SELECT * FROM database.table
```

Este tipo de consulta requiere un recurso, específicamente un Timestream LiveAnalytics `table`, para poder obtener los datos especificados de la tabla.

Recursos

Los administradores pueden usar AWS JSON políticas para especificar quién tiene acceso a qué. Es decir, qué entidad principal puede realizar acciones en qué recursos y en qué condiciones.

El elemento `Resource` JSON de política especifica el objeto o los objetos a los que se aplica la acción. Las instrucciones deben contener un elemento `Resource` o `NotResource`. Como práctica recomendada, especifique un recurso mediante su nombre de recurso de Amazon (ARN). Puede hacerlo para acciones que admitan un tipo de recurso específico, conocido como permisos de nivel de recurso.

Para las acciones que no admiten permisos de nivel de recurso, como las operaciones de descripción, utilice un carácter comodín (*) para indicar que la instrucción se aplica a todos los recursos.

```
"Resource": "*"
```

En Timestream, para LiveAnalytics bases de datos y tablas, se puede utilizar como `Resource` elemento de IAM permisos.

El recurso Timestream para la LiveAnalytics base de datos tiene lo siguiente: ARN

```
arn:${Partition}:timestream:${Region}:${Account}:database/${DatabaseName}
```

El recurso Timestream for LiveAnalytics table tiene lo siguiente: ARN

```
arn:${Partition}:timestream:${Region}:${Account}:database/${DatabaseName}/table/
${TableName}
```

Para obtener más información sobre el formato deARNs, consulte Amazon Resource Names (ARNs) y AWS Service Namespaces.

Por ejemplo, para especificar el `database` espacio de claves en la declaración, utilice lo siguiente: ARN

```
"Resource": "arn:aws:timestream:us-east-1:123456789012:database/mydatabase"
```

Para especificar todas las bases de datos que pertenecen a una cuenta específica, utilice el comodín (*):

```
"Resource": "arn:aws:timestream:us-east-1:123456789012:database/*"
```

Algunas LiveAnalytics acciones de Timestream, como las de creación de recursos, no se pueden realizar en un recurso específico. En dichos casos, debe utilizar el carácter comodín (*).

```
"Resource": "*"
```

Claves de condición

Timestream for LiveAnalytics no proporciona ninguna clave de condición específica del servicio, pero sí admite el uso de algunas claves de condición globales. Para ver todas las claves de condición AWS globales, consulte las claves de contexto de condición AWS globales en la Guía del usuario. IAM

Ejemplos

Para ver ejemplos de Timestream para políticas LiveAnalytics basadas en la identidad, consulte. Amazon Timestream LiveAnalytics para ejemplos de políticas basadas en la identidad

Cronología de las políticas basadas en recursos LiveAnalytics

Timestream for LiveAnalytics no admite políticas basadas en recursos. Para ver un ejemplo de una página detallada de política basada en recursos, consulte https://docs.aws.amazon.com/lambda/latest/dg/access-control-resource-based.html.

Autorización basada en Timestream para las etiquetas LiveAnalytics

Puede administrar el acceso a su flujo temporal de LiveAnalytics recursos mediante etiquetas. Para administrar el acceso a recursos en función de etiquetas, proporcione información

de etiquetas en el elemento de condición de una política utilizando las claves de condición `timestream:ResourceTag/key-name`, `aws:RequestTag/key-name` o `aws:TagKeys`. Para obtener más información sobre cómo etiquetar Timestream para los recursos, consulte. LiveAnalytics the section called “Etiquetado de recursos”

Para ver ejemplos de políticas basadas en identidad para limitar el acceso a un recurso en función de las etiquetas de ese recurso, consulte Secuencia temporal de acceso a los LiveAnalytics recursos basada en etiquetas.

Secuencia temporal de los roles LiveAnalytics IAM

Un IAMrol es una entidad de tu AWS cuenta que tiene permisos específicos.

Usar credenciales temporales con Timestream para LiveAnalytics

Puede usar credenciales temporales para iniciar sesión con la federación, asumir un IAM rol o asumir un rol multicuenta. Para obtener credenciales de seguridad temporales, puede llamar a AWS STS API operaciones como AssumeRoleo GetFederationToken.

Roles vinculados al servicio

Timestream for no LiveAnalytics admite funciones vinculadas al servicio.

Roles de servicio

Timestream for LiveAnalytics no admite funciones de servicio.

AWS políticas gestionadas para Amazon Timestream Live Analytics

Una política AWS gestionada es una política independiente creada y administrada por. AWS AWS Las políticas administradas están diseñadas para proporcionar permisos para muchos casos de uso comunes, de modo que pueda empezar a asignar permisos a usuarios, grupos y funciones.

Ten en cuenta que es posible que las políticas AWS administradas no otorguen permisos con privilegios mínimos para tus casos de uso específicos, ya que están disponibles para que los usen todos los AWS clientes. Se recomienda definir políticas administradas por el cliente específicas para sus casos de uso a fin de reducir aún más los permisos.

No puedes cambiar los permisos definidos en AWS las políticas administradas. Si AWS actualiza los permisos definidos en una política AWS administrada, la actualización afecta a todas las identidades

principales (usuarios, grupos y roles) a las que está asociada la política. AWS es más probable que actualice una política AWS administrada cuando Servicio de AWS se lance una nueva o cuando haya nuevas API operaciones disponibles para los servicios existentes.

Para obtener más información, consulte las políticas AWS administradas en la Guía del IAM usuario.

Temas

- AWS política gestionada: AmazonTimestreamReadOnlyAccess
- AWS política gestionada: AmazonTimestreamConsoleFullAccess
- AWS política gestionada: AmazonTimestreamFullAccess
- Timestream Live Analytics actualiza las políticas gestionadas AWS

AWS política gestionada: AmazonTimestreamReadOnlyAccess

Puede asociar `AmazonTimestreamReadOnlyAccess` a los usuarios, grupos y roles. La política proporciona acceso de solo lectura a Amazon Timestream.

Detalles del permiso

Esta política incluye el siguiente permiso:

- `Amazon Timestream`— Proporciona acceso de solo lectura a Amazon Timestream. Esta política también otorga permiso para cancelar cualquier consulta en curso.

Para revisar el JSON formato de esta política, consulte AmazonTimestreamReadOnlyAccess.

AWS política gestionada: AmazonTimestreamConsoleFullAccess

Puede asociar `AmazonTimestreamConsoleFullAccess` a los usuarios, grupos y roles.

La política proporciona acceso completo para gestionar Amazon Timestream mediante. AWS Management Console Esta política también otorga permisos para determinadas AWS KMS operaciones y operaciones a fin de gestionar las consultas guardadas.

Detalles del permiso

Esta política incluye los permisos siguientes:

- `Amazon Timestream`— Otorga a los directores acceso completo a Amazon Timestream.
- `AWS KMS`— Permite a los directores enumerar los alias y describir las claves.
- `Amazon S3`— Permite a los directores enumerar todos los buckets de Amazon S3.
- `Amazon SNS`— Permite a los directores enumerar los SNS temas de Amazon.
- `IAM`— Permite a los directores IAM enumerar las funciones.
- `DBQMS`: permite a las entidades principales acceder, crear, eliminar, describir y actualizar consultas. El servicio de metadatos de consultas de bases de datos (dbqms) es un servicio exclusivamente interno. Proporciona las consultas recientes y guardadas para el editor de consultas de múltiples Servicios de AWS, incluida Amazon Timestream. AWS Management Console

Para revisar esta política en su JSON formato, consulte. AmazonTimestreamConsoleFullAccess

AWS política gestionada: AmazonTimestreamFullAccess

Puede asociar `AmazonTimestreamFullAccess` a los usuarios, grupos y roles.

La política proporciona acceso completo a Amazon Timestream. Esta política también otorga permisos para determinadas AWS KMS operaciones.

Detalles del permiso

Esta política incluye los permisos siguientes:

- `Amazon Timestream`— Otorga a los directores acceso completo a Amazon Timestream.
- `AWS KMS`— Permite a los directores enumerar los alias y describir las claves.
- `Amazon S3`— Permite a los directores enumerar todos los buckets de Amazon S3.

Para revisar el JSON formato de esta política, consulte. AmazonTimestreamFullAccess

Timestream Live Analytics actualiza las políticas gestionadas AWS

Consulte los detalles sobre las actualizaciones de las políticas AWS gestionadas de Timestream Live Analytics desde que este servicio comenzó a rastrear estos cambios. Para recibir alertas automáticas sobre los cambios en esta página, suscríbase al RSS feed de la página del historial de documentos de Timestream Live Analytics.

Cambio	Descripción	Fecha
AmazonTimestreamReadOnlyAccess: actualización de una política actual	Se agregó la `timestream:DescribeAccountSettings` acción a la política `AmazonTimestreamReadOnlyAccess` gestionada existente. Esta acción se utiliza para describir Cuenta de AWS la configuración. Timestream Live Analytics también ha actualizado esta política gestionada añadiendo un `Sid` campo. La actualización de la política no afecta al uso de la política `AmazonTimestreamReadOnlyAccess` gestionada.	3 de junio de 2024
AmazonTimestreamReadOnlyAccess: actualización de una política actual	Se agregaron las `timestream:ListBatchLoadTasks` acciones `timestream:DescribeBatchLoadTask` y a la política `AmazonTimestreamReadOnlyAccess` gestionada existente. Estas acciones se utilizan al enumerar y describir las tareas de carga por lotes.	24 de febrero de 2023

Cambio	Descripción	Fecha
	La actualización de la política no afecta al uso de la política `AmazonTimestreamReadOnlyAccess` gestionada.	
AmazonTimestreamReadOnlyAccess: actualización de una política actual	Se agregaron las `timestream:ListScheduledQueries` acciones `timestream:DescribeScheduledQuery` y a la política `AmazonTimestreamReadOnlyAccess` administrada existente. Estas acciones se utilizan al enumerar y describir las consultas programadas existentes. La actualización de la política no afecta al uso de la política `AmazonTimestreamReadOnlyAccess` gestionada.	29 de noviembre de 2021

Cambio	Descripción	Fecha
AmazonTimestreamConsoleFullAccess: actualización de una política actual	Se agregó la `s3:ListAllMyBuckets` acción a la política `AmazonTimestreamConsoleFullAccess` administrada existente. Esta acción se utiliza cuando se especific a un bucket de Amazon S3 para que Timestream registre los errores de escritura en almacenes magnéticos. La actualización de la política no afecta al uso de la política `AmazonTimestreamConsoleFullAccess` administrada.	29 de noviembre de 2021
AmazonTimestreamFullAccess: actualización de una política actual	Se agregó la `s3:ListAllMyBuckets` acción a la política `AmazonTimestreamFullAccess` administrada existente. Esta acción se utiliza cuando se especifica un bucket de Amazon S3 para que Timestream registre los errores de escritura en almacenes magnéticos. La actualización de la política no afecta al uso de la política `AmazonTimestreamFullAccess` administrada.	29 de noviembre de 2021

Cambio	Descripción	Fecha
AmazonTimestreamConsoleFullAccess: actualización de una política actual	Se eliminaron las acciones redundantes de la política `AmazonTimestreamConsoleFullAccess` gestionada existente. Anteriormente, esta política incluía una acción redundante. `dbqms:DescribeQueryHistory` La política actualizada elimina la acción redundante. La actualización de la política no afecta al uso de la política `AmazonTimestreamConsoleFullAccess` gestionada.	23 de abril de 2021
Timestream Live Analytics comenzó a rastrear los cambios	Timestream Live Analytics comenzó a realizar un seguimiento de los cambios en sus políticas gestionadas. AWS	21 de abril de 2021

Amazon Timestream LiveAnalytics para ejemplos de políticas basadas en la identidad

De forma predeterminada, IAM los usuarios y los roles no tienen permiso para crear o modificar Timestream para los recursos. LiveAnalytics Tampoco pueden realizar tareas con AWS Management Console, CQLSH AWS CLI, o. AWS API IAMEl administrador debe crear IAM políticas que concedan a los usuarios y roles permisos para realizar API operaciones específicas en los recursos específicos que necesitan. A continuación, el administrador debe adjuntar esas políticas a los IAM usuarios o grupos que requieran esos permisos.

Para obtener información sobre cómo crear una política IAM basada en la identidad con estos documentos de JSON política de ejemplo, consulte Creación de políticas en la JSON pestaña de la Guía del IAMusuario.

Temas

Prácticas recomendadas sobre las políticas

Las políticas basadas en la identidad determinan si alguien puede crear, acceder o eliminar Timestream para los recursos de tu cuenta. LiveAnalytics Estas acciones pueden generar costes adicionales para su Cuenta de AWS. Siga estas directrices y recomendaciones al crear o editar políticas basadas en identidades:

- Comience con las políticas AWS administradas y avance hacia los permisos con privilegios mínimos: para empezar a conceder permisos a sus usuarios y cargas de trabajo, utilice las políticas AWS administradas que otorgan permisos para muchos casos de uso comunes. Están disponibles en su. Cuenta de AWS Le recomendamos que reduzca aún más los permisos definiendo políticas administradas por el AWS cliente que sean específicas para sus casos de uso. Para obtener más información, consulte las políticas AWS gestionadas o las políticas AWS gestionadas para las funciones laborales en la Guía del IAM usuario.
- Aplique permisos con privilegios mínimos: cuando establezca permisos con IAM políticas, conceda solo los permisos necesarios para realizar una tarea. Para ello, debe definir las acciones que se pueden llevar a cabo en determinados recursos en condiciones específicas, también conocidos como permisos de privilegios mínimos. Para obtener más información sobre cómo IAM aplicar permisos, consulte Políticas y permisos IAM en la IAM Guía del usuario.
- Utilice las condiciones en IAM las políticas para restringir aún más el acceso: puede añadir una condición a sus políticas para limitar el acceso a las acciones y los recursos. Por ejemplo, puede escribir una condición de política para especificar que todas las solicitudes deben enviarse medianteSSL. También puedes usar condiciones para conceder el acceso a las acciones del servicio si se utilizan a través de una acción específica Servicio de AWS, por ejemplo AWS

CloudFormation. Para obtener más información, consulte los elementos IAM JSON de la política: Condición en la Guía del IAM usuario.

- Utilice IAM Access Analyzer para validar sus IAM políticas y garantizar permisos seguros y funcionales: IAM Access Analyzer valida las políticas nuevas y existentes para que se ajusten al lenguaje de las políticas (JSON) y IAM a las IAM mejores prácticas. IAMAccess Analyzer proporciona más de 100 comprobaciones de políticas y recomendaciones prácticas para ayudarle a crear políticas seguras y funcionales. Para obtener más información, consulte Validar políticas con IAM Access Analyzer en la Guía del IAM usuario.
- Requerir autenticación multifactorial (MFA): si se encuentra en una situación en la que se requieren IAM usuarios o un usuario raíz Cuenta de AWS, actívela MFA para aumentar la seguridad. Para solicitarlo MFA cuando se convoque a API las operaciones, añada MFA condiciones a sus políticas. Para obtener más información, consulte APIAcceso seguro con MFA en la Guía del IAM usuario.

Para obtener más información sobre las prácticas recomendadasIAM, consulte las prácticas recomendadas de seguridad IAM en la Guía del IAM usuario.

Uso de Timestream para consola LiveAnalytics

Timestream for no LiveAnalytics requiere permisos específicos para acceder a Amazon Timestream para consola. LiveAnalytics Necesita al menos permisos de solo lectura para enumerar y ver detalles sobre el Timestream de los recursos de su cuenta. LiveAnalytics AWS Si creas una política basada en la identidad que sea más restrictiva que los permisos mínimos requeridos, la consola no funcionará según lo previsto para las entidades (IAMusuarios o roles) que cuenten con esa política.

Cómo permitir a los usuarios consultar sus propios permisos

En este ejemplo, se muestra cómo se puede crear una política que permita a IAM los usuarios ver las políticas integradas y administradas asociadas a su identidad de usuario. Esta política incluye permisos para completar esta acción en la consola o mediante programación mediante la tecla o. AWS CLI AWS API

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

```
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

Operaciones habituales en Timestream para LiveAnalytics

A continuación, se muestran ejemplos de IAM políticas que permiten realizar operaciones comunes en el Timestream for Service. LiveAnalytics

Temas

- ¿Permitiendo todas las operaciones
- Permitir operaciones SELECT
- Permite SELECT realizar operaciones en varios recursos
- Permitir operaciones de metadatos
- Permitir operaciones INSERT
- Permitir CRUD operaciones
- Cancele las consultas y seleccione los datos sin especificar los recursos
- Cree, describa, elimine y describa una base de datos

- Limite las bases de datos de la lista por etiqueta {"Owner": "${username}"}
- Listar todas las tablas de una base de datos
- Crear, describir, eliminar, actualizar y seleccionar una tabla
- Limitar una consulta por tabla

¿Permitiendo todas las operaciones

El siguiente es un ejemplo de política que permite realizar todas las operaciones en Timestream durante. LiveAnalytics

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "timestream:*"
            ],
            "Resource": "*"
        }
    ]
}
```

Permitir operaciones SELECT

El siguiente ejemplo de política permite realizar consultas de `SELECT` estilo en un recurso específico.

> **Note**
>
> `<account_ID>`Sustitúyelo por el ID de tu cuenta de Amazon.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "timestream:Select",
                "timestream:DescribeTable",
```

```
                "timestream:ListMeasures"
            ],
            "Resource": "arn:aws:timestream:us-east-1:<account_ID>:database/sampleDB/
table/DevOps"
        },
        {
            "Effect": "Allow",
            "Action": [
                "timestream:DescribeEndpoints",
                "timestream:SelectValues",
                "timestream:CancelQuery"
            ],
            "Resource": "*"
        }
    ]
}
```

Permite SELECT realizar operaciones en varios recursos

El siguiente ejemplo de política permite realizar consultas de SELECT estilo en varios recursos.

> **Note**
>
> <account_ID>Sustitúyelo por el ID de tu cuenta de Amazon.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "timestream:Select",
                "timestream:DescribeTable",
                "timestream:ListMeasures"
            ],
            "Resource": [
                "arn:aws:timestream:us-east-1:<account_ID>:database/sampleDB/table/
DevOps",
                "arn:aws:timestream:us-east-1:<account_ID>:database/sampleDB/table/
DevOps1",
                "arn:aws:timestream:us-east-1:<account_ID>:database/sampleDB/table/
DevOps2"
```

```
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "timestream:DescribeEndpoints",
                "timestream:SelectValues",
                "timestream:CancelQuery"
            ],
            "Resource": "*"
        }
    ]
}
```

Permitir operaciones de metadatos

El siguiente ejemplo de política permite al usuario realizar consultas de metadatos, pero no permite al usuario realizar operaciones que lean o escriban datos reales en Timestream. LiveAnalytics

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "timestream:DescribeEndpoints",
                "timestream:DescribeTable",
                "timestream:ListMeasures",
                "timestream:SelectValues",
                "timestream:ListTables",
                "timestream:ListDatabases",
                "timestream:CancelQuery"
            ],
            "Resource": "*"
        }
    ]
}
```

Permitir operaciones INSERT

El siguiente ejemplo de política permite a un usuario realizar una `INSERT` operación `database/sampleDB/table/DevOps` en su cuenta`<account_id>`.

> **Note**
>
> <account_ID>Sustitúyelo por el ID de tu cuenta de Amazon.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "timestream:WriteRecords"
            ],
            "Resource": [
                "arn:aws:timestream:us-east-1:<account_id>:database/sampleDB/table/
DevOps"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "timestream:DescribeEndpoints"
            ],
            "Resource": "*",
            "Effect": "Allow"
        }
    ]
}
```

Permitir CRUD operaciones

El siguiente ejemplo de política permite a un usuario realizar CRUD operaciones en Timestream para. LiveAnalytics

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "timestream:DescribeEndpoints",
                "timestream:CreateTable",
                "timestream:DescribeTable",
```

```
                "timestream:CreateDatabase",
                "timestream:DescribeDatabase",
                "timestream:ListTables",
                "timestream:ListDatabases",
                "timestream:DeleteTable",
                "timestream:DeleteDatabase",
                "timestream:UpdateTable",
                "timestream:UpdateDatabase"
            ],
            "Resource": "*"
        }
    ]
}
```

Cancele las consultas y seleccione los datos sin especificar los recursos

El siguiente ejemplo de política permite al usuario cancelar consultas y realizar `Select` consultas sobre datos que no requieren la especificación de recursos:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "timestream:SelectValues",
                "timestream:CancelQuery"
            ],
            "Resource": "*"
        }
    ]
}
```

Cree, describa, elimine y describa una base de datos

El siguiente ejemplo de política permite a un usuario crear, describir, eliminar y describir una base de datos`sampleDB`:

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

```
            "Action": [
                "timestream:CreateDatabase",
                "timestream:DescribeDatabase",
                "timestream:DeleteDatabase",
                "timestream:UpdateDatabase"
            ],
            "Resource": "arn:aws:timestream:us-east-1:<account_ID>:database/sampleDB"
        }
    ]
}
```

Limite las bases de datos de la lista por etiqueta **`{"Owner": "${username}"}`**

El siguiente ejemplo de política permite al usuario enumerar todas las bases de datos etiquetadas con un par clave-valor`{"Owner": "${username}"}`:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "timestream:ListDatabases"
            ],
            "Resource": "arn:aws:timestream:us-east-1:<account_ID>:database/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Owner": "${aws:username}"
                }
            }
        }
    ]
}
```

Listar todas las tablas de una base de datos

El siguiente ejemplo de política para enumerar todas las tablas de la base de datos`sampleDB`:

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

```
            "Action": [
                "timestream:ListTables"
            ],
            "Resource": "arn:aws:timestream:us-east-1:<account_ID>:database/sampleDB/"
        }
    ]
}
```

Crear, describir, eliminar, actualizar y seleccionar una tabla

El siguiente ejemplo de política permite al usuario crear tablas, describir tablas, eliminar tablas, actualizar tablas y realizar `Select` consultas sobre las tablas de la base `DevOps` de datos`sampleDB`:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "timestream:CreateTable",
                "timestream:DescribeTable",
                "timestream:DeleteTable",
                "timestream:UpdateTable",
                "timestream:Select"
            ],
            "Resource": "arn:aws:timestream:us-east-1:<account_ID>:database/sampleDB/
table/DevOps"
        }
    ]
}
```

Limitar una consulta por tabla

El siguiente ejemplo de política permite al usuario consultar todas las tablas excepto las de la base `DevOps` de datos`sampleDB`:

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

```
            "Action": [
                "timestream:Select"
            ],
            "Resource": "arn:aws:timestream:us-east-1:<account_ID>:database/sampleDB/
table/*"
        },
        {
            "Effect": "Deny",
            "Action": [
                "timestream:Select"
            ],
            "Resource": "arn:aws:timestream:us-east-1:<account_ID>:database/sampleDB/
table/DevOps"
        }
    ]
}
```

Secuencia temporal de acceso a los LiveAnalytics recursos basada en etiquetas

Puede utilizar las condiciones de su política basada en la identidad para controlar el acceso a Timestream de los recursos en función de las etiquetas. LiveAnalytics En esta sección se presentan algunos ejemplos.

En el siguiente ejemplo se muestra cómo puede crear una política que conceda permisos a un usuario para ver una tabla si el `Owner` de la tabla contiene el valor del nombre de usuario de ese usuario.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ReadOnlyAccessTaggedTables",
            "Effect": "Allow",
            "Action": "timestream:Select",
            "Resource": "arn:aws:timestream:us-east-2:111122223333:database/mydatabase/
table/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Owner": "${aws:username}"
                }
            }
        }
    ]
```

```
}
```

Puedes adjuntar esta política a los IAM usuarios de tu cuenta. Si un usuario llamado `richard-roe` intenta ver la secuencia temporal de una LiveAnalytics tabla, la tabla debe estar etiquetada `Owner=richard-roe` o. `owner=richard-roe` De lo contrario, se le deniega el acceso. La clave de la etiqueta de condición `Owner` coincide con los nombres de las claves de condición `Owner` y `owner` porque no distinguen entre mayúsculas y minúsculas. Para obtener más información, consulte Elementos IAM JSON de política: condición en la Guía del IAMusuario.

La siguiente política otorga permisos a un usuario para crear tablas con etiquetas si la etiqueta aprobada en la solicitud tiene una clave `Owner` y un valor`username`:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "CreateTagTableUser",
            "Effect": "Allow",
            "Action": [
                "timestream:Create",
                "timestream:TagResource"
            ],
            "Resource": "arn:aws:timestream:us-east-2:111122223333:database/mydatabase/
table/*",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "aws:RequestTag/Owner": "${aws:username}"
                }
            }
        }
    ]
}
```

La siguiente política permite el uso de la `DescribeDatabase` API en cualquier base de datos que tenga la `env` etiqueta establecida en`test`: `dev`

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

```
        "timestream:DescribeEndpoints"
      ],
      "Resource": "*"
    },
    {
      "Sid": "AllowDevTestAccess",
      "Effect": "Allow",
      "Action": [
        "timestream:DescribeDatabase"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "timestream:tag/env": [
              "dev",
              "test"
          ]
        }
      }
    }
  ]
}
{ "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowTagAccessForDevResources",
      "Effect": "Allow",
      "Action": [
        "timestream:TagResource"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:RequestTag/env": [
            "test",
            "dev"
          ]
        }
      }
    }
  ]
}
```

Esta política usa una `Condition` clave para permitir que una etiqueta que tenga la clave `env` y un valor de `testqa`,, se agregue a un recurso. `dev`

Consultas programadas

Listar, eliminar, actualizar y ejecutar ScheduledQuery

El siguiente ejemplo de política permite a un usuario enumerar, eliminar, actualizar y ejecutar consultas programadas.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "timestream:DeleteScheduledQuery",
                "timestream:ExecuteScheduledQuery",
                "timestream:UpdateScheduledQuery",
                "timestream:ListScheduledQueries",
                "timestream:DescribeEndpoints"
            ],
            "Resource": "*"
        }
    ]
}
```

CreateScheduledQuery utilizando una KMS clave gestionada por el cliente

El siguiente ejemplo de política permite al usuario crear una consulta programada cifrada mediante una KMS clave gestionada por el cliente; *<keyid for ScheduledQuery>*.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "iam:PassRole"
            ],
            "Resource": [
                "arn:aws:iam::123456789012:role/ScheduledQueryExecutionRole"
            ],
            "Effect": "Allow"
```

```
        },
        {
            "Action": [
                "timestream:CreateScheduledQuery",
                "timestream:DescribeEndpoints"
            ],
            "Resource": "*",
            "Effect": "Allow"
        },
        {
            "Action": [
                "kms:DescribeKey",
                "kms:GenerateDataKey"
            ],
            "Resource": "arn:aws:kms:us-west-2:123456789012:key/<keyid for
 ScheduledQuery>",
            "Effect": "Allow"
        }
    ]
}
```

DescribeScheduledQuery utilizar una KMS clave gestionada por el cliente

El siguiente ejemplo de política permite al usuario describir una consulta programada que se creó con una KMS clave gestionada por el cliente; *<keyid for ScheduledQuery>*.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "timestream:DescribeScheduledQuery",
                "timestream:DescribeEndpoints"
            ],
            "Resource": "*",
            "Effect": "Allow"
        },
        {
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": "arn:aws:kms:us-west-2:123456789012:key/<keyid for
 ScheduledQuery>",
```

```
            "Effect": "Allow"
        }
    ]
}
```

Permisos de función de ejecución (mediante una KMS clave gestionada por el cliente para consultas programadas ySSE, KMS para informes de errores)

Adjunte el siguiente ejemplo de política al IAM rol especificado en el `ScheduledQueryExecutionRoleArn` parámetro: el `CreateScheduledQuery` API que usa la KMS clave administrada por el cliente para el cifrado de consultas programadas y el `SSE-KMS` cifrado para los informes de errores.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "kms:GenerateDataKey",
            ],
            "Resource": "arn:aws:kms:us-west-2:123456789012:key/<keyid for
 ScheduledQuery>",
            "Effect": "Allow"
        },
        {
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:us-west-2:123456789012:key/<keyid for database-1>",
                "arn:aws:kms:us-west-2:123456789012:key/<keyid for database-n>",
                "arn:aws:kms:us-west-2:123456789012:key/<keyid for ScheduledQuery>"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "sns:Publish"
            ],
            "Resource": [
                "arn:aws:sns:us-west-2:123456789012:scheduled-query-notification-topic-
*"
            ],
```

```
            "Effect": "Allow"
        },
        {
            "Action": [
                "timestream:Select",
                "timestream:SelectValues",
                "timestream:WriteRecords"
            ],
            "Resource": "*",
            "Effect": "Allow"
        },
        {
            "Action": [
                "s3:PutObject",
                "s3:GetBucketAcl"
            ],
            "Resource": [
                "arn:aws:s3:::scheduled-query-error-bucket",
                "arn:aws:s3:::scheduled-query-error-bucket/*"
            ],
            "Effect": "Allow"
        }
    ]
}
```

Rol de ejecución: relación de confianza

La siguiente es la relación de confianza para el IAM rol especificado en el `ScheduledQueryExecutionRoleArn` parámetro de `CreateScheduledQuery`API.

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

```
}
```

Permitir el acceso a todas las consultas programadas creadas en una cuenta

Adjunte el siguiente ejemplo de política a la IAM función especificada en el `ScheduledQueryExecutionRoleArn` parámetro, de `CreateScheduledQuery`API, para permitir el acceso a todas las consultas programadas creadas en la cuenta *`Account_ID`*.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "timestream.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "Account_ID"
                },
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:timestream:us-
west-2:Account_ID:scheduled-query/*"
                }
            }
        }
    ]
}
```

Permita el acceso a todas las consultas programadas con un nombre específico

Adjunte el siguiente ejemplo de política al IAM rol especificado en el `ScheduledQueryExecutionRoleArn` parámetro, del `CreateScheduledQuery`API, para permitir el acceso a todas las consultas programadas con un nombre que comience por *`Scheduled_Query_Name`*, dentro de la cuenta *`Account_ID`*.

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

```
            "Principal": {
                "Service": "timestream.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "Account_ID"
                },
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:timestream:us-
west-2:Account_ID:scheduled-query/Scheduled_Query_Name*"
                }
            }
        }
    ]
}
```

Solución de problemas de identidad y acceso en Amazon Timestream LiveAnalytics

Utilice la siguiente información para ayudarle a diagnosticar y solucionar los problemas más comunes que pueden surgir al trabajar con Timestream para y. LiveAnalytics IAM

Temas

- No estoy autorizado a realizar ninguna acción en Timestream para LiveAnalytics
- No estoy autorizado a realizar el iam: PassRole
- Quiero permitir que personas ajenas a mi AWS cuenta accedan a mi Timestream para obtener recursos LiveAnalytics

No estoy autorizado a realizar ninguna acción en Timestream para LiveAnalytics

Si AWS Management Console le indica que no está autorizado a realizar una acción, debe ponerse en contacto con su administrador para obtener ayuda. El administrador es la persona que le proporcionó las credenciales de inicio de sesión.

El siguiente ejemplo de error se produce cuando el `mateojackson` IAM usuario intenta utilizar la consola para ver los detalles de un *`table`* pero no tiene `timestream:`*`Select`* permisos para la tabla.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 timestream:Select on resource: mytable
```

En este caso, Mateo pide a su administrador que actualice sus políticas de forma que pueda obtener acceso al recurso *mytable* mediante la acción `timestream:Select`.

No estoy autorizado a realizar el iam: PassRole

Si recibes un mensaje de error que indica que no estás autorizado a realizar la `iam:PassRole` acción, debes actualizar tus políticas para que puedas transferir una función a Timestream. LiveAnalytics

Algunas Servicios de AWS permiten transferir una función existente a ese servicio en lugar de crear una nueva función de servicio o una función vinculada al servicio. Para ello, debe tener permisos para transferir el rol al servicio.

El siguiente ejemplo de error se produce cuando un IAM usuario denominado `marymajor` intenta utilizar la consola para realizar una acción en Timestream. LiveAnalytics Sin embargo, la acción requiere que el servicio cuente con permisos que otorguen un rol de servicio. Mary no tiene permisos para transferir el rol al servicio.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

En este caso, las políticas de Mary se deben actualizar para permitirle realizar la acción `iam:PassRole`.

Si necesita ayuda, póngase en contacto con su administrador. AWS El administrador es la persona que le proporcionó las credenciales de inicio de sesión.

Quiero permitir que personas ajenas a mi AWS cuenta accedan a mi Timestream para obtener recursos LiveAnalytics

Puede crear un rol que los usuarios de otras cuentas o las personas externas a la organización puedan utilizar para acceder a sus recursos. Puede especificar una persona de confianza para que asuma el rol. En el caso de los servicios que admiten políticas basadas en recursos o listas de control de acceso (ACLs), puedes usar esas políticas para permitir que las personas accedan a tus recursos.

Para más información, consulte lo siguiente:

- Para saber si Timestream for LiveAnalytics admite estas funciones, consulte. Cómo funciona Amazon Timestream for LiveAnalytics con IAM

- Para obtener información sobre cómo proporcionar acceso a los recursos de su propiedad, consulte Proporcionar acceso a un IAM usuario en otro de su Cuenta de AWS propiedad en la Guía del IAMusuario. Cuentas de AWS
- Para obtener información sobre cómo proporcionar acceso a tus recursos a terceros Cuentas de AWS, consulta Cómo permitir el acceso a recursos que Cuentas de AWS son propiedad de terceros en la Guía del IAM usuario.
- Para obtener información sobre cómo proporcionar acceso mediante la federación de identidades, consulte Proporcionar acceso a usuarios autenticados externamente (federación de identidades) en la Guía del IAM usuario.
- Para saber la diferencia entre el uso de roles y políticas basadas en recursos para el acceso entre cuentas, consulte el acceso a recursos entre cuentas IAM en la Guía del usuario. IAM

Registro y supervisión en Timestream para LiveAnalytics

El monitoreo es una parte importante del mantenimiento de la confiabilidad, la disponibilidad y el rendimiento de Timestream para LiveAnalytics y sus soluciones. AWS Debe recopilar los datos de supervisión de todas las partes de la AWS solución para poder depurar más fácilmente una falla multipunto en caso de que se produzca. Sin embargo, antes de empezar a monitorear Timestream LiveAnalytics, debe crear un plan de monitoreo que incluya respuestas a las siguientes preguntas:

- ¿Cuáles son los objetivos de la supervisión?
- ¿Qué recursos va a supervisar?
- ¿Con qué frecuencia va a supervisar estos recursos?
- ¿Qué herramientas de monitoreo va a utilizar?
- ¿Quién se encargará de realizar las tareas de monitoreo?
- ¿Quién debería recibir una notificación cuando surjan problemas?

El siguiente paso es establecer una línea base para el flujo temporal normal del LiveAnalytics rendimiento en su entorno, midiendo el rendimiento en distintos momentos y bajo diferentes condiciones de carga. A medida que supervise Timestream LiveAnalytics, almacene los datos de supervisión históricos para poder compararlos con los datos de rendimiento actuales, identificar los patrones de rendimiento normales y las anomalías de rendimiento, y diseñar métodos para abordar los problemas.

Para establecer un punto de referencia debe, como mínimo, monitorizar los elementos siguientes:

- Errores del sistema, para que pueda determinar si alguna solicitud ha provocado un error.

Temas

Herramientas de monitoreo

AWS proporciona varias herramientas que puede utilizar para supervisar Timestream. LiveAnalytics Puede configurar algunas de estas herramientas para que monitoricen por usted, pero otras herramientas requieren intervención manual. Le recomendamos que automatice las tareas de monitorización en la medida de lo posible.

Temas

Herramientas de monitoreo automatizadas

Puede utilizar las siguientes herramientas de supervisión automática para vigilar Timestream LiveAnalytics e informar cuando algo vaya mal:

- Amazon CloudWatch Alarms: observe una sola métrica durante un período de tiempo que especifique y realice una o más acciones en función del valor de la métrica en relación con un umbral determinado durante varios períodos de tiempo. La acción es una notificación enviada a un tema de Amazon Simple Notification Service (AmazonSNS) o a una política de Amazon EC2 Auto Scaling. CloudWatch las alarmas no invocan acciones simplemente porque se encuentran en un estado determinado; el estado debe haber cambiado y se ha mantenido durante un número específico de períodos. Para obtener más información, consulte Monitorización con Amazon CloudWatch.

Herramientas de monitoreo manuales

Otra parte importante de la supervisión de Timestream LiveAnalytics consiste en supervisar manualmente los elementos que las CloudWatch alarmas no cubren. Los AWS Management

Console paneles Timestream for LiveAnalytics, CloudWatch Trusted Advisor, y otros proporcionan una at-a-glance vista del estado de su entorno. AWS

- La página de CloudWatch inicio muestra lo siguiente:
  - Alarmas y estado actual
  - Gráficos de alarmas y recursos
  - Estado de los servicios

  Además, puede CloudWatch hacer lo siguiente:
  - Crear paneles personalizados para monitorizar los servicios que le interesan
  - Realizar un gráfico con los datos de las métricas para resolver problemas y descubrir tendencias
  - Busque y explore todas sus métricas AWS de recursos
  - Crear y editar las alarmas de notificación de problemas

Registrar la transmisión temporal de las llamadas con LiveAnalytics API AWS CloudTrail

Timestream for LiveAnalytics está integrado con AWS CloudTrail un servicio que proporciona un registro de las acciones realizadas por un usuario, un rol o un AWS servicio de Timestream for. LiveAnalytics CloudTrail captura el lenguaje de definición de datos (DDL) y utiliza API Timestream como eventos. LiveAnalytics Las llamadas que se capturan incluyen las llamadas del Timestream para la LiveAnalytics consola y las llamadas en código al Timestream para las operaciones. LiveAnalytics API Si crea un registro, puede habilitar la entrega continua de CloudTrail eventos a un bucket de Amazon Simple Storage Service (Amazon S3), incluidos los eventos de Timestream for. LiveAnalytics Si no configura una ruta, podrá ver los eventos más recientes en la CloudTrail consola, en el historial de eventos. Con la información recopilada por usted CloudTrail, puede determinar la solicitud que se realizó a Timestream LiveAnalytics, la dirección IP desde la que se realizó la solicitud, quién la hizo, cuándo se realizó y detalles adicionales.

Para obtener más información CloudTrail, consulte la Guía del AWS CloudTrail usuario.

Cronograma para obtener información en LiveAnalytics CloudTrail

CloudTrail está activado en su AWS cuenta al crear la cuenta. Cuando se produce una actividad en Timestream for LiveAnalytics, esa actividad se registra en un CloudTrail evento junto con otros eventos de AWS servicio en el historial de eventos. Puede ver, buscar y descargar los últimos

eventos de la cuenta de AWS . Para obtener más información, consulte Visualización de eventos con el historial de CloudTrail eventos.

> **Warning**
>
> Actualmente, Timestream for LiveAnalytics genera CloudTrail eventos para toda la administración y `Query` API las operaciones, pero no genera eventos para `WriteRecords` y. `DescribeEndpoints` APIs

Para tener un registro continuo de los eventos de tu AWS cuenta, incluidos los eventos de Timestream LiveAnalytics, crea un registro. Un rastro permite CloudTrail entregar archivos de registro a un bucket de Amazon S3. De forma predeterminada, cuando crea una ruta en la consola, la ruta se aplica a todas AWS las regiones. La ruta registra los eventos de todas las regiones de la AWS partición y envía los archivos de registro al bucket de Amazon S3 que especifique. Además, puede configurar otros AWS servicios para analizar más a fondo los datos de eventos recopilados en los CloudTrail registros y actuar en función de ellos.

Para obtener más información, consulte los siguientes temas en la Guía del usuario de AWS CloudTrail :

- Introducción a la creación de registros de seguimiento
- CloudTrail Integraciones y servicios compatibles
- Configuración de Amazon SNS Notifications para CloudTrail
- Recibir archivos de CloudTrail registro de varias regiones
- Recibir archivos de CloudTrail registro de varias cuentas
- Registrar eventos de datos

Cada entrada de registro o evento contiene información sobre quién generó la solicitud. La información de identidad del usuario lo ayuda a determinar lo siguiente:

- Si la solicitud se realizó con credenciales de usuario root o AWS Identity and Access Management (IAM)
- si la solicitud se realizó con credenciales de seguridad temporales de un rol o fue un usuario federado
- Si la solicitud la realizó otro AWS servicio

Para obtener más información, consulte el CloudTrail userIdentityElemento.

Para `Query` API eventos:

- Cree una ruta que reciba todos los eventos o seleccione los eventos con Timestream como tipo `AWS::Timestream::Database` de LiveAnalytics recurso o. `AWS::Timestream::Table`
- `Query`APIlas solicitudes que no acceden a ninguna base de datos o tabla o que den lugar a una excepción de validación debido a un formato incorrecto de la cadena de consulta se registran CloudTrail con un tipo de recurso `AWS::Timestream::Database` y un ARN valor de:

  ```
  arn:aws:timestream:(region):(accountId):database/NO_RESOURCE_ACCESSED
  ```

  Estos eventos se envían solo a las rutas que reciben eventos con un tipo `AWS::Timestream::Database` de recurso.

Resiliencia en Amazon Timestream Live Analytics

La infraestructura AWS global se basa en AWS regiones y zonas de disponibilidad. AWS Las regiones proporcionan varias zonas de disponibilidad aisladas y separadas físicamente, que están conectadas mediante redes de baja latencia, alto rendimiento y alta redundancia. Con las zonas de disponibilidad, puede diseñar y utilizar aplicaciones y bases de datos que realizan una conmutación por error automática entre las zonas sin interrupciones. Las zonas de disponibilidad tienen una mayor disponibilidad, tolerancia a errores y escalabilidad que las infraestructuras tradicionales de uno o varios centros de datos.

Para obtener más información sobre AWS las regiones y las zonas de disponibilidad, consulte Infraestructura global.AWS

Para obtener información sobre la funcionalidad de protección de datos de Timestream disponible en AWS Backup, consulte. Trabajando con AWS Backup

Seguridad de la infraestructura en Amazon Timestream Live Analytics

Como servicio gestionado, Amazon Timestream Live Analytics está protegido por los procedimientos de seguridad de red global que se describen en AWS el documento técnico Amazon Web Services: Overview of Security Processes.

APILas llamadas AWS publicadas se utilizan para acceder a Timestream Live Analytics a través de la red. Los clientes deben ser compatibles con Transport Layer Security (TLS) 1.0 o una versión

posterior. Recomendamos la versión TLS 1.2 o una versión posterior. Los clientes también deben admitir conjuntos de cifrado con total confidencialidad (PFS), como Ephemeral Diffie-Hellman () o Elliptic Curve Ephemeral Diffie-Hellman (DHE). ECDHE La mayoría de los sistemas modernos como Java 7 y posteriores son compatibles con estos modos.

Además, las solicitudes deben firmarse con un identificador de clave de acceso y una clave de acceso secreta que esté asociada a un director. IAM También puede utilizar AWS Security Token Service (AWS STS) para generar credenciales de seguridad temporales para firmar solicitudes.

Timestream Live Analytics está diseñado de forma que el tráfico esté aislado en la AWS región específica en la que reside la instancia de Timestream Live Analytics.

Análisis de configuración y vulnerabilidad en Timestream

La configuración y los controles de TI son una responsabilidad compartida entre usted AWS y usted, nuestro cliente. Para obtener más información, consulte el modelo de responsabilidad AWS compartida. Además del modelo de responsabilidad compartida, Timestream para LiveAnalytics los usuarios debe tener en cuenta lo siguiente:

- Es responsabilidad del cliente colocar parches a sus aplicaciones cliente con las dependencias relevantes del lado del cliente.
- Los clientes deberían considerar la posibilidad de realizar pruebas de penetración, si procede (consulte las pruebas de https://aws.amazon.com/security/penetración/.)

Respuesta a incidentes en Timestream para LiveAnalytics

Los incidentes de Amazon Timestream LiveAnalytics para el servicio se indican en el Personal Health Dashboard. Puede obtener más información sobre el panel de control aquí. AWS Health

Timestream for LiveAnalytics admite el uso de informes. AWS CloudTrail Para obtener más información, consulte Registrar la transmisión temporal de las llamadas con LiveAnalytics API AWS CloudTrail.

VPCpuntos finales ()AWS PrivateLink

Puede establecer una conexión privada entre Amazon Timestream VPC y usted creando un punto de enlace LiveAnalytics de interfaz. VPC Los puntos finales de la interfaz funcionan con una tecnología que le permite acceder de forma privada a Timestream LiveAnalytics APIs sin una puerta de enlace

a Internet, NAT dispositivo, VPN conexión o conexión Direct AWS Connect. AWS PrivateLink Sus instancias VPC no necesitan direcciones IP públicas para comunicarse con Timestream. LiveAnalytics APIs El tráfico entre usted VPC y Timestream for LiveAnalytics no sale de la red de Amazon.

Cada punto de conexión de la interfaz está representado por una o más interfaces de red elásticas en las subredes. Para obtener más información sobre los puntos de VPC enlace de la interfaz, consulte los puntos de VPCenlace de la interfaz (AWS PrivateLink) en la Guía VPCdel usuario de Amazon.

Para empezar con Timestream para los puntos de conexión LiveAnalytics y los VPC puntos de enlace, hemos proporcionado información sobre aspectos específicos de Timestream para los VPC puntos de conexión, la creación de un VPC punto final de interfaz para LiveAnalytics Timestream, la creación de una política de puntos finales para LiveAnalytics Timestream y el uso del cliente de Timestream (para LiveAnalytics escritura o consulta) con los puntos de VPC enlace. SDK VPC

Temas

- Cómo funcionan los puntos finales con Timestream VPC
- Crear un punto final de interfaz para Timestream para VPC LiveAnalytics
- Crear una política de VPC puntos finales para Timestream para LiveAnalytics

Cómo funcionan los puntos finales con Timestream VPC

Cuando creas un VPC punto final para acceder a Timestream Write o Timestream QuerySDK, todas las solicitudes se dirigen a puntos finales de la red de Amazon y no acceden a la Internet pública. Más específicamente, tus solicitudes se envían a los puntos finales de escritura y consulta de la celda a la que se ha asignado tu cuenta para una región determinada. Para obtener más información sobre la arquitectura móvil de Timestream y los puntos finales específicos de cada celda, puedes consultar. Arquitectura celular Por ejemplo, supongamos que su cuenta se ha asignado a `cell1` in `us-west-2` y que ha configurado los puntos finales de la VPC interfaz para las escrituras () y las consultas (). `ingest-cell1.timestream.us-west-2.amazonaws.com` `query-cell1.timestream.us-west-2.amazonaws.com` En este caso, cualquier solicitud de escritura que se envíe mediante estos puntos de conexión permanecerá íntegramente dentro de la red de Amazon y no accederá a la Internet pública.

Consideraciones sobre los puntos finales de Timestream VPC

Tenga en cuenta lo siguiente al crear un VPC punto final para Timestream:

- Antes de configurar un VPC punto final de interfaz para Timestream LiveAnalytics, asegúrese de revisar las propiedades y limitaciones del punto final de interfaz en la Guía VPCdel usuario de Amazon.
- Timestream for LiveAnalytics permite realizar llamadas a todas sus acciones desde su API cuenta. VPC
- VPCLas políticas de puntos finales son compatibles con Timestream para. LiveAnalytics De forma predeterminada, se permite el acceso total a Timestream for a través del LiveAnalytics punto final. Para obtener más información, consulta Cómo controlar el acceso a los servicios con VPC puntos de conexión en la Guía del VPC usuario de Amazon.
- Debido a la arquitectura de Timestream, el acceso a las acciones de escritura y consulta requiere la creación de dos puntos finales de VPC interfaz, uno para cada uno. SDK Además, debe especificar un punto final de celda (solo podrá crear un punto final para la celda de Timestream a la que esté asignado). Encontrará información detallada en la sección de creación de un VPC punto final de interfaz para Timestream de esta guía. LiveAnalytics

Ahora que ya sabe cómo LiveAnalytics funciona Timestream for con los VPC puntos finales, cree un punto final de interfaz VPC para Timestream for. LiveAnalytics

Crear un punto final de interfaz para Timestream para VPC LiveAnalytics

Puede crear un VPCpunto final de interfaz para el LiveAnalytics servicio Timestream for mediante la VPC consola de Amazon o el AWS Command Line Interface ()AWS CLI. Para crear un VPC punto final para Timestream, complete los pasos específicos de Timestream que se describen a continuación.

> **Note**
>
> Antes de completar los pasos que se indican a continuación, asegúrese de comprender las consideraciones específicas de los puntos finales de Timestream. VPC

Crear un nombre de servicio de VPC punto final utilizando su celda de Timestream

Debido a la arquitectura única de Timestream, se deben crear puntos finales de VPC interfaz independientes para cada uno SDK (escritura y consulta). Además, debe especificar un punto final de celda de Timestream (solo podrá crear un punto final para la celda de Timestream a la que esté

asignado). Para usar los VPC puntos de conexión de la interfaz para conectarse directamente a Timestream desde su interior, complete los pasos que se indican a continuación: VPC

1. En primer lugar, busque un terminal celular de Timestream disponible. Para encontrar un punto final de celda disponible, usa la `DescribeEndpoints`acción (disponible mediante escritura y consultaAPIs) para enumerar los puntos finales de celda disponibles en tu cuenta de Timestream. Consulta el ejemplo para obtener más información.
2. Una vez que haya seleccionado un punto final de celda para usarlo, cree una cadena de punto final de VPC interfaz para Timestream Write o Query: API
   - Para la escritura: API

     ```
     com.amazonaws.<region>.timestream.ingest-<cell>
     ```

   - Para la consultaAPI:

     ```
     com.amazonaws.<region>.timestream.query-<cell>
     ```

   where *<region>* es un código de AWS región válido y *<cell>* es una de las direcciones de punto final de la celda (por ejemplo, `cell1` `ocell2`) devueltas por la DescribeEndpoints acción en el objeto Endpoints. Consulte el ejemplo para obtener más información.
3. Ahora que ha creado un nombre de servicio de VPC punto final, cree un punto final de interfaz. Cuando se le pida que proporcione un nombre de servicio de VPC punto final, utilice el nombre del servicio de VPC punto final que creó en el paso 2.

Ejemplo: crear el nombre de su servicio de VPC punto final

En el siguiente ejemplo, la `DescribeEndpoints` acción se ejecuta AWS CLI mediante la función Write API in the `us-west-2` region:

```
aws timestream-write describe-endpoints --region us-west-2
```

Este comando devolverá el siguiente resultado:

```
{
    "Endpoints": [
        {
            "Address": "ingest-cell1.timestream.us-west-2.amazonaws.com",
            "CachePeriodInMinutes": 1440
```

```
        }
    ]
}
```

En este caso, *cell1* es el *<cell>* , y *us-west-2* es el *<region>*. Por lo tanto, el nombre del servicio de VPC punto final resultante tendrá el siguiente aspecto:

```
com.amazonaws.us-west-2.timestream.ingest-cell1
```

Ahora que ha creado un VPC punto final de interfaz para Timestream LiveAnalytics, cree una política de VPC punto final para Timestream for. LiveAnalytics

Crear una política de VPC puntos finales para Timestream para LiveAnalytics

Puede adjuntar una política de puntos finales a su VPC punto final que controle el acceso a Timestream durante. LiveAnalytics La política especifica la siguiente información:

- La entidad principal que puede realizar acciones.
- Las acciones que se pueden realizar.
- Los recursos en los que se pueden llevar a cabo las acciones.

Para obtener más información, consulta Cómo controlar el acceso a los servicios con VPC puntos de conexión en la Guía del VPC usuario de Amazon.

Ejemplo: política de VPC puntos finales para Timestream para las acciones LiveAnalytics

El siguiente es un ejemplo de una política de punto final para Timestream for. LiveAnalytics Cuando se adjunta a un punto final, esta política otorga acceso al Timestream indicado para LiveAnalytics realizar acciones (en este caso `ListDatabases`) a todos los directores de todos los recursos.

```
{
   "Statement":[
      {
         "Principal":"*",
         "Effect":"Allow",
         "Action":[
            "timestream:ListDatabases"
         ],
         "Resource":"*"
```

```
        }
    ]
}
```

Prácticas recomendadas de seguridad para Amazon Timestream para LiveAnalytics

Amazon Timestream LiveAnalytics for proporciona una serie de características de seguridad que debe tener en cuenta a la hora de desarrollar e implementar sus propias políticas de seguridad. Las siguientes prácticas recomendadas son directrices generales y no suponen una solución de seguridad completa. Puesto que es posible que estas prácticas recomendadas no sean adecuadas o suficientes para el entorno, considérelas como consideraciones útiles en lugar de como normas.

Temas

- Cronograma para las mejores prácticas de seguridad preventiva LiveAnalytics

Cronograma para las mejores prácticas de seguridad preventiva LiveAnalytics

Las siguientes prácticas recomendadas pueden ayudarle a anticipar y prevenir los incidentes de seguridad en Timestream for. LiveAnalytics

Cifrado en reposo

Timestream for LiveAnalytics cifra en reposo todos los datos de usuario almacenados en tablas mediante claves de cifrado almacenadas en ().AWS Key Management ServiceAWS KMS Esto proporciona una capa adicional de protección de datos al proteger los datos del acceso no autorizado al almacenamiento subyacente.

Timestream for LiveAnalytics utiliza una única clave predeterminada del servicio (AWS propiaCMK) para cifrar todas las tablas. Si esta clave no existe, se crea para usted. Las claves predeterminadas de servicio no se pueden deshabilitar. Para obtener más información, consulte Timestream for LiveAnalytics Encryption at Rest.

Utilice IAM funciones para autenticar el acceso a Timestream para LiveAnalytics

Para que los usuarios, las aplicaciones y otros AWS servicios puedan acceder a Timestream LiveAnalytics, deben incluir credenciales válidas AWS en sus solicitudes. AWS API No debes almacenar AWS las credenciales directamente en la aplicación o EC2 la instancia. Estas son las credenciales a largo plazo que no rotan automáticamente, por lo tanto, podrían tener un impacto

empresarial significativo si se comprometen. Un IAM rol le permite obtener claves de acceso temporales que se pueden usar para acceder a AWS servicios y recursos.

Para obtener más información, consulte Roles de IAM.

Utilice IAM las políticas de Timestream como autorización básica LiveAnalytics

Al conceder permisos, usted decide quién los obtiene, para LiveAnalytics APIs qué Timestream los va a obtener y las acciones específicas que desea permitir en esos recursos. La implementación de privilegios mínimos es la clave a la hora de reducir los riesgos de seguridad y el impacto que podrían causar los errores o los intentos malintencionados.

Adjunta políticas de permisos a las IAM identidades (es decir, a los usuarios, grupos y roles) y, de este modo, otorga permisos para realizar operaciones en Timestream con respecto a los recursos. LiveAnalytics

Para hacerlo, utilice lo siguiente:

- AWS políticas gestionadas (predefinidas)
- Políticas administradas por el cliente
- autorización basada en etiquetas

Tenga en cuenta el cifrado del lado del cliente

Si almacena datos sensibles o confidenciales en Timestream LiveAnalytics, puede que desee cifrarlos lo más cerca posible de su origen para que estén protegidos durante todo su ciclo de vida. El cifrado de los datos confidenciales en tránsito y en reposo ayuda a garantizar que los datos en texto plano no estén disponibles para terceros.

Trabajar con otros servicios de

Amazon Timestream LiveAnalytics para la integración con una variedad AWS de servicios y herramientas populares de terceros. Actualmente, Timestream for LiveAnalytics admite integraciones con lo siguiente:

Temas

- Amazon DynamoDB
- AWS Lambda
- AWS IoT Core

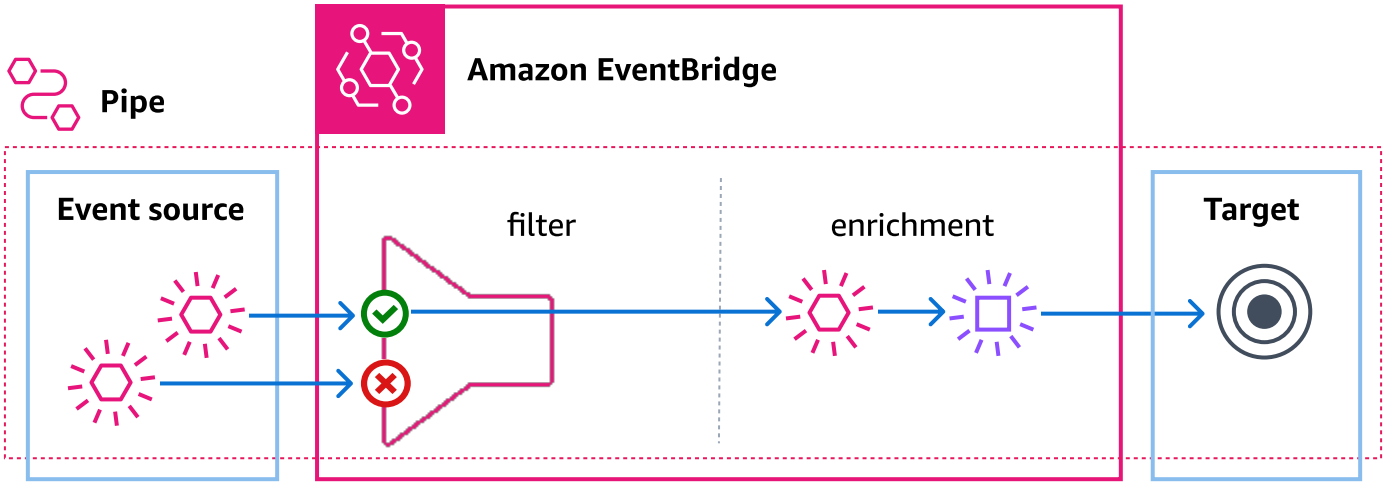
- Amazon Managed Service para Apache Flink
- Amazon Kinesis
- Amazon MQ
- Amazon MSK
- Amazon QuickSight
- Amazon SageMaker
- Amazon SQS
- Utilización DBeaver para trabajar con Amazon Timestream
- Grafana
- Utilización SquaredUp para trabajar con Amazon Timestream
- Telegraf de código abierto
- JDBC
- ODBC
- VPCpuntos finales ()AWS PrivateLink

Amazon DynamoDB

Uso de EventBridge Pipes para enviar datos de DynamoDB a Timestream

Puede usar EventBridge Pipes para enviar datos desde una transmisión de DynamoDB a una tabla de Amazon Timestream for. LiveAnalytics

Los pipes están diseñados para la point-to-point integración entre las fuentes y los destinos compatibles, y permiten realizar transformaciones y enriquecimientos avanzados. Los tubos reducen la necesidad de conocimientos especializados y códigos de integración a la hora de desarrollar arquitecturas basadas en eventos. Para configurar una canalización, elija el origen, agregue filtros opcionales, defina el enriquecimiento opcional y elija el destino de los datos del evento.



Para obtener más información sobre EventBridge Pipes, consulte EventBridge Pipes en la Guía del EventBridge usuario. Para obtener información sobre cómo configurar una canalización para entregar eventos a una tabla Amazon Timestream LiveAnalytics for, EventBridge consulte las especificaciones del objetivo de Pipes.

AWS Lambda

Puede crear funciones Lambda que interactúen con Timestream for. LiveAnalytics Por ejemplo, puede crear una función Lambda que se ejecute a intervalos regulares para ejecutar una consulta en Timestream y enviar una SNS notificación en función de los resultados de la consulta que cumplan uno o varios criterios. Para obtener más información sobre Lambda, consulte la documentación de Lambda AWS .

Temas

- Cree funciones AWS Lambda con Amazon Timestream para Python LiveAnalytics
- Cree funciones AWS Lambda con Amazon Timestream para con LiveAnalytics JavaScript
- Cree funciones AWS Lambda con Amazon Timestream for with Go LiveAnalytics
- Cree funciones AWS Lambda con Amazon Timestream para C# LiveAnalytics

Cree funciones AWS Lambda con Amazon Timestream para Python LiveAnalytics

Para crear funciones AWS Lambda con Amazon Timestream LiveAnalytics para Python, siga los pasos que se indican a continuación.

1. Cree un IAM rol para que Lambda asuma que concederá los permisos necesarios para acceder al servicio Timestream, como se describe en. Proporcione Timestream para el acceso LiveAnalytics
2. Edite la relación de confianza del IAM rol para agregar el servicio Lambda. Puede usar los siguientes comandos para actualizar un rol existente para que AWS Lambda pueda asumirlo:

   a. Cree el documento de política de confianza:

```
cat > Lambda-Role-Trust-Policy.json << EOF
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "lambda.amazonaws.com"
        ]
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF
```

   b. Actualice el rol del paso anterior con el documento de confianza

```
aws iam update-assume-role-policy --role-name <name_of_the_role_from_step_1> --
policy-document file://Lambda-Role-Trust-Policy.json
```

Las referencias relacionadas están en TimestreamWritey TimestreamQuery.

Cree funciones AWS Lambda con Amazon Timestream para con LiveAnalytics JavaScript

Para crear funciones de AWS Lambda con Amazon Timestream o JavaScript con, siga las LiveAnalytics instrucciones que se describen aquí.

Las referencias relacionadas se encuentran en Timestream Write Client (para la versión 3) y Timestream Query Client (AWS SDKpara la versión JavaScript 3). AWS SDK JavaScript

Cree funciones AWS Lambda con Amazon Timestream for with Go LiveAnalytics

Para crear funciones de AWS Lambda con Amazon Timestream LiveAnalytics for with Go, siga las instrucciones que se describen aquí.

Las referencias relacionadas se encuentran en timestreamwrite y timestreamquery.

Cree funciones AWS Lambda con Amazon Timestream para C# LiveAnalytics

Para crear funciones AWS Lambda con Amazon Timestream o con C#, siga las LiveAnalytics instrucciones que se describen aquí.

Las referencias relacionadas se encuentran en Amazon. TimestreamWritey Amazon. TimestreamQuery.

AWS IoT Core

Puede recopilar datos de dispositivos de IoT mediante AWS IoT Core y enrutar los datos a Amazon Timestream mediante acciones de reglas de IoT Core. AWS Las acciones de las reglas de IoT especifican qué hacer cuando se activa una regla. Puede definir acciones para enviar datos a una tabla de Amazon Timestream o a una base de datos de Amazon DynamoDB e invocar una función Lambda. AWS

La acción Timestream de las reglas de IoT se utiliza para insertar datos de los mensajes entrantes directamente en Timestream. La acción analiza los resultados de la SQL declaración IoT Core y almacena los datos en Timestream. Los nombres de los campos del conjunto de SQL resultados devuelto se utilizan como la medida: :nombre y el valor del campo es la medida: :valor.

Por ejemplo, considere la SQL declaración y la carga útil del mensaje de ejemplo:

```
SELECT temperature, humidity from 'iot/topic'
```

```
{
  "dataFormat": 5,
  "rssi": -88,
  "temperature": 24.04,
  "humidity": 43.605,
  "pressure": 101082,
  "accelerationX": 40,
  "accelerationY": -20,
  "accelerationZ": 1016,
  "battery": 3007,
```

```
  "txPower": 4,
  "movementCounter": 219,
  "device_id": 46216,
  "device_firmware_sku": 46216
}
```

Si se crea una acción de regla básica de IoT para Timestream con la SQL declaración anterior, se agregarán dos registros a Timestream con nombres de medida de temperatura y humedad y valores de medición de 24,04 y 43,605, respectivamente.

Puede modificar el nombre de la medida de un registro que se va a añadir a Timestream utilizando el operador AS de la sentencia. SELECT La siguiente SQL declaración creará un registro con el nombre del mensaje temp en lugar de temperature.

El tipo de datos de la medida se deduce del tipo de datos del valor de la carga útil del mensaje. JSONlos tipos de datos, como entero, doble, booleano y cadena, se asignan a los tipos de datos Timestream de,, y, respectivamente. BIGINT DOUBLE BOOLEAN VARCHAR También se pueden forzar los datos a tipos de datos específicos mediante la función cast (). Puede especificar la marca de tiempo de la medida. Si la marca de tiempo se deja en blanco, se utiliza la hora a la que se procesó la entrada.

Para obtener más información, consulte la documentación sobre las reglas y acciones de Timestream

Para crear una acción de regla de IoT Core para almacenar datos en Timestream, siga los pasos que se indican a continuación:

Temas

- Requisitos previos
- Mediante la consola
- Usando el CLI
- Aplicación de muestra
- Tutorial en vídeo

Requisitos previos

1. Cree una base de datos en Amazon Timestream siguiendo las instrucciones que se describen en. Creación de una base de datos de

2. Cree una tabla en Amazon Timestream siguiendo las instrucciones que se describen en. Creación de una tabla

Mediante la consola

1. Use la consola AWS de administración de AWS IoT Core para crear una regla. Para ello, haga clic en Administrar > Enrutamiento de mensajes > Reglas y, a continuación, en Crear regla.
2. Defina el nombre de la regla con el nombre de su elección y SQL con el texto que se muestra a continuación

```
SELECT temperature as temp, humidity from 'iot/topic'
```

3. Selecciona Timestream en la lista de acciones
4. Especifique los nombres de la base de datos, las tablas y las dimensiones de Timestream junto con la función para escribir los datos en Timestream. Si el rol no existe, puede crear uno haciendo clic en Crear roles
5. Para probar la regla, siga las instrucciones que se muestran aquí.

Usando el CLI

Si no ha instalado la interfaz de línea de AWS comandos (AWS CLI), hágalo desde aquí.

1. Guarda la siguiente carga útil de reglas en un JSON archivo llamado timestream_rule.json. Reemplazar *arn:aws:iam::123456789012:role/TimestreamRole* con tu rol arn, que otorga acceso a AWS IoT para almacenar datos en Amazon Timestream

```
{
    "actions": [
            {
                "timestream": {
                    "roleArn": "arn:aws:iam::123456789012:role/TimestreamRole",
                    "tableName": "devices_metrics",
                    "dimensions": [
                        {
                            "name": "device_id",
                            "value": "${clientId()}"
                        },
                        {
                            "name": "device_firmware_sku",
```

```
                            "value": "My Static Metadata"
                        }
                    ],
                    "databaseName": "record_devices"
                }
            }
    ],
    "sql": "select * from 'iot/topic'",
    "awsIotSqlVersion": "2016-03-23",
    "ruleDisabled": false
}
```

2. Cree una regla temática con el siguiente comando

```
aws iot create-topic-rule --rule-name timestream_test --topic-rule-payload file://
<path/to/timestream_rule.json> --region us-east-1
```

3. Recupere los detalles de la regla temática mediante el siguiente comando

```
aws iot get-topic-rule --rule-name timestream_test
```

4. Guarda la siguiente carga útil del mensaje en un archivo llamado timestream_msg.json

```
{
  "dataFormat": 5,
  "rssi": -88,
  "temperature": 24.04,
  "humidity": 43.605,
  "pressure": 101082,
  "accelerationX": 40,
  "accelerationY": -20,
  "accelerationZ": 1016,
  "battery": 3007,
  "txPower": 4,
  "movementCounter": 219,
  "device_id": 46216,
  "device_firmware_sku": 46216
}
```

5. Pruebe la regla con el siguiente comando

```
aws iot-data publish --topic 'iot/topic' --payload file://<path/to/
timestream_msg.json>
```

Aplicación de muestra

Para ayudarle a empezar a utilizar Timestream con AWS IoT Core, hemos creado una aplicación de muestra totalmente funcional que crea los artefactos necesarios en AWS IoT Core y Timestream para crear una regla temática y una aplicación de muestra para publicar datos sobre el tema.

1. Clone el GitHub repositorio de la aplicación de ejemplo para la integración de AWS IoT Core siguiendo las instrucciones de GitHub
2. Siga las instrucciones de la AWS CloudFormation plantilla READMEpara crear los artefactos necesarios en Amazon Timestream AWS e IoT Core y para publicar mensajes de muestra sobre el tema.

Tutorial en vídeo

En este vídeo se explica cómo funciona IoT Core con Timestream.

Amazon Managed Service para Apache Flink

Puede usar Apache Flink para transferir sus datos de series temporales desde Amazon Managed Service para Apache FlinkMSK, Amazon, Apache Kafka y otras tecnologías de streaming directamente a Amazon Timestream for. LiveAnalytics Hemos creado un ejemplo de conector de datos de Apache Flink para Timestream. También hemos creado una aplicación de muestra para enviar datos a Amazon Kinesis, de forma que los datos puedan pasar de Kinesis a Managed Service for Apache Flink y, finalmente, a Amazon Timestream. Todos estos artefactos están disponibles en. GitHub Este tutorial en vídeo describe la configuración.

> **Note**
>
> Java 11 es la versión recomendada para usar la aplicación Managed Service for Apache Flink. Si tiene varias versiones de Java, asegúrese de exportar Java 11 a su variable de HOME entorno JAVA _.

Temas

- Aplicación de muestra
- Tutorial en vídeo

Aplicación de muestra

Para empezar, siga el siguiente procedimiento:

1. Cree una base de datos en Timestream con el nombre `kdaflink` siguiendo las instrucciones descritas en Creación de una base de datos de
2. Cree una tabla en Timestream con el nombre `kinesisdata1` siguiendo las instrucciones descritas en Creación de una tabla
3. Cree una transmisión de datos de Amazon Kinesis con el nombre `TimestreamTestStream` siguiendo las instrucciones que se describen en Creación de una transmisión
4. Clone el GitHub repositorio del conector de datos Apache Flink para Timestream siguiendo las instrucciones de GitHub
5. Para compilar, ejecutar y usar la aplicación de ejemplo, siga las instrucciones del conector de datos de ejemplo de Apache Flink README
6. Compila el servicio gestionado para la aplicación Apache Flink siguiendo las instrucciones para compilar el código de la aplicación
7. Cargue el archivo binario de la aplicación Managed Service for Apache Flink siguiendo las instrucciones para cargar el código de streaming de Apache Flink

   a. Después de hacer clic en Crear aplicación, haga clic en el enlace del IAM rol de la aplicación

   b. Adjunte las IAM políticas de AmazonKinesisReadOnlyAccessy AmazonTimestreamFullAccess.

   > **Note**
   >
   > Las IAM políticas anteriores no se limitan a recursos específicos y no son adecuadas para su uso en producción. En el caso de un sistema de producción, considere la posibilidad de utilizar políticas que restrinjan el acceso a recursos específicos.

8. Clone el GitHub repositorio de la aplicación de ejemplo escribiendo datos en Kinesis siguiendo las instrucciones de GitHub

9. Siga las instrucciones de README para ejecutar la aplicación de ejemplo para escribir datos en Kinesis.
10. Ejecute una o más consultas en Timestream para asegurarse de que los datos se envían desde Kinesis a Managed Service para Apache Flink to Timestream siguiendo las instrucciones para Creación de una tabla

Tutorial en vídeo

En este vídeo se explica cómo utilizar Timestream con Managed Service para Apache Flink.

Amazon Kinesis

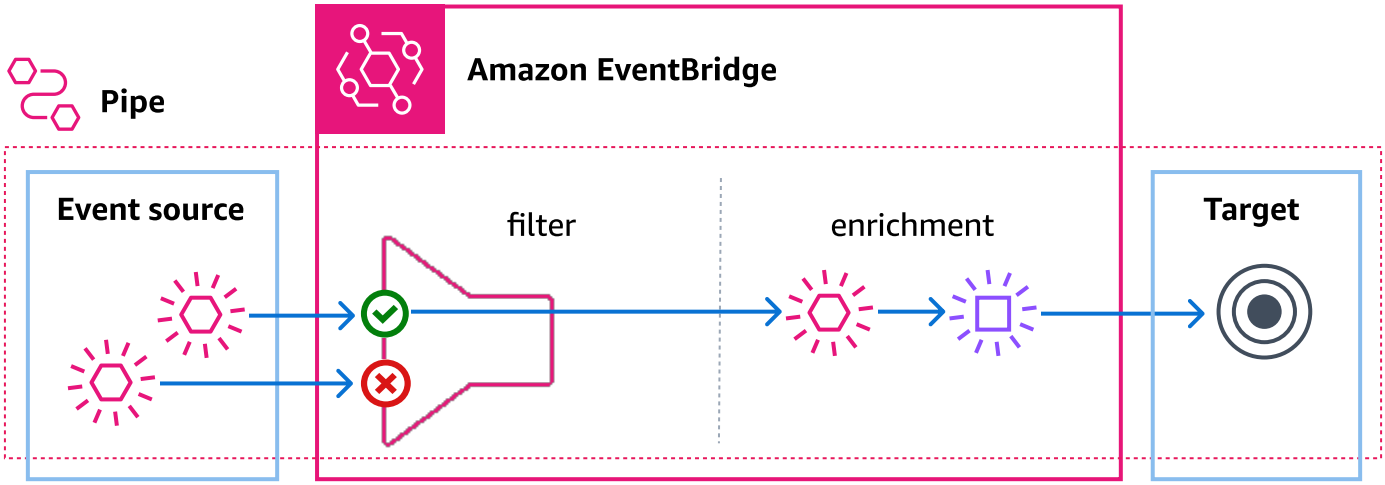
Usando Amazon Managed Service para Apache Flink

Puede enviar datos desde Kinesis Data Streams a Timestream para utilizar el conector de datos de Timestream ejemplo LiveAnalytics para Managed Service for Apache Flink. Consulte Apache Flink Amazon Managed Service para Apache Flink para obtener más información.

Uso de EventBridge Pipes para enviar datos de Kinesis a Timestream

Puede usar EventBridge Pipes para enviar datos desde una transmisión de Kinesis a una tabla de Amazon Timestream for. LiveAnalytics

Los pipes están diseñados para la point-to-point integración entre las fuentes y los destinos compatibles, y permiten realizar transformaciones y enriquecimientos avanzados. Los tubos reducen la necesidad de conocimientos especializados y códigos de integración a la hora de desarrollar arquitecturas basadas en eventos. Para configurar una canalización, elija el origen, agregue filtros opcionales, defina el enriquecimiento opcional y elija el destino de los datos del evento.



Esta integración le permite aprovechar el potencial de las capacidades de análisis de datos Timestream de series temporales y, al mismo tiempo, simplificar el proceso de ingesta de datos.

El uso de EventBridge Pipes with Timestream ofrece las siguientes ventajas:

- Ingestión de datos en tiempo real: transmita datos desde Kinesis directamente a Timestream LiveAnalytics para, de este modo, realizar análisis y supervisión en tiempo real.
- Integración perfecta: utilice EventBridge Pipes para gestionar el flujo de datos sin necesidad de complejas integraciones personalizadas.
- Filtrado y transformación mejorados: filtre o transforme los registros de Kinesis antes de almacenarlos Timestream para cumplir con sus requisitos específicos de procesamiento de datos.
- Escalabilidad: gestione flujos de datos de alto rendimiento y garantice un procesamiento de datos eficiente con funciones integradas de paralelismo y procesamiento por lotes.

Configuración

Para configurar un EventBridge Pipe para transmitir datos de Kinesis a Timestream, siga estos pasos:

1. Creación de un flujo de Kinesis

   Asegúrese de tener una transmisión de datos de Kinesis activa desde la que desee ingerir datos.
2. Cree una Timestream base de datos y una tabla

   Configure la Timestream base de datos y la tabla donde se almacenarán los datos.

3. Configure la EventBridge tubería:
   - Fuente: seleccione su transmisión de Kinesis como fuente.
   - Destino: elija Timestream como destino.
   - Configuración de procesamiento por lotes: defina la ventana de procesamiento por lotes y el tamaño del lote para optimizar el procesamiento de datos y reducir la latencia.

> **Important**
>
> Al configurar una canalización, recomendamos comprobar la exactitud de todas las configuraciones mediante la ingesta de algunos registros. Tenga en cuenta que la creación correcta de una canalización no garantiza que la canalización sea correcta y que los datos fluyan sin errores. Es posible que se produzcan errores de tiempo de ejecución, como una tabla incorrecta, un parámetro de ruta dinámica incorrecto o Timestream un registro no válido tras aplicar el mapeo, que se detectarán cuando los datos reales fluyan por la canalización.

Las siguientes configuraciones determinan la velocidad a la que se ingieren los datos:

- BatchSize: El tamaño máximo del lote que se enviará a Timestream. LiveAnalytics Rango: 0 - 100. La recomendación es mantener este valor en 100 para obtener el máximo rendimiento.
- MaximumBatchingWindowInSeconds: El tiempo máximo de espera para llenarse batchSize antes de que el lote se envíe a Timestream para su destino. LiveAnalytics En función de la velocidad de entrada de eventos, esta configuración decidirá el retraso de la ingestión. Se recomienda mantener este valor en menos de 10 segundos para seguir enviando los datos prácticamente en tiempo real. Timestream
- ParallelizationFactor: el número de lotes de cada fragmento que se van a procesar simultáneamente. La recomendación es utilizar el valor máximo de 10 para obtener el máximo rendimiento y una ingesta casi en tiempo real.

  Si varios objetivos leen tu transmisión, utiliza una función de distribución mejorada para disponer de un consumidor dedicado a tu canal y lograr un alto rendimiento. Para obtener más información, consulta Cómo desarrollar consumidores con mayor capacidad de distribución Kinesis Data Streams API en la Guía delKinesis Data Streams usuario.

> **Note**
>
> El rendimiento máximo que se puede lograr está limitado por las ejecuciones simultáneas de tuberías por cuenta.

La siguiente configuración garantiza la prevención de la pérdida de datos:

- DeadLetterConfig: Se recomienda configurarlo siempre DeadLetterConfig para evitar cualquier pérdida de datos en los casos en que los eventos no se hayan podido transferir a Timestream LiveAnalytics debido a errores del usuario.

Optimice el rendimiento de su canal con los siguientes ajustes de configuración, que ayudan a evitar que los registros se ralenticen o bloqueen.

- MaximumRecordAgeInSeconds: Los registros más antiguos que este no se procesarán y se moverán directamente a ellos. DLQ Recomendamos establecer este valor para que no sea superior al período de retención del almacén de memoria configurado en la Timestream tabla de destino.
- MaximumRetryAttempts: el número de reintentos de un registro antes de enviarlo a DeadLetterQueue. La recomendación es configurarlo en 10. Esto debería ayudar a solucionar cualquier problema transitorio y, en el caso de problemas persistentes, el registro se moverá al resto de la transmisión DeadLetterQueue y se desbloqueará.
- OnPartialBatchItemFailure: En el caso de las fuentes que admiten el procesamiento parcial por lotes, le recomendamos que lo habilite y lo configure como AUTOMATIC _ BISECT para volver a intentar los registros fallidos antes de descartarlos o enviarlos a. DLQ

Ejemplo de configuración

A continuación, se muestra un ejemplo de cómo configurar un EventBridge Pipe para transmitir datos desde una transmisión de Kinesis a una Timestream tabla:

Example IAM actualizaciones de políticas para Timestream

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

```
            "Effect": "Allow",
            "Action": [
                "timestream:WriteRecords"
            ],
            "Resource": [
                "arn:aws:timestream:us-east-1:123456789012:database/my-database/table/
my-table"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "timestream:DescribeEndpoints"
            ],
            "Resource": "*"
        }
    ]
}
```

Example Configuración de transmisiones de Kinesis

```
{
  "Source": "arn:aws:kinesis:us-east-1:123456789012:stream/my-kinesis-stream",
  "SourceParameters": {
    "KinesisStreamParameters": {
        "BatchSize": 100,
        "DeadLetterConfig": {
            "Arn": "arn:aws:sqs:us-east-1:123456789012:my-sqs-queue"
        },
       "MaximumBatchingWindowInSeconds": 5,
        "MaximumRecordAgeInSeconds": 1800,
        "MaximumRetryAttempts": 10,
        "StartingPosition": "LATEST",
       "OnPartialBatchItemFailure": "AUTOMATIC_BISECT"
    }
  }
}
```

Example Timestream configuración de destino

```
{
    "Target": "arn:aws:timestream:us-east-1:123456789012:database/my-database/table/my-
table",
```

```
    "TargetParameters": {
        "TimestreamParameters": {
            "DimensionMappings": [
                {
                    "DimensionName": "sensor_id",
                    "DimensionValue": "$.data.device_id",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "DimensionName": "sensor_type",
                    "DimensionValue": "$.data.sensor_type",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "DimensionName": "sensor_location",
                    "DimensionValue": "$.data.sensor_loc",
                    "DimensionValueType": "VARCHAR"
                }
            ],
            "MultiMeasureMappings": [
                {
                    "MultiMeasureName": "readings",
                    "MultiMeasureAttributeMappings": [
                        {
                            "MultiMeasureAttributeName": "temperature",
                            "MeasureValue": "$.data.temperature",
                            "MeasureValueType": "DOUBLE"
                        },
                        {
                            "MultiMeasureAttributeName": "humidity",
                            "MeasureValue": "$.data.humidity",
                            "MeasureValueType": "DOUBLE"
                        },
                        {
                            "MultiMeasureAttributeName": "pressure",
                            "MeasureValue": "$.data.pressure",
                            "MeasureValueType": "DOUBLE"
                        }
                    ]
                }
            ],
            "SingleMeasureMappings": [],
            "TimeFieldType": "TIMESTAMP_FORMAT",
            "TimestampFormat": "yyyy-MM-dd HH:mm:ss.SSS",
```

```
            "TimeValue": "$.data.time",
            "VersionValue": "$.approximateArrivalTimestamp"
        }
    }
}
```

Transformación de eventos

EventBridge Las tuberías le permiten transformar los datos antes de que lleguen a su destino. Timestream Puede definir reglas de transformación para modificar los Kinesis registros entrantes, como cambiar los nombres de los campos.

Suponga que su Kinesis flujo contiene datos de temperatura y humedad. Puede utilizar una EventBridge transformación para cambiar el nombre de estos campos antes de insertarlos en Timestream ellos.

Prácticas recomendadas

Almacenamiento en lotes y almacenamiento en búfer

- Configure la ventana y el tamaño del procesamiento por lotes para equilibrar la latencia de escritura y la eficiencia del procesamiento.
- Utilice una ventana de procesamiento por lotes para acumular suficientes datos antes de procesarlos, lo que reduce la sobrecarga de los lotes pequeños frecuentes.

Procesamiento paralelo

Utilice la ParallelizationFactorconfiguración para aumentar la simultaneidad, especialmente para transmisiones de alto rendimiento. Esto garantiza que se puedan procesar simultáneamente varios lotes de cada fragmento.

Transformación de datos

Aproveche las capacidades de transformación de EventBridge Pipes para filtrar y mejorar los registros antes de almacenarlos en ellos Timestream. Esto puede ayudar a alinear los datos con sus requisitos analíticos.

Seguridad

- Asegúrese de que las IAM funciones utilizadas en EventBridge Pipes tengan los permisos necesarios para leer Kinesis y escribir Timestream en ellas.

- Utilice medidas de cifrado y control de acceso para proteger los datos en tránsito y en reposo.

Fallos de depuración

- Desactivación automática de tuberías

  Las tuberías se desactivarán automáticamente en aproximadamente 2 horas si el objetivo no existe o tiene problemas con los permisos
- Limitaciones

  Las tuberías tienen la capacidad de retroceder automáticamente y volver a intentarlo hasta que se haya reducido el acelerador.
- Habilitar los registros

  Le recomendamos que habilite los registros a ERROR nivel e incluya datos de ejecución para obtener más información sobre los errores. En caso de producirse un error, estos registros contendrán request/response sent/received datos de Timestream. Esto le ayuda a comprender el error asociado y, si es necesario, a volver a procesar los registros después de corregirlo.

Monitorización

Le recomendamos que configure alarmas en los siguientes lugares para detectar cualquier problema con el flujo de datos:

- Antigüedad máxima del registro en la fuente
  - `GetRecords.IteratorAgeMilliseconds`
- Métricas de fallos en tuberías
  - `ExecutionFailed`
  - `TargetStageFailed`
- Timestream APIErrores de escritura
  - `UserErrors`

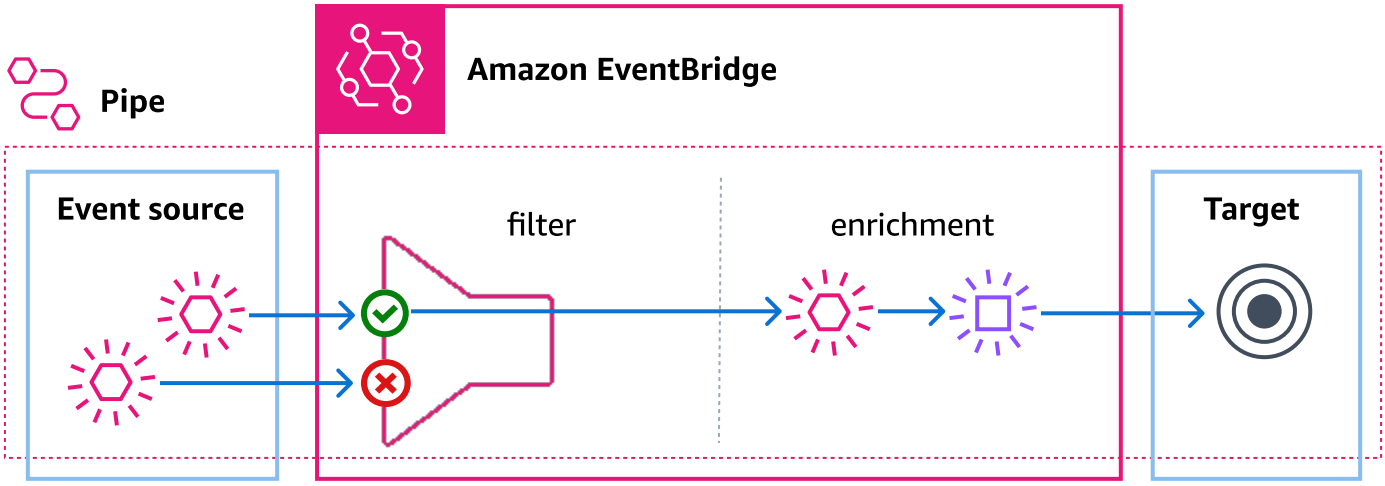
Para obtener métricas de supervisión adicionales, consulte Supervisión EventBridge en la Guía del EventBridge usuario.

Amazon MQ

Uso de EventBridge Pipes para enviar datos de Amazon MQ a Timestream

Puedes usar EventBridge Pipes para enviar datos desde un agente de Amazon MQ a una tabla de Amazon Timestream for. LiveAnalytics

Los tubos están diseñados para la point-to-point integración entre las fuentes y los destinos compatibles, y permiten realizar transformaciones y enriquecimientos avanzados. Los tubos reducen la necesidad de conocimientos especializados y códigos de integración a la hora de desarrollar arquitecturas basadas en eventos. Para configurar una canalización, elija el origen, agregue filtros opcionales, defina el enriquecimiento opcional y elija el destino de los datos del evento.



Para obtener más información sobre EventBridge Pipes, consulte EventBridge Pipes en la Guía del EventBridge usuario. Para obtener información sobre cómo configurar una canalización para entregar eventos a una tabla Amazon Timestream LiveAnalytics for, EventBridge consulte las especificaciones del objetivo de Pipes.

Amazon MSK

Uso del servicio gestionado para que Apache Flink envíe Amazon MSK datos a Timestream para LiveAnalytics

Puede enviar datos de Amazon MSK a Timestream creando un conector de datos similar al conector de Timestream datos de ejemplo para Managed Service for Apache Flink. Para obtener más información, consulte Amazon Managed Service para Apache Flink.

Uso de Kafka Connect para enviar MSK datos de Amazon a Timestream para LiveAnalytics

Puede utilizar Kafka Connect para incorporar sus datos de series temporales Amazon MSK directamente a Timestream for. LiveAnalytics

Hemos creado un ejemplo de conector Kafka Sink para. Timestream También hemos creado un ejemplo de plan de jMeter pruebas de Apache para publicar datos sobre un tema de Kafka, de modo que los datos puedan fluir desde el tema a través del conector Timestream Kafka Sink hasta un flujo temporal para una tabla. LiveAnalytics Todos estos artefactos están disponibles en. GitHub

> **Note**
>
> Java 11 es la versión recomendada para usar el conector Timestream Kafka Sink. Si tiene varias versiones de Java, asegúrese de exportar Java 11 a su variable de entorno JAVA _HOME.

Crear una aplicación de muestra

Para empezar, siga el procedimiento que se indica a continuación.

1. En Timestream for LiveAnalytics, cree una base de datos con el nombre. `kafkastream`

   Consulte el procedimiento ??? para obtener instrucciones detalladas.
2. En Timestream for LiveAnalytics, cree una tabla con el nombre. `purchase_history`

   Consulte el procedimiento ??? para obtener instrucciones detalladas.
3. Siga las instrucciones incluidas en el para crear lo siguiente:, y.

   - Un Amazon MSK clúster
   - Una Amazon EC2 instancia que está configurada como una máquina cliente de Kafka Producer
   - Un tema de Kafka

   Consulte los requisitos previos del proyecto kafka_ingestor para obtener instrucciones detalladas.
4. Clona el repositorio de Kafka Sink Connector.Timestream

Consulte Clonar un repositorio en él GitHub para obtener instrucciones detalladas.

5. Compila el código del plugin.

   Consulte Connector: compilar desde el código fuente en GitHub adelante para obtener instrucciones detalladas.

6. Cargue los siguientes archivos en un bucket de S3: siga las instrucciones que se describen en.

   - El archivo jar (kafka-connector-timestream-> VERSION <- jar-with-dependencies .jar) del directorio `/target`
   - El archivo de esquema json de ejemplo,`purchase_history.json`.

   Consulte Carga de objetos en la Guía del Amazon S3 usuario para obtener instrucciones detalladas.

7. Cree dos puntos VPC finales. El MSK Conector utilizaría estos puntos finales para acceder a los recursos que utiliza. AWS PrivateLink

   - Uno para acceder al depósito Amazon S3
   - Uno para acceder a la tabla Timestream for LiveAnalytics .

   Consulte VPCEndpoints para obtener instrucciones detalladas.

8. Crea un complemento personalizado con el archivo jar cargado.

   Consulte los complementos en la guía para Amazon MSK desarrolladores para obtener instrucciones detalladas.

9. Cree una configuración de trabajo personalizada con el JSON contenido descrito en los parámetros de configuración de trabajadores, siguiendo las instrucciones que se describen en

   Consulte Crear una configuración de trabajo personalizada en la Guía para Amazon MSK desarrolladores para obtener instrucciones detalladas.

10. Cree un IAM rol de ejecución de servicios.

    Consulte Rol IAM de servicio para obtener instrucciones detalladas.

11. Cree un Amazon MSK conector con el complemento personalizado, la configuración de trabajador personalizada y la IAM función de ejecución del servicio creados en los pasos anteriores y con el ejemplo de configuración del conector.

Consulte Crear un conector en la guía para Amazon MSK desarrolladores para obtener instrucciones detalladas.

Asegúrese de actualizar los valores de los siguientes parámetros de configuración con los valores correspondientes. Consulte los parámetros de configuración del conector para obtener más información.

- `aws.region`
- `timestream.schema.s3.bucket.name`
- `timestream.ingestion.endpoint`

La creación del conector tarda entre 5 y 10 minutos en completarse. La canalización estará lista cuando su estado cambie a. `Running`

12. Publique un flujo continuo de mensajes para escribir datos sobre el tema de Kafka creado.

    Consulte Cómo usarlo para obtener instrucciones detalladas.

13. Ejecute una o más consultas para asegurarse de que los datos se envían desde la Amazon MSK tabla MSK Connect to the Timestream for LiveAnalytics .

    Consulte el procedimiento ??? para obtener instrucciones detalladas.

Recursos adicionales de

El blog, Ingestión de datos sin servidor en tiempo real desde sus clústeres de Kafka a Timestream para usar LiveAnalytics Kafka Connect, explica la configuración de una end-to-end canalización con el conector Timestream for LiveAnalytics Kafka Sink, empezando por una máquina cliente de Kafka que utiliza el plan de jMeter pruebas de Apache para publicar miles de mensajes de muestra en un tema de Kafka hasta verificar los registros ingeridos en una tabla Timestream for. LiveAnalytics

Amazon QuickSight

Puede usar Amazon QuickSight para analizar y publicar paneles de datos que contengan sus datos de Amazon Timestream. En esta sección, se describe cómo crear una nueva conexión a una fuente de QuickSight datos, modificar los permisos, crear nuevos conjuntos de datos y realizar un análisis. Este tutorial en vídeo describe cómo trabajar con Timestream y Amazon. QuickSight

> **Note**
>
> Todos los conjuntos de datos de Amazon QuickSight son de solo lectura. No puedes realizar ningún cambio en tus datos reales de Timestream si utilizas Amazon QuickSight para eliminar la fuente de datos, el conjunto de datos o los campos.

Temas

Acceso a Amazon Timestream desde QuickSight

Para poder continuar, Amazon QuickSight necesita estar autorizado para conectarse a Amazon Timestream. Si las conexiones no están habilitadas, recibirá un mensaje de error cuando intente conectarse. Un QuickSight administrador puede autorizar las conexiones a AWS los recursos. Para autorizar una conexión desde QuickSight Timestream, siga el procedimiento descrito en Using Other AWS Services: Scoping Down Access y seleccione Amazon Timestream en el paso 5.

Cree una nueva conexión de fuente de datos para Timestream QuickSight

> **Note**
>
> La conexión entre Amazon QuickSight y Amazon Timestream se cifra en tránsito SSL mediante TLS (1.2). No puede crear una conexión sin cifrar.

1. Asegúrese de haber configurado los permisos adecuados para que Amazon acceda QuickSight a Amazon Timestream, tal y como se describe en. Acceso a Amazon Timestream desde QuickSight
2. Comience por crear un nuevo conjunto de datos. Elija Conjuntos de datos en el panel de navegación y, a continuación, elija Nuevo conjunto de datos.

3. Seleccione la tarjeta fuente de datos de Timestream.
4. En Nombre de fuente de datos, introduzca un nombre para su conexión a la fuente de datos de Timestream, por ejemplo. `US Timestream Data`

   > **Note**
   >
   > Como puede crear muchos conjuntos de datos a partir de una conexión a Timestream, es mejor que el nombre sea sencillo.

5. Seleccione Validar conexión para comprobar que puede conectarse correctamente a Timestream.

   > **Note**
   >
   > Validar la conexión solo valida la posibilidad de conectarse. Sin embargo, no valida una tabla o consulta específica.

6. Elija Crear origen de datos para continuar.
7. En Base de datos, elija Seleccionar... para ver la lista de opciones disponibles. Elige la que quieras usar.
8. Selecciona Seleccionar para continuar.
9. Seleccione una de las siguientes opciones:

   - Para importar los datos al QuickSight motor en memoria (denominadoSPICE), selecciona Importar a SPICE para obtener un análisis más rápido.
   - QuickSight Para poder ejecutar una consulta con tus datos cada vez que actualices el conjunto de datos o utilices el análisis o el panel, selecciona Consultar directamente tus datos.

10. Seleccione Editar/obtener vista y, a continuación, Guardar para guardar el conjunto de datos y cerrarlo.

Edita los permisos de la conexión a la fuente de QuickSight datos de Timestream

El siguiente procedimiento describe cómo ver, añadir y revocar los permisos de otros QuickSight usuarios para que puedan acceder a la misma fuente de datos de Timestream. Las personas deben ser usuarios activos QuickSight antes de poder agregarlas.

> **Note**
>
> En QuickSight, las fuentes de datos tienen dos niveles de permisos: usuario y propietario.
>
> - Elija el usuario para permitir el acceso de lectura.
> - Elija el propietario para permitir que el usuario edite, comparta o elimine esta fuente QuickSight de datos.

1. Asegúrese de haber configurado los permisos adecuados para que Amazon acceda QuickSight a Amazon Timestream, tal y como se describe en. Acceso a Amazon Timestream desde QuickSight
2. Elija Conjuntos de datos a la izquierda y, a continuación, desplácese hacia abajo para encontrar la tarjeta de origen de datos para su conexión a Timestream. Por ejemplo, `US Timestream Data`.
3. Elija la tarjeta de origen `Timestream` de datos.
4. Elija `Share data source`. Aparece una lista de los permisos actuales.
5. (Opcional) Para editar los permisos, puede elegir `user` o`owner`.
6. (Opcional) Para revocar los permisos, elija`Revoke access`. Las personas a las que revoques no pueden crear nuevos conjuntos de datos a partir de esta fuente de datos. Sin embargo, sus conjuntos de datos existentes seguirán teniendo acceso a esta fuente de datos.
7. Para agregar permisos, elija y`Invite users`, a continuación, siga estos pasos para agregar un usuario:
   a. Agregue personas para que puedan usar la misma fuente de datos.
   b. Para cada una de ellas, elija la `Permission` que desee aplicar.
8. Cuando haya terminado, seleccione `Close`.

Cree un nuevo QuickSight conjunto de datos para Timestream

1. Asegúrese de haber configurado los permisos adecuados para que Amazon acceda QuickSight a Amazon Timestream, tal y como se describe en. Acceso a Amazon Timestream desde QuickSight
2. Elija Conjuntos de datos a la izquierda y, a continuación, desplácese hacia abajo para encontrar la tarjeta de origen de datos para su conexión a Timestream. Si tiene muchas fuentes de datos,

puede utilizar la barra de búsqueda situada en la parte superior de la página para encontrarlas con una coincidencia parcial con el nombre.

3. Elija la tarjeta de fuente de datos de Timestream. A continuación, selecciona Crear conjunto de datos.
4. En Base de datos, elija Seleccionar para ver la lista de opciones disponibles. Elige la base de datos que quieres usar.
5. En Tablas, elija la tabla que desea utilizar.
6. Elija Editar/obtener vista previa.
7. (Opcional) Para añadir más datos, selecciona Añadir datos en la parte superior derecha.
   a. Selecciona Cambiar fuente de datos y elige una fuente de datos diferente.
   b. Siga las instrucciones de la interfaz de usuario para terminar de añadir datos.
   c. Tras añadir nuevos datos al mismo conjunto de datos, seleccione Configurar esta unión (los dos puntos rojos). Configure una unión para cada tabla adicional.
   d. Si desea añadir campos calculados, seleccione Añadir campo calculado.
   e. Para usar Sagemaker, elija Aumentar con. SageMaker Esta opción solo está disponible en la edición QuickSight Enterprise.
   f. Desactive los campos que desee omitir.
   g. Actualice los tipos de datos que desee cambiar.
8. Cuando haya terminado, elija Guardar para guardar y cerrar el conjunto de datos.

Cree un nuevo análisis para Timestream

1. Asegúrese de haber configurado los permisos adecuados para que Amazon acceda QuickSight a Amazon Timestream, tal y como se describe en. Acceso a Amazon Timestream desde QuickSight
2. Seleccione Análisis a la izquierda.
3. Seleccione una de las siguientes opciones:
   - Para crear un análisis nuevo, elija Nuevo análisis en la derecha.
   - Para añadir el conjunto de datos de Timestream a un análisis existente, abra el análisis que desee editar. Elija el icono del lápiz situado en la parte superior izquierda y, a continuación, seleccione Añadir conjunto de datos.
4. Inicie la primera visualización de datos seleccionando los campos de la izquierda.

5. Para obtener más información, consulte Trabajar con análisis - Amazon QuickSight

Tutorial en vídeo

En este vídeo se explica cómo Amazon QuickSight trabaja con Timestream.

Amazon SageMaker

Puede utilizar Amazon SageMaker Notebooks para integrar sus modelos de aprendizaje automático con Amazon Timestream. Para ayudarle a empezar, hemos creado un ejemplo de SageMaker bloc de notas que procesa los datos de Timestream. Los datos se insertan en Timestream desde una aplicación Python multihilo que envía datos de forma continua. El código fuente del ejemplo de SageMaker Notebook y de la aplicación Python de ejemplo están disponibles en GitHub.

1. Cree una base de datos y una tabla siguiendo las instrucciones descritas en Creación de una base de datos de y Creación de una tabla
2. Clona el GitHub repositorio de la aplicación de ejemplo de Python con varios subprocesos siguiendo las instrucciones de GitHub
3. Clone el GitHub repositorio del ejemplo de Timestream SageMaker Notebook siguiendo las instrucciones de. GitHub
4. Ejecute la aplicación para incorporar datos de forma continua a Timestream siguiendo las instrucciones del README
5. Siga las instrucciones para crear un bucket de Amazon S3 para Amazon SageMaker tal y como se describe aquí.
6. Crea una SageMaker instancia de Amazon con la última versión de boto3 instalada: además de las instrucciones que se describen aquí, sigue los pasos que se indican a continuación:
   a. En la página Crear una instancia de bloc de notas, haz clic en Configuración adicional
   b. Haga clic en Configuración del ciclo de vida (opcional) y seleccione Crear una nueva configuración del ciclo de vida
   c. En el cuadro Crear el asistente de configuración del ciclo de vida, haga lo siguiente:
      i. Introduzca el nombre que desee en la configuración, p. ej. `on-start`
      ii. En el script Start Notebook, copia y pega el contenido del script de Github
      iii. `PACKAGE=scipy`Sustitúyalo por `PACKAGE=boto3` en el script pegado.
7. Haga clic en Crear configuración

8. Vaya al IAM servicio en la consola AWS de administración y busque el rol de SageMaker ejecución recién creado para la instancia del portátil.
9. Adjunte la IAM política `AmazonTimestreamFullAccess` para la función de ejecución.

> **Note**
>
> La `AmazonTimestreamFullAccess` IAM política no se limita a recursos específicos y no es adecuada para su uso en producción. En el caso de un sistema de producción, considere la posibilidad de utilizar políticas que restrinjan el acceso a recursos específicos.

10. Cuando el estado de la instancia de bloc de notas sea InService, selecciona Abrir Jupyter para lanzar un SageMaker bloc de notas para la instancia
11. Cargue los archivos **`timestreamquery.py`** y colóquelos **`Timestream_SageMaker_Demo.ipynb`** en el bloc de notas pulsando el botón Cargar
12. Haga clic en `Timestream_SageMaker_Demo.ipynb`.

> **Note**
>
> Si aparece una ventana emergente que indica que no se ha encontrado el núcleo, seleccione conda_python3 y haga clic en Establecer núcleo.

13. Modifique`DB_NAME`, `TABLE_NAME`bucket, y `ENDPOINT` para que coincidan el nombre de la base de datos, el nombre de la tabla, el nombre del bucket de S3 y la región de los modelos de entrenamiento.
14. Seleccione el icono de reproducción para ejecutar las celdas individuales
15. Cuando llegue a la celda`Leverage Timestream to find hosts with average CPU utilization across the fleet`, asegúrese de que la salida devuelva al menos 2 nombres de host.

> **Note**
>
> Si hay menos de 2 nombres de host en la salida, es posible que deba volver a ejecutar la aplicación Python de ejemplo que ingiere datos en Timestream con un número mayor de subprocesos y a escala de host.

16. Cuando llegues a la celda`Train a Random Cut Forest (RCF) model using the CPU utilization history`, cámbiala en `train_instance_type` función de los recursos que necesites para tu trabajo de formación
17. Cuando llegue a la celda`Deploy the model for inference`, cámbiela en `instance_type` función de las necesidades de recursos para su trabajo de inferencia

> **Note**
>
> Entrenar el modelo puede tardar unos minutos. Cuando se complete el entrenamiento, verá el mensaje Completado: Trabajo de entrenamiento completado en la salida de la celda.

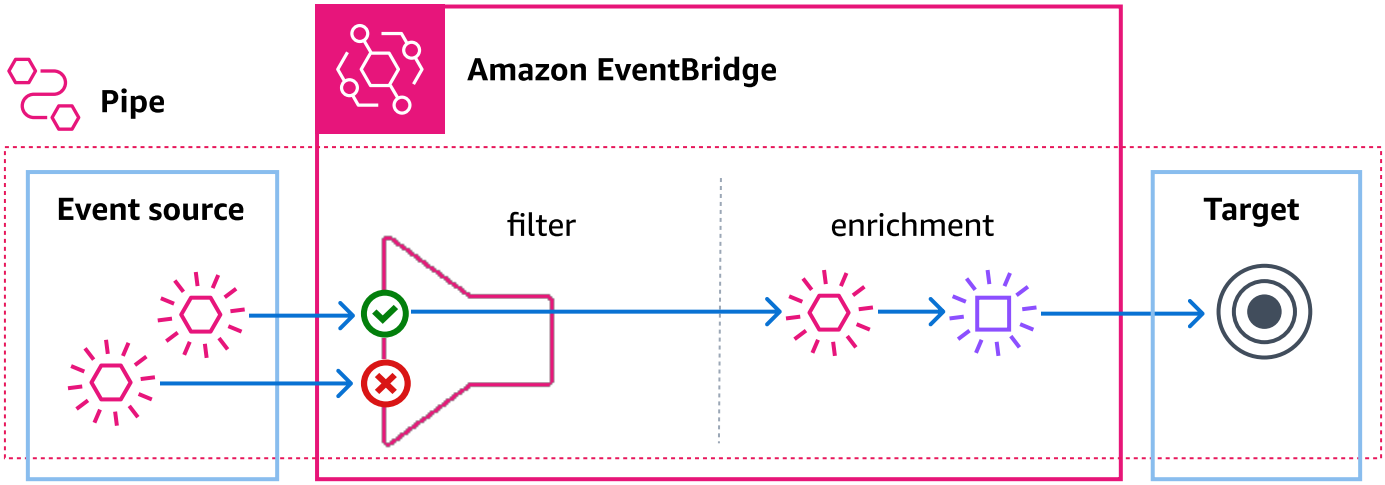
18. Ejecuta la celda `Stop and delete the endpoint` para limpiar los recursos. También puedes detener y eliminar la instancia de la SageMaker consola

Amazon SQS

Uso de EventBridge Pipes para enviar SQS datos de Amazon a Timestream

Puedes usar EventBridge Pipes para enviar datos desde una SQS cola de Amazon a una tabla de Amazon LiveAnalytics Timestream for.

Los tubos están diseñados para la point-to-point integración entre las fuentes y los destinos compatibles, y permiten realizar transformaciones y enriquecimientos avanzados. Los tubos reducen la necesidad de conocimientos especializados y códigos de integración a la hora de desarrollar arquitecturas basadas en eventos. Para configurar una canalización, elija el origen, agregue filtros opcionales, defina el enriquecimiento opcional y elija el destino de los datos del evento.



Para obtener más información sobre EventBridge Pipes, consulte EventBridge Pipes en la Guía del EventBridge usuario. Para obtener información sobre cómo configurar una canalización para entregar eventos a una tabla Amazon Timestream LiveAnalytics for, EventBridge consulte las especificaciones del objetivo de Pipes.

Utilización DBeaver para trabajar con Amazon Timestream

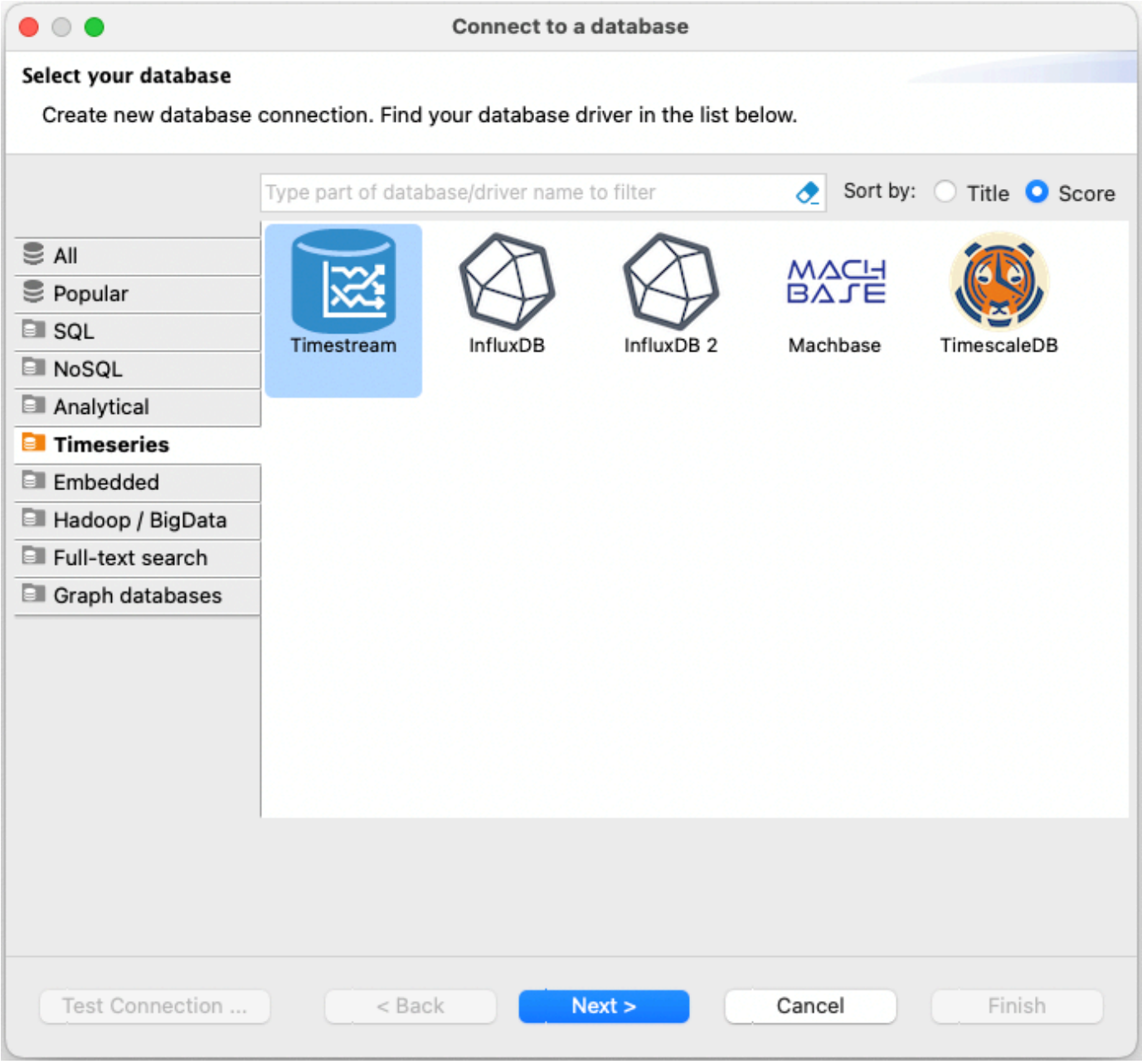
DBeaveres un SQL cliente universal gratuito que se puede utilizar para administrar cualquier base de datos que tenga un JDBC controlador. Se usa ampliamente entre los desarrolladores y administradores de bases de datos debido a sus sólidas capacidades de visualización, edición y administración de datos.

Con las opciones DBeaver de conectividad en la nube, puede conectarse DBeaver a Amazon Timestream de forma nativa. DBeaverproporciona una interfaz completa e intuitiva para trabajar con datos de series temporales directamente desde una aplicación. DBeaver Al usar sus credenciales, también le brinda acceso completo a cualquier consulta que pueda ejecutar desde otra interfaz de consulta. Incluso te permite crear gráficos para una mejor comprensión y visualización de los resultados de las consultas.

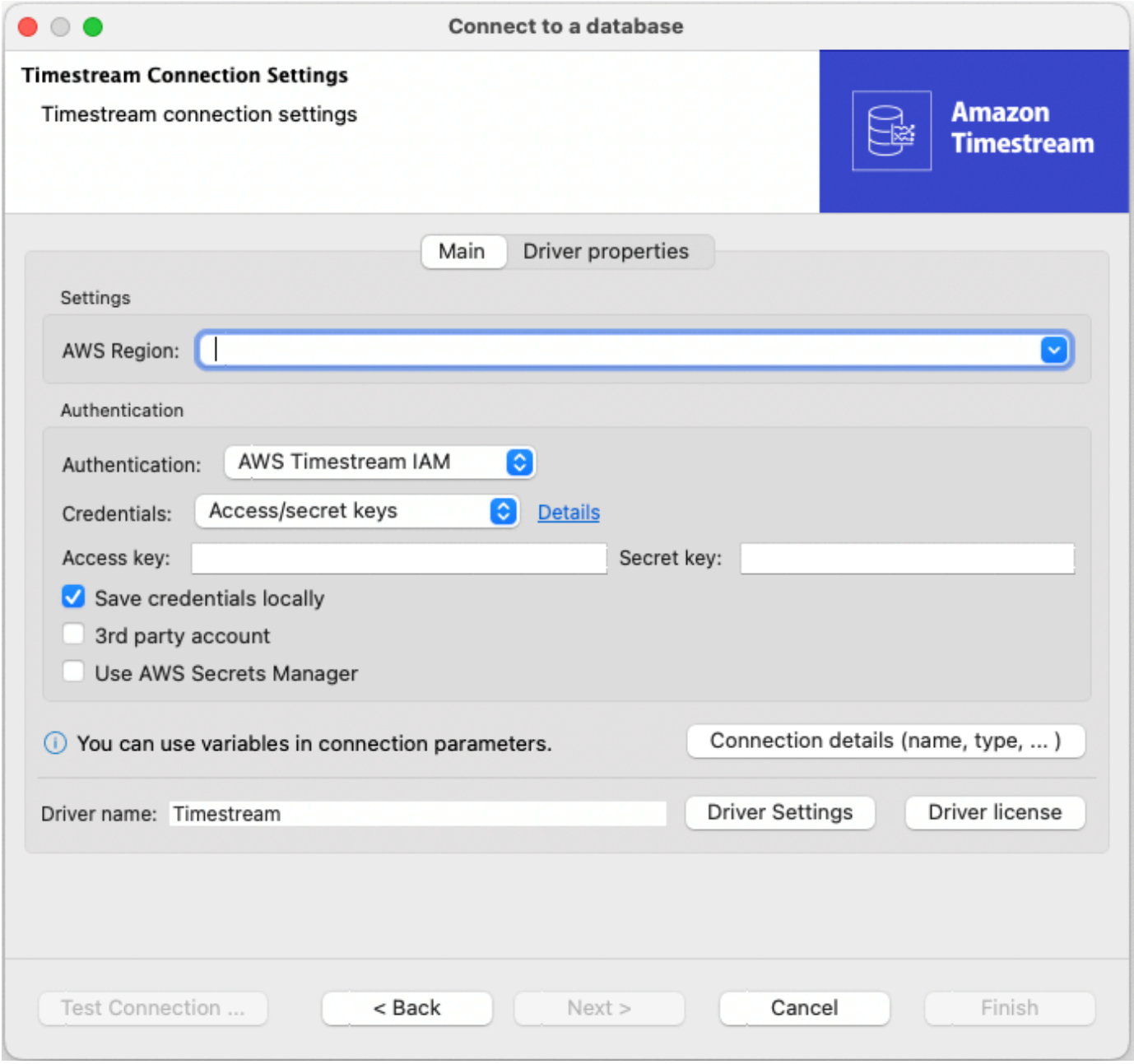
Preparándose DBeaver para trabajar con Timestream

Realice los siguientes pasos para configurar el funcionamiento DBeaver con Timestream:

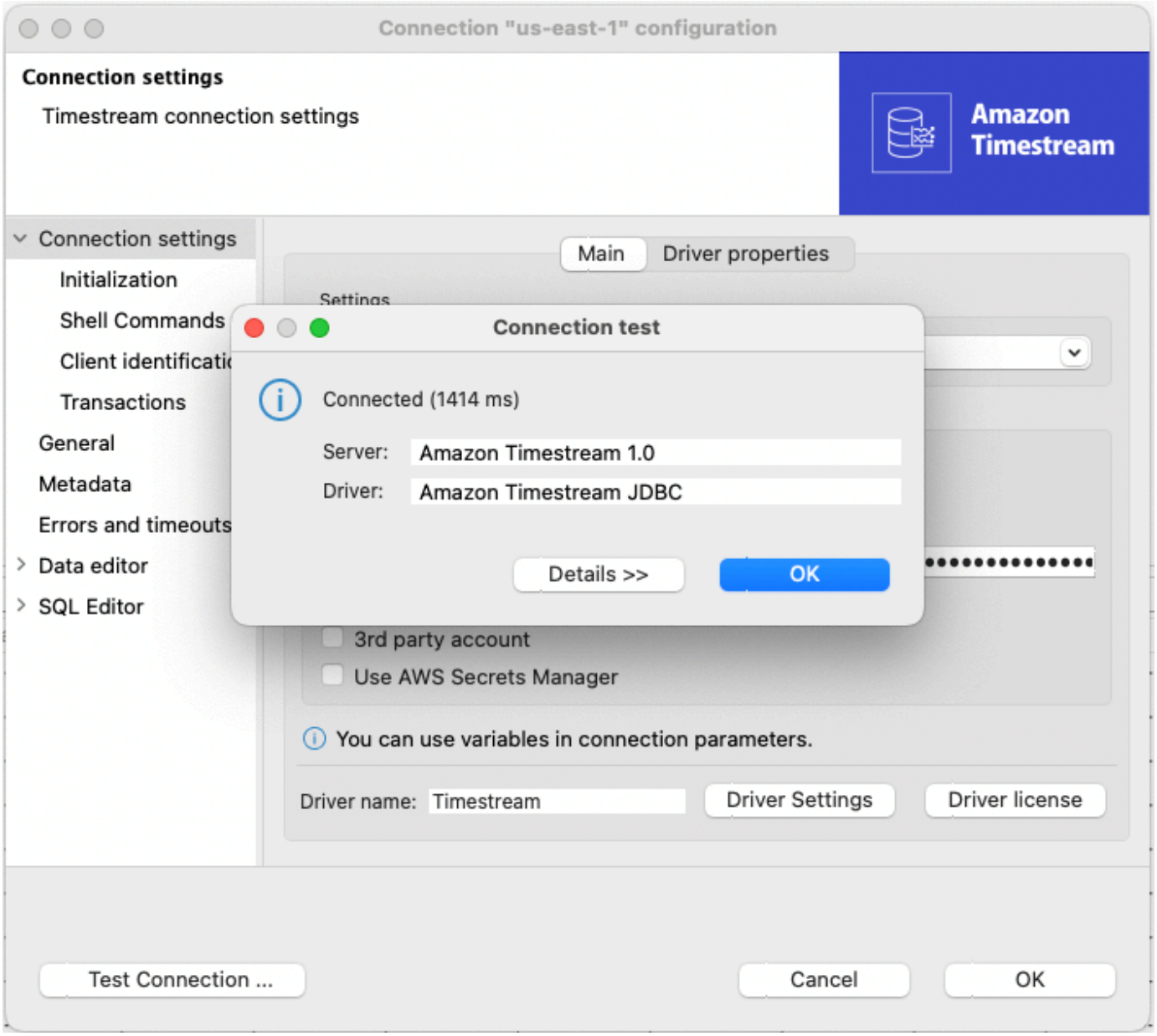
1. Descárguelo e instálelo DBeaver en su máquina local.
2. InicieDBeaver, navegue hasta el área de selección de la base de datos, elija Timeseries en el panel izquierdo y, a continuación, seleccione el icono Timestream en el panel derecho:



3. En la ventana Configuración de conexión de Timestream, introduzca toda la información necesaria para conectarse a la base de datos de Amazon Timestream. Asegúrese de que las claves de usuario que introduzca tengan los permisos necesarios para acceder a su base de datos de Timestream. Además, asegúrate de mantener la información y las claves que introduzcas de forma DBeaver segura y privada, como ocurre con cualquier información confidencial.



4. Pruebe la conexión para asegurarse de que todo está configurado correctamente:



5. Si la prueba de conexión se realiza correctamente, ahora puede interactuar con su base de datos de Amazon Timestream del mismo modo que lo haría con cualquier otra base de datos. DBeaver Por ejemplo, puede ir al SQL editor o a la vista de diagrama ER para ejecutar las consultas:

6. DBeavertambién proporciona potentes herramientas de visualización de datos. Para utilizarlos, ejecute la consulta y, a continuación, seleccione el icono del gráfico para visualizar el conjunto de resultados. La herramienta de representación gráfica puede ayudarle a comprender mejor las tendencias de los datos a lo largo del tiempo.

Al combinar Amazon Timestream DBeaver con, se crea un entorno eficaz para gestionar los datos de series temporales. Puede integrarlo sin problemas en su flujo de trabajo actual para mejorar la productividad y la eficiencia.

Grafana

Puede visualizar los datos de sus series temporales y crear alertas con Grafana. Para ayudarlo a comenzar con la visualización de datos, hemos creado un panel de ejemplo en Grafana que visualiza los datos enviados a Timestream desde una aplicación de Python y un tutorial en vídeo que describe la configuración.

Temas

- Aplicación de muestra
- Tutorial en vídeo

Aplicación de muestra

1. Cree una base de datos y una tabla en Timestream siguiendo las instrucciones que se describen en para obtener más información. Creación de una base de datos de

   > **Note**
   >
   > El nombre de la base de datos y el nombre de la tabla predeterminados para el panel de Grafana se establecen en GrafanaDB y, respectivamente. grafanaTable Utilice estos nombres para minimizar la configuración.

2. Instale Python 3.7 o superior
3. Instalar y configurar Timestream Python SDK
4. Clona el GitHub repositorio de la aplicación Python de subprocesos múltiples que ingiere datos de forma continua en Timestream siguiendo las instrucciones de GitHub
5. Ejecute la aplicación para incorporar datos de forma continua a Timestream siguiendo las instrucciones de la README
6. Complete Comenzar con Amazon Managed Grafana o complete Instalar Grafana.
7. Si vas a instalar Grafana en lugar de utilizar Amazon Managed Grafana, completa Instalar el complemento Timestream para Grafana.

8. Abre el panel de control de Grafana con el navegador que prefieras. Si has instalado Grafana localmente, puedes seguir las instrucciones que se describen en la documentación de Grafana para iniciar sesión
9. Después de iniciar Grafana, vaya a Fuentes de datos, haga clic en Agregar fuente de datos, busque Timestream y seleccione la fuente de datos Timestream
10. Configure el proveedor de autenticación y la región y haga clic en Guardar y probar
11. Configure las macros predeterminadas
    a. Establezca $__database con el nombre de su base de datos de Timestream (por ejemplo, GrafanaDB)
    b. Establezca $__table con el nombre de su tabla de Timestream (por ejemplo) grafanaTable
    c. Establece $__measure como la medida más utilizada de la tabla
12. Haz clic en Guardar y probar
13. Haga clic en la pestaña Paneles
14. Haga clic en Importar para importar el panel
15. Haga doble clic en el panel de control de la aplicación de muestra
16. Haga clic en la configuración del panel
17. Seleccione variables
18. Cambie dbName y haga coincidir tableName los nombres de la base de datos y la tabla de Timestream
19. Haga clic en Guardar
20. Actualice el panel
21. Para crear alertas, sigue las instrucciones descritas en la documentación de Grafana para crear una regla de alertas gestionadas de Grafana
22. Para solucionar problemas de alertas, siga las instrucciones descritas en la documentación de Grafana para la solución de problemas
23. Para obtener información adicional, consulte la documentación de Grafana

Tutorial en vídeo

En este vídeo se explica cómo funciona Grafana con Timestream.

Utilización SquaredUp para trabajar con Amazon Timestream

SquaredUpes una plataforma de observabilidad que se integra con Amazon Timestream. Puede usar el intuitivo diseñador SquaredUp de paneles de control para visualizar, analizar y monitorear sus datos de series temporales. Los paneles se pueden compartir de forma pública o privada, y se pueden crear canales de notificación para avisarle cuando cambie el estado de salud de un monitor.

Uso SquaredUp con Amazon Timestream

1. Regístrese SquaredUpy comience de forma gratuita.
2. Añada una fuente de AWS datos.
3. Cree un mosaico de panel que utilice el flujo de datos de Timestream Query.
4. Si lo desea, habilite la supervisión del mosaico, cree un canal de notificaciones o comparta el panel de forma pública o privada.
5. Si lo desea, cree otros mosaicos para ver sus datos de Timestream junto con los datos de sus otras herramientas de monitoreo y observabilidad.

Telegraf de código abierto

Puedes usar el complemento Timestream como LiveAnalytics salida de Telegraf para escribir métricas en Timestream directamente desde Telegraf de código abierto. LiveAnalytics

En esta sección se explica cómo instalar Telegraf con el complemento Timestream para la salida, cómo ejecutar Telegraf con el complemento Timestream para la LiveAnalytics salida y cómo funciona Telegraf de código abierto con Timestream for. LiveAnalytics LiveAnalytics

Temas

- Instalación de LiveAnalytics Telegraf con el complemento Timestream como complemento de salida
- Ejecutar Telegraf con el complemento Timestream como salida LiveAnalytics
- Mapeo de las métricas de Telegraf/InfluxDB con el Timestream para el modelo LiveAnalytics

Instalación de LiveAnalytics Telegraf con el complemento Timestream como complemento de salida

A partir de la versión 1.16, el complemento Timestream para la LiveAnalytics salida está disponible en la versión oficial de Telegraf. Para instalar el complemento de salida en la mayoría de los principales sistemas operativos, siga los pasos descritos en la documentación de Telegraf. InfluxData Para realizar la instalación en el sistema operativo Amazon Linux 2, siga las instrucciones que aparecen a continuación.

Instalación de Telegraf con el LiveAnalytics complemento Timestream for output en Amazon Linux 2

Para instalar Telegraf con el complemento Timestream Output en Amazon Linux 2, lleve a cabo los siguientes pasos.

1. Instale Telegraf mediante el administrador de paquetes. `yum`

```
cat <<EOF | sudo tee /etc/yum.repos.d/influxdb.repo
[influxdb]
name = InfluxDB Repository - RHEL \$releasever
baseurl = https://repos.influxdata.com/rhel/\$releasever/\$basearch/stable
enabled = 1
gpgcheck = 1
gpgkey = https://repos.influxdata.com/influxdb.key
EOF
```

2. Ejecute el siguiente comando de la .

```
sudo sed -i "s/\$releasever/$(rpm -E %{rhel})/g" /etc/yum.repos.d/influxdb.repo
```

3. Instale e inicie Telegraf.

```
sudo yum install telegraf
sudo service telegraf start
```

Ejecutar Telegraf con el complemento Timestream como salida LiveAnalytics

Puede seguir las instrucciones que aparecen a continuación para ejecutar Telegraf con el complemento Timestream for. LiveAnalytics

1. Genera un ejemplo de configuración con Telegraf.

```
telegraf --section-filter agent:inputs:outputs --input-filter cpu:mem --output-
filter timestream config > example.config
```

2. Cree una base de datos en Timestream mediante la consola de administración, o. CLISDKs
3. En el `example.config` archivo, añada el nombre de la base de datos editando la siguiente clave en la `[[outputs.timestream]]` sección.

```
database_name = "yourDatabaseNameHere"
```

4. De forma predeterminada, Telegraf creará una tabla. Si desea crear una tabla manualmente, `create_table_if_not_exists` configúrela `false` y siga las instrucciones para crear una tabla mediante la consola de administración CLI, o. SDKs
5. En el archivo example.config, configure las credenciales en la `[[outputs.timestream]]` sección. Las credenciales deben permitir las siguientes operaciones.

```
timestream:DescribeEndpoints
timestream:WriteRecords
```

> **Note**
>
> Si deja esta `create_table_if_not_exists` configuración en`true`, incluya:
>
> ```
> timestream:CreateTable
> ```

> **Note**
>
> Si lo has `describe_database_on_start` configurado`true`, incluye lo siguiente.
>
> ```
> timestream:DescribeDatabase
> ```

6. Puede editar el resto de la configuración según sus preferencias.
7. Cuando haya terminado de editar el archivo de configuración, ejecute Telegraf con lo siguiente.

```
./telegraf --config example.config
```

8. Las métricas deberían aparecer en unos segundos, según la configuración del agente. También debería ver las nuevas tablas, cpu y mem, en la consola de Timestream.

Mapeo de las métricas de Telegraf/InfluxDB con el Timestream para el modelo LiveAnalytics

Al escribir datos de Telegraf en Timestream for LiveAnalytics, los datos se mapean de la siguiente manera.

- La marca de tiempo se escribe como campo de tiempo.
- Las etiquetas se escriben como dimensiones.
- Los campos se escriben como medidas.
- Las medidas se escriben principalmente como nombres de tablas (más información a continuación).

El plugin Timestream for LiveAnalytics output para Telegraf ofrece múltiples opciones para organizar y almacenar datos en Timestream for. LiveAnalytics Esto se puede describir con un ejemplo que comienza con los datos en formato de protocolo de línea.

```
weather,location=us-midwest,season=summer temperature=82,humidity=71
1465839830100400200 airquality,location=us-west no2=5,pm25=16
1465839830100400200
```

A continuación se describen los datos.

- Los nombres de las mediciones son `weather` y`airquality`.
- Las etiquetas son `location` y`season`.
- Los campos son `temperature`y`humidity`,`no2`, y`pm25`.

Temas

- Almacenar los datos en varias tablas
- Almacenar los datos en una sola tabla

Almacenar los datos en varias tablas

Puede optar por crear una tabla independiente por medición y almacenar cada campo en una fila independiente por tabla.

La configuración es`mapping_mode = "multi-table"`.

- El Timestream para el LiveAnalytics adaptador creará dos tablas, a saber, y`weather`. `airquality`
- Cada fila de la tabla contendrá un solo campo.

El flujo temporal resultante para LiveAnalytics las tablas `weather` y`airquality`, tendrá este aspecto.

`weather`

hora	location	temporada	measure_name	measure_value::bigint
2016-06-13 17:43:50	Estados Unidos-Medio Oeste	verano	temperature	82
2016-06-13 17:43:50	Estados Unidos-Medio Oeste	verano	humedad	71

`airquality`

hora	location	measure_name	measure_value::bigint
2016-06-13 17:43:50	Estados Unidos-Medio Oeste	no 2	5
2016-06-13 17:43:50	Estados Unidos-Medio Oeste	pm25	16

Almacenar los datos en una sola tabla

Puede optar por almacenar todas las medidas en una sola tabla y almacenar cada campo en una fila de tabla independiente.

La configuración esmapping_mode = "single-table". Hay dos configuraciones adicionales cuando se usasingle-table, single_table_name ysingle_table_dimension_name_for_telegraf_measurement_name.

- El complemento Timestream para la LiveAnalytics salida creará una sola tabla con el nombre *<single_table_name>* que incluye un *<single_table_dimension_name_for_telegraf_measurement_name>* columna.
- La tabla puede contener varios campos en una sola fila de la tabla.

El Timestream de la LiveAnalytics tabla resultante tendrá este aspecto.

weather

hora	location	temporada	***<single_table_dimension_name_for_telegraf_measurement_name>***	measure_name	measure_value::bigint
2016-06-13 17:43:50	Estados Unidos-Medio Oeste	verano	clima	temperature	82
2016-06-13 17:43:50	Estados Unidos-Medio Oeste	verano	clima	humedad	71
2016-06-13 17:43:50	Estados Unidos-Medio Oeste	verano	calidad del aire	no 2	5
2016-06-13 17:43:50	Estados Unidos-Medio Oeste	verano	clima	pm25	16

JDBC

Puede utilizar una conexión de conectividad de bases de datos Java (JDBC) para conectar Timestream for LiveAnalytics a sus herramientas de inteligencia empresarial y otras aplicaciones, como SQL Workbench. El Timestream for LiveAnalytics JDBC Driver actualmente es compatible SSO con Okta y Microsoft Azure AD.

Temas

- Configuración del JDBC controlador para Timestream para LiveAnalytics
- Propiedades de conexión
- JDBCURLejemplos
- Configuración de Timestream para la autenticación de inicio de sesión LiveAnalytics JDBC único con Okta
- Configuración de Timestream para la autenticación de inicio de sesión LiveAnalytics JDBC único con Microsoft Azure AD

Configuración del JDBC controlador para Timestream para LiveAnalytics

Siga los pasos que se indican a continuación para configurar el JDBC controlador.

Temas

- Secuencia temporal para el conductor LiveAnalytics JDBC JARs
- Secuencia temporal de la clase y el LiveAnalytics JDBC formato del conductor URL
- Aplicación de muestra

Secuencia temporal para el conductor LiveAnalytics JDBC JARs

Puede obtener el Timestream del LiveAnalytics JDBC controlador mediante una descarga directa o añadiendo el controlador como una dependencia de Maven.

- Como descarga directa:. Para descargar directamente el LiveAnalytics JDBC controlador Timestream for, complete los siguientes pasos:

  1. Navegue hasta /releases https://github.com/awslabs/ amazon-timestream-driver-jdbc
  2. Puede usarlo `amazon-timestream-jdbc-1.0.1-shaded.jar` directamente con sus herramientas y aplicaciones de inteligencia empresarial

3. `amazon-timestream-jdbc-1.0.1-javadoc.jar`Descárguelo en el directorio que prefiera.
4. En el directorio en el que realizó la descarga`amazon-timestream-jdbc-1.0.1-javadoc.jar`, ejecute el siguiente comando para extraer los archivos JavadocHTML:

```
jar -xvf amazon-timestream-jdbc-1.0.1-javadoc.jar
```

- Como dependencia de Maven: para añadir el Timestream para el LiveAnalytics JDBC controlador como una dependencia de Maven, complete los siguientes pasos:

1. Navega hasta el `pom.xml` archivo de tu aplicación y ábrelo en el editor que prefieras.
2. Añada el JDBC controlador como una dependencia al `pom.xml` archivo de la aplicación:

```
<!-- https://mvnrepository.com/artifact/software.amazon.timestream/amazon-
timestream-jdbc -->
<dependency>
    <groupId>software.amazon.timestream</groupId>
    <artifactId>amazon-timestream-jdbc</artifactId>
    <version>1.0.1</version>
</dependency>
```

Secuencia temporal de la clase y el LiveAnalytics JDBC formato del conductor URL

La clase de conductor de Timestream for driver es: LiveAnalytics JDBC

```
software.amazon.timestream.jdbc.TimestreamDriver
```

El JDBC controlador Timestream requiere el siguiente formato: JDBC URL

```
jdbc:timestream:
```

Para especificar las propiedades de la base de datos mediante el JDBCURL, utilice el siguiente formato: URL

```
jdbc:timestream://
```

Aplicación de muestra

Para ayudarlo a comenzar a usar Timestream for LiveAnalytics withJDBC, hemos creado una aplicación de muestra completamente funcional en. GitHub

1. Cree una base de datos con datos de muestra siguiendo las instrucciones que se describen aquí.
2. Clone el GitHub repositorio de la aplicación de muestra JDBC siguiendo las instrucciones de GitHub.
3. Siga las instrucciones de la READMEpara empezar a utilizar la aplicación de muestra.

Propiedades de conexión

El Timestream para el LiveAnalytics JDBC controlador admite las siguientes opciones:

Temas

- Opciones de autenticación básicas
- Opción de información de cliente estándar
- Opción de configuración del controlador
- SDKopción
- Opción de configuración del punto final
- Opciones de proveedores de credenciales
- SAMLopciones de autenticación basadas en Okta
- SAMLopciones de autenticación basadas en Azure AD

> **Note**
>
> Si no se proporciona ninguna de las propiedades, Timestream para el LiveAnalytics JDBC conductor utilizará la cadena de credenciales predeterminada para cargar las credenciales.

> **Note**
>
> Todas las claves de propiedad distinguen mayúsculas de minúsculas.

Opciones de autenticación básicas

En la siguiente tabla se describen las opciones de autenticación básica disponibles.

Opción	Descripción	Predeterminado
AccessKeyId	El identificador de la clave de acceso del AWS usuario.	NONE
SecretAccessKey	La clave de acceso secreta del AWS usuario.	NONE
SessionToken	El token de sesión temporal necesario para acceder a una base de datos con la autenticación multifactor (MFA) habilitada.	NONE

Opción de información de cliente estándar

En la siguiente tabla se describe la opción de información de cliente estándar.

Opción	Descripción	Predeterminado
ApplicationName	El nombre de la aplicación que utiliza actualmente la conexión. `ApplicationName` se utiliza con fines de depuración y no se comunicará a Timestream para su reparación. LiveAnalytics	El nombre de la aplicación detectado por el controlador.

Opción de configuración del controlador

En la siguiente tabla se describe la opción de configuración del controlador.

Opción	Descripción	Predeterminado
EnableMetaDataPreparedStatement	Habilita Timestream para que el LiveAnalytics JDBC conductor devuelva los metadatos`PreparedStatements`, pero esto conllevará un coste adicional si Timestream recupera los LiveAnalytics metadatos.	FALSE
Región	La región de la base de datos.	us-east-1

SDKopción

En la siguiente tabla se describe la SDK opción.

Opción	Descripción	Predeterminado
RequestTimeout	El tiempo en milisegundos que AWS SDK esperará una solicitud de consulta antes de que se agote el tiempo de espera. Un valor no positivo desactiva el tiempo de espera de la solicitud.	0
SocketTimeout	El tiempo en milisegundos que AWS SDK se esperará a que los datos se transfieran a través de una conexión abierta antes de que se agote el tiempo de espera. El valor no debe ser negativo. Un valor de `0` desactiva el tiempo de espera del socket.	50000

Opción	Descripción	Predeterminado
MaxRetryCountClient	El número máximo de reintentos para los errores que se pueden volver a intentar con códigos de error de 5XX en. SDK El valor no debe ser negativo.	NONE
MaxConnections	El número máximo permitido de HTTP conexiones abiertas simultáneamente al Timestrea m para el servicio. LiveAnaly tics El valor debe ser positivo.	50

Opción de configuración del punto final

En la siguiente tabla se describe la opción de configuración del punto final.

Opción	Descripción	Predeterminado
Punto de conexión	El punto final del Timestream for LiveAnalytics Service.	NONE

Opciones de proveedores de credenciales

En la siguiente tabla se describen las opciones de proveedores de credenciales disponibles.

Opción	Descripción	Predeterminado
AwsCredentialsProviderClass	Uno de los `PropertiesFileCredentialsProvider` dos `InstanceProfileCredentialsProvider` para usar en la autenticación.	NONE

Opción	Descripción	Predeterminado
CustomCredentialsFilePath	La ruta a un archivo de propiedades que contiene las credenciales AWS de seguridad `accessKey` `ysecretKey` . Esto solo es obligatorio si `AwsCredentialsProv iderClass` se especific a como`Propertie sFileCredentialsPr ovider` .	NONE

SAMLopciones de autenticación basadas en Okta

En la siguiente tabla se describen las opciones de autenticación SAML basadas disponibles para Okta.

Opción	Descripción	Predeterminado
IdpName	El nombre del proveedor de identidad (Idp) que se utilizará para la autenticación SAML basada. Uno de `Okta` o. `AzureAD`	NONE
IdpHost	El nombre de host del Idp especificado.	NONE
IdpUserName	El nombre de usuario de la cuenta de Idp especificada.	NONE
IdpPassword	La contraseña de la cuenta de Idp especificada.	NONE
OktaApplicationID	El identificador único proporcio nado por Okta y asociado al	NONE

Opción	Descripción	Predeterminado
	Timestream de la solicitud . LiveAnalytics `AppId`se encuentra en el `entityID` campo proporcionado en los metadatos de la aplicació n. Considere el siguiente ejemplo: `entityID = http://www.okta.co m//IdpAppID`	
Función ARN	El nombre del recurso de Amazon (ARN) del rol que asume la persona que llama.	NONE
Idp ARN	El nombre del recurso de Amazon (ARN) del SAML proveedor IAM que describe al Idp.	NONE

SAMLopciones de autenticación basadas en Azure AD

En la siguiente tabla se describen las opciones de autenticación SAML basadas disponibles para Azure AD.

Opción	Descripción	Predeterminado
IdpName	El nombre del proveedor de identidad (Idp) que se utilizará para la autenticación SAML basada. Uno de `Okta` o. `AzureAD`	NONE
IdpHost	El nombre de host del Idp especificado.	NONE

Opción	Descripción	Predeterminado
IdpUserName	El nombre de usuario de la cuenta de Idp especificada.	NONE
IdpPassword	La contraseña de la cuenta de Idp especificada.	NONE
AADApplicationID	El identificador único de la aplicación registrada en Azure AD.	NONE
AADClientSecret	El secreto de cliente asociado a la aplicación registrada en Azure AD que se utiliza para autorizar la obtención de los tokens.	NONE
AADTenant	El ID de inquilino de Azure AD.	NONE
Idp ARN	El nombre del recurso de Amazon (ARN) del SAML proveedor IAM que describe al Idp.	NONE

JDBCURLejemplos

En esta sección se describe cómo crear una JDBC conexión URL y se proporcionan ejemplos. Para especificar las propiedades de conexión opcionales, utilice el siguiente URL formato:

```
jdbc:timestream://PropertyName1=value1;PropertyName2=value2...
```

> **Note**
>
> Todas las propiedades de conexión son opcionales. Todas las claves de propiedad distinguen mayúsculas de minúsculas.

A continuación se muestran algunos ejemplos de JDBC conexiónURLs.

Ejemplo con opciones de autenticación básicas y región:

```
jdbc:timestream://
AccessKeyId=<myAccessKeyId>;SecretAccessKey=<mySecretAccessKey>;SessionToken=<mySessionToken
east-1
```

Ejemplo con información del cliente, región y SDK opciones:

```
jdbc:timestream://ApplicationName=MyApp;Region=us-
east-1;MaxRetryCountClient=10;MaxConnections=5000;RequestTimeout=20000
```

Conéctese mediante la cadena de proveedores de credenciales predeterminada con las AWS credenciales configuradas en las variables de entorno:

```
jdbc:timestream
```

Conéctese mediante la cadena de proveedores de credenciales predeterminada con las AWS credenciales configuradas en la conexión: URL

```
jdbc:timestream://
AccessKeyId=<myAccessKeyId>;SecretAccessKey=<mySecretAccessKey>;SessionToken=<mySessionToken
```

Conéctese utilizando PropertiesFileCredentialsProvider como método de autenticación:

```
jdbc:timestream://
AwsCredentialsProviderClass=PropertiesFileCredentialsProvider;CustomCredentialsFilePath=<pat
 to properties file>
```

Conéctese utilizando InstanceProfileCredentialsProvider como método de autenticación:

```
jdbc:timestream://AwsCredentialsProviderClass=InstanceProfileCredentialsProvider
```

Conéctese utilizando las credenciales de Okta como método de autenticación:

```
jdbc:timestream://
IdpName=Okta;IdpHost=<host>;IdpUserName=<name>;IdpPassword=<password>;OktaApplicationID=<id>
```

Conéctese con las credenciales de Azure AD como método de autenticación:

```
jdbc:timestream:// 
IdpName=AzureAD;IdpUserName=<name>;IdpPassword=<password>;AADApplicationID=<id>;AADClientSec
```

Conéctese con un punto final específico:

```
jdbc:timestream://Endpoint=abc.us-east-1.amazonaws.com;Region=us-east-1
```

Configuración de Timestream para la autenticación de inicio de sesión LiveAnalytics JDBC único con Okta

Timestream for LiveAnalytics es compatible con Timestream para la autenticación de inicio de sesión único con Okta LiveAnalytics JDBC. Para usar Timestream para la autenticación de inicio de sesión LiveAnalytics JDBC único con Okta, complete cada una de las secciones que se indican a continuación.

Temas

- Requisitos previos
- AWS federación de cuentas en Okta
- Configuración de Okta para SAML

Requisitos previos

Asegúrese de cumplir los siguientes requisitos previos antes de usar Timestream para la autenticación de inicio de sesión único con Okta: LiveAnalytics JDBC

- Los permisos de administrador son necesarios AWS para crear el proveedor de identidades y los roles.
- Una cuenta de Okta (vaya a https://www.okta.com/login/para crear una cuenta).
- Acceso a Amazon LiveAnalytics Timestream para.

Ahora que ha completado los requisitos previos, puede continuar con. AWS federación de cuentas en Okta

AWS federación de cuentas en Okta

El Timestream for LiveAnalytics JDBC Driver es compatible con la federación de AWS cuentas en Okta. Para configurar la federación de AWS cuentas en Okta, siga estos pasos:

1. Inicie sesión en el panel de administración de Okta de la siguiente manera: URL

```
https://<company-domain-name>-admin.okta.com/admin/apps/active
```

> **Note**
>
> Sustituye < company-domain-name > por tu nombre de dominio.

2. Tras iniciar sesión correctamente, selecciona Añadir aplicación y busca AWS Account Federation.
3. Selecciona Añadir
4. Cambie el inicio de URL sesión por el apropiadoURL.
5. Elija Siguiente.
6. Elija SAML2.0 como método de inicio de sesión
7. Elija los metadatos del proveedor de identidad para abrir el archivo de metadatosXML. Guarde el archivo localmente.
8. Deje en blanco todas las demás opciones de configuración.
9. Seleccione Listo.

Ahora que ha completado la federación de AWS cuentas en Okta, puede continuar Configuración de Okta para SAML con.

Configuración de Okta para SAML

1. Elija la pestaña Sign On (Iniciar sesión). Elige la vista.
2. Pulse el botón Instrucciones de configuración en la sección Configuración.

Búsqueda del documento de metadatos de Okta

1. Para buscar el documento, vaya a:

```
https://<domain>-admin.okta.com/admin/apps/active
```

> **Note**
>
> <domain>es el nombre de dominio exclusivo de tu cuenta de Okta.

2. Elija la aplicación AWS Account Federation
3. Seleccione la pestaña Iniciar sesión

Configuración de Timestream para la autenticación de inicio de sesión LiveAnalytics JDBC único con Microsoft Azure AD

Timestream for LiveAnalytics admite Timestream para la autenticación de inicio de sesión LiveAnalytics JDBC único con Microsoft Azure AD. Para usar Timestream para la autenticación de inicio de sesión LiveAnalytics JDBC único con Microsoft Azure AD, complete cada una de las secciones que se indican a continuación.

Temas

- Requisitos previos
- Configuración de Azure AD
- Configurar el proveedor IAM de identidad y las funciones en AWS

Requisitos previos

Asegúrese de cumplir los siguientes requisitos previos antes de usar Timestream para la autenticación de inicio de sesión LiveAnalytics JDBC único con Microsoft Azure AD:

- Permisos de administrador AWS para crear el proveedor de identidades y las funciones.
- Una cuenta de Azure Active Directory (vaya a https://azure.microsoft.com/en-ca/services/active-directory/ para crear una cuenta)
- Acceso a Amazon LiveAnalytics Timestream para.

Configuración de Azure AD

1. Inicie sesión en Azure Portal
2. Elija Azure Active Directory en la lista de servicios de Azure. Esto redirigirá a la página del directorio predeterminado.

3. Selecciona Aplicaciones empresariales en la sección Administrar de la barra lateral
4. Selecciona + Nueva aplicación.
5. Busca y selecciona Amazon Web Services.
6. Selecciona el inicio de sesión único en la sección Administrar de la barra lateral
7. Elija SAMLcomo método de inicio de sesión único
8. En la sección SAML Configuración básica, introduce lo siguiente tanto URL para el identificador como para la respuesta: URL

```
https://signin.aws.amazon.com/saml
```

9. Seleccione Save.
10. Descargue los metadatos de la federación XML en la sección del certificado de SAML firma. Esto se utilizará más adelante al crear el proveedor de IAM identidad
11. Vuelva a la página del directorio predeterminado y elija Registros de aplicaciones en Administrar.
12. Seleccione Timestream para en la sección Todas LiveAnalytics las aplicaciones. La página se redirigirá a la página de descripción general de la aplicación

> **Note**
>
> Anote el ID de la aplicación (cliente) y el ID del directorio (inquilino). Estos valores son necesarios para crear una conexión.

13. Elija Certificados y secretos
14. En Secretos de cliente, crea un nuevo secreto de cliente con + Nuevo secreto de cliente.

> **Note**
>
> Anote el secreto de cliente generado, ya que es obligatorio al crear una conexión a Timestream para. LiveAnalytics

15. En la barra lateral, en Administrar, selecciona permisos API
16. En los permisos configurados, utilice Añadir un permiso para conceder a Azure AD permiso para iniciar sesión en Timestream. LiveAnalytics Elija Microsoft Graph en la página Solicitar API permisos.
17. Elija Permisos delegados y seleccione el permiso User.Read

18. Elija Añadir permisos
19. Selecciona Otorgar el consentimiento del administrador para el directorio predeterminado

Configurar el proveedor IAM de identidad y las funciones en AWS

Complete cada sección siguiente para configurar Timestream IAM para la autenticación de inicio de sesión LiveAnalytics JDBC único con Microsoft Azure AD:

Temas

- Cree un proveedor de identidad SAML
- Crea un IAM rol
- Cree una IAM política
- Aprovisionando

Cree un proveedor de identidad SAML

Para crear un proveedor de SAML identidad para Timestream para la autenticación de inicio de sesión LiveAnalytics JDBC único con Microsoft Azure AD, complete los siguientes pasos:

1. Inicie sesión en la consola de administración AWS
2. Elija Servicios y IAMseleccione Seguridad, identidad y conformidad
3. Elija los proveedores de identidad en Administración de acceso
4. Elija Crear proveedor y elija SAMLel tipo de proveedor. Introduzca el nombre del proveedor. En este ejemplo se utilizará zureADProvider A.
5. Cargue el XML archivo de metadatos de la federación descargado anteriormente
6. Seleccione Siguiente y, a continuación, seleccione Crear.
7. Al finalizar, la página se redirigirá de nuevo a la página de proveedores de identidad

Crea un IAM rol

Para crear un IAM rol para Timestream para la autenticación de inicio de sesión LiveAnalytics JDBC único con Microsoft Azure AD, complete los siguientes pasos:

1. En la barra lateral, selecciona Roles en Administración de acceso
2. Seleccione Create role (Crear función)

3. Elija la federación SAML 2.0 como entidad de confianza
4. Elija el proveedor de Azure AD
5. Elija Permitir el acceso programático y a AWS la consola de administración
6. Elija Next: Permissions (Siguiente: permisos)
7. Adjunte las políticas de permisos o continúe con Next:Tags
8. Agregue etiquetas opcionales o continúe con Next:Review
9. Escriba un Role name. En este ejemplo se utilizará A zureSAMLRole
10. Proporcione una descripción de la función
11. Elija Crear rol para completar

Cree una IAM política

Para crear una IAM política para Timestream para la autenticación de inicio de sesión LiveAnalytics JDBC único con Microsoft Azure AD, complete los siguientes pasos:

1. En la barra lateral, selecciona Políticas en Administración de acceso
2. Elija Crear política y seleccione la pestaña JSON
3. Agregue la siguiente política

```
{
"Version": "2012-10-17",
"Statement": [
    {
          "Effect": "Allow",
          "Action": [
                "iam:ListRoles",
                "iam:ListAccountAliases"
           ],
           "Resource": "*"
      }
]
}
```

4. Elija Create Policy
5. Escriba un nombre para la política. En este ejemplo se utilizará TimestreamAccessPolicy.
6. Elija Crear política
7. En la barra lateral, selecciona Funciones en Administración de acceso.

8. Elija el rol de Azure AD creado anteriormente y elija Adjuntar políticas en Permisos.
9. Seleccione la política de acceso creada anteriormente.

Aprovisionando

Para aprovisionar el proveedor de identidades de Timestream para la autenticación de inicio de sesión LiveAnalytics JDBC único con Microsoft Azure AD, complete los siguientes pasos:

1. Vuelva a Azure Portal
2. Elija Azure Active Directory en la lista de servicios de Azure. Esto redirigirá a la página del directorio predeterminado
3. Selecciona Aplicaciones empresariales en la sección Administrar de la barra lateral
4. Elija Aprovisionamiento
5. Elija el modo automático para el método de aprovisionamiento
6. En Credenciales de administrador, introduzca su AwsAccessKeyID para clientsecret y Secret SecretAccessKeyToken
7. Establezca el estado del aprovisionamiento en Activado
8. Seleccione guardar. Esto permite a Azure AD cargar las IAM funciones necesarias
9. Una vez que se complete el estado del ciclo actual, elija Usuarios y grupos en la barra lateral
10. Selecciona + Añadir usuario
11. Elija el usuario de Azure AD al que desea proporcionar acceso a Timestream LiveAnalytics
12. Elija el rol de IAM Azure AD y el proveedor de identidad de Azure correspondiente creado en AWS
13. Elija Asignar

ODBC

El ODBCcontrolador de código abierto para Amazon LiveAnalytics Timestream SQL proporciona una interfaz relacional con LiveAnalytics Timestream para desarrolladores y permite la conectividad desde herramientas de inteligencia empresarial (BI) como Power BI Desktop y Microsoft Excel. El LiveAnalytics ODBC controlador Timestream está disponible actualmente en Windows, macOS y Linux, y también es compatible SSO con Okta y Microsoft Azure Active Directory (AD).

Para obtener más información, consulte Amazon Timestream LiveAnalytics ODBC para ver la documentación del conductor sobre. GitHub

Temas

Configuración del Timestream para el conductor LiveAnalytics ODBC

Configura el acceso a Timestream desde tu cuenta LiveAnalytics AWS

Si aún no has configurado tu AWS cuenta para que funcione con Timestream LiveAnalytics, sigue las instrucciones que se indican en. Acceder a Timestream para LiveAnalytics

Instala el controlador en tu sistema ODBC

Descargue el instalador del ODBC controlador Timestream adecuado para su sistema desde el ODBC GitHubrepositorio y siga las instrucciones de instalación aplicables a su sistema:.

- Guía de instalación de Windows
- Guía de instalación de macOS
- Guía de instalación de Linux

Configure un nombre de fuente de datos (DSN) para el ODBC controlador

Siga las instrucciones de la guía de DSN configuración del sistema:

- DSNConfiguración de Windows
- DSNConfiguración de macOS
- DSNConfiguración de Linux

Configure su aplicación de inteligencia empresarial (BI) para que funcione con el ODBC controlador

Estas son las instrucciones para configurar varias aplicaciones de BI habituales para que funcionen con el ODBC controlador:

- Configuración de Microsoft Power BI.

- Configuración de Microsoft Excel
- Configuración de Tableau

Para otras aplicaciones

Sintaxis de la cadena de conexión y opciones del ODBC controlador

La sintaxis para especificar las opciones de cadena de conexión para el ODBC controlador es la siguiente:

```
DRIVER={Amazon Timestream ODBC Driver};(option)=(value);
```

Las opciones disponibles son las siguientes:

Opciones de conexión del controlador

- **`Driver`**(obligatorio): el controlador con el que se utilizaODBC.

  El valor predeterminado es Amazon Timestream.
- **`DSN`**— El nombre de la fuente de datos (DSN) que se utilizará para configurar la conexión.

  El valor predeterminado es `NONE`.
- **`Auth`**— El modo de autenticación. Este debe ser uno de los siguientes:
  - `AWS_PROFILE`— Utilice la cadena de credenciales predeterminada.
  - `IAM`— Usa AWS IAM credenciales.
  - `AAD`— Utilice el proveedor de identidades de Azure Active Directory (AD).
  - `OKTA`— Utilice el proveedor de identidades Okta.

  El valor predeterminado es `AWS_PROFILE`.

Opciones de configuración del punto final

- **`EndpointOverride`**— La anulación del punto final del Timestream for Service. LiveAnalytics Se trata de una opción avanzada que anula la región. Por ejemplo:

  ```
  query-cell2.timestream.us-east-1.amazonaws.com
  ```
- **`Region`**— La región de firma del Timestream for LiveAnalytics Service Endpoint.

El valor predeterminado es `us-east-1`.

Opción de proveedor de credenciales

- **`ProfileName`**— El nombre del perfil en el archivo de AWS configuración.

  El valor predeterminado es `NONE`.

AWS IAMopciones de autenticación

- **`UID`**o **`AccessKeyId`**: el identificador de la clave de acceso del AWS usuario. Si `AccessKeyId` se proporcionan ambos `UID` y en la cadena de conexión, se utilizará el `UID` valor a menos que esté vacío.

  El valor predeterminado es `NONE`.
- **`PWD`**o **`SecretKey`**: la clave de acceso secreta del AWS usuario. Si `SecretKey` se proporcionan ambos `PWD` y en la cadena de conexión, se utilizará el `PWD` valor con a menos que esté vacío.

  El valor predeterminado es `NONE`.
- **`SessionToken`**— El token de sesión temporal necesario para acceder a una base de datos con la autenticación multifactorial (MFA) habilitada. No incluya un final `=` en la entrada.

  El valor predeterminado es `NONE`.

SAMLopciones de autenticación basadas en Okta

- **`IdPHost`**— El nombre de host del IdP especificado.

  El valor predeterminado es `NONE`.
- **`UID`**o **`IdPUserName`**: el nombre de usuario de la cuenta de IdP especificada. Si `IdPUserName` se proporcionan ambos `UID` y en la cadena de conexión, se utilizará el `UID` valor a menos que esté vacío.

  El valor predeterminado es `NONE`.
- **`PWD`**o **`IdPPassword`**— La contraseña de la cuenta de IdP especificada. Si `IdPPassword` se proporcionan ambos `PWD` y en la cadena de conexión, se utilizará el `PWD` valor a menos que esté vacío.

El valor predeterminado es NONE.

- **OktaApplicationID**— El identificador único proporcionado por Okta y asociado al Timestream for Application. LiveAnalytics El identificador de la aplicación (AppId) se encuentra en el `entityID` campo proporcionado en los metadatos de la aplicación. Un ejemplo es:

  ```
  entityID="http://www.okta.com//(IdPAppID)
  ```

  El valor predeterminado es NONE.
- **RoleARN**— El nombre del recurso de Amazon (ARN) del rol que asume la persona que llama.

  El valor predeterminado es NONE.
- **IdPARN**— El nombre del recurso de Amazon (ARN) del SAML proveedor IAM que describe el IdP.

  El valor predeterminado es NONE.

SAMLopciones de autenticación basadas en Azure Active Directory

- **UID**o **IdPUserName**: el nombre de usuario de la cuenta de IdP especificada.

  El valor predeterminado es NONE.
- **PWD**o **IdPPassword**— La contraseña de la cuenta de IdP especificada.

  El valor predeterminado es NONE.
- **AADApplicationID**— El identificador único de la aplicación registrada en Azure AD.

  El valor predeterminado es NONE.
- **AADClientSecret**— El secreto del cliente asociado a la aplicación registrada en Azure AD que se utiliza para autorizar la obtención de los tokens.

  El valor predeterminado es NONE.
- **AADTenant**— El ID de inquilino de Azure AD.

  El valor predeterminado es NONE.
- **RoleARN**— El nombre del recurso de Amazon (ARN) del rol que asume la persona que llama.

  El valor predeterminado es NONE.
- **IdPARN**— El nombre del recurso de Amazon (ARN) del SAML proveedor IAM que describe el IdP.

El valor predeterminado es `NONE`.

AWS SDKOpciones (avanzadas)

- **`RequestTimeout`**— El tiempo en milisegundos que AWS SDK espera una solicitud de consulta antes de que se agote el tiempo de espera. Cualquier valor no positivo desactiva el tiempo de espera de la solicitud.

  El valor predeterminado es `3000`.
- **`ConnectionTimeout`**— El tiempo en milisegundos que AWS SDK espera a que se transfieran los datos a través de una conexión abierta antes de que se agote el tiempo de espera. Un valor de 0 desactiva el tiempo de espera de la conexión. Este valor no debe ser negativo.

  El valor predeterminado es `1000`.
- **`MaxRetryCountClient`**— El número máximo de reintentos para los errores que se pueden volver a intentar con códigos de error de 5xx en. SDK El valor no debe ser negativo.

  El valor predeterminado es `0`.
- **`MaxConnections`**— El número máximo de HTTP conexiones abiertas simultáneamente al servicio Timestream. El valor debe ser positivo.

  El valor predeterminado es `25`.

ODBCOpciones de registro del conductor

- **`LogLevel`**— El nivel de registro para el registro del conductor. Debe ser uno de los siguientes:
  - (0OFF).
  - (1ERROR).
  - (2WARNING).
  - (3INFO).
  - (4DEBUG).

  El valor predeterminado es 1 (ERROR).

  Advertencia: el conductor puede registrar información personal al utilizar el modo de DEBUG registro.

- **LogOutput**— Carpeta en la que almacenar el archivo de registro.

  El valor predeterminado es:

  - Windows:`%USERPROFILE%`, o si no está disponible,`%HOMEDRIVE%%HOMEPATH%`.
  - macOS y Linux: `$HOME` o si no está disponible, el campo `pw_dir` del valor `getpwuid(getuid())` devuelto por la función.

SDKopciones de registro

El nivel de AWS SDK registro es independiente del nivel de registro de Timestream para el LiveAnalytics ODBC conductor. Establecer uno no afecta al otro.

El nivel de SDK registro se establece mediante la variable de entornoTS_AWS_LOG_LEVEL. Los valores válidos son:

- `OFF`
- `ERROR`
- `WARN`
- `INFO`
- `DEBUG`
- `TRACE`
- `FATAL`

Si no `TS_AWS_LOG_LEVEL` está establecido, el nivel de SDK registro se establece en el valor predeterminado, que es`WARN`.

Conexión a través de un proxy

El ODBC controlador admite la conexión a Amazon Timestream LiveAnalytics a través de un proxy. Para utilizar esta función, configure las siguientes variables de entorno en función de la configuración del proxy:

- **TS_PROXY_HOST**— el host del proxy.
- **TS_PROXY_PORT**— El número de puerto del proxy.
- **TS_PROXY_SCHEME**— El esquema de proxy, ya sea `http` o`https`.
- **TS_PROXY_USER**— El nombre de usuario para la autenticación mediante proxy.

- **TS_PROXY_PASSWORD**— La contraseña de usuario para la autenticación mediante proxy.
- **TS_PROXY_SSL_CERT_PATH**— El archivo de SSL certificado que se utilizará para conectarse a un HTTPS proxy.
- **TS_PROXY_SSL_CERT_TYPE**— El tipo de SSL certificado del cliente proxy.
- **TS_PROXY_SSL_KEY_PATH**— El archivo de clave privada que se utilizará para conectarse a un HTTPS proxy.
- **TS_PROXY_SSL_KEY_TYPE**— El tipo de archivo de clave privada utilizado para conectarse a un HTTPS proxy.
- **TS_PROXY_SSL_KEY_PASSWORD**— La contraseña del archivo de clave privada que se utiliza para conectarse a un proxy. HTTPS

Ejemplos de cadenas de conexión para el Timestream para el conductor LiveAnalytics ODBC

Ejemplo de conexión al ODBC conductor con credenciales IAM

```
Driver={Amazon Timestream ODBC Driver};Auth=IAM;AccessKeyId=(your access key
 ID);secretKey=(your secret key);SessionToken=(your session token);Region=us-east-2;
```

Ejemplo de conexión al ODBC conductor con un perfil

```
Driver={Amazon Timestream ODBC Driver};ProfileName=(the profile name);region=us-west-2;
```

El controlador intentará conectarse con las credenciales proporcionadas en la variable de entorno o`~/.aws/credentials`, si se especifica un archivo en la variable de entornoAWS_SHARED_CREDENTIALS_FILE, con las credenciales de ese archivo.

Ejemplo de conexión al ODBC controlador con Okta

```
driver={Amazon Timestream ODBC Driver};auth=okta;region=us-west-2;idPHost=(your host at
 Okta);idPUsername=(your user name);idPPassword=(your password);OktaApplicationID=(your
 Okta AppId);roleARN=(your role ARN);idPARN=(your Idp ARN);
```

Ejemplo de conexión al ODBC controlador con Azure Active Directory () AAD

```
driver={Amazon Timestream ODBC Driver};auth=aad;region=us-west-2;idPUsername=(your
 user name);idPPassword=(your password);aadApplicationID=(your AAD
```

```
 AppId);aadClientSecret=(your AAD client secret);aadTenant=(your AAD
 tenant);roleARN=(your role ARN);idPARN=(your idP ARN);
```

Ejemplo de conexión al ODBC controlador con un punto final específico y un nivel de registro de 2 (WARNING)

```
Driver={Amazon Timestream ODBC Driver};Auth=IAM;AccessKeyId=(your access
 key ID);secretKey=(your secret key);EndpointOverride=ingest.timestream.us-
west-2.amazonaws.com;Region=us-east-2;LogLevel=2;
```

Solución de problemas de conexión con el ODBC controlador

> **Note**
>
> Si el nombre de usuario y la contraseña ya están especificados en elDSN, no es necesario volver a especificarlos cuando el administrador del ODBC conductor los solicite.

Se `Re-writing (connection string option) (have you specified it several times?` produce un código de error `01S02` con un mensaje cuando se pasa una opción de cadena de conexión más de una vez en la cadena de conexión. Si se especifica una opción más de una vez, se produce un error. Al realizar una conexión con una cadena de conexión DSN y una cadena de conexión, si ya se ha especificado una opción de conexión en laDSN, no la vuelva a especificar en la cadena de conexión.

VPCpuntos finales ()AWS PrivateLink

Puede establecer una conexión privada entre Amazon Timestream VPC y usted creando un punto de enlace LiveAnalytics de interfaz. VPC Para obtener más información, consulte VPCpuntos finales ()AWS PrivateLink.

Prácticas recomendadas

Para aprovechar al máximo las ventajas de Amazon Timestream LiveAnalytics for, siga las prácticas recomendadas que se describen a continuación.

> **Note**
>
> Al ejecutar proof-of-concept aplicaciones, tenga en cuenta la cantidad de datos que la aplicación acumulará en unos meses o años al evaluar el rendimiento y la escala de Timestream. LiveAnalytics A medida que sus datos aumenten con el tiempo, observará que el rendimiento de Timestream for LiveAnalytics prácticamente no ha cambiado, ya que su arquitectura sin servidor puede aprovechar enormes cantidades de paralelismo para procesar volúmenes de datos más grandes y escalar automáticamente para adaptarse a las necesidades de su aplicación.

Temas

Modelado de datos

Amazon Timestream LiveAnalytics for está diseñado para recopilar, almacenar y analizar datos de series temporales de aplicaciones y dispositivos que emiten una secuencia de datos con una marca de tiempo. Para obtener un rendimiento óptimo, los datos que se envían a Timestream LiveAnalytics deben tener características temporales y el tiempo debe ser un componente esencial de los datos.

Timestream for le LiveAnalytics brinda la flexibilidad de modelar sus datos de diferentes maneras para adaptarlos a los requisitos de su aplicación. En esta sección, abordamos varios de estos patrones y proporcionamos pautas para que pueda optimizar sus costos y su rendimiento. Familiarícese con los LiveAnalytics conceptos clave de Timestream, como las dimensiones y las medidas. En esta sección, obtendrá más información sobre:

A la hora de decidir si crear una o varias tablas para almacenar datos, tenga en cuenta lo siguiente:

- Qué datos incluir en la misma tabla o cuándo desea separar los datos en varias tablas y bases de datos.
- Cómo elegir entre Timestream para registros de LiveAnalytics múltiples medidas o registros de una sola medida, y las ventajas de modelar con registros de múltiples medidas, especialmente cuando su aplicación rastrea múltiples mediciones al mismo tiempo y en un instante.
- Qué atributos modelar como dimensiones o como medidas.
- Cómo utilizar de forma eficaz los atributos del nombre de la medida para optimizar la latencia de las consultas.

Temas

Tabla única frente a tablas múltiples

Al modelar los datos en la aplicación, otro aspecto importante es cómo modelar los datos en tablas y bases de datos. Las bases de datos y las tablas de Timestream LiveAnalytics son abstracciones para controlar el acceso, especificar KMS claves, períodos de retención, etc. Timestream permite particionar LiveAnalytics automáticamente los datos y está diseñado para escalar los recursos de forma que se adapten a la carga, el almacenamiento y las consultas, y a los requisitos de sus aplicaciones.

Una tabla de Timestream for LiveAnalytics puede ampliarse a petabytes de datos almacenados, decenas de gigabytes/segundo de escrituras de datos y las consultas pueden procesar cientos de ellas por hora. TBs Las consultas de Timestream for LiveAnalytics pueden abarcar varias tablas y bases de datos, lo que proporciona uniones y uniones que permiten acceder sin problemas a los datos de varias tablas y bases de datos. Por lo tanto, la escala de los datos o los volúmenes de solicitudes no suelen ser la principal preocupación a la hora de decidir cómo organizar los datos en Timestream. LiveAnalytics A continuación, se incluyen algunas consideraciones importantes a la hora de decidir qué datos colocar en la misma tabla o en tablas diferentes o tablas de bases de datos diferentes.

- Las políticas de retención de datos (retención del almacenamiento de memoria, retención del almacenamiento magnético, etc.) se admiten con la granularidad de una tabla. Por lo tanto, los datos que requieren políticas de retención diferentes deben estar en tablas diferentes.
- AWS KMS las claves que se utilizan para cifrar los datos se configuran a nivel de base de datos. Por lo tanto, los diferentes requisitos de clave de cifrado implican que los datos deberán estar en bases de datos diferentes.
- Timestream for LiveAnalytics admite el control de acceso basado en los recursos con la granularidad de tablas y bases de datos. Tenga en cuenta sus requisitos de control de acceso al decidir qué datos escribir en la misma tabla o en tablas diferentes.
- Tenga en cuenta los límites del número de dimensiones, nombres de medidas y nombres de atributos de medidas múltiples a la hora de decidir qué datos se almacenan en qué tabla.
- Tenga en cuenta la carga de trabajo de las consultas y los patrones de acceso a la hora de decidir cómo organizar los datos, ya que la latencia de las consultas y la facilidad de redacción de las consultas dependerán de ello.
  - Si guardas los datos que consultas con frecuencia en la misma tabla, esto suele facilitar la redacción de las consultas y, con frecuencia, evitar tener que escribir combinaciones, uniones o expresiones de tabla comunes. Esto también suele provocar una latencia de consulta más baja. Puede usar predicados en los nombres de las dimensiones y las medidas para filtrar los datos relevantes para las consultas.

    Por ejemplo, considere un caso en el que almacena datos de dispositivos ubicados en seis continentes. Si tus consultas acceden con frecuencia a datos de todos los continentes para obtener una vista global agregada, almacenar los datos de estos continentes en la misma tabla hará que las consultas sean más fáciles de escribir. Por otro lado, si almacenas datos en tablas diferentes, puedes combinar los datos en la misma consulta; sin embargo, tendrás que escribir una consulta para unir los datos de todas las tablas.
  - Timestream for LiveAnalytics utiliza el particionamiento y la indexación adaptables en los datos. Por lo tanto, a las consultas solo se les cobra por los datos que son relevantes para sus consultas. Por ejemplo, si tienes una tabla que almacena datos de un millón de dispositivos en seis continentes, si tu consulta tiene predicados del formulario `WHERE device_id = 'abcdef'` O`WHERE continent = 'North America'`, las consultas solo se cobran por los datos del dispositivo o del continente.
  - Siempre que sea posible, si utiliza el nombre de la medida para separar los datos de la misma tabla que no se emiten al mismo tiempo o que no se consultan con frecuencia, si utiliza predicados como `WHERE measure_name = 'cpu'` en su consulta, no solo obtiene

las ventajas de la medición, sino que Timestream for también LiveAnalytics puede eliminar eficazmente las particiones que no tienen el nombre de medida utilizado en el predicado de la consulta. Esto le permite almacenar datos relacionados con diferentes nombres de medidas en la misma tabla sin que ello afecte a la latencia de las consultas ni a los costes, y evita la dispersión de los datos en varias tablas. El nombre de la medida se usa básicamente para particionar los datos y eliminar las particiones irrelevantes para la consulta.

Registros de medidas múltiples frente a registros de medida única

Timestream for LiveAnalytics le permite escribir datos con varias medidas por registro (medidas múltiples) o una sola medida por registro (medida única).

Registros de medidas múltiples

En muchos casos de uso, un dispositivo o una aplicación que estés rastreando puede emitir varias métricas o eventos al mismo tiempo. En esos casos, puedes almacenar todas las métricas emitidas en la misma marca de tiempo en el mismo registro de múltiples medidas. Es decir, todas las medidas almacenadas en el mismo registro de múltiples medidas aparecen como columnas diferentes en la misma fila de datos.

Imagine, por ejemplo, que su aplicación emite métricas como la CPU, la memoria o el disk_iops desde un dispositivo medidas en el mismo instante. El siguiente es un ejemplo de una tabla de este tipo en la que varias métricas emitidas al mismo tiempo se almacenan en la misma fila. Verá que dos hosts emiten las métricas una vez por segundo.

Hostname	measure_name	Tiempo	cpu	Memoria	disk_iops
Host-24GJU	métricas	2021-12-01 19:00:00	35	54,9	38,2
Host-24GJU	métricas	2021-12-01 19:00:01	36	58	39
Host-28GJU	métricas	2021-12-01 19:00:00	15	55	92
Host-28GJU	métricas	2021-12-01 19:00:01	16	50	40

Registros de medida única

Los registros de medida única son adecuados cuando sus dispositivos emiten diferentes métricas en diferentes períodos de tiempo o si utiliza una lógica de procesamiento personalizada (que emite metrics/events at different time periods (for instance, when a device's reading/state cambios). Como cada medida tiene una marca de tiempo única, las medidas se pueden almacenar en sus propios registros en Timestream for. LiveAnalytics Por ejemplo, consideremos un sensor de IoT, que rastrea la temperatura y la humedad del suelo, que emite un registro solo cuando detecta un cambio con respecto a la entrada reportada anteriormente. En el siguiente ejemplo, se proporciona un ejemplo de la emisión de dichos datos mediante registros de una sola medida.

device_id	measure_name	Tiempo	measure_value::double	measure_value::bigint
sensor-sea478	temperature	2021-12-01 19:22:32	35	NULL
sensor-sea478	temperature	2021-12-01 18:07:51	36	NULL
sensor-sea478	hidratación	2021-12-01 19:05:30	NULL	21
sensor-sea478	hidratación	2021-12-01 19:00:01	NULL	23

Comparación de registros de una sola medida y de múltiples medidas

Timestream for le LiveAnalytics proporciona la flexibilidad de modelar sus datos como registros de una sola medida o de múltiples medidas, en función de los requisitos y las características de su aplicación. Una sola tabla puede almacenar registros de una sola medida y de múltiples medidas, si así lo desean los requisitos de su aplicación. En general, cuando la aplicación emite varias medidas o eventos al mismo tiempo y al mismo tiempo, se recomienda modelar los datos como registros de múltiples medidas para un acceso eficiente a los datos y un almacenamiento de datos rentable.

Por ejemplo, si se piensa en un caso DevOps práctico para realizar un seguimiento de las métricas y los eventos de cientos de miles de servidores, cada servidor emite periódicamente 20 métricas y 5 eventos, de forma que los eventos y las métricas se emiten al mismo tiempo y en un instante. Estos

datos se pueden modelar con registros de una sola medida o con registros de múltiples medidas (consulte el generador de datos de código abierto para ver el esquema resultante). Para este caso de uso, modelar los datos mediante registros de múltiples medidas en comparación con registros de una sola medida da como resultado:

- Medición de ingestión: los registros de medición múltiple producen aproximadamente un 40% menos de bytes de ingestión escritos.
- Procesamiento por lotes de ingestión: los registros que se miden varias veces dan como resultado el envío de lotes de datos más grandes, lo que implica que los clientes necesitan menos subprocesos y menos para procesar la ingestión. CPU
- Medición del almacenamiento: los registros con múltiples medidas reducen aproximadamente 8 veces la capacidad de almacenamiento, lo que se traduce en un importante ahorro de almacenamiento tanto en memoria como en almacenamiento magnético.
- Latencia de consulta: los registros de medidas múltiples dan como resultado una latencia de consulta más baja para la mayoría de los tipos de consultas en comparación con los registros de medición única.
- Bytes medidos por consulta: en el caso de las consultas que escanean menos de 10 MB de datos, los registros de medida única y de múltiples medidas son comparables. Para las consultas que acceden a una sola medida y escanean más de 10 MB de datos, los registros de una sola medida suelen tener como resultado bytes medidos más bajos. En el caso de las consultas que hacen referencia a 3 o más medidas, los registros de varias medidas dan como resultado una medición de bytes más baja.
- Facilidad de expresar consultas de varias medidas: cuando las consultas hacen referencia a varias medidas, modelar los datos con registros de varias medidas hace que sea más fácil escribir consultas más compactas.

Los factores anteriores variarán en función del número de métricas que realices el seguimiento, del número de dimensiones que tengan tus datos, etc. Si bien en el ejemplo anterior se proporcionan algunos datos concretos, en muchos escenarios de aplicación y casos de uso se han observado casos de uso en los que, si la aplicación emite varias medidas al mismo tiempo, es más eficaz almacenar los datos como registros de múltiples medidas. Además, los registros de múltiples medidas proporcionan la flexibilidad de los tipos de datos y almacenan varios valores más como contexto (por ejemplo, almacenan solicitudes y marcas de tiempo adicionalesIDs, que se analizarán más adelante).

Tenga en cuenta que un registro de múltiples medidas también puede modelar medidas dispersas, como en el ejemplo anterior, para registros de una sola medida: puede usar measure_name para almacenar el nombre de la medida y usar un nombre de atributo genérico de múltiples medidas, como value_double para almacenar `DOUBLE` medidas, value_bigint para almacenar `BIGINT` medidas, value_timestamp para almacenar `TIMESTAMP` valores adicionales, etc.

Dimensiones y medidas

Una tabla en Timestream LiveAnalytics le permite almacenar dimensiones (identificando los atributos del dispositivo o los datos que está almacenando) y medidas (las métricas/valores que está rastreando). Consulte Timestream para ver LiveAnalytics los conceptos para obtener más información. A medida que modele su aplicación en Timestream LiveAnalytics, la forma en que mapee sus datos en dimensiones y medidas afectará a la latencia de ingesta y consulta. Las siguientes son pautas sobre cómo modelar los datos como dimensiones y medidas que puede aplicar a su caso de uso.

Elección de dimensiones

Los datos que identifican la fuente que envía los datos de series temporales se adaptan perfectamente a las dimensiones, que son atributos que no cambian con el tiempo. Por ejemplo, si tiene un servidor que emite métricas, los atributos que lo identifican, como el nombre de host, la región, el rack o la zona de disponibilidad, son aptos para las dimensiones. Del mismo modo, para un dispositivo de IoT con múltiples sensores que informan datos de series temporales, la identificación del dispositivo, la identificación del sensor, etc., son candidatas para las dimensiones.

Si escribe los datos como registros de varias medidas, las dimensiones y los atributos de varias medidas aparecen como columnas en la tabla cuando realiza DESCRIBE o ejecuta una SELECT declaración en la tabla. Por lo tanto, al escribir las consultas, puede utilizar libremente las dimensiones y medidas de la misma consulta. Sin embargo, al crear el registro de escritura para ingerir datos, tenga en cuenta lo siguiente al elegir qué atributos se especifican como dimensiones y cuáles son valores de medida:

- Los nombres de las dimensiones, los valores, el nombre de la medida y la marca de tiempo identifican de forma exclusiva los datos de la serie temporal. Timestream for LiveAnalytics utiliza este identificador único para deduplicar automáticamente los datos. Es decir, si Timestream for LiveAnalytics recibe dos puntos de datos con los mismos valores: nombres de dimensiones, valores de dimensión, nombre de medida y marca de tiempo, si los valores tienen el mismo número de versión, Timestream for deduplica. LiveAnalytics Si la nueva solicitud de escritura tiene una versión inferior a la de los datos ya existentes en Timestream, se rechaza la solicitud

de escritura. LiveAnalytics Si la nueva solicitud de escritura tiene una versión superior, el nuevo valor sobrescribe el valor anterior. Por lo tanto, la forma en que elija los valores de las dimensiones afectará a este comportamiento de deduplicación.

- Los nombres y valores de las dimensiones no se pueden actualizar, el valor de la medida sí. Por lo tanto, es mejor modelar cualquier dato que pueda necesitar actualizaciones como valores de medida. Por ejemplo, si tiene una máquina en la fábrica cuyo color puede cambiar, puede modelar el color como un valor de medición, a menos que desee utilizar el color también como atributo de identificación necesario para la deduplicación. Es decir, los valores de medición se pueden usar para almacenar atributos que solo cambian lentamente con el tiempo.

Tenga en cuenta que una tabla en Timestream LiveAnalytics no limita el número de combinaciones únicas de nombres y valores de dimensiones. Por ejemplo, puede almacenar miles de millones de estas combinaciones de valores únicos en una tabla. Sin embargo, como verá en los ejemplos siguientes, la elección cuidadosa de las dimensiones y las medidas puede optimizar considerablemente la latencia de las solicitudes, especialmente en el caso de las consultas.

Único IDs en dimensiones

Si el escenario de su aplicación requiere que almacene un identificador único para cada punto de datos (por ejemplo, un ID de solicitud, un ID de transacción o un ID de correlación), modelar el atributo de ID como un valor de medida permitirá mejorar considerablemente la latencia de las consultas. Al modelar los datos con registros de varias medidas, el ID aparece en la misma fila en el contexto del resto de los datos de dimensiones y series temporales, para que las consultas puedan seguir utilizándolos de forma eficaz. Por ejemplo, si consideramos un caso de DevOps uso en el que cada punto de datos emitido por un servidor tiene un atributo de ID de solicitud único, modelar el ID de solicitud como un valor de medida da como resultado una latencia de consulta hasta 4 veces menor en distintos tipos de consulta, en lugar de modelar el ID de solicitud único como una dimensión.

Puede utilizar una analogía similar para los atributos que no son totalmente únicos para cada punto de datos, pero que tienen cientos de miles o millones de valores únicos. Puede modelar esos atributos como dimensiones o valores de medida. Debería modelarlo como una dimensión si los valores son necesarios para la deduplicación en la ruta de escritura, como se ha explicado anteriormente, o si lo utiliza con frecuencia como predicado (por ejemplo, en la `WHERE` cláusula con un predicado de igualdad sobre un valor de ese atributo, por ejemplo, si su aplicación rastrea millones de dispositivos) en sus consultas. `device_id = 'abcde'`

Gran variedad de tipos de datos con registros de múltiples medidas

Los registros de múltiples medidas le proporcionan la flexibilidad necesaria para modelar sus datos de forma eficaz. Los datos que se almacenan en un registro de varias medidas aparecen en la tabla como columnas similares a las dimensiones, lo que proporciona la misma facilidad a la hora de consultar las dimensiones y los valores de las medidas. Ha visto algunos de estos patrones en los ejemplos descritos anteriormente. A continuación, encontrará patrones adicionales para utilizar de forma eficaz los registros de múltiples medidas para adaptarse a los casos de uso de su aplicación.

Los registros de medidas múltiples admiten los atributos de los tipos de datos `DOUBLE`, `BIGINT`, `VARCHAR`, `BOOLEAN`, y. `TIMESTAMP` Por lo tanto, se adaptan naturalmente a diferentes tipos de atributos:

- Información de ubicación: por ejemplo, si desea realizar un seguimiento de la ubicación (expresada como latitud y longitud), modelarla como un atributo de múltiples medidas reducirá la latencia de las consultas en comparación con almacenarlas como `VARCHAR` dimensiones, especialmente si tiene predicados sobre la latitud y las longitudes.
- Varias marcas de tiempo en un registro: si el escenario de su aplicación requiere que realice un seguimiento de varias marcas de tiempo para un registro de serie temporal, puede modelarlas como atributos adicionales en el registro de medidas múltiples. Este patrón se puede usar para almacenar datos con marcas de tiempo futuras o pasadas. Tenga en cuenta que todos los registros seguirán utilizando la marca de tiempo de la columna de tiempo para particionar, indexar e identificar de forma única un registro.

En concreto, si tiene datos numéricos o marcas de tiempo en las que se utilizan predicados en la consulta, modelar esos atributos como atributos de múltiples medidas y no como dimensiones reducirá la latencia de la consulta. Esto se debe a que, al modelar dichos datos con los tipos de datos enriquecidos compatibles con los registros de múltiples medidas, puede expresar los predicados mediante tipos de datos nativos en lugar de convertir los valores VARCHAR a otro tipo de datos si ha modelado dichos datos como dimensiones.

Uso del nombre de la medida con registros de varias medidas

Las tablas de Timestream LiveAnalytics admiten un atributo (o columna) especial llamado nombre de medida. Debe especificar un valor para este atributo para cada registro para el que escriba en Timestream. LiveAnalytics En el caso de los registros de una sola medida, es normal utilizar el nombre de la métrica (por ejemplo, CPU, memoria para las métricas del servidor, o temperatura, presión, para las métricas de los sensores). Cuando se utilizan registros de múltiples medidas, dado que los atributos de un registro de múltiples medidas tienen nombres (y estos nombres se convierten

en nombres de columnas en la tabla), CPU, memoria o temperatura, la presión puede convertirse en nombres de atributos de múltiples medidas. Por lo tanto, una pregunta natural es cómo utilizar eficazmente el nombre de la medida.

Timestream for LiveAnalytics utiliza los valores del atributo del nombre de la medida para particionar e indexar los datos. Por lo tanto, si una tabla tiene varios nombres de medidas diferentes y si las consultas utilizan esos valores como predicados de consulta, Timestream for LiveAnalytics puede utilizar su particionamiento e indexación personalizados para eliminar los datos que no son relevantes para las consultas. Por ejemplo, si la tabla tiene nombres de medidas de CPU y memoria, y la consulta tiene un predicado`WHERE measure_name = 'cpu'`, Timestream for LiveAnalytics puede recortar eficazmente los datos de los nombres de medidas que no son relevantes para la consulta, por ejemplo, las filas con memoria de nombres de medidas en este ejemplo. Esta reducción se aplica incluso cuando se utilizan nombres de medidas con registros de varias medidas. Puede utilizar el atributo de nombre de medida de forma eficaz como atributo de partición de una tabla. El nombre de la medida junto con los nombres y valores de las dimensiones y el tiempo se utilizan para dividir los datos en una tabla Timestream for. LiveAnalytics Tenga en cuenta los límites de la cantidad de nombres de medida únicos permitidos en una tabla de flujo temporal. LiveAnalytics Tenga en cuenta también que el nombre de una medida también está asociado a un tipo de datos de valor de medida; por ejemplo, un nombre de medida individual solo puede asociarse a un tipo de valor de medida. Ese tipo puede ser uno de los `DOUBLE` siguientes: `BIGINT``BOOLEAN`,`VARCHAR`, y`MULTI`. Los registros de varias medidas almacenados con un nombre de medida tendrán el tipo de datos como`MULTI`. Como un único registro de múltiples medidas puede almacenar múltiples métricas con diferentes tipos de datos (`DOUBLE`,`BIGINT`, `VARCHAR``BOOLEAN`, y`TIMESTAMP`), puede asociar datos de diferentes tipos en un registro de múltiples medidas.

En las siguientes secciones se describen algunos ejemplos diferentes de cómo se puede utilizar eficazmente el atributo de nombre de la medida para agrupar distintos tipos de datos en la misma tabla.

Sensores de IoT que reportan calidad y valor

Supongamos que tiene una aplicación que monitorea los datos de los sensores de IoT. Cada sensor registra diferentes medidas, como la temperatura o la presión. Además de los valores reales, los sensores también indican la calidad de las mediciones, que es una medida de la precisión de la lectura y una unidad de medición. Como la calidad, la unidad y el valor se emiten juntos, se pueden modelar como registros de múltiples medidas, como se muestra en los siguientes datos de ejemplo, donde device_id es una dimensión, la calidad, el valor y la unidad son atributos de medidas múltiples:

device_id	measure_name	Tiempo	Calidad	Valor	Unidad
sensor-sea478	temperature	2021-12-01 19:22:32	92	35	c
sensor-sea478	temperature	2021-12-01 18:07:51	93	34	c
sensor-sea478	pressure	2021-12-01 19:05:30	98	31	psi
sensor-sea478	pressure	2021-12-01 19:00:01	24	132	psi

Este enfoque le permite combinar las ventajas de los registros de múltiples medidas con la partición y depuración de los datos mediante los valores del nombre de la medida. Si las consultas hacen referencia a una sola medida, por ejemplo, a la temperatura, puede incluir un predicado del nombre de la medida en la consulta. El siguiente es un ejemplo de una consulta de este tipo, que también proyecta la unidad para las mediciones cuya calidad es superior a 90.

```
SELECT device_id, time, value AS temperature, unit
FROM db.table
WHERE time > ago(1h)
    AND measure_name = 'temperature'
    AND quality > 90
```

El uso del predicado measure_name en la consulta permite a Timestream eliminar eficazmente las particiones y los datos que no son relevantes LiveAnalytics para la consulta, lo que mejora la latencia de la consulta.

También es posible almacenar todas las métricas en el mismo registro de múltiples medidas si todas las métricas se emiten en la misma fecha o si se consultan varias métricas juntas en la misma consulta. Por ejemplo, puede crear un registro de múltiples medidas con los atributos temperature_quality, temperature_value, temperature_unit, pressure_quality, pressure_value, pressure_unit, etc. Muchos de los puntos discutidos anteriormente sobre el modelado de datos mediante registros de una sola medida frente a registros de múltiples medidas se aplican a la decisión de cómo modelar los datos. Tenga en cuenta los patrones de acceso a las consultas y la

forma en que se generan los datos para elegir un modelo que optimice los costes, la latencia de ingesta y consulta y la facilidad de redacción de las consultas.

Diferentes tipos de métricas en la misma tabla

Otro caso de uso en el que puede combinar registros de varias medidas con valores de nombres de medidas es modelar diferentes tipos de datos que se emiten de forma independiente desde el mismo dispositivo. Considere el caso práctico de la DevOps monitorización: los servidores emiten dos tipos de datos: métricas emitidas con regularidad y eventos irregulares. Un ejemplo de este enfoque es el esquema descrito en el generador de datos que modela un caso de DevOps uso. En este caso, puede almacenar los diferentes tipos de datos emitidos desde el mismo servidor en la misma tabla utilizando diferentes nombres de medida. Por ejemplo, todas las métricas que se emiten al mismo tiempo y en un instante se almacenan con las métricas de los nombres de las medidas. Todos los eventos que se emiten en un instante de tiempo diferente al de las métricas se almacenan con los eventos con nombres de medidas. El esquema de medidas de la tabla (por ejemplo, el resultado de la `SHOW MEASURES` consulta) es:

measure_name	data_type	Dimensiones
events	múltiple	[{"data_type» :"varchar», "dimension_name» :"availability_zone "}, {" data_type» :"varchar», "dimension_name» :"microservice_name "}, {" data_type» :"varchar», "dimension_name» :"instance_name "}, {" data_type» :"varchar», «dimension_name» :"nombre_proceso "}, {" data_type» :"varchar», "dimension_name» :"jdk_version "}, {" data_type» :"varchar», "dimension_name» :"celda "}, {" data_type» :"varchar», "dimension_name» :"region "}, {" data_type» :« varchar», «dimension_name» :"silo "}]

measure_name	data_type	Dimensiones
métricas	múltiple	[{"data_type» :"varchar», "dimension_name» :"availability_zone "}, {" data_type» :"varchar», "dimension_name» :"microservice_name "}, {" data_type» :"varchar», "dimension_name» :"instance_name "}, {" data_type» :"varchar», «dimension_name» :"os_version "}, {" data_type» :"varchar», "dimension_name» :"cell "}, {" data_type» :"varchar», "dimension_name» :"region "}, {" data_type» :"varchar», "dimension_name» :"silo "}, {" data_type» :"varchar», «nombre_dimensión» :"tipo_instancia "}]

En este caso, puede ver que los eventos y las métricas también tienen diferentes conjuntos de dimensiones, donde los eventos tienen diferentes dimensiones jdk_version y process_name, mientras que las métricas tienen dimensiones instance_type y os_version.

El uso de diferentes nombres de medida te permite escribir consultas con predicados para obtener solo las métricas. `WHERE measure_name = 'metrics'` Además, tener todos los datos emitidos desde la misma instancia en la misma tabla implica que también puedes escribir una consulta más sencilla con el predicado instance_name para obtener todos los datos de esa instancia. Por ejemplo, un predicado del formulario `WHERE instance_name = ‘instance-1234’` sin el predicado measure_name devolverá todos los datos de una instancia de servidor específica.

Recomendaciones para particionar registros de múltiples medidas

> **Important**
>
> ¡Esta sección está obsoleta!
> Estas recomendaciones están desactualizadas. El particionamiento ahora se controla mejor mediante claves de partición definidas por el cliente.

Hemos observado que hay un número creciente de cargas de trabajo en el ecosistema de series temporales que requieren ingerir y almacenar enormes cantidades de datos y, al mismo tiempo, requieren respuestas a consultas de baja latencia cuando se accede a los datos mediante un conjunto de valores de dimensión de alta cardinalidad.

Debido a estas características, las recomendaciones de esta sección resultarán útiles para las cargas de trabajo de los clientes que tengan lo siguiente.

- Ha adoptado o desea adoptar registros con varias medidas.
- Espere recibir un gran volumen de datos en el sistema que se almacenarán durante períodos prolongados.
- Requieren tiempos de respuesta de baja latencia para sus principales patrones de acceso (consulta).
- Tenga en cuenta que los patrones de consultas más importantes implican algún tipo de condición de filtrado en el predicado. Esta condición de filtrado se basa en una dimensión de alta cardinalidad. Por ejemplo, considere los eventos o agregaciones por UserId DeviceId, ID de servidor, nombre de host, etc.

En estos casos, un nombre único para todas las medidas de varias medidas no servirá de nada, ya que nuestro motor utiliza un nombre de varias medidas para particionar los datos y tener un único valor limita la ventaja de partición que se obtiene. La partición de estos registros se basa principalmente en dos dimensiones. Supongamos que el tiempo está en el eje x y es un hash de nombres de dimensiones y `measure_name` en el eje y. `measure_name`En estos casos, funciona casi como una clave de partición.

Nuestra recomendación es la siguiente.

- Al modelar tus datos para casos de uso como el que mencionamos, usa un patrón `measure_name` que sea derivado directamente de tu patrón de acceso a las consultas principales. Por ejemplo:

  - Su caso de uso requiere realizar un seguimiento del rendimiento de las aplicaciones y de la QoE desde el punto de vista del usuario final. Esto también podría ser el seguimiento de las mediciones de un solo servidor o dispositivo de IoT.
  - Si está realizando consultas y filtrando por UserId, necesitará encontrar, en el momento de la ingesta, la mejor manera de asociarse`measure_name`. UserId
  - Como una tabla de múltiples medidas solo puede contener 8192 nombres de medidas diferentes, sea cual sea la fórmula que se adopte, no debería generar más de 8192 valores diferentes.
- Un enfoque que hemos aplicado con éxito a los valores de cadena es aplicar un algoritmo de hash al valor de cadena. A continuación, realice la operación de módulo con el valor absoluto del resultado del hash y 8192.

```
measure_name = getMeasureName(UserId)
int getMeasureName(value) {
    hash_value =  abs(hash(value))
    return hash_value % 8192
}
```

- También hemos añadido este signo `abs()` para eliminar la posibilidad de que los valores oscilen entre -8192 y 8192. Esto debe realizarse antes de la operación del módulo.
- Con este método, las consultas pueden ejecutarse en una fracción del tiempo que tardarían en ejecutarse en un modelo de datos sin particiones.
- Cuando consultes los datos, asegúrate de incluir una condición de filtrado en el predicado que utilice el valor recién derivado de measure_name. Por ejemplo:
  -
```
SELECT * FROM your_database.your_table
WHERE host_name = 'Host-1235' time BETWEEN '2022-09-01'
    AND '2022-09-18'
    AND measure_name = (SELECT
 cast(abs(from_big_endian_64(xxhash64(CAST('HOST-1235' AS varbinary))))%8192 AS
 varchar))
```
  - Esto minimizará el número total de particiones escaneadas para obtener datos que, con el tiempo, se traducirán en consultas más rápidas.

Tenga en cuenta que si desea aprovechar las ventajas de este esquema de particiones, el hash debe calcularse en el lado del cliente y pasarse a Timestream for LiveAnalytics como un valor

estático para el motor de consultas. El ejemplo anterior proporciona una forma de validar que el motor pueda resolver el hash generado cuando sea necesario.

hora	host_name	location	tipo_servidor	cpu_usage	memoria_disponible	cpu_temp
2022-09-07 21:48:44.000000000	host-1235	us-east1	5,8 xl	55	16.2	78
R2022-09-07 21:48:44.000000000	host-3587	us-west1	5,8 xl	62	18.1	81
2022-09-07 21:48:45.000 000000	host-258743	eu-central	5,8 xl	88	9.4	91
2022-09-07 21:48:45.000000000	host-35654	us-east2	5,8 xl	29	24	54
R2022-09-07 21:48:45.000000000	host-254	us-west1	5,8 xl	44	32	48

Para generar los asociados `measure_name` siguiendo nuestra recomendación, hay dos rutas que dependen de tu patrón de ingesta.

1. Para la ingesta por lotes de datos históricos: puedes añadir la transformación al código de escritura si vas a utilizar tu propio código para el proceso por lotes.

   Basado en el ejemplo anterior.

```
        List<String> hosts = new ArrayList<>();
```

```
        hosts.add("host-1235");
        hosts.add("host-3587");
        hosts.add("host-258743");
        hosts.add("host-35654");
        hosts.add("host-254");

        for (String h: hosts){
            ByteBuffer buf2 = ByteBuffer.wrap(h.getBytes());
            partition = abs(hasher.hash(buf2, 0L)) % 8192;
            System.out.println(h + " - " + partition);

        }
```

Salida

```
host-1235 - 6445
host-3587 - 6399
host-258743 - 640
host-35654 - 2093
host-254 - 7051
```

Conjunto de datos resultante

hora	host_name	location	measure_name	tipo_servidor	cpu_usage	memoria_disponible	cpu_temp
2022-09-07 21:48:44.000000000	host-1235	us-east1	6445	5,8 xl	55	16.2	78
R2022-09-07 21:48:44.000000000	host-3587	us-west1	6399	5,8 xl	62	18.1	81

hora	host_name	location	measure_name	tipo_servidor	cpu_usage	memoria_disponible	cpu_temp
2022-09-07 21:48:45.000000000	host-258743	eu-central	640	5,8 xl	88	9.4	91
2022-09-07 21:48:45.000000000	host-356554	us-east2	2093	5,8 xl	29	24	54
R2022-09-07 21:48:45.000000000	host-254	us-west1	7051	5,8 xl	44	32	48

2. Para la ingesta en tiempo real: es necesario generar los datos durante el `measure_name` vuelo a medida que van llegando los datos.

En ambos casos, te recomendamos que pruebes tu algoritmo de generación de hash en ambos extremos (ingesta y consulta) para asegurarte de que estás obteniendo los mismos resultados.

Estos son algunos ejemplos de código en base a los que generar el valor hash. `host_name`

Example Python

```
>>> import xxhash
>>> from bitstring import BitArray
>>> b=xxhash.xxh64('HOST-ID-1235').digest()
>>> BitArray(b).int % 8192
### 3195
```

Example Go

```
package main
```

```
import (
    "bytes"
    "fmt"
    "github.com/cespare/xxhash"
)

func main() {
    buf := bytes.NewBufferString("HOST-ID-1235")
    x := xxhash.New()
    x.Write(buf.Bytes())
    // convert unsigned integer to signed integer before taking mod
    fmt.Printf("%f\n", abs(int64(x.Sum64())) % 8192)
}

func abs(x int64) int64 {
    if (x < 0) {
        return -x
    }
    return x
}
```

Example Java

```
import java.nio.ByteBuffer;

import net.jpountz.xxhash.XXHash64;

public class test {
    public static void main(String[] args) {
        XXHash64 hasher = net.jpountz.xxhash.XXHashFactory.fastestInstance().hash64();

        String host = "HOST-ID-1235";
        ByteBuffer buf = ByteBuffer.wrap(host.getBytes());

        Long result = Math.abs(hasher.hash(buf, 0L));
        Long partition = result % 8192;

        System.out.println(result);
        System.out.println(partition);
    }
}
```

Example dependencia en Maven

```
        <dependency>
            <groupId>net.jpountz.lz4</groupId>
            <artifactId>lz4</artifactId>
            <version>1.3.0</version>
        </dependency>
```

Seguridad

- Para tener acceso continuo a Timestream LiveAnalytics, asegúrese de que las claves de cifrado estén protegidas y no se revoquen ni hagan inaccesibles.
- APISupervise los registros de acceso desde. AWS CloudTrail Audite y revoque cualquier patrón de acceso anómalo de usuarios no autorizados.
- Siga las pautas adicionales que se describen en. Prácticas recomendadas de seguridad para Amazon Timestream para LiveAnalytics

Configuración de Amazon Timestream para LiveAnalytics

Configure el período de retención de datos para el almacén de memoria y el almacén magnético para que se adapte a los requisitos de procesamiento de datos, almacenamiento, rendimiento de consultas y costos.

- Configure la retención de datos del almacén de memoria para que se adapte a los requisitos de la aplicación para procesar los datos que llegan tarde. Los datos que llegan tarde son datos entrantes con una marca de tiempo anterior a la hora actual. Se emite desde recursos que agrupan eventos durante un período de tiempo antes de enviar los datos a Timestream LiveAnalytics, y también desde recursos con conectividad intermitente, por ejemplo, un sensor de IoT que está en línea de forma intermitente.
- Si prevé que los datos que llegan tarde lleguen ocasionalmente con marcas de tiempo antes de la retención en el almacén de memoria, debe habilitar la escritura magnética en su tabla. Una vez que hayas configurado una tabla, MagneticStoreWritesProperties esta aceptará los datos con marca de tiempo anteriores a la retención del almacén de memoria, pero dentro del período de retención del almacén magnético. EnableMagneticStoreWrites
- Tenga en cuenta las características de las consultas que planea ejecutar en Timestream, LiveAnalytics como los tipos de consultas, la frecuencia, el intervalo de tiempo y los requisitos de rendimiento. Esto se debe a que el almacén de memoria y el almacén magnético están

optimizados para diferentes escenarios. El almacén de memoria está optimizado para point-in-time consultas rápidas que procesan pequeñas cantidades de datos recientes enviados a Timestream. LiveAnalytics El almacén magnético está optimizado para consultas analíticas rápidas que procesan volúmenes medianos o grandes de datos enviados a Timestream. LiveAnalytics

- El período de retención de datos también debe estar influenciado por los requisitos de costo de su sistema.

  Por ejemplo, imagine un escenario en el que el límite de retraso de los datos para su solicitud sea de 2 horas y sus aplicaciones envíen muchas consultas que procesen datos de un día, una semana o un mes. En ese caso, tal vez desee configurar un período de retención más corto para el almacén de memoria (de 2 a 3 horas) y permitir que fluyan más datos al almacén magnético, dado que el almacén magnético está optimizado para consultas analíticas rápidas.

Comprenda el impacto de aumentar o disminuir el período de retención de datos del almacén de memoria y del almacén magnético de una tabla existente.

- Al reducir el período de retención del almacén de memoria, los datos se mueven del almacén de memoria al almacén magnético y esta transferencia de datos es permanente. Timestream for LiveAnalytics no recupera los datos del almacén magnético para llenarlos en el almacén de memoria. Al reducir el período de retención del almacén magnético, los datos se eliminan del sistema y la eliminación de datos es permanente.
- Al aumentar el período de retención del almacén de memoria o del almacén magnético, el cambio se aplicará a los datos que se envíen a Timestream a LiveAnalytics partir de ese momento. Timestream for LiveAnalytics no recupera los datos del almacén magnético para rellenar el almacén de memoria. Por ejemplo, si el período de retención del almacén de memoria se estableció inicialmente en 2 horas y luego se aumentó a 24 horas, el almacén de memoria tardará 22 horas en contener 24 horas de datos.

Escribe

- Asegúrese de que la marca de tiempo de los datos entrantes no sea anterior a la retención de datos configurada para el almacén de memoria ni posterior al período de ingesta futuro definido en. Cuotas Si se envían datos con una marca de tiempo fuera de estos límites, Timestream los rechazará, a LiveAnalytics menos que habilites la escritura magnética en la tabla. Si habilita la escritura en almacenamiento magnético, asegúrese de que la marca de tiempo de los datos entrantes no sea anterior a la retención de datos configurada para el almacenamiento magnético.

- Si prevé que los datos lleguen tarde, active la grabación magnética en la tabla. Esto permitirá la ingesta de datos con marcas de tiempo que estén fuera del período de retención del almacén de memoria, pero que aún se encuentren dentro del período de retención del almacenamiento magnético. Para configurarlo, actualice la `EnableMagneticStoreWrites` marca de la tabla`MagneticStoreWritesProperties`. Esta propiedad es falsa de forma predeterminada. Tenga en cuenta que las escrituras en el almacén magnético no estarán disponibles inmediatamente para consultarlas. Estarán disponibles en un plazo de 6 horas.
- Dirija las cargas de trabajo de alto rendimiento al almacén de memoria asegurándose de que las marcas de tiempo de los datos ingeridos se encuentren dentro de los límites de retención del almacén de memoria. Las escrituras en el almacén magnético están limitadas a un número máximo de particiones del almacén magnético activas que pueden recibir ingestas simultáneas para una base de datos. Puedes ver esta `ActiveMagneticStorePartitions` métrica en. CloudWatch Para reducir las particiones de almacenamiento magnético activas, trate de reducir el número de series y la duración del tiempo en que se ingieren simultáneamente los depósitos magnéticos.
- Al enviar datos a Timestream LiveAnalytics, agrupe varios registros en una sola solicitud para optimizar el rendimiento de la ingesta de datos.
  - Resulta beneficioso agrupar los registros de la misma serie temporal y los registros con el mismo nombre de medida.
  - Batch tantos registros como sea posible en una sola solicitud, siempre y cuando las solicitudes estén dentro de los límites de servicio definidos enCuotas.
  - Utilice atributos comunes siempre que sea posible para reducir los costes de transferencia e ingesta de datos. Para obtener más información, consulte WriteRecords API.
- Si encuentra errores parciales en el lado del cliente al escribir datos en Timestream LiveAnalytics, puede volver a enviar el lote de registros que no se pudo ingerir una vez que haya abordado la causa del rechazo.
- Los datos ordenados por marcas de tiempo tienen un mejor rendimiento de escritura.
- Amazon Timestream LiveAnalytics for está diseñado para adaptarse automáticamente a las necesidades de su aplicación. Cuando Timestream for LiveAnalytics Notes se dispare en las solicitudes de escritura de su aplicación, es posible que esta experimente algún nivel de limitación del almacenamiento de memoria inicial. Si su aplicación experimenta una reducción del almacenamiento de memoria, continúe enviando datos a Timestream a la misma velocidad (o a una mayor) LiveAnalytics velocidad para permitir que Timestream se escale automáticamente para satisfacer las necesidades de su LiveAnalytics aplicación. Si observa una reducción del almacenamiento magnético, debe reducir la velocidad de ingestión del

almacenamiento magnético hasta que disminuya el número de acumulaciones magnéticas. `ActiveMagneticStorePartitions`

Carga por lotes

Las mejores prácticas para la carga de lotes se describen enMejores prácticas de carga por lotes.

Consultas

A continuación se sugieren las prácticas recomendadas para las consultas con Amazon LiveAnalytics Timestream for.

- Incluya solo los nombres de medidas y dimensiones esenciales para realizar la consulta. Añadir columnas superfluas aumentará los escaneos de datos, lo que repercutirá en el rendimiento de las consultas.
- Antes de implementar la consulta en producción, le recomendamos que revise la información sobre las consultas para asegurarse de que el ajuste espacial y temporal es óptimo. Para obtener más información, consulte Uso de la información sobre consultas para optimizar las consultas en Amazon Timestream.
- Siempre que sea posible, transfiera el cálculo de datos a Timestream para LiveAnalytics utilizar las funciones agregadas y escalares integradas en la cláusula y la SELECT cláusula, según proceda, a fin de mejorar el WHERE rendimiento de las consultas y reducir los costes. Consulte SELECT y Funciones de agregación.
- Siempre que sea posible, utilice funciones aproximadas. Por ejemplo, utilice APPROX _ DISTINCT en lugar de COUNT (DISTINCTcolumn_name) para optimizar el rendimiento de las consultas y reducir el coste de las consultas. Consulte Funciones de agregación.
- Use una CASE expresión para realizar agregaciones complejas en lugar de seleccionar elementos de la misma tabla varias veces. Consulte La declaración CASE.
- Siempre que sea posible, incluya un intervalo de tiempo en la WHERE cláusula de la consulta. Esto optimiza el rendimiento y los costes de las consultas. Por ejemplo, si solo necesita los datos de la última hora en su conjunto de datos, incluya un predicado de tiempo como tiempo > hace 1 hora (1 h). Consulte SELECT y Intervalo y duración.
- Cuando una consulta acceda a un subconjunto de medidas de una tabla, incluye siempre los nombres de las medidas en la WHERE cláusula de la consulta.

- Siempre que sea posible, utilice el operador de igualdad al comparar dimensiones y medidas en la WHERE cláusula de una consulta. Un predicado de igualdad en los nombres de las dimensiones y las medidas permite mejorar el rendimiento de las consultas y reducir los costes de las consultas.
- Siempre que sea posible, evite utilizar las funciones de la WHERE cláusula para optimizar los costes.
- Absténgase de usar LIKE la cláusula varias veces. En su lugar, utilice expresiones regulares cuando filtre varios valores en una columna de cadena. Consulte Regular expression functions (Funciones de expresión regular).
- Utilice únicamente las columnas necesarias en la cláusula GROUP BY de una consulta.
- Si el resultado de la consulta debe estar en un orden específico, especifique explícitamente ese orden en la cláusula ORDER BY de la consulta más externa. Si el resultado de la consulta no requiere orden, evite usar una cláusula ORDER BY para mejorar el rendimiento de la consulta.
- Usa una LIMIT cláusula si solo necesitas las N primeras filas de la consulta.
- Si utiliza una cláusula ORDER BY para ver los N valores superiores o inferiores, utilice una LIMIT cláusula para reducir los costes de la consulta.
- Usa el token de paginación de la respuesta devuelta para recuperar los resultados de la consulta. Para obtener más información, consulte Query.
- Si ha empezado a ejecutar una consulta y se da cuenta de que la consulta no devolverá los resultados que busca, cancele la consulta para ahorrar costes. Para obtener más información, consulte CancelQuery.
- Si su aplicación sufre limitaciones, continúe enviando datos a Amazon Timestream a la misma velocidad para permitir que Amazon Timestream LiveAnalytics for se escale automáticamente LiveAnalytics para satisfacer las necesidades de rendimiento de consultas de su aplicación.
- Si los requisitos de simultaneidad de consultas de sus aplicaciones superan los límites predeterminados de Timestream, el límite de consultas aumentará. LiveAnalytics AWS Support

Consultas programadas

Las consultas programadas le ayudan a optimizar sus paneles al calcular previamente algunas estadísticas agregadas de toda la flota. Por lo tanto, una pregunta natural es cómo analizar su caso de uso e identificar qué resultados debe precalcular y cómo utilizar estos resultados almacenados en una tabla derivada para crear su panel de control. El primer paso de este proceso consiste en identificar qué paneles se deben precalcular. A continuación se presentan algunas pautas de alto nivel:

- Tenga en cuenta los bytes escaneados por las consultas que se utilizan para rellenar los paneles, la frecuencia con la que se recargan los paneles y el número de usuarios simultáneos que cargarían estos paneles. Debería empezar por los paneles que se cargan con más frecuencia y por escanear cantidades significativas de datos. Los dos primeros paneles del ejemplo del panel agregado, así como el panel agregado del ejemplo detallado, son buenos ejemplos de este tipo de paneles.
- Tenga en cuenta qué cálculos se utilizan repetidamente. Si bien es posible crear una consulta programada para cada panel y cada valor de variable utilizado en el panel, puede optimizar considerablemente sus costes y el número de consultas programadas si busca formas de utilizar un solo cálculo para calcular previamente los datos necesarios para varios paneles.
- Tenga en cuenta la frecuencia de las consultas programadas para actualizar los resultados materializados en la tabla derivada. Debería analizar la frecuencia con la que se actualiza un cuadro de mando en comparación con el intervalo de tiempo que se consulta en un cuadro de mando y el intervalo de tiempo utilizado en el cálculo previo y en los paneles de los cuadros de mando. Por ejemplo, si un panel que muestra los agregados por hora de los últimos días solo se actualiza una vez cada pocas horas, puede configurar las consultas programadas para que solo se actualicen una vez cada 30 minutos o una hora. Por otro lado, si tiene un panel que traza los agregados por minuto y se actualiza aproximadamente cada minuto, querrá que las consultas programadas actualicen los resultados cada minuto o cada pocos minutos.
- Considera qué patrones de consulta se pueden optimizar aún más (tanto desde el punto de vista del coste como de la latencia de las consultas) mediante consultas programadas. Por ejemplo, al calcular los valores de dimensión únicos que se utilizan con frecuencia como variables en los paneles, o al devolver el último punto de datos emitido por un sensor o el primer punto de datos emitido por un sensor después de una fecha determinada, etc. Algunos de estos patrones de ejemplo se analizan en esta guía.

Las consideraciones anteriores repercutirán considerablemente en sus ahorros al mover el panel de control para consultar las tablas derivadas, en la actualización de los datos de los paneles y en el coste incurrido por las consultas programadas.

Aplicaciones cliente e integraciones compatibles

Ejecute su aplicación cliente desde la misma región que Timestream para LiveAnalytics reducir las latencias de la red y los costes de transferencia de datos. Para obtener más información sobre cómo trabajar con otros servicios, consulte. Trabajar con otros servicios de Los siguientes son otros vínculos útiles.

- Mejores prácticas para AWS el desarrollo con el AWS SDK for Java
- Mejores prácticas para trabajar con AWS Lambda funciones
- Prácticas recomendadas para Amazon Managed Service para Apache Flink
- Mejores prácticas para crear cuadros de mando en Grafana

General

- Asegúrese de seguir el marco de The AWS Well-Architected Framework cuando utilice Timestream para. LiveAnalytics Este documento técnico proporciona orientación sobre las mejores prácticas en materia de excelencia operativa, seguridad, confiabilidad, eficiencia del rendimiento y optimización de costos.

Medición y optimización de costes

Con Amazon Timestream LiveAnalytics for, solo pagas por lo que usas. Transmite el tiempo de los LiveAnalytics contadores por separado para las escrituras, los datos almacenados y los datos escaneados mediante consultas. El precio de cada dimensión de medición se especifica en la página de precios. Puedes calcular tu factura mensual con la calculadora de precios de Amazon Timestream. LiveAnalytics

En esta sección se describe cómo funciona la medición para las escrituras, el almacenamiento y las consultas en Timestream for. LiveAnalytics También se proporcionan ejemplos de escenarios y cálculos. Además, se incluye una lista de las mejores prácticas para la optimización de costes. Puede seleccionar uno de los siguientes temas:

Temas

- Escribe
- Almacenamiento
- Consultas
- Optimización de costos
- Monitorización con Amazon CloudWatch

Escribe

El tamaño de escritura de cada evento de serie temporal se calcula como la suma del tamaño de la marca de tiempo y uno o más nombres de dimensiones, valores de dimensiones, nombres de medidas y valores de medidas. El tamaño de la marca de tiempo es de 8 bytes. El tamaño de los nombres de las dimensiones, los valores de las dimensiones y los nombres de las medidas corresponde a la longitud de los UTF 8 bytes codificados de la cadena que representa cada nombre de dimensión, valor de dimensión y nombre de medida. El tamaño del valor de la medida depende del tipo de datos. Tiene 1 byte para el tipo de datos booleano, 8 bytes para bigint y double y la longitud de los UTF -8 bytes codificados para las cadenas. Cada escritura se cuenta en unidades de 1 KiB.

A continuación se proporcionan dos ejemplos de cálculos:

Temas

Calcular el tamaño de escritura de un evento de serie temporal

Considere un evento de serie temporal que represente la CPU utilización de una EC2 instancia, como se muestra a continuación:

Tiempo	región	az	vpc	Hostname	measure_name	measure_value::double
1602983435238563000	us-east-1	1d	vpc-1a2b3c4d	Host-24GJU	utilización de la CPU	35,0

El tamaño de escritura del evento de la serie temporal se puede calcular de la siguiente manera:

- tiempo = 8 bytes
- primera dimensión = 15 bytes (`region+us-east-1`)
- segunda dimensión = 4 bytes (`az+1d`)

- tercera dimensión = 15 bytes (`vpc+vpc-1a2b3c4d`)
- cuarta dimensión = 18 bytes (`hostname+host-24Gju`)
- nombre de la medida = 15 bytes (`cpu_utilization`)
- valor de la medida = 8 bytes

Tamaño de escritura del evento de la serie temporal = 83 bytes

Calcular el número de escrituras

Ahora consideremos 100 EC2 instancias, similares a la instancia descrita enCalcular el tamaño de escritura de un evento de serie temporal, que emiten métricas cada 5 segundos. El total de escrituras mensuales de las EC2 instancias variará en función del número de eventos de series temporales que existan por escritura y de si se utilizan atributos comunes al agrupar los eventos de series temporales. Se proporciona un ejemplo del cálculo del total de escrituras mensuales para cada uno de los siguientes escenarios:

Temas

- Un evento de serie temporal por escritura
- Procesamiento por lotes de eventos de series temporales en una escritura
- Procesamiento por lotes de eventos de series temporales y uso de atributos comunes en una escritura

Un evento de serie temporal por escritura

Si cada escritura contiene solo un evento de serie temporal, el total de escrituras mensuales se calcula de la siguiente manera:

- 100 eventos de series temporales = 100 escrituras cada 5 segundos
- x 12 escrituras/minuto = 1200 escrituras
- x 60 minutos/hora = 72.000 escrituras
- x 24 horas/día = 1.728.000 escrituras
- x 30 días/mes = 51.840.000 escrituras

Total de escrituras mensuales = 51.840.000

Procesamiento por lotes de eventos de series temporales en una escritura

Dado que cada escritura se mide en unidades de 1 KB, una escritura puede contener un lote de 12 eventos de series temporales (998 bytes) y el total de escrituras mensuales se calcula de la siguiente manera:

- 100 eventos de series temporales = 9 escrituras (12 eventos de series temporales por escritura) cada 5 segundos
- x 12 escrituras/minuto = 108 escrituras
- x 60 minutos/hora = 6.480 escrituras
- x 24 horas/día = 155 520 escrituras
- x 30 días/mes = 4.665.600 escrituras

Total de escrituras mensuales = 4.665.600

Procesamiento por lotes de eventos de series temporales y uso de atributos comunes en una escritura

Si la región, az, vpc y el nombre de la medida son comunes en 100 EC2 instancias, los valores comunes se pueden especificar solo una vez por escritura y se denominan atributos comunes. En este caso, el tamaño de los atributos comunes es de 52 bytes y el tamaño de los eventos de la serie temporal es de 27 bytes. Dado que cada escritura se mide en unidades de 1 KiB, una escritura puede contener 36 eventos de series temporales y atributos comunes, y el total de escrituras mensuales se calcula de la siguiente manera:

- 100 eventos de series temporales = 3 escrituras (36 eventos de series temporales por escritura) cada 5 segundos
- x 12 escrituras/minuto = 36 escrituras
- x 60 minutos/hora = 2.160 escrituras
- x 24 horas/día = 51.840 escrituras
- x 30 días/mes = 1.555.200 escrituras

Total de escrituras mensuales = 1.555.200

> **Note**
>
> Debido al uso del procesamiento por lotes, a los atributos comunes y al redondeo de las escrituras a unidades de 1 KB, el tamaño de almacenamiento de los eventos de series temporales puede ser diferente del tamaño de la escritura.

Almacenamiento

El tamaño de almacenamiento de cada evento de serie temporal en el almacén de memoria y en el almacén magnético se calcula como la suma del tamaño de la marca temporal, los nombres de las dimensiones, los nombres de las medidas y los valores de las medidas. El tamaño de la marca de tiempo es de 8 bytes. El tamaño de los nombres de las dimensiones, los valores de las dimensiones y los nombres de las medidas corresponde a la longitud de los UTF -8 bytes codificados de cada cadena que representa el nombre de la dimensión, el valor de la dimensión y el nombre de la medida. El tamaño del valor de la medida depende del tipo de datos. Tiene 1 byte para los tipos de datos booleanos, 8 bytes para bigint y double, y la longitud de los UTF -8 bytes codificados para las cadenas. Cada medida se almacena como un registro independiente en Amazon Timestream LiveAnalytics, es decir, si su evento de serie temporal tiene cuatro medidas, habrá cuatro registros para ese evento de serie temporal en el almacenamiento.

Si tenemos en cuenta el ejemplo del evento de serie temporal que representa la CPU utilización de una EC2 instancia (consulteCalcular el tamaño de escritura de un evento de serie temporal), el tamaño de almacenamiento del evento de serie temporal se calcula de la siguiente manera:

- tiempo = 8 bytes
- primera dimensión = 15 bytes (`region+us-east-1`)
- segunda dimensión = 4 bytes (`az+1d`)
- tercera dimensión = 15 bytes (`vpc+vpc-1a2b3c4d`)
- cuarta dimensión = 18 bytes (`hostname+host-24Gju`)
- nombre de la medida = 15 bytes (`cpu_utilization`)
- valor de la medida = 8 bytes

Tamaño de almacenamiento del evento de la serie temporal = 83 bytes

> **Note**
>
> El almacén de memoria se mide en GB por hora y el almacenamiento magnético se mide en GB por mes.

Consultas

Las consultas se cobran en función de la duración de las unidades de cómputo de Timestream (TCUs) utilizadas por la aplicación en TCU horas, tal como se especifica en la página de precios de Amazon Timestream. El motor de consultas Amazon Timestream LiveAnalytics for elimina los datos irrelevantes al procesar una consulta. Las consultas con proyecciones y predicados que incluyen rangos de tiempo, nombres de medidas o nombres de dimensiones permiten al motor de procesamiento de consultas reducir una cantidad significativa de datos y ayudan a reducir los costos de las consultas.

Optimización de costos

Para optimizar el coste de las escrituras, el almacenamiento y las consultas, utilice las siguientes prácticas recomendadas con Amazon LiveAnalytics Timestream para:

- Batch varios eventos de series temporales por escritura para reducir el número de solicitudes de escritura.
- Considere la posibilidad de utilizar registros de medidas múltiples, que le permiten escribir varias medidas de series temporales en una sola solicitud de escritura y almacenar los datos de forma más compacta. Esto reduce la cantidad de solicitudes de escritura, así como el costo de almacenamiento de datos y el costo de las consultas.
- Utilice atributos comunes con el procesamiento por lotes para agrupar más eventos de series temporales por escritura para reducir aún más el número de solicitudes de escritura.
- Configure la retención de datos del almacén de memoria para que se ajuste a los requisitos de la aplicación para procesar los datos que llegan tarde. Los datos que llegan tarde son datos entrantes con una marca de tiempo anterior a la hora actual y fuera del período de retención del almacén de memoria.
- Configure la retención de datos del almacén magnético para que se adapte a sus requisitos de almacenamiento de datos a largo plazo.
- Al escribir las consultas, incluya solo los nombres de medidas y dimensiones esenciales para la consulta. La adición de columnas superfluas aumentará los escaneos de datos y, por lo

tanto, también aumentará el costo de la consulta. Le recomendamos que revise los datos de las consultas para evaluar la eficacia de la poda de las dimensiones y medidas incluidas.

- Siempre que sea posible, incluye un intervalo de tiempo en la WHERE cláusula de tu consulta. Por ejemplo, si solo necesitas los datos de la última hora en tu conjunto de datos, incluye un predicado temporal como `time > ago(1h)`.
- Cuando una consulta acceda a un subconjunto de medidas de una tabla, incluya siempre los nombres de las medidas en la WHERE cláusula de la consulta.
- Si ha empezado a ejecutar una consulta y se da cuenta de que la consulta no devolverá los resultados que busca, cancele la consulta para ahorrar costes.

Monitorización con Amazon CloudWatch

Puedes monitorizar Timestream para usar LiveAnalytics Amazon CloudWatch, que recopila y procesa datos sin procesar de Timestream para LiveAnalytics convertirlos en métricas legibles. near-real-time Estas estadísticas se mantienen durante dos semanas, de forma que pueda acceder a información histórica y disponer de una mejor perspectiva sobre el desempeño de su aplicación web o servicio. De forma predeterminada, el Timestream de los datos LiveAnalytics métricos se envía automáticamente en períodos de 1 minuto o 15 minutos. CloudWatch Para obtener más información, consulta ¿Qué es Amazon CloudWatch? en la Guía del CloudWatch usuario de Amazon.

Temas

- ¿Cómo utilizo Timestream para LiveAnalytics las métricas?
- Secuencia temporal de LiveAnalytics métricas y dimensiones
- Crear CloudWatch alarmas para monitorear Timestream para LiveAnalytics

¿Cómo utilizo Timestream para LiveAnalytics las métricas?

Las métricas que informa Timestream LiveAnalytics proporcionan información que se puede analizar de diferentes maneras. En la siguiente lista se indican algunos usos frecuentes de las métricas. Se trata de sugerencias que puede usar como punto de partida y no de una lista completa.

¿Cómo...?	Métricas relevantes
How can I determine if any system errors occurred?	Puede monitorear SystemErrors para determinar si alguna solicitud resultó en un código de error del servidor. Normalmente, esta métrica debe ser igual a cero. De no ser así, es posible que desee investigarlo.
How can I monitor the amount of data in the memory store?	Puede supervisar MemoryCumulativeBytesMetered durante el período de tiempo especificado para controlar la cantidad de datos almacenados en la memoria almacenada en bytes. Esta métrica se emite cada hora y puede realizar un seguimiento de los bytes almacenados en una cuenta, así como de la granularidad de la base de datos. El almacén de memoria se mide en GB por hora (el coste de almacenar 1 GB de datos durante una hora). Por lo tanto, si multiplicas el valor por hora por el precio en GB por hora en tu región, obtendrás el coste incurrido por hora. MemoryCumulativeBytesMetered Dimensiones: operación (almacenamiento), nombre métrico DatabaseName
How can I monitor the amount of data in the magnetic store?	Puede supervisar MagneticCumulativeBytesMetered durante el período de tiempo especificado para controlar la cantidad de datos almacenados en el almacén magnético en bytes. Esta métrica se emite cada hora y puede realizar un seguimiento de los bytes almacenados en una cuenta, así como de la granularidad de la base de datos. El almacén de memoria se mide en GB al mes (el coste de almacenar 1 GB de datos durante un mes). Por lo tanto, si multiplicas el valor por hora por el precio en GB al mes en tu región, obtendrás el coste incurrido por hora. MagneticCumulativeBytesMetered Por ejemplo, si el valor de MagneticCumulativeBytesMetered es 107374182400 bytes (100 GB), el cargo por hora de 1 GB de datos en Magnetic Store = (0,03) (precio us-east-1)/(30,4*24). Si multiplicas este valor por el valor en GB, obtendrás aproximadamente 0,004$ por hora. MagneticCumulativeBytesMetered

¿Cómo...?	Métricas relevantes
	Dimensiones: operación (almacenamiento), nombre métrico DatabaseName
How can I monitor the data scanned by queries?	Puede monitorear `CumulativeBytesMetered` durante el período de tiempo especificado para monitorear los datos escaneados por las consultas (en bytes) enviadas a Timestrea m. LiveAnalytics Esta métrica se emite después de la ejecución de la consulta y permite realizar un seguimiento de los datos escaneados con el nivel de detalle de la cuenta y la base de datos. Puede calcular el coste de la consulta para un período determinado multiplicando el valor de la métrica por el precio por GB escaneado en su región. Los bytes escaneados por las consultas programadas se contabilizan en esta métrica. Dimensiones: operación (consulta) DatabaseName, nombre de la métrica
How can I monitor the data scanned by scheduled queries?	Puede supervisar `CumulativeBytesMetered` durante el período de tiempo especificado para supervisar los datos escaneados por las consultas programadas (en bytes) ejecutada s por Timestream. LiveAnalytics Esta métrica se emite después de la ejecución de la consulta y permite realizar un seguimiento de los datos escaneados con el nivel de detalle de la cuenta y la base de datos. Puede calcular el coste de la consulta para un período determinado multiplicando el valor de la métrica por el precio por GB escaneado en su región. **Note** Los bytes medidos también se tienen en cuenta en la consulta. `CumulativeBytesMetered` Dimensiones: operación (TriggeredScheduledQuery) DatabaseN ame, nombre de la métrica

¿Cómo...?	Métricas relevantes
How can I monitor the number of records ingested?	Puede supervisar el NumberOfRecords período de tiempo especificado para controlar el número de registros ingeridos . Puede realizar un seguimiento de los bytes almacenados en una cuenta, así como de la granularidad de la base de datos. También puede utilizar esta métrica para supervisar las escrituras realizadas por las consultas programadas cuando los resultados de las consultas se escriben en una tabla independi ente. Cuando se usa WriteRecords API, la métrica se emite para cada WriteRecords solicitud, siendo la dimensión CloudWatc h OperaciónWriteRecords . Cuando se utiliza BatchLoad o ScheduledQuery APIs, la métrica se emite a intervalos determinados por el servicio hasta que se completa la tarea. La dimensión de CloudWatch operación de esta métrica es BatchLoad oScheduledQuery , según la que API se utilice. Dimensiones: operación (WriteRecords, BatchLoad, o ScheduledQuery) DatabaseName, nombre de la métrica

¿Cómo...?	Métricas relevantes
How can I monitor the cost of records ingested?	Puede monitorizar CumulativeBytesMetered para monitorizar el número de bytes ingeridos que devengan costes. Puede realizar un seguimiento de los bytes almacenados en una cuenta, así como de la granularidad de la base de datos. Los registros ingeridos se miden en bytes acumulados. Si multiplica el valor de los precios CumulativeBytesMetered por Write en su región, obtendrá el coste de ingesta en el que se incurre. Cuando se utiliza WriteRecords API, esta métrica se emite para cada WriteRecords solicitud, siendo la dimensión CloudWatch Operación. WriteRecords Cuando se utiliza BatchLoad o ScheduledQuery API, la métrica se emite a intervalos determinados por el servicio hasta que se completa la tarea. La dimensión de CloudWatch operación de esta métrica es BatchLoad o ScheduledQuery depende de la que API se utilice. Dimensiones: operación (WriteRecords, BatchLoad, o ScheduledQuery) DatabaseName, nombre de la métrica
How can I monitor the Timestream Compute Units (TCUs) used in my account?	Puede supervisar QueryTCU durante el período de tiempo especificado para supervisar las unidades de cálculo consumida s en la carga de trabajo de consultas en la cuenta. Esta métrica se emite con unidades de cálculo máximas y mínimas por minuto durante la carga de trabajo de consultas activas de la cuenta. Unidades: Count Estadísticas válidas: mínimo, máximo Métrica: ResourceCount Dimensiones:Service: Timestream ,Resource: QueryTCU,Type: Resource, Class: OnDemand

Secuencia temporal de LiveAnalytics métricas y dimensiones

Cuando interactúas con Timestream for LiveAnalytics, envía las siguientes métricas y dimensiones a Amazon. CloudWatch Todas las métricas se agregan y notifican cada minuto. Puede utilizar los siguientes procedimientos para ver las métricas de Timestream for. LiveAnalytics

Para ver las métricas mediante la consola CloudWatch

Las métricas se agrupan en primer lugar por el espacio de nombres de servicio y, a continuación, por las diversas combinaciones de dimensiones dentro de cada espacio de nombres.

1. Abra la CloudWatch consola en https://console.aws.amazon.com/cloudwatch/.
2. De ser necesario, cambie la región. En la barra de navegación, elige la región en la que residen tus AWS recursos. Para obtener más información, consulte Puntos de enlace del servicio de AWS.
3. En el panel de navegación, seleccione Métricas.
4. En la pestaña Todas las métricas, elija `AWS/Timestream for LiveAnalytics`.

Para ver las métricas mediante el AWS CLI

- En el símbolo del sistema, ejecute el siguiente comando.

```
aws cloudwatch list-metrics --namespace "AWS/Timestream"
```

Dimensiones de Timestream para las métricas LiveAnalytics

Las métricas de Timestream LiveAnalytics se clasifican según los valores de la cuenta, el nombre de la tabla o la operación. Puede utilizar la CloudWatch consola para recuperar Timestream para LiveAnalytics los datos de cualquiera de las dimensiones de la siguiente tabla:

Dimensión	Descripción
`DatabaseName`	Esta dimensión limita los datos a un flujo temporal específico o para la base de datos. LiveAnalytics Este valor puede ser cualquier base de datos de la región actual y de la cuenta corriente AWS

Dimensión	Descripción
`Operation`	Esta dimensión limita los datos a uno de los flujos de tiempo para LiveAnalytics las operaciones, como`Storage`, `WriteRecords` `BatchLoad` , o. `ScheduledQuery` Consulte la API referencia Timestream for LiveAnalytics Query para ver una lista de los valores disponibles.
`TableName`	Esta dimensión limita los datos a una tabla específica de una base de datos Timestream for. LiveAnalyticss

Important

`CumulativeBytesMetered`, `UserErrors` y `SystemErrors` las métricas solo tienen la `Operation` dimensión. `SuccessfulRequestLatency`las métricas siempre tienen `Operation` una dimensión, pero también pueden tener las `TableName` dimensiones `DatabaseName` y, según el valor de`Operation`. Esto se debe a que Timestream para las operaciones a LiveAnalytics nivel de tabla tiene `DatabaseName` y `TableName` como dimensiones, pero las operaciones a nivel de cuenta no.

Secuencia temporal de las métricas LiveAnalytics

Note

Amazon CloudWatch agrega todo el siguiente flujo temporal de las LiveAnalytics métricas en intervalos de un minuto.

Métricas generales

Métrica	Descripción
`SuccessfulRequestLatency`	Las solicitudes realizadas a Timestream satisfactoriamente LiveAnalytics durante el período de tiempo especificado. Successfu

Métrica	Descripción
	lRequestLatency puede proporcionar dos tipos diferentes de información: • El tiempo transcurrido para que las solicitud es se acepten (mínimo, máximo, suma o promedio). • El número de solicitudes realizadas correctamente (SampleCount). SuccessfulRequestLatency refleja la actividad solo dentro de Timestream LiveAnalytics y no tiene en cuenta la latencia de la red ni la actividad del lado del cliente. Unidades: `Milliseconds` Dimensiones • `DatabaseName` • `TableName` • `Operation` Estadísticas válidas: • `Minimum` • `Maximum` • `Average` • `SampleCount` • `P10` • `p50` • `p90` • `p95` • `p99`

Métricas de escritura y almacenamiento

Métrica	Descripción
`MagneticStoreRejectedRecordCount`	El número de registros escritos del almacén magnético que se rechazaron de forma asíncrona. Esto puede suceder si el nuevo registro tiene una versión inferior a la versión actual o si el nuevo registro tiene una versión igual a la versión actual pero con datos diferentes. Unidades: `Count` Dimensiones • `DatabaseName` • `TableName` • `Operation` Estadísticas válidas: • `Sum` • `SampleCount`
`MagneticStoreRejectedUpload UserFailures`	El número de informes de registros rechazados por Magnetic Store que no se cargaron debido a errores del usuario. Esto puede deberse a que IAM los permisos no están configurados correctamente o a que se haya eliminado un bucket de S3. Unidades: `Count` Dimensiones • `DatabaseName` • `TableName`

Métrica	Descripción
	• `Operation` Estadísticas válidas: • `Sum` • `SampleCount`
`MagneticStoreRejectedUploadSystemFailures`	El número de informes de registros rechazados por el almacén magnético que no se cargaron debido a errores del sistema. Unidades: `Count` Dimensiones • `DatabaseName` • `TableName` • `Operation` Estadísticas válidas: • `Sum` • `SampleCount`

Métrica	Descripción
`ActiveMagneticStorePartitions`	El número de particiones de almacenamiento magnético que ingieren datos de forma activa en un momento dado. Unidades: `Count` Dimensiones • `DatabaseName` • `Operation` Estadísticas válidas: • `Sum` • `SampleCount`

Métrica	Descripción
`MagneticStorePendingRecordsLatency`	Las más antiguas escriben en un almacén magnético que no está disponible para su consulta. Los registros escritos en el almacén magnético estarán disponibles para su consulta en un plazo de 6 horas. Unidades: `Milliseconds` Dimensiones • `DatabaseName` • `TableName` • `Operation` Estadísticas válidas: • `Minimum` • `Maximum` • `Average` • `SampleCount` • `P10` • `p50` • `p90` • `p95` • `p99`

Métrica	Descripción
`MemoryCumulativeBytesMetered`	La cantidad de datos almacenados en el almacén de memoria, en bytes Unidades: `Bytes` Dimensiones: `Operation` Estadísticas válidas: • `Average`
`MagneticCumulativeBytesMetered`	La cantidad de datos almacenados en el almacén magnético, en bytes Unidades: `Bytes` Dimensiones: `Operation` Estadísticas válidas: • `Average`
`CumulativeBytesMetered`	La cantidad de datos medida mediante la ingesta a Timestream, en bytes. LiveAnalytics Unidades: `Bytes` Dimensiones: `Operation` Estadísticas válidas: `Sum`
`NumberOfRecords`	El número de registros ingeridos en Timestrea m para. LiveAnalytics Unidades: `Count` Dimensiones: `Operation` Estadísticas válidas: `Sum`

Métricas de consulta

Métrica	Descripción
`CumulativeBytesMetered`	La cantidad de datos escaneados por las consultas enviadas a Timestream LiveAnaly tics, en bytes. Unidades: `Bytes` Dimensiones: `Operation` Estadísticas válidas: • `Sum`
`ResourceCount`	Las unidades de cómputo de flujo temporal (TCUs) consumidas para la carga de trabajo de consultas en la cuenta. Esta métrica se emite con unidades de cálculo máximas y mínimas por minuto durante la carga de trabajo de consultas activas de la cuenta. Unidades: `Count` Estadísticas válidas: mínimo, máximo Dimensiones:`Service: Timestrea m` ,`Resource: QueryTCU`,`Type: Resource`, `Class: OnDemand`

Métricas de error

Métrica	Descripción
`SystemErrors`	Las solicitudes a Timestream para LiveAnaly tics eso generan un SystemError durante el período de tiempo especificado. A SystemError suele indicar un error de servicio interno.

Métrica	Descripción
	Unidades: `Count` Dimensiones: `Operation` Estadísticas válidas: • `Sum` • `SampleCount`
`UserErrors`	Las solicitudes a Timestream al respecto LiveAnalytics generan un InvalidRequest error durante el período de tiempo especificado. A InvalidRequest suele indicar un error del lado del cliente, como una combinación de parámetros no válida, un intento de actualiza r una tabla inexistente o una firma de solicitud incorrecta. UserErrors representa la suma de las solicitudes no válidas para la AWS región actual y la cuenta corriente. AWS Unidades: `Count` Dimensiones: `Operation` Estadísticas válidas: • `Sum` • `SampleCount`

Important

No todas las estadísticas, tales como `Average` o `Sum`, son aplicables a todas las métricas. Sin embargo, todos estos valores están disponibles a través de Timestream para la LiveAnalytics consola, mediante la CloudWatch consola o AWS SDKs para todas las métricas. AWS CLI

Crear CloudWatch alarmas para monitorear Timestream para LiveAnalytics

Puedes crear una CloudWatch alarma de Amazon para Timestream para LiveAnalytics que envíe un mensaje de Amazon Simple Notification Service SNS (Amazon) cuando la alarma cambie de estado. Una alarma vigila una métrica determinada durante el periodo especificado. Realiza una o varias acciones según el valor de la métrica con respecto a un umbral dado durante varios períodos de tiempo. La acción es una notificación enviada a un SNS tema de Amazon o a una política de Auto Scaling.

Las alarmas invocan acciones únicamente en caso de cambios de estado sostenidos. CloudWatch las alarmas no invocan acciones simplemente porque se encuentran en un estado determinado. El estado debe haber cambiado y debe mantenerse durante el número de periodos especificado.

Para obtener más información sobre la creación de CloudWatch alarmas, consulte Uso de Amazon CloudWatch Alarms en la Guía del CloudWatch usuario de Amazon.

Resolución de problemas

Esta sección contiene información sobre la solución de problemas de Timestream para. LiveAnalytics

Temas

- Manejo de los aceleradores WriteRecords
- Gestión de registros rechazados
- Solución de problemas desde Timestream para UNLOAD LiveAnalytics
- Secuencia temporal para códigos de error LiveAnalytics específicos

Manejo de los aceleradores WriteRecords

Las solicitudes de escritura de su almacén de memoria a Timestream pueden verse limitadas a medida que Timestream se amplíe para adaptarse a las necesidades de ingesta de datos de su aplicación. Si sus aplicaciones encuentran excepciones de limitación, debe seguir enviando datos con el mismo rendimiento (o superior) para permitir que Timestream se adapte automáticamente a las necesidades de su aplicación.

Es posible que las solicitudes de escritura del almacén magnético a Timestream se limiten si se supera el límite máximo de particiones del almacén magnético que reciben entradas. Verás un mensaje de aceleración que te indicará que compruebes la métrica de Cloudwatch para esta base de datos`ActiveMagneticStorePartitions`. Este acelerador puede tardar hasta 6 horas en

resolverse. Para evitar esta aceleración, debe utilizar el almacén de memoria para cualquier carga de trabajo de ingestión de alto rendimiento. Para la ingesta de almacenamiento magnético, puede centrarse en la ingesta en un menor número de particiones limitando el número de series y la duración de la ingesta

Para obtener más información sobre las prácticas recomendadas de ingesta de datos, consulte. Escribe

Gestión de registros rechazados

Si Timestream rechaza los registros, recibirá un aviso `RejectedRecordsException` con los detalles del rechazo. Consulte Gestión de errores de escritura para obtener más información sobre cómo extraer esta información de la WriteRecords respuesta.

Todos los rechazos se incluirán en esta respuesta, con la excepción de las actualizaciones del almacén magnético, en las que la versión del nuevo disco sea inferior o igual a la versión del registro existente. En este caso, Timestream no actualizará el registro existente que tiene la versión superior. Timestream rechazará el nuevo registro con una versión inferior o igual y escribirá estos errores de forma asíncrona en su bucket de S3. Para recibir estos informes de errores asíncronos, debe configurar la propiedad en su tabla. `MagneticStoreRejectedDataLocation` `MagneticStoreWriteProperties`

Solución de problemas desde Timestream para UNLOAD LiveAnalytics

A continuación se ofrece una guía para la solución de problemas relacionados con el UNLOAD comando.

Categoría	Mensaje de error	Cómo solucionar problemas
Longitud de la clave S3	UNLOADSi se utiliza el prefijo S3 [%s] proporcionado en el destino, la clave del archivo de resultados superará la longitud de clave permitida por S3. Consulte la documentación para obtener más información.	Al exportar los resultados de una consulta mediante la `UNLOAD` sentencia, la longitud de la clave S3, que consiste en la suma de la longitud del nombre y el prefijo del bucket de S3, supera la longitud máxima de clave de S3 admitida. Se recomienda

Categoría	Mensaje de error	Cómo solucionar problemas
		reducir la longitud del prefijo o del nombre del bucket.
	UNLOADSi se utiliza partition ed_by [%s], la clave del archivo de resultados superará la longitud de clave permitida en S3. Consulte la documentación para obtener más información.	Al exportar los resultados de una consulta mediante la UNLOAD sentencia, la longitud de la clave S3 que utiliza la columna partitioned_by supera la longitud máxima de clave S3 admitida. Recomendamos particionar con una columna alternativa o reducir la longitud de la partitioned_column (si es posible).
	UNLOADSi se usa el prefijo S3 [%s] junto con partition ed_by [%s], la clave del archivo de resultados superará la longitud de clave permitida por S3. Consulte la documentación para obtener más información.	Al exportar los resultados de una consulta mediante la UNLOAD sentencia, la longitud de la clave S3, compuesta por la suma de la longitud del nombre del bucket de S3, el prefijo y el nombre de la columna partitioned_by, supera la longitud máxima de clave de S3 admitida. Se recomienda reducir la longitud del prefijo y el nombre del bucket o utilizar una columna alternativa para particionar los datos.

Categoría	Mensaje de error	Cómo solucionar problemas
	La clave de objeto de S3 generada: %s es demasiado larga. Consulte la documentación para obtener más información.	Al procesar la consulta con la UNLOAD sentencia, uno de los valores de la columna particionada supera la longitud máxima de clave S3 admitida. La columna y el valor de la partición se encuentran en la clave de objeto generada.
S3 acelera	Hemos detectado que Amazon S3 está limitando las escrituras desde UNLOAD el comando. Consulte la documentación de Amazon Timestream para obtener más información.	Consulte la documentación de S3 aquí. La velocidad de API llamadas de S3 podría reducirse cuando varios lectores o grabadores acceden a la misma carpeta. Audite el volumen de llamadas hasta el bucket proporcionado. Si utilizas el mismo depósito para varias UNLOAD consultas simultáneas, intenta utilizar distintos grupos para el mismo. Si utilizas el mismo depósito para varias operaciones distintas de Timestream LiveAnalytics UNLOAD, considera la posibilidad de mover los UNLOAD resultados a un depósito diferente.

Secuencia temporal para códigos de error LiveAnalytics específicos

Esta sección contiene los códigos de error específicos de Timestream para. LiveAnalytics

Secuencia temporal de errores de escritura LiveAnalytics API

InternalServerException

HTTPCódigo de estado: 500

ThrottlingException

HTTPCódigo de estado: 429

ValidationException

HTTPCódigo de estado: 400

ConflictException

HTTPCódigo de estado: 409

AccessDeniedException

No tiene acceso suficiente para realizar esta acción.

HTTPCódigo de estado: 403

ServiceQuotaExceededException

HTTPCódigo de estado: 402

ResourceNotFoundException

HTTPCódigo de estado: 404

RejectedRecordsException

HTTPCódigo de estado: 419

InvalidEndpointException

HTTPCódigo de estado: 421

Secuencia temporal de errores de consulta LiveAnalytics API

ValidationException

HTTPCódigo de estado: 400

QueryExecutionException

HTTPCódigo de estado: 400

ConflictException

HTTPCódigo de estado: 409

ThrottlingException

HTTPCódigo de estado: 429

InternalServerException

HTTPCódigo de estado: 500

InvalidEndpointException

HTTPCódigo de estado: 421

Cuotas

En este tema se describen las cuotas actuales, también denominadas límites, en Amazon LiveAnalytics Timestream para. Cada una de las cuotas se aplica a una sola región, a no ser que se especifique otra cosa.

Temas

Cuotas predeterminadas

La siguiente tabla contiene el flujo temporal de las LiveAnalytics cuotas y los valores predeterminados.

displayName	Descripción	defaultValue
Bases de datos por cuenta	El número máximo de bases de datos que se pueden crear por. Cuenta de AWS	500
Tablas por cuenta	El número máximo de tablas que se pueden crear por Cuenta de AWS.	50000
Velocidad de aceleración para CRUD APIs	El número máximo de API solicitudes de Create/Update/List/Describe/Delete database/table/scheduled consulta permitidas por segundo por cuenta, en la región actual.	1
Consultas programadas por cuenta	El número máximo de consultas programadas que puede crear por cada consulta Cuenta de AWS.	10000
Recuento máximo de particiones de almacenamiento magnético activas	El número máximo de particiones de almacenamiento magnético activas por base de datos. Una partición puede permanecer activa hasta seis horas después de ser ingerida.	250
maxQueryTCU	La consulta máxima TCUs que puedes establecer para tu cuenta.	1 000

Límites de los servicios

La siguiente tabla contiene el intervalo temporal de los límites de LiveAnalytics servicio y los valores predeterminados. Para editar la retención de datos de una tabla desde la consola, consulte Editar una tabla.

displayName	Descripción	defaultValue
Periodo de ingesta futuro en minutos	Tiempo de espera máximo (en minutos) de los datos de serie temporal en comparación con la hora actual del sistema. Por ejemplo, si el período de ingestión futuro es de 15 minutos, Timestream for LiveAnalytics aceptará datos que estén hasta 15 minutos por delante de la hora actual del sistema.	15
Periodo mínimo de retención en el almacén en memoria en horas	Tiempo mínimo (en horas) durante el que se deben conservar los datos en el almacén en memoria por tabla.	1
Periodo máximo de retención en el almacén en memoria en horas	Tiempo máximo (en horas) durante el que se pueden conservar los datos en el almacén en memoria por tabla.	8766
Periodo mínimo de retención en el almacén magnético en días	Tiempo mínimo (en días) durante el que se deben conservar los datos en el almacén magnético por tabla.	1

displayName	Descripción	defaultValue
Periodo máximo de duración de la retención del almacén magnético en días	Tiempo máximo (en días) durante el que se pueden conservar los datos en el almacén magnético. Este valor equivale a 200 años.	73.000
Periodo de retención predeterminado para el almacenamiento magnético en días	El valor predeterminado (en días) para el que se conservan los datos en el almacén magnético por tabla. Este valor equivale a 200 años.	73.000
Periodo de retención predeterminado para el almacén de memoria en horas	El tiempo predeterminado (en horas) durante el que se conservan los datos en el almacén de memoria.	6
Dimensiones por tabla	Número máximo de dimension es por tabla.	128
Mida los nombres por tabla	El número máximo de nombres de medidas únicos por tabla.	8192
Tamaño del par “nombre de la dimensión-valor de la dimensión” por serie	Tamaño máximo del par “nombre de la dimensión-valor de la dimensión” por serie.	2 kilobytes
Tamaño de registro máximo	El tamaño máximo de un registro.	2 kilobytes
Registros por WriteRecords API solicitud	El número máximo de registros de una WriteRecords API solicitud.	100

displayName	Descripción	defaultValue
Longitud del nombre de la dimensión	El número máximo de bytes para un nombre de dimensión.	60 bytes
Longitud del nombre de la medida	El número máximo de bytes para un nombre de medida.	256 bytes
Longitud del nombre de la base de datos	El número máximo de bytes para un nombre de base de datos.	256 bytes
Longitud del nombre de la tabla	El número máximo de bytes para un nombre de tabla.	256 bytes
QueryString longitud en KiB	La longitud máxima (en KiB) de una cadena de consulta en UTF -8 caracteres codificados para una consulta.	256
Duración de ejecución de las consultas en horas	Duración máxima de ejecución (en horas) de una consulta. Se agotará el tiempo de espera de las consultas que tarden más.	1
Información sobre la consulta	El número máximo de API solicitudes de consulta permitidas con la información de consultas habilitada por segundo por cuenta, en la región actual.	1
Tamaño de los metadatos del resultado de la consulta	Tamaño máximo de los metadatos del resultado de una consulta.	100 kilobytes
Tamaño de los datos del resultado de la consulta	Tamaño máximo de los datos del resultado de una consulta.	5 Gigabytes

displayName	Descripción	defaultValue
Medidas por registro de múltiples medidas	Número máximo de medidas por registro de múltiples medidas.	256
Tamaño del valor de la medida por registro de múltiples medidas	Tamaño máximo de los valores de las medidas por registro de múltiples medidas.	2048
Medidas únicas en todos los registros de múltiples medidas por tabla	Medidas únicas de todos los registros de múltiples medidas definidos en una sola tabla.	1024
Unidades de cómputo Timestream () TCUs por cuenta	El máximo predeterminado TCUs por cuenta.	200

Tipos de datos compatibles

En la siguiente tabla se describen los tipos de datos admitidos para los valores de medida y dimensión.

Descripción	Secuencia temporal del valor LiveAnalytics
Tipos de datos compatibles para los valores de las medidas.	Big int, double, string, boolean, MULTI Timestamp
Tipos de datos compatibles para los valores de dimensión.	Cadena

Carga por lotes

Las cuotas actuales, también denominadas límites, dentro de la carga por lotes son las siguientes.

Descripción	Flujo temporal de valor LiveAnalytics
Tamaño máximo de la tarea de carga por lotes	El tamaño máximo de la tarea de carga por lotes no puede superar los 100 GB.
Cantidad de archivos	Una tarea de carga por lotes no puede tener más de 100 archivos.
Tamaño máximo de archivo	El tamaño máximo de archivo en una tarea de carga por lotes no puede superar los 5 GB.
CSVtamaño de fila de archivos	Una fila de un CSV archivo no puede superar los 16 MB. Se trata de un límite estricto que no se puede aumentar.
Tareas activas de carga por lotes	Una tabla no puede tener más de 5 tareas de carga por lotes activas y una cuenta no puede tener más de 10 tareas de carga por lotes activas. Timestream for LiveAnalytics limitará las nuevas tareas de carga por lotes hasta que haya más recursos disponibles.

Restricciones en la nomenclatura

En la siguiente tabla se describen las restricciones de nomenclatura.

Descripción	Secuencia temporal del valor LiveAnalytics
La longitud máxima del nombre de una dimensión .	60 bytes
La longitud máxima del nombre de una medida.	256 bytes
La longitud máxima del nombre de una tabla o de una base de datos.	256 bytes
Nombre de tabla y base de datos	• Le recomendamos que no utiliceIdentificadores del sistema. • Puede contener a-z A-Z 0-9 _ (guión bajo) - (guión). (punto).

Descripción	Secuencia temporal del valor LiveAnalytics
	• Todos los nombres deben estar codificados como UTF -8 y distinguen entre mayúsculas y minúsculas. **Note** Los nombres de tablas y bases de datos se comparan mediante la representación binaria UTF -8. Esto significa que la comparación de ASCII caracteres distingue entre mayúsculas y minúsculas.
Nombre de la medida	• No debe contener Identificadores del sistema ni dos puntos ':'. • No debe empezar con un prefijo reservado (`ts_`,`measure_value`). **Note** Los nombres de tablas y bases de datos se comparan mediante la representación binaria UTF -8. Esto significa que la comparación de ASCII caracteres distingue entre mayúsculas y minúsculas.

Descripción	Secuencia temporal del valor LiveAnalytics
Nombre de la dimensión	• No debe contener Identificadores del sistema dos puntos «:» ni comillas dobles («). • No debe empezar con un prefijo reservado (`ts_`,`measure_value`). • No debe contener los caracteres Unicode [0,31] enumerados aquí ni «\ u2028" o «\ u2029". **Note** Los nombres de dimensiones y medidas se comparan mediante una representación binaria de -8. UTF Esto significa que la comparación de ASCII caracteres distingue entre mayúsculas y minúsculas.
Todos los nombres de las columnas	Los nombres de las columnas no se pueden duplicar. Como los registros de varias medidas representan las dimensiones y las medidas como columnas, el nombre de una dimensión no puede ser el mismo que el nombre de una medida. Los nombres distinguen mayúsculas de minúsculas.

Palabras clave reservadas

Todas las siguientes son palabras clave reservadas:

- ALTER
- AND
- AS
- BETWEEN
- BY
- CASE
- CAST
- CONSTRAINT

- CREATE
- CROSS
- CUBE
- CURRENT_DATE
- CURRENT_TIME
- CURRENT_TIMESTAMP
- CURRENT_USER
- DEALLOCATE
- DELETE
- DESCRIBE
- DISTINCT
- DROP
- ELSE
- END
- ESCAPE
- EXCEPT
- EXECUTE
- EXISTS
- EXTRACT
- FALSE
- FOR
- FROM
- FULL
- GROUP
- GROUPING
- HAVING
- IN
- INNER

- INSERT
- INTERSECT
- INTO
- IS
- JOIN
- LEFT
- LIKE
- LOCALTIME
- LOCALTIMESTAMP
- NATURAL
- NORMALIZE
- NOT
- NULL
- ON
- OR
- ORDER
- OUTER
- PREPARE
- RECURSIVE
- RIGHT
- ROLLUP
- SELECT
- TABLE
- THEN
- TRUE
- UESCAPE
- UNION
- UNNEST
- USING

- VALUES
- WHEN
- WHERE
- WITH

Identificadores del sistema

Reservamos los nombres de columna «measure_value», «ts_non_existent_col» y «time» para que sean Timestream para los identificadores del sistema. LiveAnalytics Además, los nombres de las columnas no pueden empezar por «ts_» o «measure_name». Los identificadores del sistema distinguen entre mayúsculas y minúsculas. Los identificadores se comparan mediante la representación binaria UTF -8. Esto significa que la comparación de los identificadores distingue entre mayúsculas y minúsculas.

> **Note**
>
> Los identificadores del sistema no se pueden usar para los nombres de dimensiones o medidas. Le recomendamos que no utilice identificadores del sistema para los nombres de bases de datos o tablas.

UNLOAD

Para conocer los límites relacionados con el `UNLOAD` comando, consulte Utilización UNLOAD para exportar los resultados de consultas a S3 desde Timestream.

Consulta el idioma de referencia

> **Note**
>
> Esta referencia del lenguaje de consulta incluye la siguiente documentación de terceros de Trino Software Foundation (anteriormente Presto Software Foundation), que está licenciada bajo la licencia Apache, versión 2.0. No puede utilizar este archivo excepto de conformidad con esta licencia. Para obtener una copia de la licencia de Apache, versión 2.0, consulte el sitio web de Apache.

Timestream for LiveAnalytics admite un lenguaje de consulta enriquecido para trabajar con sus datos. A continuación, puede ver los tipos de datos, los operadores, las funciones y las construcciones disponibles.

También puedes empezar de inmediato con el lenguaje de consultas de Timestream en la sección. Consultas de ejemplo

Temas

Tipos de datos compatibles

El lenguaje de consulta de Timestream for LiveAnalytics admite los siguientes tipos de datos.

> **Note**
>
> Los tipos de datos compatibles con las escrituras se describen en Tipos de datos.

Tipo de datos	Descripción
`int`	Representa un entero de 32 bits.
`bigint`	Representa un entero con signo de 64 bits.
`boolean`	Uno de los dos valores de verdad de la lógica `True` y`False`.
`double`	Representa un tipo de datos de precisión variable de 64 bits. Implementa el IEEEestándar 754 para aritmética binaria de punto flotante. **Note** El lenguaje de consulta es para leer datos. Hay funciones para `Infinity` valores `NaN` dobles que se pueden utilizar en las consultas. Pero no puede escribir esos valores en Timestream.
`varchar`	Datos de caracteres de longitud variable con un tamaño máximo de 2 KB.
`array[T,...]`	Contiene uno o más elementos de un tipo de datos especific ado *T*donde, *T* puede ser cualquiera de los tipos de datos admitidos en Timestream.
`row(T,...)`	Contiene uno o más campos con nombre del tipo de datos *T*. Los campos pueden ser de cualquier tipo de datos compatibl e con Timestream y se accede a ellos con el operador de referencia de campos de puntos: `.`

Tipo de datos	Descripción
`date`	Representa una fecha en el formulario*YYYY*-*MM*-*DD*. ¿Dónde *YYYY* es el año, *MM* es el mes, y *DD* es el día, respectivamente. El rango admitido es de `1970-01-01` a`2262-04-11` . Ejemplo: `1971-02-03`
`time`	Representa la hora del día en UTC. El `time` tipo de datos se representa de la siguiente manera: *HH*.*MM*.*SS*.*sssssssss* . admite una precisión de nanosegundos. Ejemplo: `17:02:07.496000000`
`timestamp`	Representa una instancia en el tiempo utilizando el tiempo de precisión de nanosegundos en. UTC *YYYY*-*MM*-*DD* *hh*:*mm*:*ss*.*sssssssss* Query admite marcas de tiempo en el rango de 0 a. `1677-09-21 00:12:44.000000000` `2262-04-11 23:47:16.854775807`

Tipo de datos	Descripción

`interval`	Representa un intervalo de tiempo como una cadena literal*X**t*, compuesta por dos partes, *X* y *t*.
	
	X es un valor numérico mayor o igual a`0`, y *t* es una unidad de tiempo, como un segundo o una hora. La unidad no está pluralizada. La unidad de tiempo *t* is debe ser uno de los siguientes literales de cadena:
	
	• `nanosecond`
	• `microsecond`
	• `millisecond`
	• `second`
	• `minute`
	• `hour`
	• `day`
	• `ns`(igual `nanosecond` que)
	• `us`(igual que`microsecond`)
	• `ms`(igual que`millisecond`)
	• `s`(igual que`second`)
	• `m`(igual que`minute`)
	• `h`(igual que`hour`)
	• `d`(igual que`day`)
	
	Ejemplos:
	
	`17s`
	
	`12second`
	
	`21hour`

Tipo de datos	Descripción
	`2d`
`timeseries[row(timestamp, T,...)]`	Representa los valores de una medida registrados durante un intervalo de tiempo como un `array` conjunto de `row` objetos. Cada uno `row` contiene uno `timestamp` y uno o más valores de medida del tipo de datos *T*donde, *T* puede ser cualquiera `bigint`, `boolean``double`, `varchar`. Las filas se ordenan en orden ascendente por. `timestamp` El tipo de datos de la serie temporal representa los valores de una medida a lo largo del tiempo.
`unknown`	Representa datos nulos.

Funcionalidad de series temporales integrada

Timestream for LiveAnalytics proporciona una funcionalidad de series temporales integrada que trata los datos de series temporales como un concepto de primera clase.

La funcionalidad integrada de series temporales se puede dividir en dos categorías: vistas y funciones.

Puedes leer más sobre cada construcción a continuación.

Temas

- Vistas de series temporales
- Funciones de series temporales

Vistas de series temporales

Timestream for LiveAnalytics admite las siguientes funciones para transformar los datos en el `timeseries` tipo de datos:

Temas

- CREATE_TIME_SERIES
- UNNEST

CREATE_TIME_SERIES

CREATE_ TIME _ SERIES es una función de agregación que toma todas las medidas sin procesar de una serie temporal (valores de tiempo y medida) y devuelve un tipo de datos de serie temporal. La sintaxis de esta función es la siguiente:

```
CREATE_TIME_SERIES(time, measure_value::<data_type>)
```

donde `<data_type>` es el tipo de datos del valor de la medida y puede ser bigint, boolean, double o varchar. El segundo parámetro no puede ser nulo.

Tenga en cuenta CPU el uso de EC2 las instancias almacenadas en una tabla denominada métricas, como se muestra a continuación:

Tiempo	región	az	vpc	instance_id	measure_name	measure_value::double
2019-12-04 19:00:00.000 000000	us-east-1	US-East-1d	vpc-1a2b3c4d	i-1234567890abcdef0	utilización de la CPU	35,0
2019-12-04 19:00:01.000 000000	us-east-1	US-East-1d	vpc-1a2b3c4d	i-1234567890abcdef0	utilización de la CPU	38.2
2019-12-04 19:00:02.000 000000	us-east-1	US-East-1d	vpc-1a2b3c4d	i-1234567890abcdef0	utilización de la CPU	45,3
2019-12-04 19:00:00.000 000000	us-east-1	US-East-1d	vpc-1a2b3c4d	i-1234567890abcdef1	utilización de la CPU	54.1

Tiempo	región	az	vpc	instance_id	measure_name	measure_value::double
2019-12-04 19:00:01.000 000000	us-east-1	US-East-1d	vpc-1a2b3c4d	i-1234567890abcdef1	utilización de la CPU	42,5
2019-12-04 19:00:02.000 000000	us-east-1	US-East-1d	vpc-1a2b3c4d	i-1234567890abcdef1	utilización de la CPU	3.7

Ejecutando la consulta:

```
SELECT region, az, vpc, instance_id, CREATE_TIME_SERIES(time, measure_value::double) as
 cpu_utilization FROM metrics
    WHERE measure_name='cpu_utilization'
    GROUP BY region, az, vpc, instance_id
```

devolverá todas las series que tengan `cpu_utilization` un valor de medida. En este caso, tenemos dos series:

región	az	vpc	instance_id	cpu_utilization
us-east-1	us-east-1d	vpc-1a2b3c4d	i-1234567890abcdef0	[{hora: 2019-12-04 19:00:00.000 000000, valor_medida: :doble: 35,0}, {hora: 2019-12-04 19:00:01.000 000000, valor_med

región	az	vpc	instance_id	cpu_utilization
				ida: :doble: 38,2}, {hora: 2019-12-0 4 19:00:02. 000 000000, valor_med ida: :doble: 45,3}]
us-east-1	us-east-1d	vpc-1a2b3c4d	i-1234567 890abcdef1	[{hora: 2019-12-0 4 19:00:00. 000 000000, valor_med ida: :doble: 35,1}, {hora: 2019-12-0 4 19:00:01. 000 000000, valor_med ida: :doble: 38,5}, {hora: 2019-12-0 4 19:00:02. 000 000000, valor_med ida: :doble: 45,7}]

UNNEST

`UNNEST`es una función de tabla `timeseries` que permite transformar los datos en un modelo plano. La sintaxis es la siguiente:

UNNESTtimeseriestransforma a en dos columnas, a saber, `time` y`value`. También puedes usar alias UNNEST como se muestra a continuación:

```
UNNEST(timeseries) AS <alias_name> (time_alias, value_alias)
```

donde <alias_name> es el alias de la tabla plana, `time_alias` es el alias de la `time` columna y `value_alias` es el alias de la `value` columna.

Por ejemplo, considere el escenario en el que algunas de las EC2 instancias de su flota están configuradas para emitir métricas en un intervalo de 5 segundos, otras emiten métricas en un intervalo de 15 segundos y necesita el promedio de las métricas de todas las instancias con una granularidad de 10 segundos durante las últimas 6 horas. Para obtener estos datos, debe transformar las métricas al modelo de series temporales mediante CREATE_ TIME _ SERIES. A continuación, puede usar INTERPOLATE_ LINEAR para obtener los valores faltantes con una granularidad de 10 segundos. A continuación, se vuelven a transformar los datos en el modelo plano utilizando y UNNEST, a continuación, se utilizan AVGpara obtener las métricas medias de todas las instancias.

```
WITH interpolated_timeseries AS (
    SELECT region, az, vpc, instance_id,
        INTERPOLATE_LINEAR(
            CREATE_TIME_SERIES(time, measure_value::double),
                SEQUENCE(ago(6h), now(), 10s)) AS interpolated_cpu_utilization
    FROM timestreamdb.metrics
    WHERE measure_name= 'cpu_utilization' AND time >= ago(6h)
    GROUP BY region, az, vpc, instance_id
)
SELECT region, az, vpc, instance_id, avg(t.cpu_util)
FROM interpolated_timeseries
CROSS JOIN UNNEST(interpolated_cpu_utilization) AS t (time, cpu_util)
GROUP BY region, az, vpc, instance_id
```

La consulta anterior demuestra el uso de UNNESTcon un alias. A continuación se muestra un ejemplo de la misma consulta sin usar un alias para UNNEST:

```
WITH interpolated_timeseries AS (
    SELECT region, az, vpc, instance_id,
        INTERPOLATE_LINEAR(
            CREATE_TIME_SERIES(time, measure_value::double),
                SEQUENCE(ago(6h), now(), 10s)) AS interpolated_cpu_utilization
```

```
    FROM timestreamdb.metrics
    WHERE measure_name= ‘cpu_utilization’ AND time >= ago(6h)
    GROUP BY region, az, vpc, instance_id
)
SELECT region, az, vpc, instance_id, avg(value)
FROM interpolated_timeseries
CROSS JOIN UNNEST(interpolated_cpu_utilization)
GROUP BY region, az, vpc, instance_id
```

Funciones de series temporales

Amazon Timestream LiveAnalytics for admite funciones de series temporales, como derivadas, integrales y correlaciones, entre otras, para obtener información más profunda a partir de los datos de series temporales. En esta sección se proporciona información sobre el uso de cada una de estas funciones, así como ejemplos de consultas. Seleccione uno de los temas siguientes para obtener más información.

Temas

- Funciones de interpolación
- Funciones derivadas
- Funciones integrales
- Funciones de correlación
- Funciones de filtrado y reducción

Funciones de interpolación

Si a los datos de la serie temporal les faltan valores para los eventos en determinados momentos, puede estimar los valores de esos eventos faltantes mediante la interpolación. Amazon Timestream admite cuatro variantes de interpolación: interpolación lineal, interpolación de spline cúbica, interpolación de última observación transferida (locf) e interpolación constante. En esta sección se proporciona información sobre el uso de Timestream para las funciones de interpolación, así como ejemplos de consultas. LiveAnalytics

Información de uso

Función	Tipo de datos de salida	Descripción
`interpolate_linear (timeseries, array[timestamp])`	serie temporal	Rellena los datos faltantes mediante interpolación lineal.
`interpolate_linear (timeseries, timestamp)`	double	Rellena los datos faltantes mediante interpolación lineal.
`interpolate_spline _cubic(timeseries, array[timestamp])`	serie temporal	Rellena los datos faltantes mediante la interpolación de ranuras cúbicas.
`interpolate_spline _cubic(timeseries, timestamp)`	double	Rellena los datos faltantes mediante la interpolación de ranuras cúbicas.
`interpolate_locf(t imeseries, array[tim estamp])`	serie temporal	Rellena los datos faltantes utilizando el último valor muestreado.
`interpolate_locf(t imeseries, timestamp )`	double	Rellena los datos faltantes utilizando el último valor muestreado.
`interpolate_fill(t imeseries, array[tim estamp], double)`	serie temporal	Rellena los datos faltantes con un valor constante.
`interpolate_fill(t imeseries, timestamp , double)`	double	Rellena los datos faltantes con un valor constante.

Consultas de ejemplo

Example

Encuentre la CPU utilización media agrupada en intervalos de 30 segundos para un EC2 host específico durante las últimas 2 horas y rellene los valores faltantes mediante la interpolación lineal:

```
WITH binned_timeseries AS (
SELECT hostname, BIN(time, 30s) AS binned_timestamp, ROUND(AVG(measure_value::double),
 2) AS avg_cpu_utilization
FROM "sampleDB".DevOps
WHERE measure_name = 'cpu_utilization'
    AND hostname = 'host-Hovjv'
    AND time > ago(2h)
GROUP BY hostname, BIN(time, 30s)
), interpolated_timeseries AS (
SELECT hostname,
    INTERPOLATE_LINEAR(
        CREATE_TIME_SERIES(binned_timestamp, avg_cpu_utilization),
            SEQUENCE(min(binned_timestamp), max(binned_timestamp), 15s)) AS
 interpolated_avg_cpu_utilization
FROM binned_timeseries
GROUP BY hostname
)
SELECT time, ROUND(value, 2) AS interpolated_cpu
FROM interpolated_timeseries
CROSS JOIN UNNEST(interpolated_avg_cpu_utilization)
```

Example

Calcule la CPU utilización media agrupada en intervalos de 30 segundos para un EC2 host específico durante las últimas 2 horas y rellene los valores faltantes mediante la interpolación en función de la última observación realizada:

```
WITH binned_timeseries AS (
SELECT hostname, BIN(time, 30s) AS binned_timestamp, ROUND(AVG(measure_value::double),
 2) AS avg_cpu_utilization
FROM "sampleDB".DevOps
WHERE measure_name = 'cpu_utilization'
    AND hostname = 'host-Hovjv'
    AND time > ago(2h)
GROUP BY hostname, BIN(time, 30s)
```

```
), interpolated_timeseries AS (
SELECT hostname,
    INTERPOLATE_LOCF(
        CREATE_TIME_SERIES(binned_timestamp, avg_cpu_utilization),
            SEQUENCE(min(binned_timestamp), max(binned_timestamp), 15s)) AS
 interpolated_avg_cpu_utilization
FROM binned_timeseries
GROUP BY hostname
)
SELECT time, ROUND(value, 2) AS interpolated_cpu
FROM interpolated_timeseries
CROSS JOIN UNNEST(interpolated_avg_cpu_utilization)
```

Funciones derivadas

Las derivadas se utilizan para calcular la tasa de cambio de una métrica determinada y se pueden utilizar para responder de forma proactiva a un evento. Por ejemplo, supongamos que calcula la derivada de la CPU utilización de EC2 instancias en los últimos 5 minutos y observa una derivada positiva significativa. Esto puede ser indicativo de un aumento de la demanda de su carga de trabajo, por lo que puede decidir activar más EC2 instancias para gestionar mejor su carga de trabajo.

Amazon Timestream admite dos variantes de funciones derivadas. En esta sección se proporciona información sobre el uso del Timestream para funciones LiveAnalytics derivadas, así como ejemplos de consultas.

Información de uso

Función	Tipo de datos de salida	Descripción
`derivative_linear( timeseries, interval)`	serie temporal	Calcula la derivada de cada punto del `timeseries` objeto especificado`interval`.
`non_negative_deriv ative_linear(times eries, interval)`	serie temporal	Igual que`derivativ e_linear(timeserie s, interval)` , pero solo devuelve valores positivos.

Consultas de ejemplo

Example

Calcule la tasa de cambio en la CPU utilización cada 5 minutos durante la última hora:

```
SELECT DERIVATIVE_LINEAR(CREATE_TIME_SERIES(time, measure_value::double), 5m) AS
 result
FROM “sampleDB”.DevOps
WHERE measure_name = 'cpu_utilization'
AND hostname = 'host-Hovjv' and time > ago(1h)
GROUP BY hostname, measure_name
```

Example

Calcule la tasa de aumento de los errores generados por uno o más microservicios:

```
WITH binned_view as (
    SELECT bin(time, 5m) as binned_timestamp, ROUND(AVG(measure_value::double), 2) as
 value
    FROM “sampleDB”.DevOps
    WHERE micro_service = 'jwt'
    AND time > ago(1h)
    AND measure_name = 'service_error'
    GROUP BY bin(time, 5m)
)
SELECT non_negative_derivative_linear(CREATE_TIME_SERIES(binned_timestamp, value), 1m)
 as rateOfErrorIncrease
FROM binned_view
```

Funciones integrales

Puedes usar integrales para encontrar el área bajo la curva por unidad de tiempo para tus eventos de series temporales. Por ejemplo, supongamos que está realizando un seguimiento del volumen de solicitudes que recibe su aplicación por unidad de tiempo. En este escenario, puedes usar la función integral para determinar el volumen total de solicitudes atendidas por intervalo especificado durante un período de tiempo específico.

Amazon Timestream admite una variante de funciones integrales. En esta sección se proporciona información sobre el uso de la función Timestream para LiveAnalytics una función integral, así como ejemplos de consultas.

Información de uso

Función	Tipo de datos de salida	Descripción

`integral_trapezoidal(timeseries(double))`
 	
`integral_trapezoidal(timeseries(double), interval day to second)`
 	
`integral_trapezoidal(timeseries(bigint))`
 	
`integral_trapezoidal(timeseries(bigint), interval day to second)`
 	
`integral_trapezoidal(timeseries(integer), interval day to second)`
 	
`integral_trapezoidal(timeseries(integer))`
	double	Aproxima la integral según lo especificado `interval day to second` para lo `timeseries` proporcionado, utilizando la regla trapezoidal. El intervalo entre el día y el segundo parámetro es opcional y el valor predeterminado es. `1s` Para obtener más información sobre los intervalos, consulteIntervalo y duración.

Consultas de ejemplo

Example

Calcule el volumen total de solicitudes atendidas cada cinco minutos durante la última hora por un host específico:

```
SELECT INTEGRAL_TRAPEZOIDAL(CREATE_TIME_SERIES(time, measure_value::double), 5m) AS
 result FROM sample.DevOps
WHERE measure_name = 'request'
AND hostname = 'host-Hovjv'
AND time > ago (1h)
GROUP BY hostname, measure_name
```

Funciones de correlación

Dadas dos series temporales de longitud similar, las funciones de correlación proporcionan un coeficiente de correlación, que explica la tendencia de las dos series temporales a lo largo del tiempo. El coeficiente de correlación varía de `-1.0` a`1.0`. `-1.0`indica que las dos series temporales tienden en direcciones opuestas a la misma velocidad, mientras que `1.0` indica que las dos series temporales tienden en la misma dirección y a la misma velocidad. Un valor de `0` indica que no hay correlación entre las dos series temporales. Por ejemplo, si el precio del petróleo aumenta y el precio de las acciones de una empresa petrolera aumenta, la tendencia del aumento del precio del petróleo y el aumento del precio de la empresa petrolera tendrá un coeficiente de correlación positivo. Un coeficiente de correlación positivo alto indicaría que los dos precios tienen una tendencia similar. Del mismo modo, el coeficiente de correlación entre los precios de los bonos y los rendimientos de los bonos es negativo, lo que indica que estos dos valores tienen una tendencia en la dirección opuesta a lo largo del tiempo.

Amazon Timestream admite dos variantes de funciones de correlación. En esta sección se proporciona información sobre el uso del Timestream para las funciones de LiveAnalytics correlación, así como ejemplos de consultas.

Información de uso

Función	Tipo de datos de salida	Descripción
`correlate_pearson( timeseries, timeseries)`	double	Calcula el coeficiente de correlación de Pearson para ambos`timeseries` . La serie temporal debe tener las mismas marcas de tiempo.

Función	Tipo de datos de salida	Descripción
`correlate_spearman (timeseries, timeseries)`	double	Calcula el coeficiente de correlación de Spearman para las dos. `timeseries` La serie temporal debe tener las mismas marcas de tiempo.

Consultas de ejemplo

Example

```
WITH cte_1 AS (
    SELECT INTERPOLATE_LINEAR(
        CREATE_TIME_SERIES(time, measure_value::double),
        SEQUENCE(min(time), max(time), 10m)) AS result
    FROM sample.DevOps
    WHERE measure_name = 'cpu_utilization'
    AND hostname = 'host-Hovjv' AND time > ago(1h)
    GROUP BY hostname, measure_name
),
cte_2 AS (
    SELECT INTERPOLATE_LINEAR(
        CREATE_TIME_SERIES(time, measure_value::double),
        SEQUENCE(min(time), max(time), 10m)) AS result
    FROM sample.DevOps
    WHERE measure_name = 'cpu_utilization'
    AND hostname = 'host-Hovjv' AND time > ago(1h)
    GROUP BY hostname, measure_name
)
SELECT correlate_pearson(cte_1.result, cte_2.result) AS result
FROM cte_1, cte_2
```

Funciones de filtrado y reducción

Amazon Timestream admite funciones para realizar operaciones de filtrado y reducción de datos de series temporales. En esta sección se proporciona información sobre el uso de Timestream para las funciones de LiveAnalytics filtrado y reducción, así como ejemplos de consultas.

Información de uso

Función	Tipo de datos de salida	Descripción
`filter(timeseries(T), function(T, Boolean))`	serie temporal (T)	Construye una serie temporal a partir de una serie temporal de entrada, utilizando valores para los que se devuelve lo pasado`function`. `true`
`reduce(timeseries(T), initialState S, inputFunction(S, T, S), outputFunction(S, R))`	R	Devuelve un valor único, reducido de la serie temporal. Se `inputFunction` invocará en cada elemento de la serie temporal en orden. Además de tomar el elemento actual, inputFunction toma el estado actual (inicialm `enteinitialState`) y devuelve el nuevo estado. Se `outputFunction` invocará para convertir el estado final en el valor resultante. `outputFunction` Puede ser una función de identidad.

Consultas de ejemplo

Example

Construya una serie temporal de CPU utilización de un anfitrión y filtre los puntos con una medición superior a 70:

```
WITH time_series_view AS (
    SELECT INTERPOLATE_LINEAR(
        CREATE_TIME_SERIES(time, ROUND(measure_value::double,2)),
            SEQUENCE(ago(15m), ago(1m), 10s)) AS cpu_user
    FROM sample.DevOps
```

```
    WHERE hostname = 'host-Hovjv' and measure_name = 'cpu_utilization'
        AND time > ago(30m)
    GROUP BY hostname
)
SELECT FILTER(cpu_user, x -> x.value > 70.0) AS cpu_above_threshold
from time_series_view
```

Example

Construya una serie temporal de CPU utilización de un anfitrión y determine la suma al cuadrado de las medidas:

```
WITH time_series_view AS (
    SELECT INTERPOLATE_LINEAR(
        CREATE_TIME_SERIES(time, ROUND(measure_value::double,2)),
            SEQUENCE(ago(15m), ago(1m), 10s)) AS cpu_user
    FROM sample.DevOps
    WHERE hostname = 'host-Hovjv' and measure_name = 'cpu_utilization'
        AND time > ago(30m)
    GROUP BY hostname
)
SELECT REDUCE(cpu_user,
    DOUBLE '0.0',
    (s, x) -> x.value * x.value + s,
    s -> s)
from time_series_view
```

Example

Construya una serie temporal de CPU utilización de un huésped y determine la fracción de muestras que están por encima del CPU umbral:

```
WITH time_series_view AS (
    SELECT INTERPOLATE_LINEAR(
        CREATE_TIME_SERIES(time, ROUND(measure_value::double,2)),
            SEQUENCE(ago(15m), ago(1m), 10s)) AS cpu_user
    FROM sample.DevOps
    WHERE hostname = 'host-Hovjv' and measure_name = 'cpu_utilization'
        AND time > ago(30m)
    GROUP BY hostname
)
SELECT ROUND(
```

```
    REDUCE(cpu_user,
      -- initial state
      CAST(ROW(0, 0) AS ROW(count_high BIGINT, count_total BIGINT)),
      -- function to count the total points and points above a certain threshold
      (s, x) -> CAST(ROW(s.count_high + IF(x.value > 70.0, 1, 0), s.count_total + 1) AS
 ROW(count_high BIGINT, count_total BIGINT)),
      -- output function converting the counts to fraction above threshold
      s -> IF(s.count_total = 0, NULL, CAST(s.count_high AS DOUBLE) / s.count_total)),
    4) AS fraction_cpu_above_threshold
from time_series_view
```

SQLsoporte

Timestream for LiveAnalytics admite algunas construcciones comunesSQL. Puede leer más a continuación.

Temas

- SELECT
- Soporte de subconsultas
- SHOWdeclaraciones
- DESCRIBEdeclaraciones
- UNLOAD

SELECT

SELECTlas declaraciones se pueden utilizar para recuperar datos de una o más tablas. El lenguaje de consulta de Timestream admite la siguiente sintaxis para SELECTlas sentencias:

```
[ WITH with_query [, ...] ]
            SELECT [ ALL | DISTINCT ] select_expr [, ...]
            [ function (expression) OVER (
            [ PARTITION BY partition_expr_list ]
            [ ORDER BY order_list ]
            [ frame_clause ] )
            [ FROM from_item [, ...] ]
            [ WHERE condition ]
            [ GROUP BY [ ALL | DISTINCT ] grouping_element [, ...] ]
            [ HAVING condition]
            [ { UNION | INTERSECT | EXCEPT } [ ALL | DISTINCT ] select ]
```

```
            [ ORDER BY order_list ]
            [ LIMIT [ count | ALL ] ]
```

where

- `function (expression)`es una de las funciones de ventana compatibles.
- `partition_expr_listes`:

```
expression | column_name [, expr_list ]
```

- `order_listes`:

```
expression | column_name [ ASC | DESC ]
[ NULLS FIRST | NULLS LAST ]
[, order_list ]
```

- `frame_clausees`:

```
ROWS | RANGE
{ UNBOUNDED PRECEDING | expression PRECEDING | CURRENT ROW } |
{BETWEEN
{ UNBOUNDED PRECEDING | expression { PRECEDING | FOLLOWING } |
CURRENT ROW}
AND
{ UNBOUNDED FOLLOWING | expression { PRECEDING | FOLLOWING } |
CURRENT ROW }}
```

- `from_itemes` uno de los siguientes:

```
table_name [ [ AS ] alias [ ( column_alias [, ...] ) ] ]
from_item join_type from_item [ ON join_condition | USING ( join_column [, ...] ) ]
```

- `join_typees` uno de los siguientes:

```
[ INNER ] JOIN
LEFT [ OUTER ] JOIN
RIGHT [ OUTER ] JOIN
FULL [ OUTER ] JOIN
```

- `grouping_elementes` uno de los siguientes:

```
()
```

```
expression
```

Soporte de subconsultas

Timestream admite subconsultas y predicados. `EXISTS` `IN` El `EXISTS` predicado determina si una subconsulta devuelve alguna fila. El `IN` predicado determina si los valores producidos por la subconsulta coinciden con los valores o la expresión de la cláusula IN. El lenguaje de consultas Timestream admite subconsultas correlacionadas y de otro tipo.

```
SELECT t.c1
FROM (VALUES 1, 2, 3, 4, 5) AS t(c1)
WHERE EXISTS
(SELECT t.c2
 FROM (VALUES 1, 2, 3) AS t(c2)
 WHERE t.c1= t.c2
)
ORDER BY t.c1
```

c1
1
2
3

```
SELECT t.c1
FROM (VALUES 1, 2, 3, 4, 5) AS t(c1)
WHERE t.c1 IN
(SELECT t.c2
 FROM (VALUES 2, 3, 4) AS t(c2)
)
ORDER BY t.c1
```

c1
2

c1
3
4

SHOWdeclaraciones

Puede ver todas las bases de datos de una cuenta mediante el `SHOW DATABASES` extracto. La sintaxis es la siguiente:

```
SHOW DATABASES [LIKE pattern]
```

donde la `LIKE` cláusula se puede utilizar para filtrar los nombres de las bases de datos.

Puede ver todas las tablas de una cuenta mediante la `SHOW TABLES` declaración. La sintaxis es la siguiente:

```
SHOW TABLES [FROM database] [LIKE pattern]
```

donde la `FROM` cláusula se puede usar para filtrar los nombres de las bases de datos y la `LIKE` cláusula se puede usar para filtrar los nombres de las tablas.

Puede ver todas las medidas de una tabla mediante la `SHOW MEASURES` sentencia. La sintaxis es la siguiente:

```
SHOW MEASURES FROM database.table [LIKE pattern]
```

donde la `FROM` cláusula se usará para especificar el nombre de la base de datos y la tabla y la `LIKE` cláusula se puede usar para filtrar los nombres de las medidas.

DESCRIBEdeclaraciones

Puede ver los metadatos de una tabla mediante la `DESCRIBE` instrucción. La sintaxis es la siguiente:

```
DESCRIBE database.table
```

donde `table` contiene el nombre de la tabla. La instrucción describe devuelve los nombres de las columnas y los tipos de datos de la tabla.

UNLOAD

Timestream for LiveAnalytics admite un UNLOAD comando como extensión de su SQL soporte. Los tipos de datos compatibles con UNLOAD se describen en. Tipos de datos compatibles Los unknown tipos time y no se aplican aUNLOAD.

```
UNLOAD (SELECT statement)
 TO 's3://bucket-name/folder'
 WITH ( option = expression [, ...] )
```

donde la opción es

```
{ partitioned_by = ARRAY[ col_name[,…] ]
 | format = [ '{ CSV | PARQUET }' ]
 | compression = [ '{ GZIP | NONE }' ]
 | encryption = [ '{ SSE_KMS | SSE_S3 }' ]
 | kms_key = '<string>'
 | field_delimiter ='<character>'
 | escaped_by = '<character>'
 | include_header = ['{true, false}']
 | max_file_size = '<value>'
}
```

SELECTdeclaración

La sentencia de consulta utilizada para seleccionar y recuperar datos de uno o más Timestream para LiveAnalytics tablas.

```
(SELECT column 1, column 2, column 3 from database.table
      where measure_name = "ABC" and timestamp between ago (1d) and now() )
```

Cláusula TO

```
TO 's3://bucket-name/folder'
```

o

```
TO 's3://access-point-alias/folder'
```

La TO cláusula de la UNLOAD declaración especifica el destino de la salida de los resultados de la consulta. Debe proporcionar la ruta completa, incluido el nombre del bucket de Amazon S3 o

Amazon S3 access-point-alias con la ubicación de la carpeta en Amazon S3 donde Timestream for LiveAnalytics escribe los objetos del archivo de salida. El bucket de S3 debe ser propiedad de la misma cuenta y estar en la misma región. Además del conjunto de resultados de la consulta, Timestream for LiveAnalytics escribe los archivos de manifiesto y metadatos en la carpeta de destino especificada.

PARTITIONEDCláusula _BY

```
partitioned_by = ARRAY [col_name[,…] , (default: none)
```

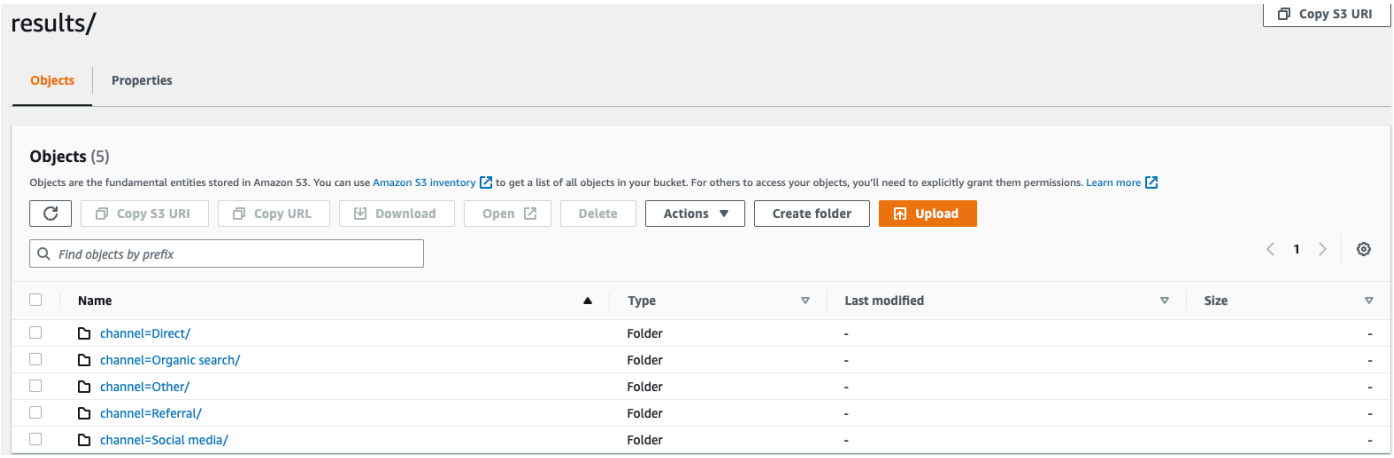
La `partitioned_by` cláusula se utiliza en las consultas para agrupar y analizar los datos de forma granular. Al exportar los resultados de la consulta al bucket de S3, puede elegir dividir los datos en función de una o más columnas de la consulta seleccionada. Al particionar los datos, los datos exportados se dividen en subconjuntos según la columna de partición y cada subconjunto se almacena en una carpeta independiente. Dentro de la carpeta de resultados que contiene los datos exportados, se crea automáticamente una subcarpeta`folder/results/partition column = partition value/`. Sin embargo, ten en cuenta que las columnas particionadas no se incluyen en el archivo de salida.

`partitioned_by`no es una cláusula obligatoria en la sintaxis. Si decide exportar los datos sin ninguna partición, puede excluir la cláusula de la sintaxis.

Example

Supongamos que está supervisando los datos del flujo de clics de su sitio web y tiene 5 canales de tráfico`direct`, a saber,`Social Media`, `Organic Search``Other`, y. `Referral` Al exportar los datos, puede optar por particionarlos mediante la columna`Channel`. Dentro de tu carpeta de datos`s3://bucketname/results`, tendrás cinco carpetas, cada una con su nombre de canal respectivo. Por ejemplo, `s3://bucketname/results/channel=Social Media/.` dentro de esta carpeta encontrarás los datos de todos los clientes que llegaron a tu sitio web a través del `Social Media` canal. Del mismo modo, dispondrá de otras carpetas para el resto de los canales.

Datos exportados particionados por columna de canales



FORMAT

```
format = [ '{ CSV | PARQUET }' , default: CSV
```

Las palabras clave para especificar el formato de los resultados de la consulta que se escriben en su bucket de S3. Puede exportar los datos como valores separados por comas (CSV) utilizando una coma (,) como delimitador predeterminado o en el formato Apache Parquet, un formato eficiente de almacenamiento en columnas abiertas para el análisis.

COMPRESSION

```
compression = [ '{ GZIP | NONE }' ], default: GZIP
```

Puede comprimir los datos exportados mediante un algoritmo de compresión GZIP o descomprimirlos especificando la opción. NONE

ENCRYPTION

```
encryption = [ '{ SSE_KMS | SSE_S3 }' ], default: SSE_S3
```

Los archivos de salida de Amazon S3 se cifran mediante la opción de cifrado que haya seleccionado. Además de sus datos, los archivos de manifiesto y metadatos también se cifran en función de la opción de cifrado que haya seleccionado. Actualmente, admitimos el KMS cifrado SSE _S3 y SSE _. SSE_S3 es un cifrado del lado del servidor en el que Amazon S3 cifra los datos mediante el cifrado estándar de cifrado avanzado () de 256 bits. AES SSE_ KMS es un cifrado del lado del servidor para cifrar los datos mediante claves administradas por el cliente.

KMS_KEY

```
kms_key = '<string>'
```

KMSLa clave es una clave definida por el cliente para cifrar los resultados de las consultas exportadas. KMSKey Management Service (AWS KMS) gestiona la AWS clave de forma segura y se utiliza para cifrar archivos de datos en Amazon S3.

FIELD_DELIMITER

```
field_delimiter ='<character>' , default: (,)
```

Al exportar los datos en CSV formato, este campo especifica un único ASCII carácter que se utiliza para separar los campos del archivo de salida, como un carácter vertical (|), una coma (,) o una tabulación (/t). El delimitador predeterminado de los CSV archivos es un carácter de coma. Si un valor de los datos contiene el delimitador elegido, el delimitador aparecerá entre comillas. Por ejemplo, si el valor de sus datos lo contiene`Time,stream`, este valor se indicará como `"Time,stream"` en los datos exportados. El carácter de comilla utilizado por Timestream LiveAnalytics son comillas dobles («).

Evite especificar el carácter de retorno (ASCII13, hexadecimal`0D`, texto '\ r') o el carácter de salto de línea (ASCII10, hexadecimal 0A, texto'\n') `FIELD_DELIMITER` si desea incluir encabezados en elCSV, ya que eso impedirá que muchos analizadores puedan analizar los encabezados correctamente en el resultado resultante. CSV

ESCAPED_BY

```
escaped_by = '<character>', default: (\)
```

Al exportar los datos en CSV formato, este campo especifica el carácter que debe tratarse como un carácter de escape en el archivo de datos escrito en el bucket de S3. El escape se produce en los siguientes escenarios:

1. Si el valor en sí contiene el carácter entre comillas («), se escapará utilizando un carácter de escape. Por ejemplo, si el valor es`Time"stream`, donde (\) es el carácter de escape configurado, se escapa como`Time\"stream`.
2. Si el valor contiene el carácter de escape configurado, se escapará. Por ejemplo, si el valor es`Time\stream`, se escapará como`Time\\stream`.

> **Note**
>
> Si la salida exportada contiene tipos de datos complejos, como matrices, filas o series temporales, se serializará como una cadena. JSON A continuación se muestra un ejemplo.

Tipo de datos	Valor real	Cómo se escapa el valor en el CSV formato [JSONcade na serializada]
Matriz	`[ 23,24,25 ]`	`"[23,24,25]"`
Fila	`( x=23.0, y=hello )`	`"{\"x\":23.0,\"y\":\"hello\"}"`
Serie temporal	`[ ( time=1970-01-01 00:00:00.000000010 , value=100.0 ), ( time=1970-01-01 00:00:00.000000012, value=120.0 ) ]`	`"[{\"time\":\"1970-01-01 00:00:00.000000010Z\",\"value\":100.0},{\"time\":\"1970-01-01 00:00:00.000000012Z\",\"value\":120.0}]"`

INCLUDE_HEADER

```
include_header = 'true' , default: 'false'
```

Al exportar los datos en CSV formato, este campo permite incluir los nombres de las columnas en la primera fila de los archivos de CSV datos exportados.

Los valores aceptados son «verdadero» y «falso» y el valor predeterminado es «falso». Las opciones de transformación de texto, por ejemplo, `escaped_by` `field_delimiter` se aplican también a los encabezados.

> **Note**
>
> Al incluir encabezados, es importante que no seleccione un carácter de retorno vertical (ASCII13, hexadecimal 0D, texto '\ r') o un carácter de salto de línea (ASCII10, hexadecimal 0A, texto'\n') como talesFIELD_DELIMITER, ya que esto impedirá que muchos analizadores puedan analizar correctamente los encabezados en el resultado resultante. CSV

MAX_FILE_SIZE

```
max_file_size = 'X[MB|GB]' , default: '78GB'
```

Este campo especifica el tamaño máximo de los archivos que la UNLOAD declaración crea en Amazon S3. La UNLOAD declaración puede crear varios archivos, pero el tamaño máximo de cada archivo escrito en Amazon S3 será aproximadamente el especificado en este campo.

El valor del campo debe estar comprendido entre 16 MB y 78 GB, ambos inclusive. Puede especificarlo en números enteros, por ejemplo12GB, o en decimales, como 0.5GB o24.7MB. El valor predeterminado es 78 GB.

El tamaño real del archivo es aproximado cuando se escribe el archivo, por lo que es posible que el tamaño máximo real no sea exactamente igual al número que especifique.

Logical operators (Operadores lógicos)

Timestream for LiveAnalytics admite los siguientes operadores lógicos.

Operador	Descripción	Ejemplo
AND	Verdadero si ambos valores son verdaderos	a AND b
OR	Verdadero si alguno de los valores es verdadero	a O b
NOT	Verdadero si el valor es falso	NOTa

- El resultado de una AND comparación puede ser NULL si uno o ambos lados de la expresión sonNULL.
- Si al menos un lado de un AND operador es FALSE el resultado de la expresión. FALSE
- El resultado de una OR comparación puede ser NULL si uno o ambos lados de la expresión son NULL iguales.
- Si al menos un lado de un OR operador es TRUE el resultado de la expresión. TRUE
- El complemento lógico de NULL esNULL.

La siguiente tabla de verdad muestra el manejo de NULL in AND yOR:

A	B	A y b	A o b
null	nulo	nulo	null
false	null	false	null
null	false	false	null
true	null	null	true
null	true	null	true
false	false	false	false
true	false	false	true
false	true	false	true
true	true	true	true

La siguiente tabla de verdad demuestra el manejo de NULL inNOT:

A	No es un
null	null
true	false

A	No es un
false	true

Operadores de comparación

Timestream for LiveAnalytics admite los siguientes operadores de comparación.

Operador	Descripción
<	Menor que
>	Mayor que
<=	Menor o igual que
>=	Mayor o igual que
=	Igualdad
<>	Desigualdad
!=	Desigualdad

Note

- El `BETWEEN` operador comprueba si un valor está dentro de un rango especificado. La sintaxis es la siguiente:

  ```
  BETWEEN min AND max
  ```

  La presencia de `NULL` una `NOT BETWEEN` sentencia `BETWEEN` o dará como resultado que la sentencia se evalúe `NULL` como
- `IS NULL` y `IS NOT NULL` los operadores comprueban si un valor es nulo (indefinido). Si se utiliza `NULL` con, `IS NULL` el resultado es verdadero.
- EnSQL, un `NULL` valor significa un valor desconocido.

Funciones de comparación

Timestream for LiveAnalytics admite las siguientes funciones de comparación.

Temas

mejor ()

La función greatest () devuelve el mayor de los valores proporcionados. Devuelve `NULL` si alguno de los valores proporcionados lo es`NULL`. La sintaxis es la siguiente.

```
greatest(value1, value2, ..., valueN)
```

mínimo ()

La función least () devuelve el menor de los valores proporcionados. Devuelve `NULL` si alguno de los valores proporcionados lo es`NULL`. La sintaxis es la siguiente.

```
least(value1, value2, ..., valueN)
```

ALL(), ANY () y SOME ()

Los `ALL` `SOME` cuantificadores `ANY` y se pueden usar junto con los operadores de comparación de la siguiente manera.

Expression	Significado
A = ALL (...)	Se evalúa como verdadero cuando A es igual a todos los valores.
A <> ALL (...)	Se evalúa como verdadero cuando A no coincide con ningún valor.
A < ALL (...)	Se evalúa como verdadero cuando A es menor que el valor más pequeño.

Expression	Significado
A = ANY (...)	Se evalúa como verdadero cuando A es igual a cualquiera de los valores.
A <> ANY (...)	Se evalúa como verdadero cuando A no coincide con uno o más valores.
A < ANY (...)	Se evalúa como verdadero cuando A es menor que el valor mayor.

Ejemplos y notas de uso

> **Note**
>
> Cuando se usa`ALL`, `ANY` o`SOME`, se `VALUES` debe usar la palabra clave si los valores de comparación son una lista de literales.

Ejemplo: **`ANY()`**

Un ejemplo de una declaración `ANY()` de consulta es el siguiente.

```
SELECT 11.7 = ANY (VALUES 12.0, 13.5, 11.7)
```

Una sintaxis alternativa para la misma operación es la siguiente.

```
SELECT 11.7 = ANY (SELECT 12.0 UNION ALL SELECT 13.5 UNION ALL SELECT 11.7)
```

En este caso, `ANY()` se evalúa como. `True`

Ejemplo: **`ALL()`**

Un ejemplo de una sentencia `ALL()` de consulta es el siguiente.

```
SELECT 17 < ALL (VALUES 19, 20, 15);
```

Una sintaxis alternativa para la misma operación es la siguiente.

```
SELECT 17 < ALL (SELECT 19 UNION ALL SELECT 20 UNION ALL SELECT 15);
```

En este caso, `ALL()` se evalúa como. `False`

Ejemplo: **`SOME()`**

Un ejemplo de una sentencia `SOME()` de consulta es el siguiente.

```
SELECT 50 >= SOME (VALUES 53, 77, 27);
```

Una sintaxis alternativa para la misma operación es la siguiente.

```
SELECT 50 >= SOME (SELECT 53 UNION ALL SELECT 77 UNION ALL SELECT 27);
```

En este caso, `SOME()` se evalúa como. `True`

Expresiones condicionales

Timestream for LiveAnalytics admite las siguientes expresiones condicionales.

Temas

- La declaración CASE
- La sentencia IF
- ¿La declaración COALESCE
- La declaración NULLIF
- La TRY declaración

La declaración CASE

La CASEsentencia busca cada expresión de valor de izquierda a derecha hasta encontrar una que sea igual `aexpression`. Si encuentra una coincidencia, se devuelve el resultado del valor coincidente. Si no se encuentra ninguna coincidencia, se devuelve el resultado de la `ELSE` cláusula, si existe; de lo contrario, `null` se devuelve. La sintaxis es la siguiente:

```
CASE expression
```

```
    WHEN value THEN result
    [ WHEN ... ]
    [ ELSE result ]
END
```

Timestream también admite la siguiente sintaxis para CASElas sentencias. En esta sintaxis, la forma «buscada» evalúa cada condición booleana de izquierda a derecha hasta que aparezca una `true` y devuelve el resultado coincidente. Si no hay condiciones`true`, se devuelve el resultado de la `ELSE` cláusula si existe; de lo contrario`null`, se devuelve. Consulte la sintaxis alternativa a continuación:

```
CASE
    WHEN condition THEN result
    [ WHEN ... ]
    [ ELSE result ]
END
```

La sentencia IF

La sentencia IF evalúa una condición como verdadera o falsa y devuelve el valor adecuado. Timestream admite las dos siguientes representaciones sintácticas para IF:

```
if(condition, true_value)
```

Esta sintaxis evalúa y devuelve `true_value` si la condición es`true`; de lo contrario, `null` se devuelve y no `true_value` se evalúa.

```
if(condition, true_value, false_value)
```

Esta sintaxis evalúa y devuelve `true_value` si la condición es; de lo contrario`true`, evalúa y devuelve. `false_value`

Ejemplos

```
SELECT
  if(true, 'example 1'),
  if(false, 'example 2'),
  if(true, 'example 3 true', 'example 3 false'),
  if(false, 'example 4 true', 'example 4 false')
```

_col0	_col1	_col2	_col3
example 1	- null	example 3 true	example 4 false

¿La declaración COALESCE

COALESCEdevuelve el primer valor no nulo de una lista de argumentos. La sintaxis es la siguiente:

```
coalesce(value1, value2[,...])
```

La declaración NULLIF

La sentencia IF evalúa una condición como verdadera o falsa y devuelve el valor apropiado. Timestream admite las dos siguientes representaciones sintácticas para IF:

NULLIFdevuelve null si `value1` es igual `avalue2`; de lo contrario, devuelve. `value1` La sintaxis es la siguiente:

```
nullif(value1, value2)
```

La TRY declaración

La TRYfunción evalúa una expresión y trata ciertos tipos de errores `null` devolviéndola. La sintaxis es la siguiente:

```
try(expression)
```

Funciones de conversión

Timestream for LiveAnalytics admite las siguientes funciones de conversión.

Temas

- cast()
- try_cast ()

cast()

La sintaxis de la función de conversión para convertir explícitamente un valor como un tipo es la siguiente.

```
cast(value AS type)
```

try_cast ()

Timestream for LiveAnalytics también admite la función try_cast, que es similar a cast, pero devuelve null si la transmisión falla. La sintaxis es la siguiente.

```
try_cast(value AS type)
```

Operadores matemáticos

Timestream for LiveAnalytics admite los siguientes operadores matemáticos.

Operador	Descripción
+	Suma
-	Resta
*	Multiplicación
/	División (la división de enteros realiza el truncamiento)
%	Módulo (resto)

Funciones matemáticas

Timestream for LiveAnalytics admite las siguientes funciones matemáticas.

Función	Tipo de datos de salida	Descripción
abs (x)	[igual que la entrada]	Devuelve el valor absoluto de x.
cbrt (x)	double	Devuelve la raíz cúbica de x.
techo (x) o techo (x)	[igual que la entrada]	Devuelve x redondeado al número entero más cercano.
grados (x)	double	Convierte el ángulo x en radianes en grados.
e ()	double	Devuelve el número de Euler constante.
exp (x)	double	Devuelve el número de Euler elevado a la potencia de x.
piso (x)	[igual que la entrada]	Devuelve x redondeado hacia abajo al entero más cercano.
from_base (cadena, raíz)	bigint	Devuelve el valor de la cadena interpretada como un número base-base.
ln (x)	double	Devuelve el logaritmo natural de x.
log2 (x)	double	Devuelve el logaritmo en base 2 de x.
log10 (x)	double	Devuelve el logaritmo en base 10 de x.
mod (n, m)	[igual que la entrada]	Devuelve el módulo (resto) de n dividido entre m.
pi ()	double	Devuelve la constante Pi.

Función	Tipo de datos de salida	Descripción
pow (x, p) o potencia (x, p)	double	Devuelve x elevado a la potencia de p.
radianes (x)	double	Convierte el ángulo x en grados en radianes.
rand () o random ()	double	Devuelve un valor pseudoale atorio en el rango de 0.0 1.0.
aleatorio (n)	[igual que la entrada]	Devuelve un número pseudoaleatorio entre 0 y n (exclusivo).
redondo (x)	[igual que la entrada]	Devuelve x redondeado al entero más cercano.
redondear (x, d)	[igual que la entrada]	Devuelve x redondeado a d decimales.

Función	Tipo de datos de salida	Descripción
signo (x)	[igual que la entrada]	Devuelve la función signum de x, es decir: • 0 si el argumento es 0 • 1 si el argumento es mayor que 0 • -1 si el argumento es menor que 0. Para argumentos dobles, la función devuelve además: • NaN si el argumento es NaN • 1 si el argumento es +Infinity • -1 si el argumento es -Infinity.
sqrt (x)	double	Devuelve la raíz cuadrada de x.
to_base (x, radix)	varchar	Devuelve la representación base-base de x.
truncar (x)	double	Devuelve x redondeada a número entero dejando caer los dígitos después de la coma decimal.
acos (x)	double	Devuelve el arco coseno de x.
cuenca (x)	double	Devuelve el arco seno de x.

Función	Tipo de datos de salida	Descripción
atan (x)	double	Devuelve el arco tangente de x.
en un 2 (y, x)	double	Devuelve el arco tangente de y/x.
cos (x)	double	Devuelve el coseno de x.
cosh (x)	double	Devuelve el coseno hiperbóli co de x.
sin (x)	double	Devuelve el seno de x.
tan (x)	double	Devuelve la tangente de x.
tanh (x)	double	Devuelve la tangente hiperbóli ca de x.
infinito ()	double	Devuelve la constante que representa el infinito positivo.
is_finite (x)	boolean	Determina si x es finito.
is_infinite (x)	boolean	Determina si x es infinito.
is_nan (x)	boolean	Determine si x es. not-a-num ber
nan ()	double	Devuelve la constante que representa not-a-number.

Operadores de cadena

Timestream for LiveAnalytics admite el `||` operador para concatenar una o más cadenas.

Funciones de cadena

> **Note**
>
> Se supone que el tipo de datos de entrada de estas funciones es varchar a menos que se especifique lo contrario.

Función	Tipo de datos de salida	Descripción
chr (n)	varchar	Devuelve el punto n del código Unicode como un varchar.
punto de código (x)	integer	Devuelve el punto de código Unicode del único carácter de str.
concat (x1,..., xN)	varchar	Devuelve la concatenación de x1, x2,..., xN.
Hamming_distance (x1, x2)	bigint	Devuelve la distancia de Hamming de x1 y x2, es decir, el número de posicione s en las que los caractere s correspondientes son diferentes. Tenga en cuenta que las dos entradas de varchar deben tener la misma longitud.
longitud (x)	bigint	Devuelve la longitud de x en caracteres.
levenshtein_distance (x1, x2)	bigint	Devuelve la distancia de edición de Levenshtein de x1 y x2, es decir, el número mínimo de ediciones de un solo carácter (inserciones,

Función	Tipo de datos de salida	Descripción
		eliminaciones o sustituciones) necesarias para cambiar x1 a x2.
inferior (x)	varchar	Convierte x en minúsculas.
carga (x1, tamaño grande, x2)	varchar	Teclas izquierdas x1 para dimensionar los caractere s con x2. Si el tamaño es inferior a x1, el resultado se trunca para ajustar el tamaño de los caracteres. El tamaño no debe ser negativo y x2 no debe estar vacío.
ltrim (x)	varchar	Elimina los espacios en blanco iniciales de x.
reemplazar (x1, x2)	varchar	Elimina todas las instancias de x2 de x1.
reemplazar (x1, x2, x3)	varchar	Sustituye todas las instancias de x2 por x3 en x1.
Reverso (x)	varchar	Devuelve x con los caracteres en orden inverso.
carretera (x1, tamaño grande, x2)	varchar	Tecla derecha x1 para dimensionar los caractere s con x2. Si el tamaño es inferior a x1, el resultado se trunca para ajustar el tamaño de los caracteres. El tamaño no debe ser negativo y x2 no debe estar vacío.

Función	Tipo de datos de salida	Descripción
recortar (x)	varchar	Elimina los espacios en blanco finales de x.
dividir (x1, x2)	array(varchar)	Divide x1 en el delimitador x2 y devuelve una matriz.
dividir (x1, x2, límite de bits)	array(varchar)	Divide x1 en el delimitador x2 y devuelve una matriz. El último elemento de la matriz siempre contiene todo lo que queda en la x1. El límite debe ser un número positivo.
split_part (x1, x2, bigint pos)	varchar	Divide x1 en el delimitador x2 y devuelve el campo varchar en pos. Los índices de campo comienzan por 1. Si pos es mayor que el número de campos, se devuelve un valor nulo.
strpos (x1, x2)	bigint	Devuelve la posición inicial de la primera instancia de x2 en x1. Las posiciones comienzan con 1. Si no se encuentra, se devuelve 0.
strpos (x1, x2, instancia de bigint)	bigint	Devuelve la posición de la enésima instancia de x2 en x1. La instancia debe ser un número positivo. Las posiciones comienzan con 1. Si no se encuentra, se devuelve 0.

Función	Tipo de datos de salida	Descripción
strrpos (x1, x2)	bigint	Devuelve la posición inicial de la última instancia de x2 en x1. Las posiciones comienzan con 1. Si no se encuentra, se devuelve 0.
strrpos (x1, x2, instancia de bigint)	bigint	Devuelve la posición de la enésima instancia de x2 en x1 empezando por el final de x1. La instancia debe ser un número positivo. Las posiciones comienzan con 1. Si no se encuentra, se devuelve 0.
posición (x2 EN x1)	bigint	Devuelve la posición inicial de la primera instancia de x2 en x1. Las posiciones comienzan con 1. Si no se encuentra, se devuelve 0.
substr (x, inicio bigint)	varchar	Devuelve el resto de x desde la posición inicial inicial. Las posiciones comienzan con 1. Una posición inicial negativa se interpreta como relativa al final de x.
substr (x, inicio bigint, lente bigint)	varchar	Devuelve una subcadena de x de longitud len desde la posición inicial start. Las posiciones comienzan con 1. Una posición inicial negativa se interpreta como relativa al final de x.

Función	Tipo de datos de salida	Descripción
recortar (x)	varchar	Elimina los espacios en blanco iniciales y finales de x.
superior (x)	varchar	Convierte x en mayúsculas.

Operadores de matriz

Timestream for LiveAnalytics admite los siguientes operadores de matriz.

Operador	Descripción
[]	Acceda a un elemento de una matriz donde el primer índice comience en 1.
||	Concatena una matriz con otra matriz o elemento del mismo tipo.

Funciones de matriz

Timestream for LiveAnalytics admite las siguientes funciones de matriz.

Función	Tipo de datos de salida	Descripción
array_distinct (x)	matriz	Elimine los valores duplicados de la matriz x. `SELECT array_distinct(ARRAY[1,2,2,3])` Ejemplo de resultado: `[ 1,2,3 ]`
array_intersect (x, y)	matriz	Devuelve una matriz de los elementos en la intersección de x e y, sin duplicados.

Función	Tipo de datos de salida	Descripción
		`SELECT array_intersect(ARRAY[1,2,3], ARRAY[3,4,5])` Ejemplo de resultado: `[ 3 ]`
array_union (x, y)	matriz	Devuelve una matriz de los elementos en la unión de x e y, sin duplicados. `SELECT array_union(ARRAY[1,2,3], ARRAY[3,4,5])` Ejemplo de resultado: `[ 1,2,3,4,5 ]`
array_except (x, y)	matriz	Devuelve una matriz de elementos en x pero no en y, sin duplicados. `SELECT array_except(ARRAY[1,2,3], ARRAY[3,4,5])` Ejemplo de resultado: `[ 1,2 ]`

Función	Tipo de datos de salida	Descripción
array_join (x, delimiter, null_replacement)	varchar	Concatena los elementos de la matriz dada mediante el delimitador y una cadena opcional para reemplazar los valores nulos. `SELECT array_joi n(ARRAY[1,2,3], ';', '')` Ejemplo de resultado: `1;2;3`
array_max (x)	igual que los elementos de la matriz	Devuelve el valor máximo de la matriz de entrada. `SELECT array_max (ARRAY[1,2,3])` Ejemplo de resultado: 3
array_min (x)	igual que los elementos de la matriz	Devuelve el valor mínimo de la matriz de entrada. `SELECT array_min (ARRAY[1,2,3])` Ejemplo de resultado: 1

Función	Tipo de datos de salida	Descripción
array_position (x, elemento)	bigint	Devuelve la posición de la primera aparición del elemento en la matriz x (o 0 si no se encuentra). `SELECT array_position(ARRAY[3,4,5,9], 5)` Ejemplo de resultado: 3
array_remove (x, elemento)	matriz	Elimine todos los elementos que sean iguales al elemento de la matriz x. `SELECT array_remove(ARRAY[3,4,5,9], 4)` Ejemplo de resultado: `[ 3,5,9 ]`
array_sort (x)	matriz	Ordena y devuelve la matriz x. Los elementos de x deben poder ordenarse. Los elementos nulos se colocarán al final de la matriz devuelta. `SELECT array_sort(ARRAY[6,8,2,9,3])` Ejemplo de resultado: `[ 2,3,6,8,9 ]`

Función	Tipo de datos de salida	Descripción
arrays_overlap (x, y)	boolean	Comprueba si las matrices x e y tienen en común algún elemento que no sea nulo. Devuelve null si no hay elementos no nulos en común pero alguna de las matrices contiene nulos. `SELECT arrays_overlap(ARRAY[6,8,2,9,3], ARRAY[6,8])` Ejemplo de resultado: `true`
cardinalidad (x)	bigint	Devuelve el tamaño de la matriz x. `SELECT cardinality(ARRAY[6,8,2,9,3])` Ejemplo de resultado: 5
concat (matriz1, matriz2,..., matrizN)	matriz	Concatena las matrices matriz1, matriz2,..., arrayN. `SELECT concat(ARRAY[6,8,2,9,3], ARRAY[11,32], ARRAY[6,8,2,0,14])` Resultado de ejemplo: `[ 6,8,2,9,3,11,32,6,8,2,0,14 ]`

Función	Tipo de datos de salida	Descripción
element_at (matriz (E), índice)	E	Devuelve el elemento de la matriz en un índice dado. Si el índice es inferior a 0, element_at accede a los elementos desde el último hasta el primero. `SELECT element_at(ARRAY[6,8,2,9,3], 1)` Ejemplo de resultado: 6
repetir (elemento, recuento)	matriz	Repita el elemento para contar los tiempos. `SELECT repeat(1, 3)` Ejemplo de resultado: `[ 1,1,1 ]`
inverso (x)	matriz	Devuelve una matriz que tiene el orden inverso al de la matriz x. `SELECT reverse(ARRAY[6,8,2,9,3])` Ejemplo de resultado: `[ 3,9,2,8,6 ]`

Función	Tipo de datos de salida	Descripción

secuencia (inicio, parada)	matriz (bigint)	Genera una secuencia de números enteros desde el principio hasta la parada, incrementándola en 1 si start es menor o igual que stop; de lo contrario, en -1.
		`SELECT sequence(3, 8)`
		Ejemplo de resultado:
		`[ 3,4,5,6,7,8 ]`

secuencia (inicio, parada, paso)	matriz (bigint)	Genera una secuencia de números enteros de principio a fin, incrementándola paso a paso.
		`SELECT sequence(3, 15, 2)`
		Ejemplo de resultado:
		`[ 3,5,7,9,11,13,15 ]`

Función	Tipo de datos de salida	Descripción
secuencia (inicio, parada)	matriz (marca de tiempo)	Genere una secuencia de marcas de tiempo desde la fecha de inicio hasta la fecha de finalización, incrementándolas en 1 día. `SELECT sequence( '2023-04-02 19:26:12.941000000', '2023-04-06 19:26:12.941000000', 1d)` Ejemplo de resultado: `[ 2023-04-02 19:26:12.941000000 ,2023-04-03 19:26:12.941000000 ,2023-04-04 19:26:12.941000000 ,2023-04-05 19:26:12.941000000 ,2023-04-06 19:26:12.941000000 ]`

Función	Tipo de datos de salida	Descripción
secuencia (inicio, parada, paso)	matriz (marca de tiempo)	Genere una secuencia de marcas de tiempo desde el principio hasta el final, incrementándolas paso a paso. El tipo de datos del paso es el intervalo. `SELECT sequence( '2023-04-02 19:26:12.941000000', '2023-04-10 19:26:12.941000000', 2d)` Ejemplo de resultado: `[ 2023-04-02 19:26:12.941000000 ,2023-04-04 19:26:12.941000000 ,2023-04-06 19:26:12.941000000 ,2023-04-08 19:26:12.941000000 ,2023-04-10 19:26:12.941000000 ]`
shuffle (x)	matriz	Genera una permutación aleatoria de la matriz x dada. `SELECT shuffle(ARRAY[6,8,2,9,3])` Ejemplo de resultado: `[ 6,3,2,9,8 ]`

Función	Tipo de datos de salida	Descripción
rebanada (x, inicio, longitud)	matriz	Subestablece la matriz x empezando por el inicio del índice (o empezando por el final si el inicio es negativo) con una longitud igual a la longitud. `SELECT slice(ARRAY[6,8,2,9,3], 1, 3)` Ejemplo de resultado: `[ 6,8,2 ]`
zip (matriz1, matriz2 [,...])	matriz (fila)	Fusiona las matrices dadas, por elementos, en una sola matriz de filas. Si los argumentos tienen una longitud irregular, los valores faltantes se rellenan con. NULL `SELECT zip(ARRAY[6,8,2,9,3], ARRAY[15, 24])` Resultado de ejemplo: `[ ( 6, 15 ),( 8, 24 ), ( 2, - ),( 9, - ),( 3, - ) ]`

Bitwise functions (Funciones Bitwise)

Timestream for LiveAnalytics admite las siguientes funciones bit a bit.

Función	Tipo de datos de salida	Descripción

bit_count (bigint, bigint)	bigint (el complemento de dos)	Devuelve el recuento de bits del primer parámetro de bigint, donde el segundo parámetro es un entero con signo de bits, como 8 o 64.
		`SELECT bit_count(19, 8)`
		Ejemplo de resultado: 3
		`SELECT bit_count(19, 2)`
		Ejemplo de resultado: `Number must be represent able with the bits specified. 19 can not be represented with 2 bits`

bitwise_and (bigint, bigint)	bigint (complemento de dos)	Devuelve el valor bit a bit de los parámetros AND de bigint. `SELECT bitwise_and(12, 7)` Ejemplo de resultado: 4
bitwise_not (bigint)	bigint (el complemento de dos)	Devuelve el valor bit a bit NOT del parámetro bigint. `SELECT bitwise_not(12)` Ejemplo de resultado: -13

Función	Tipo de datos de salida	Descripción
bitwise_or (bigint, bigint)	bigint (complemento de dos)	Devuelve el OR bit a bit de los parámetros de bigint. `SELECT bitwise_or(12, 7)` Ejemplo de resultado: 15
bitwise_xor (bigint, bigint)	bigint (el complemento de dos)	Devuelve el valor bit a bit de los parámetros XOR de bigint. `SELECT bitwise_xor(12, 7)` Ejemplo de resultado: 11

Regular expression functions (Funciones de expresión regular)

La expresión regular funciona en Timestream para LiveAnalytics admitir la sintaxis de patrones de Java. Timestream for LiveAnalytics admite las siguientes funciones de expresiones regulares.

Función	Tipo de datos de salida	Descripción
regexp_extract_all (cadena, patrón)	array(varchar)	Devuelve las subcadenas que coinciden con el patrón de expresión regular de la cadena. `SELECT regexp_extract_all('example expect complex', 'ex\w')` Ejemplo de resultado: `[ exa,exp ]`

Función	Tipo de datos de salida	Descripción
regexp_extract_all (cadena, patrón, grupo)	array(varchar)	Busca todas las aparicion es del patrón de expresión regular en una cadena y devuelve el grupo numérico del grupo capturador. `SELECT regexp_ex tract_all('example expect complex', '(ex) (\w)', 2)` Ejemplo de resultado: `[ a,p ]`
regexp_extract (cadena, patrón)	varchar	Devuelve la primera subcadena que coincide con el patrón de expresión regular de la cadena. `SELECT regexp_ex tract('example expect', 'ex\w')` Ejemplo de resultado: `exa`

Función	Tipo de datos de salida	Descripción
regexp_extract (cadena, patrón, grupo)	varchar	Busca la primera aparición del patrón de expresión regular en la cadena y devuelve el grupo numérico del grupo que lo capturó. `SELECT regexp_extract('example expect', '(ex)(\w)', 2)` Ejemplo de resultado: a
regexp_like (cadena, patrón)	boolean	Evalúa el patrón de expresión regular y determina si está contenido dentro de una cadena. Esta función es similar al LIKE operador, excepto en que el patrón solo debe estar contenido dentro de la cadena, en lugar de tener que coincidir con toda la cadena. En otras palabras, realiza una operación de contención en lugar de una operación de coincidencia. Puedes hacer coincidir toda la cadena anclando el patrón con ^ y $. `SELECT regexp_like('example', 'ex')` Ejemplo de resultado: `true`

Función	Tipo de datos de salida	Descripción
regexp_replace (cadena, patrón)	varchar	Elimina de la cadena todas las instancias de la subcadena que coincidan con el patrón de expresión regular. `SELECT regexp_replace('example expect', 'expect')` Ejemplo de resultado: `example`
regexp_replace (cadena, patrón, reemplazo)	varchar	Reemplaza cada instancia de la subcadena que coincida con el patrón de expresion es regulares de la cadena por un reemplazo. Se puede hacer referencia a los grupos de captura en lugar de usar $g para un grupo numerado o $ {name} para un grupo con nombre. Se puede incluir un signo de dólar ($) en la pieza sustituida por una barra invertida (\ $). `SELECT regexp_replace('example expect', 'expect', 'surprise')` Ejemplo de resultado: `example surprise`

Función	Tipo de datos de salida	Descripción
regexp_replace (cadena, patrón, función)	varchar	Reemplaza todas las instancias de la subcadena que coincidan con el patrón de expresión regular en la cadena mediante la función. La función de expresión lambda se invoca para cada coincidencia y los grupos de captura se pasan como una matriz. Los números de los grupos de captura comienzan por el número uno; no hay ningún grupo para toda la coincidencia (si lo necesita, ponga paréntesis entre paréntesis toda la expresión). `SELECT regexp_replace('example', '(\w)', x -> upper(x[1]))` Ejemplo de resultado: `EXAMPLE`

Función	Tipo de datos de salida	Descripción
regexp_split (cadena, patrón)	array(varchar)	Divide la cadena utilizando el patrón de expresión regular y devuelve una matriz. Se conservan las cadenas vacías finales. `SELECT regexp_split('example', 'x')` Ejemplo de resultado: `[ e,ample ]`

Operadores de fecha y hora

> **Note**
>
> Timestream for no LiveAnalytics admite valores de hora negativos. Cualquier operación que dé como resultado un tiempo negativo genera un error.

Timestream for LiveAnalytics admite las siguientes operaciones en `timestampsdates`, y. `intervals`

Operador	Descripción
+	Suma
-	Resta

Temas

- Operaciones
- Suma
- Resta

Operaciones

El tipo de resultado de una operación se basa en los operandos. `3s`Se pueden utilizar literales de intervalo como `1day` y.

```
SELECT date '2022-05-21' + interval '2' day
```

```
SELECT date '2022-05-21' + 2d
```

```
SELECT date '2022-05-21' + 2day
```

Ejemplo de resultado para cada uno: `2022-05-23`

Las unidades de intervalo incluyen `second` `minutehour`,`day`,`week`,`month`, `yyear`. Sin embargo, en algunos casos no todas son aplicables. Por ejemplo, los segundos, los minutos y las horas no se pueden sumar ni restar de una fecha.

```
SELECT interval '4' year + interval '2' month
```

Ejemplo de resultado: `4-2`

```
SELECT typeof(interval '4' year + interval '2' month)
```

Ejemplo de resultado: `interval year to month`

El tipo de resultado de las operaciones de intervalo puede ser `'interval year to month'` o `'interval day to second'` depender de los operandos. Los intervalos se pueden sumar o restar de y. `dates` `timestamps` Sin embargo, un `date` o `timestamp` no se puede sumar ni restar de un o. `date` `timestamp` Para buscar intervalos o duraciones relacionados con fechas o marcas horarias, consulte `date_diff` y funciones relacionadas en. Intervalo y duración

Suma

Example

```
SELECT date '2022-05-21' + interval '2' day
```

Ejemplo de resultado: `2022-05-23`

Example

```
SELECT typeof(date '2022-05-21' + interval '2' day)
```

Ejemplo de resultado: `date`

Example

```
SELECT interval '2' year + interval '4' month
```

Ejemplo de resultado: `2-4`

Example

```
SELECT typeof(interval '2' year + interval '4' month)
```

Ejemplo de resultado: `interval year to month`

Resta

Example

```
SELECT timestamp '2022-06-17 01:00' - interval '7' hour
```

Ejemplo de resultado: `2022-06-16 18:00:00.000000000`

Example

```
SELECT typeof(timestamp '2022-06-17 01:00' - interval '7' hour)
```

Ejemplo de resultado: `timestamp`

Example

```
SELECT interval '6' day - interval '4' hour
```

Ejemplo de resultado: `5 20:00:00.000000000`

Example

```
SELECT typeof(interval '6' day - interval '4' hour)
```

Ejemplo de resultado: `interval day to second`

Funciones de fecha y hora

> **Note**
>
> Timestream for no LiveAnalytics admite valores de hora negativos. Cualquier operación que dé como resultado un tiempo negativo genera un error.

Timestream for LiveAnalytics utiliza la UTC zona horaria para la fecha y la hora. Timestream admite las siguientes funciones de fecha y hora.

Temas

- General y conversión
- Intervalo y duración
- Formatear y analizar
- Extracción

General y conversión

Timestream for LiveAnalytics admite las siguientes funciones generales y de conversión de fecha y hora.

Función	Tipo de datos de salida	Descripción
fecha_actual	date	Devuelve la fecha actual en. UTC No se utilizan paréntesis. `SELECT current_date` Ejemplo de resultado: `2022-07-07`

Función	Tipo de datos de salida	Descripción
		Note También es una palabra clave reservada. Para obtener una lista de palabras clave reservadas, consulte Palabras clave reservadas.
hora_actual	hora	Devuelve la hora actual en. UTC No se utilizan paréntesis. `SELECT current_time` Ejemplo de resultado: `17:41:52.827000000` **Note** También es una palabra clave reservada. Para obtener una lista de palabras clave reservadas, consulte Palabras clave reservadas.

Función	Tipo de datos de salida	Descripción
current_timestamp o now ()	Marca de tiempo	Devuelve la marca de tiempo actual en. UTC `SELECT current_timestamp` Ejemplo de resultado: `2022-07-07 17:42:32.939000000` **Note** También es una palabra clave reservada. Para obtener una lista de palabras clave reservadas, consulte Palabras clave reservadas.
current_timezone ()	varchar El valor será '.' UTC	Timestream usa la UTC zona horaria para la fecha y la hora. `SELECT current_timezone()` Ejemplo de resultado: UTC
fecha (varchar (x)), fecha (marca de tiempo)	date	`SELECT date(TIMESTAMP '2022-07-07 17:44:43.771000000')` Ejemplo de resultado: `2022-07-07`

Función	Tipo de datos de salida	Descripción
last_day_of_month (marca de tiempo), last_day_of_month (fecha)	date	`SELECT last_day_of_month(TIMESTAMP '2022-07-07 17:44:43.771000000')` Ejemplo de resultado: `2022-07-31`
from_iso8601_timestamp (cadena)	Marca de tiempo	Analiza la marca de tiempo 8601 en un formato de marca de tiempo interno. ISO `SELECT from_iso8601_timestamp('2022-06-17T08:04:05.000000000+05:00')` Ejemplo de resultado: `2022-06-17 03:04:05.000000000`
from_iso8601_date (cadena)	date	Analiza la cadena de fecha ISO 8601 para convertir la en un formato de marca de tiempo interno para las 00:00:00 de la fecha especificada. UTC `SELECT from_iso8601_date('2022-07-17')` Resultado de ejemplo: `2022-07-17`

Función	Tipo de datos de salida	Descripción
to_iso8601 (marca de tiempo), to_iso8601 (fecha)	varchar	Devuelve una cadena ISO con formato 8601 para la entrada. `SELECT to_iso8601(from_iso8601_date('2022-06-17'))` Ejemplo de resultado: `2022-06-17`
from_milisegundos (bigint)	Marca de tiempo	`SELECT from_milliseconds(1)` Ejemplo de resultado: `1970-01-01 00:00:00.001000000`
from_nanoseconds (bigint)	Marca de tiempo	`select from_nanoseconds(300000001)` Ejemplo de resultado: `1970-01-01 00:00:00.300000001`
from_unixtime (doble)	Marca de tiempo	Devuelve una marca de tiempo que corresponde al unixtime proporcionado. `SELECT from_unixtime(1)` Ejemplo de resultado: `1970-01-01 00:00:01.000000000`

Función	Tipo de datos de salida	Descripción
hora local	hora	Devuelve la hora actual enUTC. No se utilizan paréntesis. `SELECT localtime` Ejemplo de resultado: `17:58:22.654000000` **Note** También es una palabra clave reservada. Para obtener una lista de palabras clave reservadas, consulte Palabras clave reservadas.

Función	Tipo de datos de salida	Descripción
marca de tiempo local	Marca de tiempo	Devuelve la marca de tiempo actual. UTC No se utilizan paréntesis. `SELECT localtimestamp` Ejemplo de resultado: `2022-07-07 17:59:04.368000000` **Note** También es una palabra clave reservada. Para obtener una lista de palabras clave reservadas, consulte Palabras clave reservadas.

Función	Tipo de datos de salida	Descripción

to_milisegundos (intervalo de un día a un segundo), to_milisegundos (marca de tiempo)	bigint	`SELECT to_millis econds(INTERVAL '2' DAY + INTERVAL '3' HOUR)`
		Ejemplo de resultado: `183600000`
		`SELECT to_millis econds(TIMESTAMP '2022-06-17 17:44:43. 771000000')`
		Ejemplo de resultado: `1655487883771`

to_nanoseconds (interval o de un día a un segundo), to_nanoseconds (marca de tiempo)	bigint	`SELECT to_nanose conds(INTERVAL '2' DAY + INTERVAL '3' HOUR)`
		Ejemplo de resultado: `183600000000000`
		`SELECT to_nanose conds(TIMESTAMP '2022-06-17 17:44:43. 771000678')`
		Ejemplo de resultado: `1655487883771000678`

Función	Tipo de datos de salida	Descripción
to_unixtime (marca de tiempo)	double	Devuelve unixtime para la marca de tiempo proporcio nada. `SELECT to_unixti me('2022-06-17 17:44:43.771000000')` Ejemplo de resultado: `1.6554878837710001E9`
date_trunc (unidad, marca de tiempo)	Marca de tiempo	Devuelve la marca de tiempo truncada a la unidad, donde la unidad es una de [segundo, minuto, hora, día, semana, mes, trimestre o año]. `SELECT date_trun c('minute', TIMESTAMP '2022-06-17 17:44:43. 771000000')` Ejemplo de resultado: `2022-06-17 17:44:00. 000000000`

Intervalo y duración

Timestream for LiveAnalytics admite las siguientes funciones de intervalo y duración para la fecha y la hora.

Función	Tipo de datos de salida	Descripción
date_add (unidad, bigint, fecha), date_add (unidad,	Marca de tiempo	Añade un bigint de unidades, donde la unidad es una de

Función	Tipo de datos de salida	Descripción
bigint, hora), date_add (varchar (x), bigint, timestamp)		las siguientes: [segundo, minuto, hora, día, semana, mes, trimestre o año]. `SELECT date_add('hour', 9, TIMESTAMP '2022-06-17 00:00:00')` Ejemplo de resultado: `2022-06-17 09:00:00.000000000`
date_diff (unidad, fecha, fecha), date_diff (unidad, hora, hora), date_diff (unidad, marca de tiempo, marca de tiempo)	bigint	Devuelve una diferencia, donde la unidad es una de las siguientes: [segundo, minuto, hora, día, semana, mes, trimestre o año]. `SELECT date_diff('day', DATE '2020-03-01', DATE '2020-03-02')` Ejemplo de resultado: 1

Función	Tipo de datos de salida	Descripción

parse_duration (cadena)	intervalo	Analiza la cadena de entrada para devolver un equivalente. `interval`
		
		`SELECT parse_duration('42.8ms')`
		
		Ejemplo de resultado: `0 00:00:00.042800000`
		
		`SELECT typeof(parse_duration('42.8ms'))`
		
		Ejemplo de resultado: `interval day to second`

Función	Tipo de datos de salida	Descripción
bin (marca de tiempo, intervalo)	Marca de tiempo	Redondea el valor entero del `timestamp` parámetro al múltiplo más cercano del valor entero del `interval` parámetro. El significado de este valor devuelto puede no ser obvio. Se calcula mediante aritmétic a de enteros dividiendo primero el entero de la marca de tiempo por el entero del intervalo y, a continuación, multiplicando el resultado por el entero del intervalo. Teniendo en cuenta que una marca de tiempo especifica un UTC punto en el tiempo como el número de fracciones de segundo que han transcurr ido desde la POSIX época (1 de enero de 1970), el valor devuelto rara vez se alineará con las unidades del calendari o. Por ejemplo, si especifica un intervalo de 30 días, todos los días transcurridos desde la época se dividen en increment os de 30 días y se devuelve el inicio del incremento de 30 días más reciente, que no tiene relación con los meses naturales.

Función	Tipo de datos de salida	Descripción

		Estos son algunos ejemplos:
		
		`bin(TIMESTAMP '2022-06-17 10:15:20', 5m) ==> 2022-06-17 10:15:00.000000000`
		`bin(TIMESTAMP '2022-06-17 10:15:20', 1d) ==> 2022-06-17 00:00:00.000000000`
		`bin(TIMESTAMP '2022-06-17 10:15:20', 10day) ==> 2022-06-17 00:00:00.000000000`
		`bin(TIMESTAMP '2022-06-17 10:15:20', 30day) ==> 2022-05-28 00:00:00.000000000`

hace (intervalo)	Marca de tiempo	Devuelve el valor correspondiente a `interval` current_timestamp. `SELECT ago(1d)` Ejemplo de resultado: `2022-07-06 21:08:53.245000000`

Función	Tipo de datos de salida	Descripción
literales de intervalo como 1h, 1d y 30m	intervalo	Los literales de intervalo son útiles para parse_duration (string). Por ejemplo, `1d` es igual que `parse_duration('1d')` . Esto permite el uso de los literales siempre que se utilice un intervalo. Por ejemplo, `ago(1d)` y `bin(<timestamp> , 1m)`.

Algunos literales de intervalo actúan como forma abreviada de parse_duration. Por ejemplo,, `parse_duration('1day')1day`, y `1d` cada uno devuelve `parse_duration('1d')` donde está el tipo. `1 00:00:00.000000000 interval day to second` Se permite el espacio en el formato proporcionado a`parse_duration`. Por ejemplo, `parse_duration('1day')` también devuelve`00:00:00.000000000`. Pero no `1 day` es un intervalo literal.

Las unidades correspondientes `interval day to second` son ns, nanosegundo, us, microsegundo, ms, milisegundo, s, segundo, m, minuto, h, hora, d y día.

También lo hay. `interval year to month` Las unidades relacionadas con el intervalo de un año a otro son y, año y mes. Por ejemplo, las `SELECT 1year` devoluciones`1-0`. `SELECT 12month`también devuelve`1-0`. `SELECT 8month`devoluciones`0-8`.

Aunque la unidad de también `quarter` está disponible para algunas funciones, como `date_trunc` y`date_add`, no `quarter` está disponible como parte de un intervalo literal.

Formatear y analizar

Timestream for LiveAnalytics admite las siguientes funciones de formato y análisis de fecha y hora.

Función	Tipo de datos de salida	Descripción
date_format (timestamp, varchar (x))	varchar	Para obtener más información sobre los especificadores de formato utilizados por esta

Función	Tipo de datos de salida	Descripción
		función, consulte # https://trino.io/docs/current/functions/datetime.html mysql-date-functions `SELECT date_format(TIMESTAMP '2019-10-20 10:20:20', '%Y-%m-%d %H:%i:%s')` Ejemplo de resultado: `2019-10-20 10:20:20`
date_parse (varchar (x), varchar (y))	Marca de tiempo	Para obtener más información sobre los especificadores de formato utilizados por esta función, consulte # https://trino.io/docs/current/functions/datetime.html mysql-date-functions `SELECT date_parse('2019-10-20 10:20:20', '%Y-%m-%d %H:%i:%s')` Ejemplo de resultado: `2019-10-20 10:20:20.000000000`

Función	Tipo de datos de salida	Descripción
format_datetime (timestamp, varchar (x))	varchar	Para obtener más información sobre la cadena de formato utilizada por esta función, consulte http://joda-time.sourceforge. net/apidocs/org/joda/time/format/DateTimeFormat.html `SELECT format_datetime(parse_datetime('1968-01-13 12', 'yyyy-MM-dd HH'), 'yyyy-MM-dd HH')` Ejemplo de resultado: `1968-01-13 12`
parse_datetime (varchar (x), varchar (y))	Marca de tiempo	Para obtener más información sobre la cadena de formato utilizada por esta función, consulte http://joda-time.sourceforge. net/apidocs/org/joda/time/format/DateTimeFormat.html `SELECT parse_datetime('2019-12-29 10:10 PST', 'uuuu-LL-dd HH:mm z')` Ejemplo de resultado: `2019-12-29 18:10:00.000000000`

Extracción

Timestream for LiveAnalytics admite las siguientes funciones de extracción de fecha y hora. La función de extracción es la base del resto de funciones prácticas.

Función	Tipo de datos de salida	Descripción
extract	bigint	Extrae un campo de una marca de tiempo, donde el campo es uno de [YEAR,, QUARTERMONTH, _OF_ WEEKDAY, DAY _OF_MONTH, DAY _OF_WEEK, DAY _OF_DOW,, YEARDOY, YEAR o]. WEEK YOW HOUR MINUTE SECOND `SELECT extract(YEAR FROM '2019-10-12 23:10:34.000000000')` Resultado de ejemplo: 2019
día (marca de tiempo), día (fecha), día (intervalo de un día a un segundo)	bigint	`SELECT day('2019-10-12 23:10:34.000000000')` Ejemplo de resultado: 12
day_of_month (marca de tiempo), day_of_month (fecha), day_of_month (intervalo de un día a otro)	bigint	`SELECT day_of_month('2019-10-12 23:10:34.000000000')` Ejemplo de resultado: 12

Función	Tipo de datos de salida	Descripción
day_of_week (marca de tiempo), day_of_week (fecha)	bigint	`SELECT day_of_week('2019-10-12 23:10:34.000000000')` Ejemplo de resultado: 6
day_of_year (marca de tiempo), day_of_year (fecha)	bigint	`SELECT day_of_year('2019-10-12 23:10:34.000000000')` Ejemplo de resultado: 285
down (marca de tiempo), dow (fecha)	bigint	Alias de day_of_week
doy (marca de tiempo), doy (fecha)	bigint	Alias para day_of_year
hora (marca de tiempo), hora (hora), hora (intervalo de un día a un segundo)	bigint	`SELECT hour('2019-10-12 23:10:34.000000000')` Ejemplo de resultado: 23
milisegundo (marca de tiempo), milisegundo (hora), milisegundo (intervalo de un día a otro)	bigint	`SELECT millisecond('2019-10-12 23:10:34.000000000')` Ejemplo de resultado: 0
minuto (marca de tiempo), minuto (hora), minuto (intervalo de un día a un segundo)	bigint	`SELECT minute('2019-10-12 23:10:34.000000000')` Ejemplo de resultado: 10

Función	Tipo de datos de salida	Descripción
mes (marca de tiempo), mes (fecha), mes (intervalo de un año a otro)	bigint	`SELECT month('2019-10-12 23:10:34.000000000')` Ejemplo de resultado: `10`
nanosegundo (marca de tiempo), nanosegundo (tiempo), nanosegundo (intervalo de un día a otro)	bigint	`SELECT nanosecond(current_timestamp)` Ejemplo de resultado: `162000000`
trimestre (marca de tiempo), trimestre (fecha)	bigint	`SELECT quarter('2019-10-12 23:10:34.000000000')` Ejemplo de resultado: `4`
segundo (marca de tiempo), segundo (hora), segundo (intervalo de un día a otro)	bigint	`SELECT second('2019-10-12 23:10:34.000000000')` Ejemplo de resultado: `34`
semana (marca de tiempo), semana (fecha)	bigint	`SELECT week('2019-10-12 23:10:34.000000000')` Ejemplo de resultado: `41`
week_of_year (marca de tiempo), week_of_year (fecha)	bigint	Alias para la semana

Función	Tipo de datos de salida	Descripción
año (marca de tiempo), año (fecha), año (intervalo de un año a otro)	bigint	`SELECT year('2019-10-12 23:10:34.000000000')` Ejemplo de resultado: 2019
year_of_week (marca de tiempo), year_of_week (fecha)	bigint	`SELECT year_of_week('2019-10-12 23:10:34.000000000')` Ejemplo de resultado: 2019
cómo (marca de tiempo), cómo (fecha)	bigint	Alias de year_of_week

Funciones de agregación

Timestream for LiveAnalytics admite las siguientes funciones de agregado.

Función	Tipo de datos de salida	Descripción
arbitrario (x)	[igual que la entrada]	Devuelve un valor arbitrario no nulo de x, si existe. `SELECT arbitrary(t.c) FROM (VALUES 1, 2, 3, 4) AS t(c)` Ejemplo de resultado: 1
array_agg (x)	array< [igual que la entrada]	Devuelve una matriz creada a partir de los x elementos de entrada.

Función	Tipo de datos de salida	Descripción
		`SELECT array_agg(t.c) FROM (VALUES 1, 2, 3, 4) AS t(c)` Ejemplo de resultado: `[ 1,2,3,4 ]`
avg (x)	double	Devuelve el promedio (media aritmética) de todos los valores de entrada. `SELECT avg(t.c) FROM (VALUES 1, 2, 3, 4) AS t(c)` Ejemplo de resultado: `2.5`
bool_and (booleano) cada (booleano)	boolean	Devuelve TRUE si todos los valores de entrada son, en caso contrario. TRUE FALSE `SELECT bool_and(t.c) FROM (VALUES true, true, false, true) AS t(c)` Ejemplo de resultado: `false`

Función	Tipo de datos de salida	Descripción
bool_or (booleano)	boolean	Devuelve TRUE si algún valor de entrada lo es, en caso contrario. TRUE FALSE `SELECT bool_or(t.c) FROM (VALUES true, true, false, true) AS t(c)` Ejemplo de resultado: `true`
contar (*) contar (x)	bigint	count (*) devuelve el número de filas de entrada. count (x) devuelve el número de valores de entrada no nulos. `SELECT count(t.c) FROM (VALUES true, true, false, true) AS t(c)` Ejemplo de resultado: `4`
count_if (x)	bigint	Devuelve el número de valores de TRUE entrada. `SELECT count_if(t.c) FROM (VALUES true, true, false, true) AS t(c)` Ejemplo de resultado: `3`

Función	Tipo de datos de salida	Descripción
geometric_mean (x)	double	Devuelve la media geométrica de todos los valores de entrada. `SELECT geometric_mean(t.c) FROM (VALUES 1, 2, 3, 4) AS t(c)` Ejemplo de resultado: `2.213363839400643`
max_by (x, y)	[igual que x]	Devuelve el valor de x asociado al valor máximo de y sobre todos los valores de entrada. `SELECT max_by(t.c1, t.c2) FROM (VALUES (('a', 1)), (('b', 2)), (('c', 3)), (('d', 4))) AS t(c1, c2)` Ejemplo de resultado: d

Función	Tipo de datos de salida	Descripción
max_by (x, y, n)	matriz< [same as x] >	Devuelve n valores de x asociados al n mayor de todos los valores de entrada de y en orden descendente de y. `SELECT max_by(t.c1, t.c2, 2) FROM (VALUES (('a', 1)), (('b', 2)), (('c', 3)), (('d', 4))) AS t(c1, c2)` Ejemplo de resultado: `[ d,c ]`
min_by (x, y)	[igual que x]	Devuelve el valor de x asociado al valor mínimo de y sobre todos los valores de entrada. `SELECT min_by(t.c1, t.c2) FROM (VALUES (('a', 1)), (('b', 2)), (('c', 3)), (('d', 4))) AS t(c1, c2)` Ejemplo de resultado: a

Función	Tipo de datos de salida	Descripción
min_by (x, y, n)	matriz< [same as x] >	Devuelve n valores de x asociados al n más pequeño de todos los valores de entrada de y en orden ascendente de y. `SELECT min_by(t.c1, t.c2, 2) FROM (VALUES (('a', 1)), (('b', 2)), (('c', 3)), (('d', 4))) AS t(c1, c2)` Ejemplo de resultado: `[ a,b ]`
máximo (x)	[igual que la entrada]	Devuelve el valor máximo de todos los valores de entrada. `SELECT max(t.c) FROM (VALUES 1, 2, 3, 4) AS t(c)` Ejemplo de resultado: 4
máximo (x, n)	matriz< [same as x] >	Devuelve n los valores más altos de todos los valores de entrada de x. `SELECT max(t.c, 2) FROM (VALUES 1, 2, 3, 4) AS t(c)` Ejemplo de resultado: `[ 4,3 ]`

Función	Tipo de datos de salida	Descripción
min (x)	[igual que la entrada]	Devuelve el valor mínimo de todos los valores de entrada. `SELECT min(t.c) FROM (VALUES 1, 2, 3, 4) AS t(c)` Ejemplo de resultado: 1
min (x, n)	matriz< [same as x] >	Devuelve los n valores más pequeños de todos los valores de entrada de x. `SELECT min(t.c, 2) FROM (VALUES 1, 2, 3, 4) AS t(c)` Ejemplo de resultado: `[ 1,2 ]`
suma (x)	[igual que la entrada]	Devuelve la suma de todos los valores de entrada. `SELECT sum(t.c) FROM (VALUES 1, 2, 3, 4) AS t(c)` Ejemplo de resultado: `10`

Función	Tipo de datos de salida	Descripción
bitwise_and_agg (x)	bigint	Devuelve el valor bit a bit de todos los valores de entrada en una representación AND del complemento de 2 segundos. `SELECT bitwise_and_agg(t.c) FROM (VALUES 1, -3) AS t(c)` Ejemplo de resultado: 1
bitwise_or_agg (x)	bigint	Devuelve el OR bit a bit de todos los valores de entrada en una representación complementaria de 2 segundos. `SELECT bitwise_or_agg(t.c) FROM (VALUES 1, -3) AS t(c)` Ejemplo de resultado: -3

Función	Tipo de datos de salida	Descripción
approx_distinct (x)	bigint	Devuelve el número aproximado de valores de entrada distintos. Esta función proporciona una aproximación del recuento (DISTINCTx). Si todos los valores de entrada son nulos, se devuelve cero. Esta función debería producir un error estándar del 2,3%, que es la desviación estándar de la distribución del error (aproximadamente normal) en todos los conjuntos posibles. No garantiza un límite superior del error para ningún conjunto de entradas específico. `SELECT approx_distinct(t.c) FROM (VALUES 1, 2, 3, 4, 8) AS t(c)` Ejemplo de resultado: 5

Función	Tipo de datos de salida	Descripción
approx_distinct (x, e)	bigint	Devuelve el número aproximado de valores de entrada distintos. Esta función proporciona una aproximación del recuento (DISTINCTx). Si todos los valores de entrada son nulos, se devuelve cero. Esta función debería producir un error estándar no superior a e, que es la desviación estándar de la distribución del error (aproximadamente normal) en todos los conjuntos posibles. No garantiza un límite superior del error para ningún conjunto de entradas específico. La implementación actual de esta función requiere que e esté en el rango de [0,0040625, 0,26000]. `SELECT approx_distinct(t.c, 0.2) FROM (VALUES 1, 2, 3, 4, 8) AS t(c)` Ejemplo de resultado: 5

Función	Tipo de datos de salida	Descripción
percentil aproximado (x, porcentaje)	[igual que x]	Devuelve el percentil aproximado de todos los valores de entrada de x en el porcentaje indicado. El valor del porcentaje debe estar entre cero y uno y debe ser constante en todas las filas de entrada. `SELECT approx_percentile(t.c, 0.4) FROM (VALUES 1, 2, 3, 4) AS t(c)` Ejemplo de resultado: 2
percentil aproximado (x, porcentajes)	matriz< [same as x] >	Devuelve el percentil aproximado de todos los valores de entrada de x en cada uno de los porcentajes especificados. Cada elemento de la matriz de porcentajes debe estar entre cero y uno, y la matriz debe ser constante en todas las filas de entrada. `SELECT approx_percentile(t.c, ARRAY[0.1, 0.8, 0.8]) FROM (VALUES 1, 2, 3, 4) AS t(c)` Ejemplo de resultado: `[ 1,4,4 ]`

Función	Tipo de datos de salida	Descripción
percentil aproximado (x, w, porcentaje)	[igual que x]	Devuelve el percentil ponderado aproximado de todos los valores de entrada de x utilizando el peso w por elemento en el porcentaje p. El peso debe ser un valor entero de al menos uno. En efecto, es un recuento de réplicas para el valor x del conjunto de percentil es. El valor de p debe estar entre cero y uno y debe ser constante en todas las filas de entrada. `SELECT approx_pe rcentile(t.c, 1, 0.1) FROM (VALUES 1, 2, 3, 4) AS t(c)` Ejemplo de resultado: 1

Función	Tipo de datos de salida	Descripción
percentil aproximado (x, w, porcentajes)	matriz< [same as x] >	Devuelve el percentil ponderado aproximado de todos los valores de entrada de x utilizando el peso w por elemento en cada uno de los porcentajes especificados en la matriz. El peso debe ser un valor entero de al menos uno. En efecto, es un recuento de réplicas para el valor x del conjunto de percentiles. Cada elemento de la matriz debe estar entre cero y uno, y la matriz debe ser constante en todas las filas de entrada. `SELECT approx_pe rcentile(t.c, 1, ARRAY[0.1, 0.8, 0.8]) FROM (VALUES 1, 2, 3, 4) AS t(c)` Ejemplo de resultado: `[ 1,4,4 ]`

Función	Tipo de datos de salida	Descripción
percentil aproximado (x, w, porcentaje, precisión)	[igual que x]	Devuelve el percentil ponderado aproximado de todos los valores de entrada de x utilizando el peso w por elemento en el porcentaje p, con un error de precisión de clasificación máximo. El peso debe ser un valor entero de al menos uno. En efecto, es un recuento de réplicas para el valor x del conjunto de percentiles. El valor de p debe estar entre cero y uno y debe ser constante en todas las filas de entrada. La precisión debe ser un valor superior a cero e inferior a uno, y debe ser constante en todas las filas de entrada. `SELECT approx_percentile(t.c, 1, 0.1, 0.5) FROM (VALUES 1, 2, 3, 4) AS t(c)` Ejemplo de resultado: 1

Función	Tipo de datos de salida	Descripción
corr (y, x)	double	Devuelve el coeficiente de correlación de los valores de entrada. `SELECT corr(t.c1, t.c2) FROM (VALUES ((1, 1)), ((2, 2)), ((3, 3)), ((4, 4))) AS t(c1, c2)` Ejemplo de resultado: `1.0`
covar_pop (y, x)	double	Devuelve la covarianza poblacional de los valores de entrada. `SELECT covar_pop(t.c1, t.c2) FROM (VALUES ((1, 1)), ((2, 2)), ((3, 3)), ((4, 4))) AS t(c1, c2)` Ejemplo de resultado: `1.25`
covar_samp (y, x)	double	Devuelve la covarianza muestral de los valores de entrada. `SELECT covar_samp(t.c1, t.c2) FROM (VALUES ((1, 1)), ((2, 2)), ((3, 3)), ((4, 4))) AS t(c1, c2)` Ejemplo de resultado: `1.6666666666666667`

Función	Tipo de datos de salida	Descripción
regr_intercept (y, x)	double	Devuelve la intersección de regresión lineal de los valores de entrada. y es el valor dependiente. x es el valor independiente. `SELECT regr_intercept(t.c1, t.c2) FROM (VALUES ((1, 1)), ((2, 2)), ((3, 3)), ((4, 4))) AS t(c1, c2)` Ejemplo de resultado: `0.0`
regr_slope (y, x)	double	Devuelve la pendiente de regresión lineal de los valores de entrada. y es el valor dependiente. x es el valor independiente. `SELECT regr_slope(t.c1, t.c2) FROM (VALUES ((1, 1)), ((2, 2)), ((3, 3)), ((4, 4))) AS t(c1, c2)` Ejemplo de resultado: `1.0`

Función	Tipo de datos de salida	Descripción
asimetría (x)	double	Devuelve la asimetría de todos los valores de entrada. `SELECT skewness(t.c1) FROM (VALUES 1, 2, 3, 4, 8) AS t(c1)` Ejemplo de resultado: `0.8978957037987335`
stddev_pop (x)	double	Devuelve la desviación estándar de la población de todos los valores de entrada. `SELECT stddev_pop(t.c1) FROM (VALUES 1, 2, 3, 4, 8) AS t(c1)` Ejemplo de resultado: `2.4166091947189146`
stddev_samp (x) stddev (x)	double	Devuelve la desviación estándar de la muestra de todos los valores de entrada. `SELECT stddev_samp(t.c1) FROM (VALUES 1, 2, 3, 4, 8) AS t(c1)` Ejemplo de resultado: `2.701851217221259`

Función	Tipo de datos de salida	Descripción
var_pop (x)	double	Devuelve la varianza poblacional de todos los valores de entrada. `SELECT var_pop(t.c1) FROM (VALUES 1, 2, 3, 4, 8) AS t(c1)` Ejemplo de resultado: `5.840000000000001`
var_samp (x) varianza (x)	double	Devuelve la varianza muestral de todos los valores de entrada. `SELECT var_samp(t.c1) FROM (VALUES 1, 2, 3, 4, 8) AS t(c1)` Ejemplo de resultado: `7.300000000000001`

Funciones de ventana

Las funciones de ventana realizan cálculos en todas las filas del resultado de la consulta. Se ejecutan después de la HAVING cláusula pero antes de la cláusula ORDER BY. La invocación de una función de ventana requiere una sintaxis especial, utilizando la OVER cláusula para especificar la ventana. Una ventana tiene tres componentes:

- La especificación de la partición, que separa las filas de entrada en diferentes particiones. Esto es análogo a la forma en que la cláusula GROUP BY separa las filas en diferentes grupos para las funciones de agregación.
- La especificación de ordenación, que determina el orden en el que la función de ventana procesará las filas de entrada.

- El marco de la ventana, que especifica una ventana deslizante de filas que la función procesará para una fila determinada. Si no se especifica el marco, el valor predeterminado es RANGE UNBOUNDEDPRECEDING, que es igual RANGE BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW que. Este marco contiene todas las filas desde el inicio de la partición hasta el último par de la fila actual.

Todas las funciones agregadas se pueden utilizar como funciones de ventana añadiendo la OVER cláusula. La función de agregado se calcula para cada fila sobre las filas del marco de ventana de la fila actual. Además de las funciones de agregado, Timestream for LiveAnalytics admite las siguientes funciones de clasificación y valor.

Función	Tipo de datos de salida	Descripción
cume_dist ()	bigint	Devuelve la distribución acumulada de un valor en un grupo de valores. El resultado es el número de filas que preceden o coinciden con la fila de la partición de la ventana dividido por el número total de filas de la partición de la ventana. Por lo tanto, cualquier valor de empate en el orden se evaluará con el mismo valor de distribución.
dense_rank ()	bigint	Devuelve el rango de un valor en un grupo de valores. Es similar a rank (), excepto que los valores de empate no producen huecos en la secuencia.
Entil (n)	bigint	Divide las filas de cada partición de ventana en n cubos que van desde 1 hasta n como máximo. Los valores

Función	Tipo de datos de salida	Descripción
		de los cubos diferirán como máximo en 1. Si el número de filas de la partición no se divide uniformemente entre el número de cubos, los valores restantes se distribuyen uno por grupo, empezando por el primer grupo.
percent_rank ()	double	Devuelve la clasificación porcentual de un valor en un grupo de valores. El resultado es (r - 1)/(n - 1) donde r es el rango () de la fila y n es el número total de filas de la partición de la ventana.
rango ()	bigint	Devuelve el rango de un valor en un grupo de valores. El rango es uno más el número de filas que preceden a la fila y que no coinciden con la fila. Por lo tanto, los valores de empate en el orden producirá n huecos en la secuencia. La clasificación se realiza para cada partición de ventana.
row_number ()	bigint	Devuelve un número secuencial único para cada fila, empezando por uno, según el orden de las filas dentro de la partición de la ventana.

Función	Tipo de datos de salida	Descripción
primer_valor (x)	[igual que la entrada]	Devuelve el primer valor de la ventana. Esta función se limita al marco de la ventana. La función toma una expresión o un objetivo como parámetro.
last_value (x)	[igual que la entrada]	Devuelve el último valor de la ventana. Esta función se limita al marco de la ventana. La función toma una expresión o un objetivo como parámetro.
nth_value (x, offset)	[igual que la entrada]	Devuelve el valor con el desfase especificado desde el inicio de la ventana. Los desfases comienzan en 1. El desplazamiento puede ser cualquier expresión escalar. Si el desplazamiento es nulo o superior al número de valores de la ventana, se devuelve un valor nulo. Es un error que el desfase sea cero o negativo. La función toma una expresión o un objetivo como primer parámetro.

Función	Tipo de datos de salida	Descripción
lead (x [, offset [, default_value]])	[igual que la entrada]	Devuelve el valor en las filas desplazadas después de la fila actual de la ventana. Los desfases comienzan en 0, que es la fila actual. El desplazamiento puede ser cualquier expresión escalar. El desfase predeterminado es 1. Si el desplazamiento es nulo o mayor que la ventana, se devuelve el valor predeterminado o, si no se especifica, se devuelve nulo. La función toma una expresión o un objetivo como primer parámetro.
lag (x [, offset [, default_value]])	[igual que la entrada]	Devuelve el valor de las filas desplazadas antes de la fila actual de la ventana. Las compensaciones comienzan en 0, que es la fila actual. El desplazamiento puede ser cualquier expresión escalar. El desfase predeterminado es 1. Si el desplazamiento es nulo o mayor que la ventana, se devuelve el valor predeterminado o, si no se especifica, se devuelve nulo. La función toma una expresión o un objetivo como primer parámetro.

Consultas de ejemplo

En esta sección se incluyen ejemplos de casos de uso del lenguaje de consulta LiveAnalytics de Timestream for.

Temas

Consultas sencillas

A continuación, se obtienen los 10 puntos de datos agregados más recientemente a una tabla.

```
SELECT * FROM <database_name>.<table_name>
ORDER BY time DESC
LIMIT 10
```

A continuación, se obtienen los 5 puntos de datos más antiguos de una medida específica.

```
SELECT * FROM <database_name>.<table_name>
WHERE measure_name = '<measure_name>'
ORDER BY time ASC
LIMIT 5
```

Lo siguiente funciona con marcas de tiempo de granularidad de nanosegundos.

```
SELECT now() AS time_now
, now() - (INTERVAL '12' HOUR) AS twelve_hour_earlier -- Compatibility with ANSI SQL
, now() - 12h AS also_twelve_hour_earlier -- Convenient time interval literals
, ago(12h) AS twelve_hours_ago -- More convenience with time functionality
, bin(now(), 10m) AS time_binned -- Convenient time binning support
, ago(50ns) AS fifty_ns_ago -- Nanosecond support
, now() + (1h + 50ns) AS hour_fifty_ns_future
```

Los valores de medida de los registros de medidas múltiples se identifican por el nombre de la columna. Los valores de medida de los registros de una sola medida se identifican por`measure_value::<data_type>`, si `<data_type>` es uno de ellos`double`, `bigint`, `boolean`,

o tal y `varchar` como se describe en. Tipos de datos compatibles Para obtener más información sobre cómo se modelan los valores de medida, consulte Tabla única frente a tablas múltiples.

A continuación, se recuperan los valores de una medida llamada a `speed` partir de registros de varias medidas con un valor de. `measure_name` `IoTMulti-stats`

```
SELECT speed FROM <database_name>.<table_name> where measure_name = 'IoTMulti-stats'
```

A continuación, se recuperan `double` valores de registros de una sola medida con un de. `measure_name` `load`

```
SELECT measure_value::double FROM <database_name>.<table_name> WHERE measure_name =
 'load'
```

Consultas con funciones de series temporales

Temas

- Ejemplo de conjunto de datos y consultas

Ejemplo de conjunto de datos y consultas

Puede usar Timestream LiveAnalytics para comprender y mejorar el rendimiento y la disponibilidad de sus servicios y aplicaciones. A continuación se muestra una tabla de ejemplo y ejemplos de consultas que se ejecutan en esa tabla.

La tabla `ec2_metrics` almacena datos de telemetría, como la CPU utilización y otras métricas de EC2 las instancias. Puedes ver la siguiente tabla.

Tiempo	región	az	Hostname	measure_name	measure_value::double	measure_value::bigint
2019-12-04 19:00:00.000000000	us-east-1	us-east-1a	parte delantera 01	utilización de la CPU	35.1	null

Tiempo	región	az	Hostname	measure_name	measure_value::double	measure_value::bigint
2019-12-04 19:00:00.000000000	us-east-1	us-east-1a	parte delantera 01	memory_utilization	5.3	null
2019-12-04 19:00:00.000000000	us-east-1	us-east-1a	parte delantera 01	network_bytes_in	null	1500
2019-12-04 19:00:00.000000000	us-east-1	us-east-1a	parte delantera 01	network_bytes_out	null	6700
2019-12-04 19:00:00.000000000	us-east-1	us-east-1b	parte delantera 02	utilización de la CPU	38,5	null
2019-12-04 19:00:00.000000000	us-east-1	us-east-1b	parte delantera 02	memory_utilization	58.4	null
2019-12-04 19:00:00.000000000	us-east-1	us-east-1b	parte delantera 02	network_bytes_in	null	23.000
2019-12-04 19:00:00.000000000	us-east-1	us-east-1b	parte delantera 02	network_bytes_out	null	12 000

Tiempo	región	az	Hostname	measure_name	measure_value::double	measure_value::bigint
2019-12-04 19:00:00.000000000	us-east-1	us-east-1c	parte delantera 03	utilización de la CPU	45.0	null
04/12/2019 19:00:00.000000000	us-east-1	us-east-1c	parte delantera 03	memory_utilization	65,8	null
2019-12-04 19:00:00.000000000	us-east-1	us-east-1c	parte delantera 03	network_bytes_in	null	15.000
2019-12-04 19:00:00.000000000	us-east-1	us-east-1c	parte delantera 03	network_bytes_out	null	836.000
2019-12-04 19:00:05.000000000	us-east-1	us-east-1a	parte delantera 01	utilización de la CPU	5.2	null
2019-12-04 19:00:05.000000000	us-east-1	us-east-1a	parte delantera 01	memory_utilization	75.0	null
2019-12-04 19:00:05.000000000	us-east-1	us-east-1a	parte delantera 01	network_bytes_in	null	1.245

Tiempo	región	az	Hostname	measure_name	measure_value::double	measure_value::bigint
2019-12-04 19:00:05.000000000	us-east-1	us-east-1a	parte delantera 01	network_bytes_out	null	68.432
2019-12-04 19:00:08.000000000	us-east-1	us-east-1b	parte delantera 02	utilización de la CPU	65,6	null
2019-12-04 19:00:08.000000000	us-east-1	us-east-1b	parte delantera 02	memory_utilization	85.3	null
2019-12-04 19:00:08.000000000	us-east-1	us-east-1b	parte delantera 02	network_bytes_in	null	1.245
2019-12-04 19:00:08.000000000	us-east-1	us-east-1b	parte delantera 02	network_bytes_out	null	68.432
2019-12-04 19:00:20.000000000	us-east-1	us-east-1c	front-end 03	utilización de la CPU	12.1	null
2019-12-04 19:00:20.000000000	us-east-1	us-east-1c	front-end 03	memory_utilization	32,0	null

Tiempo	región	az	Hostname	measure_name	measure_value::double	measure_value::bigint
2019-12-04 19:00:20.000 000000	us-east-1	us-east-1c	front-end 03	network_bytes_in	null	1.400
2019-12-04 19:00:20.000 000000	us-east-1	us-east-1c	front-end 03	network_bytes_out	null	345
2019-12-04 19:00:10.000 000000	us-east-1	us-east-1a	parte delantera 01	utilización de la CPU	15.3	null
2019-12-04 19:00:10.000 000000	us-east-1	us-east-1a	parte delantera 01	memory_utilization	35,4	null
2019-12-04 19:00:10.000 000000	us-east-1	us-east-1a	parte delantera 01	network_bytes_in	null	23
2019-12-04 19:00:10.000 000000	us-east-1	us-east-1a	parte delantera 01	network_bytes_out	null	0
2019-12-04 19:00:16 .000 000000	us-east-1	us-east-1b	parte delantera 02	utilización de la CPU	44.0	null

Tiempo	región	az	Hostname	measure_name	measure_value::double	measure_value::bigint
2019-12-04 19:00:16.000000000	us-east-1	us-east-1b	parte delantera 02	memory_utilization	64.2	null
2019-12-04 19:00:16.000000000	us-east-1	us-east-1b	parte delantera 02	network_bytes_in	null	1.450
2019-12-04 19:00:16.000000000	us-east-1	us-east-1b	parte delantera 02	network_bytes_out	null	200
2019-12-04 19:00:40.000000000	us-east-1	us-east-1c	front-end 03	utilización de la CPU	6.4	null
2019-12-04 19:00:40.000000000	us-east-1	us-east-1c	front-end 03	memory_utilization	86.3	null
2019-12-04 19:00:40.000000000	us-east-1	us-east-1c	front-end 03	network_bytes_in	null	300
2019-12-04 19:00:40.000000000	us-east-1	us-east-1c	front-end 03	network_bytes_out	null	423

Encuentre el CPU uso promedio de p90, p95 y p99 de un EC2 host específico en las últimas 2 horas:

```
SELECT region, az, hostname, BIN(time, 15s) AS binned_timestamp,
    ROUND(AVG(measure_value::double), 2) AS avg_cpu_utilization,
    ROUND(APPROX_PERCENTILE(measure_value::double, 0.9), 2) AS p90_cpu_utilization,
    ROUND(APPROX_PERCENTILE(measure_value::double, 0.95), 2) AS p95_cpu_utilization,
    ROUND(APPROX_PERCENTILE(measure_value::double, 0.99), 2) AS p99_cpu_utilization
FROM "sampleDB".DevOps
WHERE measure_name = 'cpu_utilization'
    AND hostname = 'host-Hovjv'
    AND time > ago(2h)
GROUP BY region, hostname, az, BIN(time, 15s)
ORDER BY binned_timestamp ASC
```

Identifique los EC2 hosts CPU cuya utilización sea superior en un 10% o más a la CPU utilización media de toda la flota durante las últimas 2 horas:

```
WITH avg_fleet_utilization AS (
    SELECT COUNT(DISTINCT hostname) AS total_host_count, AVG(measure_value::double) AS
 fleet_avg_cpu_utilization
    FROM "sampleDB".DevOps
    WHERE measure_name = 'cpu_utilization'
        AND time > ago(2h)
), avg_per_host_cpu AS (
    SELECT region, az, hostname, AVG(measure_value::double) AS avg_cpu_utilization
    FROM "sampleDB".DevOps
    WHERE measure_name = 'cpu_utilization'
        AND time > ago(2h)
    GROUP BY region, az, hostname
)
SELECT region, az, hostname, avg_cpu_utilization, fleet_avg_cpu_utilization
FROM avg_fleet_utilization, avg_per_host_cpu
WHERE avg_cpu_utilization > 1.1 * fleet_avg_cpu_utilization
ORDER BY avg_cpu_utilization DESC
```

Encuentre la CPU utilización media agrupada en intervalos de 30 segundos para un EC2 host específico durante las últimas 2 horas:

```
SELECT BIN(time, 30s) AS binned_timestamp, ROUND(AVG(measure_value::double), 2) AS
 avg_cpu_utilization
FROM "sampleDB".DevOps
WHERE measure_name = 'cpu_utilization'
```

```
    AND hostname = 'host-Hovjv'
    AND time > ago(2h)
GROUP BY hostname, BIN(time, 30s)
ORDER BY binned_timestamp ASC
```

Encuentre la CPU utilización media agrupada en intervalos de 30 segundos para un EC2 host específico durante las últimas 2 horas y rellene los valores faltantes mediante la interpolación lineal:

```
WITH binned_timeseries AS (
    SELECT hostname, BIN(time, 30s) AS binned_timestamp,
 ROUND(AVG(measure_value::double), 2) AS avg_cpu_utilization
    FROM "sampleDB".DevOps
    WHERE measure_name = 'cpu_utilization'
        AND hostname = 'host-Hovjv'
        AND time > ago(2h)
    GROUP BY hostname, BIN(time, 30s)
), interpolated_timeseries AS (
    SELECT hostname,
        INTERPOLATE_LINEAR(
            CREATE_TIME_SERIES(binned_timestamp, avg_cpu_utilization),
                SEQUENCE(min(binned_timestamp), max(binned_timestamp), 15s)) AS
 interpolated_avg_cpu_utilization
    FROM binned_timeseries
    GROUP BY hostname
)
SELECT time, ROUND(value, 2) AS interpolated_cpu
FROM interpolated_timeseries
CROSS JOIN UNNEST(interpolated_avg_cpu_utilization)
```

Calcule la CPU utilización media agrupada en intervalos de 30 segundos para un EC2 host específico durante las últimas 2 horas y rellene los valores faltantes mediante la interpolación en función de la última observación realizada:

```
WITH binned_timeseries AS (
    SELECT hostname, BIN(time, 30s) AS binned_timestamp,
 ROUND(AVG(measure_value::double), 2) AS avg_cpu_utilization
    FROM "sampleDB".DevOps
    WHERE measure_name = 'cpu_utilization'
        AND hostname = 'host-Hovjv'
        AND time > ago(2h)
    GROUP BY hostname, BIN(time, 30s)
), interpolated_timeseries AS (
```

```
    SELECT hostname,
        INTERPOLATE_LOCF(
            CREATE_TIME_SERIES(binned_timestamp, avg_cpu_utilization),
                SEQUENCE(min(binned_timestamp), max(binned_timestamp), 15s)) AS
 interpolated_avg_cpu_utilization
    FROM binned_timeseries
    GROUP BY hostname
)
SELECT time, ROUND(value, 2) AS interpolated_cpu
FROM interpolated_timeseries
CROSS JOIN UNNEST(interpolated_avg_cpu_utilization)
```

Consultas con funciones agregadas

A continuación se muestra un ejemplo de conjunto de datos de un escenario de IoT para ilustrar las consultas con funciones agregadas.

Temas

- Datos de ejemplo
- Consultas de ejemplo

Datos de ejemplo

Timestream le permite almacenar y analizar los datos de los sensores de IoT, como la ubicación, el consumo de combustible, la velocidad y la capacidad de carga de una o más flotas de camiones, para permitir una gestión eficaz de la flota. A continuación se muestra el esquema y algunos de los datos de una tabla iot_trucks que almacena la telemetría, como la ubicación, el consumo de combustible, la velocidad y la capacidad de carga de los camiones.

Tiempo	truck_id	Make	Modelo	Flota	capacida _combus ble	capacida _carga	measure ame	measure alue::dou ble	measure_v alue::var char
04/12/20 9 19:00:00 000 000000	1234567	GMC	Astro	Alpha (Alfa)	100	500	lectura de combusti le	65.2	null

Tiempo	truck_id	Make	Modelo	Flota	capacida_combustible	capacida_carga	measure_name	measure_value::double	measure_value::varchar
2019-12-4 19:00:00.000000000	1234567	GMC	Astro	Alpha (Alfa)	100	500	carga	400,0	null
2019-12-4 19:00:00.000000000	1234567	GMC	Astro	Alpha (Alfa)	100	500	speed	90.2	null
2019-12-4 19:00:00.000000000	1234567	GMC	Astro	Alpha (Alfa)	100	500	location	null	47,6062 N, 122,3321 W
2019-12-4 19:00:00.000000000	1234567	Kenworth	W900	Alpha (Alfa)	150	1 000	lectura de combustible	10.1	null
2019-12-4 19:00:00.000000000	1234567	Kenworth	W900	Alpha (Alfa)	150	1 000	carga	950,3	null

Tiempo	truck_id	Make	Modelo	Flota	capacida _combus ble	capacida _carga	measure ame	measure alue::dou ble	measure_v alue::var char
2019-12-4 19:00:00 000 000000	1234567	Kenworth	W900	Alpha (Alfa)	150	1 000	speed	50.8	null
2019-12-4 19:00:00 000 000000	1234567	Kenworth	W900	Alpha (Alfa)	150	1 000	location	null	40.7128 grados N, 74.0060 grados W

Consultas de ejemplo

Obtenga una lista de todos los atributos y valores de los sensores que se están monitoreando para cada camión de la flota.

```
SELECT
    truck_id,
    fleet,
    fuel_capacity,
    model,
    load_capacity,
    make,
    measure_name
FROM "sampleDB".IoT
GROUP BY truck_id, fleet, fuel_capacity, model, load_capacity, make, measure_name
```

Obtenga la lectura de combustible más reciente de cada camión de la flota en las últimas 24 horas.

```
WITH latest_recorded_time AS (
    SELECT
        truck_id,
        max(time) as latest_time
```

```
    FROM "sampleDB".IoT
    WHERE measure_name = 'fuel-reading'
    AND time >= ago(24h)
    GROUP BY truck_id
)
SELECT
    b.truck_id,
    b.fleet,
    b.make,
    b.model,
    b.time,
    b.measure_value::double as last_reported_fuel_reading
FROM
latest_recorded_time a INNER JOIN "sampleDB".IoT b
ON a.truck_id = b.truck_id AND b.time = a.latest_time
WHERE b.measure_name = 'fuel-reading'
AND b.time > ago(24h)
ORDER BY b.truck_id
```

Identifique los camiones que han estado funcionando con poco combustible (menos del 10%) en las últimas 48 horas:

```
WITH low_fuel_trucks AS (
    SELECT time, truck_id, fleet, make, model, (measure_value::double/
cast(fuel_capacity as double)*100) AS fuel_pct
    FROM "sampleDB".IoT
    WHERE time >= ago(48h)
    AND (measure_value::double/cast(fuel_capacity as double)*100) < 10
    AND measure_name = 'fuel-reading'
),
other_trucks AS (
SELECT time, truck_id, (measure_value::double/cast(fuel_capacity as double)*100) as
 remaining_fuel
    FROM "sampleDB".IoT
    WHERE time >= ago(48h)
    AND truck_id IN (SELECT truck_id FROM low_fuel_trucks)
    AND (measure_value::double/cast(fuel_capacity as double)*100) >= 10
    AND measure_name = 'fuel-reading'
),
trucks_that_refuelled AS (
    SELECT a.truck_id
    FROM low_fuel_trucks a JOIN other_trucks b
    ON a.truck_id = b.truck_id AND b.time >= a.time
```

```
)
SELECT DISTINCT truck_id, fleet, make, model, fuel_pct
FROM low_fuel_trucks
WHERE truck_id NOT IN (
    SELECT truck_id FROM trucks_that_refuelled
)
```

Calcula la carga media y la velocidad máxima de cada camión durante la última semana:

```
SELECT
    bin(time, 1d) as binned_time,
    fleet,
    truck_id,
    make,
    model,
    AVG(
        CASE WHEN measure_name = 'load' THEN measure_value::double ELSE NULL END
    ) AS avg_load_tons,
    MAX(
        CASE WHEN measure_name = 'speed' THEN measure_value::double ELSE NULL END
    ) AS max_speed_mph
FROM "sampleDB".IoT
WHERE time >= ago(7d)
AND measure_name IN ('load', 'speed')
GROUP BY fleet, truck_id, make, model, bin(time, 1d)
ORDER BY truck_id
```

Obtenga la eficiencia de carga de cada camión durante la semana pasada:

```
WITH average_load_per_truck AS (
    SELECT
        truck_id,
        avg(measure_value::double)  AS avg_load
    FROM "sampleDB".IoT
    WHERE measure_name = 'load'
    AND time >= ago(7d)
    GROUP BY truck_id, fleet, load_capacity, make, model
),
truck_load_efficiency AS (
    SELECT
        a.truck_id,
        fleet,
        load_capacity,
```

```
        make,
        model,
        avg_load,
        measure_value::double,
        time,
        (measure_value::double*100)/avg_load as load_efficiency -- ,
 approx_percentile(avg_load_pct, DOUBLE '0.9')
    FROM "sampleDB".IoT a JOIN average_load_per_truck b
    ON a.truck_id = b.truck_id
    WHERE a.measure_name = 'load'
)
SELECT
    truck_id,
    time,
    load_efficiency
FROM truck_load_efficiency
ORDER BY truck_id, time
```

APIreferencia

Esta sección contiene la documentación de API referencia de Amazon Timestream.

Timestream tiene dosAPIs: Query y Write.

- La función Write API permite realizar operaciones como la creación de tablas, el etiquetado de recursos y la escritura de registros en Timestream.
- La consulta API le permite realizar operaciones de consulta.

> **Note**
>
> Ambas APIs incluyen la DescribeEndpoints acción. Tanto para Query como para Write, la DescribeEndpoints acción es idéntica.

Puede obtener más información sobre cada una de ellas API a continuación, junto con los tipos de datos, los errores comunes y los parámetros.

> **Note**
>
> Para ver los códigos de error comunes a todos los AWS servicios, consulte la sección AWS Support.

Temas

Acciones

Amazon Timestream Write admite las siguientes acciones:

- UpdateTable
- WriteRecords

Amazon Timestream Query admite las siguientes acciones:

- CancelQuery
- CreateScheduledQuery
- DeleteScheduledQuery
- DescribeAccountSettings
- DescribeEndpoints
- DescribeScheduledQuery
- ExecuteScheduledQuery
- ListScheduledQueries
- ListTagsForResource
- PrepareQuery
- Query
- TagResource
- UntagResource
- UpdateAccountSettings
- UpdateScheduledQuery

Amazon Timestream Write

Amazon Timestream Write admite las siguientes acciones:

- CreateBatchLoadTask
- CreateDatabase
- CreateTable
- DeleteDatabase
- DeleteTable
- DescribeBatchLoadTask
- DescribeDatabase

- DescribeEndpoints
- DescribeTable
- ListBatchLoadTasks
- ListDatabases
- ListTables
- ListTagsForResource
- ResumeBatchLoadTask
- TagResource
- UntagResource
- UpdateDatabase
- UpdateTable
- WriteRecords

CreateBatchLoadTask

Servicio: Amazon Timestream Write

Crea una nueva tarea de carga por lotes de Timestream. Una tarea de carga por lotes procesa los datos de una CSV fuente en una ubicación de S3 y los escribe en una tabla de flujo temporal. En una tarea de carga por lotes se define un mapeo del origen al destino. Los errores y los eventos se escriben en un informe en una ubicación de S3. En el caso del informe, si no se especifica la AWS KMS clave, el informe se cifrará con una clave gestionada por S3 cuando `SSE_S3` sea posible. De lo contrario, se generará un error. Para obtener más información, consulte Claves administradas de AWS. Se aplican cuotas de servicio. Para obtener más información, consulte el ejemplo de código.

Sintaxis de la solicitud

```
{
   "ClientToken": "string",
   "DataModelConfiguration": {
      "DataModel": {
         "DimensionMappings": [
            {
               "DestinationColumn": "string",
               "SourceColumn": "string"
            }
         ],
         "MeasureNameColumn": "string",
         "MixedMeasureMappings": [
            {
               "MeasureName": "string",
               "MeasureValueType": "string",
               "MultiMeasureAttributeMappings": [
                  {
                     "MeasureValueType": "string",
                     "SourceColumn": "string",
                     "TargetMultiMeasureAttributeName": "string"
                  }
               ],
               "SourceColumn": "string",
               "TargetMeasureName": "string"
            }
         ],
         "MultiMeasureMappings": {
            "MultiMeasureAttributeMappings": [
               {
```

```
                    "MeasureValueType": "string",
                    "SourceColumn": "string",
                    "TargetMultiMeasureAttributeName": "string"
                 }
              ],
              "TargetMultiMeasureName": "string"
           },
           "TimeColumn": "string",
           "TimeUnit": "string"
        },
        "DataModelS3Configuration": {
           "BucketName": "string",
           "ObjectKey": "string"
        }
     },
     "DataSourceConfiguration": {
        "CsvConfiguration": {
           "ColumnSeparator": "string",
           "EscapeChar": "string",
           "NullValue": "string",
           "QuoteChar": "string",
           "TrimWhiteSpace": boolean
        },
        "DataFormat": "string",
        "DataSourceS3Configuration": {
           "BucketName": "string",
           "ObjectKeyPrefix": "string"
        }
     },
     "RecordVersion": number,
     "ReportConfiguration": {
        "ReportS3Configuration": {
           "BucketName": "string",
           "EncryptionOption": "string",
           "KmsKeyId": "string",
           "ObjectKeyPrefix": "string"
        }
     },
     "TargetDatabaseName": "string",
     "TargetTableName": "string"
}
```

Parámetros de la solicitud

Para obtener información sobre los parámetros comunes a todas las acciones, consulte Parámetros comunes.

La solicitud acepta los siguientes datos en JSON formato.

ClientToken

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 64.

Obligatorio: no

DataModelConfiguration

Tipo: objeto DataModelConfiguration

Obligatorio: no

DataSourceConfiguration

Define los detalles de configuración de la fuente de datos para una tarea de carga por lotes.

Tipo: objeto DataSourceConfiguration

Obligatorio: sí

RecordVersion

Tipo: largo

Obligatorio: no

ReportConfiguration

Informe de la configuración de una tarea de carga por lotes. Contiene detalles sobre dónde se almacenan los informes de errores.

Tipo: objeto ReportConfiguration

Obligatorio: sí

TargetDatabaseName

Base de datos de Timestream de destino para una tarea de carga por lotes.

Tipo: cadena

Patrón: `[a-zA-Z0-9_.-]+`

Obligatorio: sí

TargetTableName

Tabla de flujo temporal de destino para una tarea de carga por lotes.

Tipo: cadena

Patrón: `[a-zA-Z0-9_.-]+`

Obligatorio: sí

Sintaxis de la respuesta

```
{
   "TaskId": "string"
}
```

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta de HTTP 200.

El servicio devuelve los siguientes datos en JSON formato.

TaskId

El ID de la tarea de carga por lotes.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 3. La longitud máxima es de 32 caracteres.

Patrón: `[A-Z0-9]+`

Errores

Para obtener información acerca de los errores comunes a todas las acciones, consulte Errores comunes.

AccessDeniedException

No está autorizado a realizar esta acción.

HTTPCódigo de estado: 400

ConflictException

Timestream no pudo procesar esta solicitud porque contiene un recurso que ya existe.

HTTPCódigo de estado: 400

InternalServerException

Timestream no pudo procesar completamente esta solicitud debido a un error interno del servidor.

HTTPCódigo de estado: 500

InvalidEndpointException

El punto final solicitado no era válido.

HTTPCódigo de estado: 400

ResourceNotFoundException

La operación intentó acceder a un recurso inexistente. Es posible que el recurso no esté especificado correctamente o que su estado no lo estéACTIVE.

HTTPCódigo de estado: 400

ServiceQuotaExceededException

Se ha superado la cuota de recursos de la instancia para esta cuenta.

HTTPCódigo de estado: 400

ThrottlingException

Un usuario realizó demasiadas solicitudes y estas superaron las cuotas de servicio. La solicitud se ha limitado.

HTTPCódigo de estado: 400

ValidationException

Solicitud no válida o mal formada.

HTTPCódigo de estado: 400

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- Interfaz de la línea de comandos de AWS
- AWS SDKpara. NET
- AWS SDKpara C++
- AWS SDKpara Go v2
- AWS SDKpara Java V2
- AWS SDKpara JavaScript V3
- AWS SDKpara PHP V3
- AWS SDKpara Python
- AWS SDKpara Ruby V3

CreateDatabase
Servicio: Amazon Timestream Write

Crea una nueva base de datos de Timestream. Si no se especifica la AWS KMS clave, la base de datos se cifrará con una AWS KMS clave gestionada por Timestream ubicada en su cuenta. Para obtener más información, consulte Claves administradas de AWS. Se aplican cuotas de servicio. Para obtener más información, consulta el ejemplo de código.

Sintaxis de la solicitud

```
{
   "DatabaseName": "string",
   "KmsKeyId": "string",
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ]
}
```

Parámetros de la solicitud

Para obtener información sobre los parámetros comunes a todas las acciones, consulte Parámetros comunes.

La solicitud acepta los siguientes datos en JSON formato.

DatabaseName

Nombre de la base de datos de Timestream.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 3. La longitud máxima es de 256 caracteres.

Patrón: `[a-zA-Z0-9_.-]+`

Obligatorio: sí

KmsKeyId

La AWS KMS clave de la base de datos. Si no se especifica la AWS KMS clave, la base de datos se cifrará con una AWS KMS clave gestionada por Timestream ubicada en su cuenta. Para obtener más información, consulte Claves administradas de AWS.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Obligatorio: no

Tags

Una lista de pares clave-valor para etiquetar la tabla.

Tipo: matriz de objetos Tag

Miembros de la matriz: número mínimo de 0 artículos. La cantidad máxima es de 200 artículos.

Obligatorio: no

Sintaxis de la respuesta

```
{
   "Database": {
      "Arn": "string",
      "CreationTime": number,
      "DatabaseName": "string",
      "KmsKeyId": "string",
      "LastUpdatedTime": number,
      "TableCount": number
   }
}
```

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta de HTTP 200.

El servicio devuelve los siguientes datos en JSON formato.

Database

La base de datos Timestream recién creada.

Tipo: objeto Database

Errores

Para obtener información acerca de los errores comunes a todas las acciones, consulte Errores comunes.

AccessDeniedException

No está autorizado a realizar esta acción.

HTTPCódigo de estado: 400

ConflictException

Timestream no pudo procesar esta solicitud porque contiene un recurso que ya existe.

HTTPCódigo de estado: 400

InternalServerException

Timestream no pudo procesar completamente esta solicitud debido a un error interno del servidor.

HTTPCódigo de estado: 500

InvalidEndpointException

El punto final solicitado no era válido.

HTTPCódigo de estado: 400

InvalidEndpointException

El punto final solicitado no era válido.

HTTPCódigo de estado: 400

ServiceQuotaExceededException

Se ha superado la cuota de recursos de la instancia para esta cuenta.

HTTPCódigo de estado: 400

ThrottlingException

Un usuario realizó demasiadas solicitudes y estas superaron las cuotas de servicio. La solicitud se ha limitado.

HTTPCódigo de estado: 400

ValidationException

Solicitud no válida o mal formada.

HTTPCódigo de estado: 400

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- Interfaz de la línea de comandos de AWS
- AWS SDKpara. NET
- AWS SDKpara C++
- AWS SDKpara Go v2
- AWS SDKpara Java V2
- AWS SDKpara JavaScript V3
- AWS SDKpara PHP V3
- AWS SDKpara Python
- AWS SDKpara Ruby V3

CreateTable

Servicio: Amazon Timestream Write

Agrega una tabla nueva a una base de datos existente en tu cuenta. En una Cuenta de AWS, los nombres de las tablas deben ser al menos únicos en cada región si están en la misma base de datos. Es posible que tenga nombres de tabla idénticos en la misma región si las tablas están en bases de datos independientes. Cuando cree la tabla, debe especificar su nombre, el nombre de la base de datos y las propiedades de retención. Se aplican cuotas de servicio. Consulte el ejemplo de código para obtener más detalles.

Sintaxis de la solicitud

```
{
   "DatabaseName": "string",
   "MagneticStoreWriteProperties": {
      "EnableMagneticStoreWrites": boolean,
      "MagneticStoreRejectedDataLocation": {
         "S3Configuration": {
            "BucketName": "string",
            "EncryptionOption": "string",
            "KmsKeyId": "string",
            "ObjectKeyPrefix": "string"
         }
      }
   },
   "RetentionProperties": {
      "MagneticStoreRetentionPeriodInDays": number,
      "MemoryStoreRetentionPeriodInHours": number
   },
   "Schema": {
      "CompositePartitionKey": [
         {
            "EnforcementInRecord": "string",
            "Name": "string",
            "Type": "string"
         }
      ]
   },
   "TableName": "string",
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
```

```
        }
    ]
}
```

Parámetros de la solicitud

Para obtener información sobre los parámetros comunes a todas las acciones, consulte Parámetros comunes.

La solicitud acepta los siguientes datos en JSON formato.

DatabaseName

Nombre de la base de datos de Timestream.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 3. La longitud máxima es de 256 caracteres.

Patrón: `[a-zA-Z0-9_.-]+`

Obligatorio: sí

MagneticStoreWriteProperties

Contiene las propiedades que se tienen que establecer en la tabla cuando se habilitan las escrituras en el almacén magnético.

Tipo: objeto MagneticStoreWriteProperties

Obligatorio: no

RetentionProperties

El tiempo durante el que deben almacenarse los datos de series temporales en el almacén de memoria y en el almacén magnético.

Tipo: objeto RetentionProperties

Obligatorio: no

Schema

El esquema de la tabla.

Tipo: objeto Schema

Obligatorio: no

TableName

Nombre de la tabla de Timestream.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 3. La longitud máxima es de 256 caracteres.

Patrón: `[a-zA-Z0-9_.-]+`

Obligatorio: sí

Tags

Una lista de pares clave-valor para etiquetar la tabla.

Tipo: matriz de objetos Tag

Miembros de la matriz: número mínimo de 0 artículos. La cantidad máxima es de 200 artículos.

Obligatorio: no

Sintaxis de la respuesta

```
{
   "Table": {
      "Arn": "string",
      "CreationTime": number,
      "DatabaseName": "string",
      "LastUpdatedTime": number,
      "MagneticStoreWriteProperties": {
         "EnableMagneticStoreWrites": boolean,
         "MagneticStoreRejectedDataLocation": {
            "S3Configuration": {
               "BucketName": "string",
               "EncryptionOption": "string",
               "KmsKeyId": "string",
               "ObjectKeyPrefix": "string"
            }
```

```
         }
      },
      "RetentionProperties": {
         "MagneticStoreRetentionPeriodInDays": number,
         "MemoryStoreRetentionPeriodInHours": number
      },
      "Schema": {
         "CompositePartitionKey": [
            {
               "EnforcementInRecord": "string",
               "Name": "string",
               "Type": "string"
            }
         ]
      },
      "TableName": "string",
      "TableStatus": "string"
   }
}
```

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta de HTTP 200.

El servicio devuelve los siguientes datos en JSON formato.

Table

La tabla Timestream recién creada.

Tipo: objeto Table

Errores

Para obtener información acerca de los errores comunes a todas las acciones, consulte Errores comunes.

AccessDeniedException

No está autorizado a realizar esta acción.

HTTPCódigo de estado: 400

ConflictException

Timestream no pudo procesar esta solicitud porque contiene un recurso que ya existe.

HTTPCódigo de estado: 400

InternalServerException

Timestream no pudo procesar completamente esta solicitud debido a un error interno del servidor.

HTTPCódigo de estado: 500

InvalidEndpointException

El punto final solicitado no era válido.

HTTPCódigo de estado: 400

InvalidEndpointException

El punto final solicitado no era válido.

HTTPCódigo de estado: 400

ResourceNotFoundException

La operación intentó acceder a un recurso inexistente. Es posible que el recurso no esté especificado correctamente o que su estado no lo estéACTIVE.

HTTPCódigo de estado: 400

ServiceQuotaExceededException

Se ha superado la cuota de recursos de la instancia para esta cuenta.

HTTPCódigo de estado: 400

ThrottlingException

Un usuario realizó demasiadas solicitudes y estas superaron las cuotas de servicio. La solicitud se ha limitado.

HTTPCódigo de estado: 400

ValidationException

Solicitud no válida o mal formada.

HTTPCódigo de estado: 400

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- Interfaz de la línea de comandos de AWS
- AWS SDKpara. NET
- AWS SDKpara C++
- AWS SDKpara Go v2
- AWS SDKpara Java V2
- AWS SDKpara JavaScript V3
- AWS SDKpara PHP V3
- AWS SDKpara Python
- AWS SDKpara Ruby V3

DeleteDatabase
Servicio: Amazon Timestream Write

Elimina una base de datos de Timestream determinada. Se trata de una operación irreversible. Una vez eliminada una base de datos, no se pueden recuperar los datos de las series temporales de sus tablas.

> **Note**
>
> Primero se deben eliminar todas las tablas de la base de datos o se ValidationException producirá un error.
> Debido a la naturaleza de los reintentos distribuidos, la operación puede arrojar un resultado exitoso o un ResourceNotFoundException. Los clientes deberían considerarlos equivalentes.

Consulte el ejemplo de código para obtener más detalles.

Sintaxis de la solicitud

```
{
   "DatabaseName": "string"
}
```

Parámetros de la solicitud

Para obtener información sobre los parámetros comunes a todas las acciones, consulte Parámetros comunes.

La solicitud acepta los siguientes datos en JSON formato.

DatabaseName

El nombre de la base de datos de Timestream que se va a eliminar.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 3. La longitud máxima es de 256 caracteres.

Obligatorio: sí

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta de HTTP 200 puntos con el cuerpo vacíoHTTP.

Errores

Para obtener información acerca de los errores comunes a todas las acciones, consulte Errores comunes.

AccessDeniedException

No está autorizado a realizar esta acción.

HTTPCódigo de estado: 400

InternalServerException

Timestream no pudo procesar completamente esta solicitud debido a un error interno del servidor.

HTTPCódigo de estado: 500

InvalidEndpointException

El punto final solicitado no era válido.

HTTPCódigo de estado: 400

ResourceNotFoundException

La operación intentó acceder a un recurso inexistente. Es posible que el recurso no esté especificado correctamente o que su estado no lo estéACTIVE.

HTTPCódigo de estado: 400

ThrottlingException

Un usuario realizó demasiadas solicitudes y estas superaron las cuotas de servicio. La solicitud se ha limitado.

HTTPCódigo de estado: 400

ValidationException

Solicitud no válida o mal formada.

HTTPCódigo de estado: 400

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- Interfaz de la línea de comandos de AWS
- AWS SDKpara. NET
- AWS SDKpara C++
- AWS SDKpara Go v2
- AWS SDKpara Java V2
- AWS SDKpara JavaScript V3
- AWS SDKpara PHP V3
- AWS SDKpara Python
- AWS SDKpara Ruby V3

DeleteTable
Servicio: Amazon Timestream Write

Elimina una tabla de flujo temporal determinada. Se trata de una operación irreversible. Tras eliminar una tabla de la base de datos de Timestream, no se pueden recuperar los datos de la serie temporal almacenados en la tabla.

> **Note**
>
> Debido a la naturaleza de los reintentos distribuidos, la operación puede arrojar resultados satisfactorios o un. ResourceNotFoundException Los clientes deberían considerarlos equivalentes.

Consulte el ejemplo de código para obtener más detalles.

Sintaxis de la solicitud

```
{
   "DatabaseName": "string",
   "TableName": "string"
}
```

Parámetros de la solicitud

Para obtener información sobre los parámetros comunes a todas las acciones, consulte Parámetros comunes.

La solicitud acepta los siguientes datos en JSON formato.

DatabaseName

El nombre de la base de datos en la que se va a eliminar la base de datos de Timestream.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 3. La longitud máxima es de 256 caracteres.

Obligatorio: sí

TableName

Nombre de la tabla Timestream que se va a eliminar.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 3. La longitud máxima es de 256 caracteres.

Obligatorio: sí

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta de HTTP 200 puntos con el cuerpo vacíoHTTP.

Errores

Para obtener información acerca de los errores comunes a todas las acciones, consulte Errores comunes.

AccessDeniedException

No está autorizado a realizar esta acción.

HTTPCódigo de estado: 400

InternalServerException

Timestream no pudo procesar completamente esta solicitud debido a un error interno del servidor.

HTTPCódigo de estado: 500

InvalidEndpointException

El punto final solicitado no era válido.

HTTPCódigo de estado: 400

ResourceNotFoundException

La operación intentó acceder a un recurso inexistente. Es posible que el recurso no esté especificado correctamente o que su estado no lo estéACTIVE.

HTTPCódigo de estado: 400

ThrottlingException

Un usuario realizó demasiadas solicitudes y estas superaron las cuotas de servicio. La solicitud se ha limitado.

HTTPCódigo de estado: 400

ValidationException

Solicitud no válida o mal formada.

HTTPCódigo de estado: 400

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- Interfaz de la línea de comandos de AWS
- AWS SDKpara. NET
- AWS SDKpara C++
- AWS SDKpara Go v2
- AWS SDKpara Java V2
- AWS SDKpara JavaScript V3
- AWS SDKpara PHP V3
- AWS SDKpara Python
- AWS SDKpara Ruby V3

DescribeBatchLoadTask
Servicio: Amazon Timestream Write

Devuelve información sobre la tarea de carga por lotes, incluidas las configuraciones, las asignaciones, el progreso y otros detalles. Se aplican cuotas de servicio. Consulte el ejemplo de código para obtener más detalles.

Sintaxis de la solicitud

```
{
   "TaskId": "string"
}
```

Parámetros de la solicitud

Para obtener información sobre los parámetros comunes a todas las acciones, consulte Parámetros comunes.

La solicitud acepta los siguientes datos en JSON formato.

TaskId

El ID de la tarea de carga por lotes.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 3. La longitud máxima es de 32 caracteres.

Patrón: `[A-Z0-9]+`

Obligatorio: sí

Sintaxis de la respuesta

```
{
   "BatchLoadTaskDescription": {
      "CreationTime": number,
      "DataModelConfiguration": {
         "DataModel": {
            "DimensionMappings": [
               {
                  "DestinationColumn": "string",
                  "SourceColumn": "string"
```

```
            }
         ],
         "MeasureNameColumn": "string",
         "MixedMeasureMappings": [
            {
               "MeasureName": "string",
               "MeasureValueType": "string",
               "MultiMeasureAttributeMappings": [
                  {
                     "MeasureValueType": "string",
                     "SourceColumn": "string",
                     "TargetMultiMeasureAttributeName": "string"
                  }
               ],
               "SourceColumn": "string",
               "TargetMeasureName": "string"
            }
         ],
         "MultiMeasureMappings": {
            "MultiMeasureAttributeMappings": [
               {
                  "MeasureValueType": "string",
                  "SourceColumn": "string",
                  "TargetMultiMeasureAttributeName": "string"
               }
            ],
            "TargetMultiMeasureName": "string"
         },
         "TimeColumn": "string",
         "TimeUnit": "string"
      },
      "DataModelS3Configuration": {
         "BucketName": "string",
         "ObjectKey": "string"
      }
   },
   "DataSourceConfiguration": {
      "CsvConfiguration": {
         "ColumnSeparator": "string",
         "EscapeChar": "string",
         "NullValue": "string",
         "QuoteChar": "string",
         "TrimWhiteSpace": boolean
      },
```

```
         "DataFormat": "string",
         "DataSourceS3Configuration": {
            "BucketName": "string",
            "ObjectKeyPrefix": "string"
         }
      },
      "ErrorMessage": "string",
      "LastUpdatedTime": number,
      "ProgressReport": {
         "BytesMetered": number,
         "FileFailures": number,
         "ParseFailures": number,
         "RecordIngestionFailures": number,
         "RecordsIngested": number,
         "RecordsProcessed": number
      },
      "RecordVersion": number,
      "ReportConfiguration": {
         "ReportS3Configuration": {
            "BucketName": "string",
            "EncryptionOption": "string",
            "KmsKeyId": "string",
            "ObjectKeyPrefix": "string"
         }
      },
      "ResumableUntil": number,
      "TargetDatabaseName": "string",
      "TargetTableName": "string",
      "TaskId": "string",
      "TaskStatus": "string"
   }
}
```

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta de HTTP 200.

El servicio devuelve los siguientes datos en JSON formato.

BatchLoadTaskDescription

Descripción de la tarea de carga por lotes.

Tipo: objeto BatchLoadTaskDescription

Errores

Para obtener información acerca de los errores comunes a todas las acciones, consulte Errores comunes.

AccessDeniedException

No está autorizado a realizar esta acción.

HTTPCódigo de estado: 400

InternalServerException

Timestream no pudo procesar completamente esta solicitud debido a un error interno del servidor.

HTTPCódigo de estado: 500

InvalidEndpointException

El punto final solicitado no era válido.

HTTPCódigo de estado: 400

ResourceNotFoundException

La operación intentó acceder a un recurso inexistente. Es posible que el recurso no esté especificado correctamente o que su estado no lo estéACTIVE.

HTTPCódigo de estado: 400

ThrottlingException

Un usuario realizó demasiadas solicitudes y estas superaron las cuotas de servicio. La solicitud se ha limitado.

HTTPCódigo de estado: 400

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- Interfaz de la línea de comandos de AWS
- AWS SDKpara. NET

- AWS SDKpara C++
- AWS SDKpara Go v2
- AWS SDKpara Java V2
- AWS SDKpara JavaScript V3
- AWS SDKpara PHP V3
- AWS SDKpara Python
- AWS SDKpara Ruby V3

DescribeDatabase
Servicio: Amazon Timestream Write

Devuelve información sobre la base de datos, incluido el nombre de la base de datos, la hora en que se creó la base de datos y el número total de tablas que se encuentran en la base de datos. Se aplican cuotas de servicio. Consulte el ejemplo de código para obtener más detalles.

Sintaxis de la solicitud

```
{
   "DatabaseName": "string"
}
```

Parámetros de la solicitud

Para obtener información sobre los parámetros comunes a todas las acciones, consulte Parámetros comunes.

La solicitud acepta los siguientes datos en JSON formato.

DatabaseName

Nombre de la base de datos de Timestream.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 3. La longitud máxima es de 256 caracteres.

Obligatorio: sí

Sintaxis de la respuesta

```
{
   "Database": {
      "Arn": "string",
      "CreationTime": number,
      "DatabaseName": "string",
      "KmsKeyId": "string",
      "LastUpdatedTime": number,
      "TableCount": number
   }
```

```
}
```

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta de HTTP 200.

El servicio devuelve los siguientes datos en JSON formato.

Database

Nombre de la tabla de Timestream.

Tipo: objeto Database

Errores

Para obtener información acerca de los errores comunes a todas las acciones, consulte Errores comunes.

AccessDeniedException

No está autorizado a realizar esta acción.

HTTPCódigo de estado: 400

InternalServerException

Timestream no pudo procesar completamente esta solicitud debido a un error interno del servidor.

HTTPCódigo de estado: 500

InvalidEndpointException

El punto final solicitado no era válido.

HTTPCódigo de estado: 400

ResourceNotFoundException

La operación intentó acceder a un recurso inexistente. Es posible que el recurso no esté especificado correctamente o que su estado no lo estéACTIVE.

HTTPCódigo de estado: 400

ThrottlingException

Un usuario realizó demasiadas solicitudes y estas superaron las cuotas de servicio. La solicitud se ha limitado.

HTTPCódigo de estado: 400

ValidationException

Solicitud no válida o mal formada.

HTTPCódigo de estado: 400

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- Interfaz de la línea de comandos de AWS
- AWS SDKpara. NET
- AWS SDKpara C++
- AWS SDKpara Go v2
- AWS SDKpara Java V2
- AWS SDKpara JavaScript V3
- AWS SDKpara PHP V3
- AWS SDKpara Python
- AWS SDKpara Ruby V3

DescribeEndpoints
Servicio: Amazon Timestream Write

Devuelve una lista de puntos finales disponibles para realizar llamadas de Timestream. API Esta API operación está disponible mediante la escritura y la consulta. APIs

Dado que los Timestream SDKs están diseñados para funcionar de forma transparente con la arquitectura del servicio, incluida la administración y el mapeo de los puntos finales del servicio, no se recomienda utilizar esta operación a menos que: API

- Está utilizando VPCendpoints () con Timestream AWS PrivateLink
- Su aplicación utiliza un lenguaje de programación que aún no es compatible SDK
- Necesita un mejor control sobre la implementación del lado del cliente

Para obtener información detallada sobre cómo y cuándo usar e implementar DescribeEndpoints, consulte The Endpoint Discovery Pattern.

Sintaxis de la respuesta

```
{
   "Endpoints": [
      {
         "Address": "string",
         "CachePeriodInMinutes": number
      }
   ]
}
```

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta de HTTP 200.

El servicio devuelve los siguientes datos en JSON formato.

Endpoints

Se devuelve un `Endpoints` objeto cuando se realiza una `DescribeEndpoints` solicitud.

Tipo: matriz de objetos Endpoint

Errores

Para obtener información acerca de los errores comunes a todas las acciones, consulte Errores comunes.

InternalServerException

Timestream no pudo procesar completamente esta solicitud debido a un error interno del servidor.

HTTPCódigo de estado: 500

ThrottlingException

Un usuario realizó demasiadas solicitudes y estas superaron las cuotas de servicio. La solicitud se ha limitado.

HTTPCódigo de estado: 400

ValidationException

Solicitud no válida o mal formada.

HTTPCódigo de estado: 400

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- Interfaz de la línea de comandos de AWS
- AWS SDKpara. NET
- AWS SDKpara C++
- AWS SDKpara Go v2
- AWS SDKpara Java V2
- AWS SDKpara JavaScript V3
- AWS SDKpara PHP V3
- AWS SDKpara Python
- AWS SDKpara Ruby V3

DescribeTable

Servicio: Amazon Timestream Write

Devuelve información sobre la tabla, incluidos el nombre de la tabla, el nombre de la base de datos, la duración de retención del almacén de memoria y del almacén magnético. Se aplican cuotas de servicio. Consulte el ejemplo de código para obtener más detalles.

Sintaxis de la solicitud

```
{
   "DatabaseName": "string",
   "TableName": "string"
}
```

Parámetros de la solicitud

Para obtener información sobre los parámetros comunes a todas las acciones, consulte Parámetros comunes.

La solicitud acepta los siguientes datos en JSON formato.

DatabaseName

Nombre de la base de datos de Timestream.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 3. La longitud máxima es de 256 caracteres.

Obligatorio: sí

TableName

Nombre de la tabla de Timestream.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 3. La longitud máxima es de 256 caracteres.

Obligatorio: sí

Sintaxis de la respuesta

```
{
```

```
   "Table": {
      "Arn": "string",
      "CreationTime": number,
      "DatabaseName": "string",
      "LastUpdatedTime": number,
      "MagneticStoreWriteProperties": {
         "EnableMagneticStoreWrites": boolean,
         "MagneticStoreRejectedDataLocation": {
            "S3Configuration": {
               "BucketName": "string",
               "EncryptionOption": "string",
               "KmsKeyId": "string",
               "ObjectKeyPrefix": "string"
            }
         }
      },
      "RetentionProperties": {
         "MagneticStoreRetentionPeriodInDays": number,
         "MemoryStoreRetentionPeriodInHours": number
      },
      "Schema": {
         "CompositePartitionKey": [
            {
               "EnforcementInRecord": "string",
               "Name": "string",
               "Type": "string"
            }
         ]
      },
      "TableName": "string",
      "TableStatus": "string"
   }
}
```

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta de HTTP 200.

El servicio devuelve los siguientes datos en JSON formato.

Table

La tabla Timestream.

Tipo: objeto Table

Errores

Para obtener información acerca de los errores comunes a todas las acciones, consulte Errores comunes.

AccessDeniedException

No está autorizado a realizar esta acción.

HTTPCódigo de estado: 400

InternalServerException

Timestream no pudo procesar completamente esta solicitud debido a un error interno del servidor.

HTTPCódigo de estado: 500

InvalidEndpointException

El punto final solicitado no era válido.

HTTPCódigo de estado: 400

ResourceNotFoundException

La operación intentó acceder a un recurso inexistente. Es posible que el recurso no esté especificado correctamente o que su estado no lo estéACTIVE.

HTTPCódigo de estado: 400

ThrottlingException

Un usuario realizó demasiadas solicitudes y estas superaron las cuotas de servicio. La solicitud se ha limitado.

HTTPCódigo de estado: 400

ValidationException

Solicitud no válida o mal formada.

HTTPCódigo de estado: 400

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- Interfaz de la línea de comandos de AWS
- AWS SDKpara. NET
- AWS SDKpara C++
- AWS SDKpara Go v2
- AWS SDKpara Java V2
- AWS SDKpara JavaScript V3
- AWS SDKpara PHP V3
- AWS SDKpara Python
- AWS SDKpara Ruby V3

ListBatchLoadTasks
Servicio: Amazon Timestream Write

Proporciona una lista de las tareas de carga por lotes, junto con el nombre, el estado, hasta cuándo se puede reanudar la tarea y otros detalles. Consulte el ejemplo de código para obtener más detalles.

Sintaxis de la solicitud

```
{
   "MaxResults": number,
   "NextToken": "string",
   "TaskStatus": "string"
}
```

Parámetros de la solicitud

Para obtener información sobre los parámetros comunes a todas las acciones, consulte Parámetros comunes.

La solicitud acepta los siguientes datos en JSON formato.

MaxResults

El número total de elementos que se van a devolver en la salida. Si el número total de artículos disponibles es superior al valor especificado, NextToken se proporciona a en la salida. Para reanudar la paginación, proporcione el NextToken valor como argumento de una API invocación posterior.

Tipo: entero

Rango válido: valor mínimo de 1. Valor máximo de 100.

Obligatorio: no

NextToken

Un token destinado a especificar dónde iniciar la paginación. Es el NextToken de una respuesta previamente truncada.

Tipo: cadena

Requerido: no

TaskStatus

Estado de la tarea de carga por lotes.

Tipo: cadena

Valores válidos: `CREATED | IN_PROGRESS | FAILED | SUCCEEDED | PROGRESS_STOPPED | PENDING_RESUME`

Obligatorio: no

Sintaxis de la respuesta

```
{
   "BatchLoadTasks": [
      {
         "CreationTime": number,
         "DatabaseName": "string",
         "LastUpdatedTime": number,
         "ResumableUntil": number,
         "TableName": "string",
         "TaskId": "string",
         "TaskStatus": "string"
      }
   ],
   "NextToken": "string"
}
```

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta de HTTP 200.

El servicio devuelve los siguientes datos en JSON formato.

BatchLoadTasks

Una lista de detalles de las tareas de carga por lotes.

Tipo: matriz de objetos BatchLoadTask

NextToken

Un token destinado a especificar dónde iniciar la paginación. Proporcione la siguiente ListBatchLoadTasksRequest.

Tipo: cadena

Errores

Para obtener información acerca de los errores comunes a todas las acciones, consulte Errores comunes.

AccessDeniedException

No está autorizado a realizar esta acción.

HTTPCódigo de estado: 400

InternalServerException

Timestream no pudo procesar completamente esta solicitud debido a un error interno del servidor.

HTTPCódigo de estado: 500

InvalidEndpointException

El punto final solicitado no era válido.

HTTPCódigo de estado: 400

ThrottlingException

Un usuario realizó demasiadas solicitudes y estas superaron las cuotas de servicio. La solicitud se ha limitado.

HTTPCódigo de estado: 400

ValidationException

Solicitud no válida o mal formada.

HTTPCódigo de estado: 400

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- Interfaz de la línea de comandos de AWS

- AWS SDKpara. NET
- AWS SDKpara C++
- AWS SDKpara Go v2
- AWS SDKpara Java V2
- AWS SDKpara JavaScript V3
- AWS SDKpara PHP V3
- AWS SDKpara Python
- AWS SDKpara Ruby V3

ListDatabases
Servicio: Amazon Timestream Write

Devuelve una lista de sus bases de datos de Timestream. Se aplican cuotas de servicio. Consulte el ejemplo de código para obtener más detalles.

Sintaxis de la solicitud

```
{
   "MaxResults": number,
   "NextToken": "string"
}
```

Parámetros de la solicitud

Para obtener información sobre los parámetros comunes a todas las acciones, consulte Parámetros comunes.

La solicitud acepta los siguientes datos en JSON formato.

MaxResults

El número total de elementos que se van a devolver en la salida. Si el número total de artículos disponibles es superior al valor especificado, NextToken se proporciona a en la salida. Para reanudar la paginación, proporcione el NextToken valor como argumento de una API invocación posterior.

Tipo: entero

Rango válido: valor mínimo de 1. Valor máximo de 20.

Obligatorio: no

NextToken

El token de paginación. Para reanudar la paginación, proporciona el NextToken valor como argumento de una invocación posteriorAPI.

Tipo: cadena

Requerido: no

Sintaxis de la respuesta

```
{
   "Databases": [
      {
         "Arn": "string",
         "CreationTime": number,
         "DatabaseName": "string",
         "KmsKeyId": "string",
         "LastUpdatedTime": number,
         "TableCount": number
      }
   ],
   "NextToken": "string"
}
```

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta de HTTP 200.

El servicio devuelve los siguientes datos en JSON formato.

Databases

Una lista de nombres de bases de datos.

Tipo: matriz de objetos Database

NextToken

El token de paginación. Este parámetro se devuelve cuando se trunca la respuesta.

Tipo: cadena

Errores

Para obtener información acerca de los errores comunes a todas las acciones, consulte Errores comunes.

AccessDeniedException

No está autorizado a realizar esta acción.

HTTPCódigo de estado: 400

InternalServerException

Timestream no pudo procesar completamente esta solicitud debido a un error interno del servidor.

HTTPCódigo de estado: 500

InvalidEndpointException

El punto final solicitado no era válido.

HTTPCódigo de estado: 400

ThrottlingException

Un usuario realizó demasiadas solicitudes y estas superaron las cuotas de servicio. La solicitud se ha limitado.

HTTPCódigo de estado: 400

ValidationException

Solicitud no válida o mal formada.

HTTPCódigo de estado: 400

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- Interfaz de la línea de comandos de AWS
- AWS SDKpara. NET
- AWS SDKpara C++
- AWS SDKpara Go v2
- AWS SDKpara Java V2
- AWS SDKpara JavaScript V3
- AWS SDKpara PHP V3
- AWS SDKpara Python
- AWS SDKpara Ruby V3

ListTables

Servicio: Amazon Timestream Write

Proporciona una lista de tablas, junto con el nombre, el estado y las propiedades de retención de cada tabla. Consulte el ejemplo de código para obtener más detalles.

Sintaxis de la solicitud

```
{
   "DatabaseName": "string",
   "MaxResults": number,
   "NextToken": "string"
}
```

Parámetros de la solicitud

Para obtener información sobre los parámetros comunes a todas las acciones, consulte Parámetros comunes.

La solicitud acepta los siguientes datos en JSON formato.

DatabaseName

Nombre de la base de datos de Timestream.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 3. La longitud máxima es de 256 caracteres.

Obligatorio: no

MaxResults

El número total de elementos que se van a devolver en la salida. Si el número total de artículos disponibles es superior al valor especificado, NextToken se proporciona a en la salida. Para reanudar la paginación, proporcione el NextToken valor como argumento de una API invocación posterior.

Tipo: entero

Rango válido: valor mínimo de 1. Valor máximo de 20.

Obligatorio: no

NextToken

El token de paginación. Para reanudar la paginación, proporciona el NextToken valor como argumento de una invocación posteriorAPI.

Tipo: cadena

Requerido: no

Sintaxis de la respuesta

```
{
   "NextToken": "string",
   "Tables": [
      {
         "Arn": "string",
         "CreationTime": number,
         "DatabaseName": "string",
         "LastUpdatedTime": number,
         "MagneticStoreWriteProperties": {
            "EnableMagneticStoreWrites": boolean,
            "MagneticStoreRejectedDataLocation": {
               "S3Configuration": {
                  "BucketName": "string",
                  "EncryptionOption": "string",
                  "KmsKeyId": "string",
                  "ObjectKeyPrefix": "string"
               }
            }
         },
         "RetentionProperties": {
            "MagneticStoreRetentionPeriodInDays": number,
            "MemoryStoreRetentionPeriodInHours": number
         },
         "Schema": {
            "CompositePartitionKey": [
               {
                  "EnforcementInRecord": "string",
                  "Name": "string",
                  "Type": "string"
               }
            ]
         },
```

```
            "TableName": "string",
            "TableStatus": "string"
        }
    ]
}
```

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta de HTTP 200.

El servicio devuelve los siguientes datos en JSON formato.

NextToken

Un token destinado a especificar dónde iniciar la paginación. Se trata NextToken de una respuesta previamente truncada.

Tipo: cadena

Tables

Una lista de tablas.

Tipo: matriz de objetos Table

Errores

Para obtener información acerca de los errores comunes a todas las acciones, consulte Errores comunes.

AccessDeniedException

No está autorizado a realizar esta acción.

HTTPCódigo de estado: 400

InternalServerException

Timestream no pudo procesar completamente esta solicitud debido a un error interno del servidor.

HTTPCódigo de estado: 500

InvalidEndpointException

El punto final solicitado no era válido.

HTTPCódigo de estado: 400

ResourceNotFoundException

La operación intentó acceder a un recurso inexistente. Es posible que el recurso no esté especificado correctamente o que su estado no lo estéACTIVE.

HTTPCódigo de estado: 400

ThrottlingException

Un usuario realizó demasiadas solicitudes y estas superaron las cuotas de servicio. La solicitud se ha limitado.

HTTPCódigo de estado: 400

ValidationException

Solicitud no válida o mal formada.

HTTPCódigo de estado: 400

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- Interfaz de la línea de comandos de AWS
- AWS SDKpara. NET
- AWS SDKpara C++
- AWS SDKpara Go v2
- AWS SDKpara Java V2
- AWS SDKpara JavaScript V3
- AWS SDKpara PHP V3
- AWS SDKpara Python
- AWS SDKpara Ruby V3

ListTagsForResource
Servicio: Amazon Timestream Write

Muestra todas las etiquetas de un recurso de Timestream.

Sintaxis de la solicitud

```
{
   "ResourceARN": "string"
}
```

Parámetros de la solicitud

Para obtener información sobre los parámetros comunes a todas las acciones, consulte Parámetros comunes.

La solicitud acepta los siguientes datos en JSON formato.

ResourceARN

El recurso Timestream con las etiquetas que se van a enumerar. Este valor es un nombre de recurso de Amazon (ARN).

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. Longitud máxima de 1011.

Obligatorio: sí

Sintaxis de la respuesta

```
{
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ]
}
```

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta de HTTP 200.

El servicio devuelve los siguientes datos en JSON formato.

Tags

Las etiquetas asociadas actualmente al recurso Timestream.

Tipo: matriz de objetos Tag

Miembros de la matriz: número mínimo de 0 artículos. La cantidad máxima es de 200 artículos.

Errores

Para obtener información acerca de los errores comunes a todas las acciones, consulte Errores comunes.

InvalidEndpointException

El punto final solicitado no era válido.

HTTPCódigo de estado: 400

ResourceNotFoundException

La operación intentó acceder a un recurso inexistente. Es posible que el recurso no esté especificado correctamente o que su estado no lo estéACTIVE.

HTTPCódigo de estado: 400

ThrottlingException

Un usuario realizó demasiadas solicitudes y estas superaron las cuotas de servicio. La solicitud se ha limitado.

HTTPCódigo de estado: 400

ValidationException

Solicitud no válida o mal formada.

HTTPCódigo de estado: 400

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- Interfaz de la línea de comandos de AWS
- AWS SDKpara. NET
- AWS SDKpara C++
- AWS SDKpara Go v2
- AWS SDKpara Java V2
- AWS SDKpara JavaScript V3
- AWS SDKpara PHP V3
- AWS SDKpara Python
- AWS SDKpara Ruby V3

ResumeBatchLoadTask
Servicio: Amazon Timestream Write

Sintaxis de la solicitud

```
{
   "TaskId": "string"
}
```

Parámetros de la solicitud

Para obtener información sobre los parámetros comunes a todas las acciones, consulte Parámetros comunes.

La solicitud acepta los siguientes datos en JSON formato.

TaskId

El ID de la tarea de carga por lotes que se va a reanudar.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 3. La longitud máxima es de 32 caracteres.

Patrón: `[A-Z0-9]+`

Obligatorio: sí

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta de HTTP 200 puntos con HTTP el cuerpo vacío.

Errores

Para obtener información acerca de los errores comunes a todas las acciones, consulte Errores comunes.

AccessDeniedException

No está autorizado a realizar esta acción.

HTTPCódigo de estado: 400

InternalServerException

Timestream no pudo procesar completamente esta solicitud debido a un error interno del servidor.

HTTPCódigo de estado: 500

InvalidEndpointException

El punto final solicitado no era válido.

HTTPCódigo de estado: 400

ResourceNotFoundException

La operación intentó acceder a un recurso inexistente. Es posible que el recurso no esté especificado correctamente o que su estado no lo estéACTIVE.

HTTPCódigo de estado: 400

ThrottlingException

Un usuario realizó demasiadas solicitudes y estas superaron las cuotas de servicio. La solicitud se ha limitado.

HTTPCódigo de estado: 400

ValidationException

Solicitud no válida o mal formada.

HTTPCódigo de estado: 400

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- Interfaz de la línea de comandos de AWS
- AWS SDKpara. NET
- AWS SDKpara C++
- AWS SDKpara Go v2
- AWS SDKpara Java V2

- AWS SDKpara JavaScript V3
- AWS SDKpara PHP V3
- AWS SDKpara Python
- AWS SDKpara Ruby V3

TagResource
Servicio: Amazon Timestream Write

Asocia un conjunto de etiquetas a un recurso de Timestream. A continuación, puede activar estas etiquetas definidas por el usuario para que aparezcan en la consola Billing and Cost Management para el seguimiento de la asignación de costes.

Sintaxis de la solicitud

```
{
   "ResourceARN": "string",
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ]
}
```

Parámetros de la solicitud

Para obtener información sobre los parámetros comunes a todas las acciones, consulte Parámetros comunes.

La solicitud acepta los siguientes datos en JSON formato.

ResourceARN

Identifica el recurso Timestream al que se deben agregar las etiquetas. Este valor es un nombre de recurso de Amazon (ARN).

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. Longitud máxima de 1011.

Obligatorio: sí

Tags

Las etiquetas que se van a asignar al recurso Timestream.

Tipo: matriz de objetos Tag

Miembros de la matriz: número mínimo de 0 artículos. La cantidad máxima es de 200 artículos.

Obligatorio: sí

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta de HTTP 200 puntos con el cuerpo vacíoHTTP.

Errores

Para obtener información acerca de los errores comunes a todas las acciones, consulte Errores comunes.

InvalidEndpointException

El punto final solicitado no era válido.

HTTPCódigo de estado: 400

ResourceNotFoundException

La operación intentó acceder a un recurso inexistente. Es posible que el recurso no esté especificado correctamente o que su estado no lo estéACTIVE.

HTTPCódigo de estado: 400

ServiceQuotaExceededException

Se ha superado la cuota de recursos de la instancia para esta cuenta.

HTTPCódigo de estado: 400

ThrottlingException

Un usuario realizó demasiadas solicitudes y estas superaron las cuotas de servicio. La solicitud se ha limitado.

HTTPCódigo de estado: 400

ValidationException

Solicitud no válida o mal formada.

HTTPCódigo de estado: 400

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- Interfaz de la línea de comandos de AWS
- AWS SDKpara. NET
- AWS SDKpara C++
- AWS SDKpara Go v2
- AWS SDKpara Java V2
- AWS SDKpara JavaScript V3
- AWS SDKpara PHP V3
- AWS SDKpara Python
- AWS SDKpara Ruby V3

UntagResource
Servicio: Amazon Timestream Write

Elimina la asociación de etiquetas de un recurso de Timestream.

Sintaxis de la solicitud

```
{
   "ResourceARN": "string",
   "TagKeys": [ "string" ]
}
```

Parámetros de la solicitud

Para obtener información sobre los parámetros comunes a todas las acciones, consulte Parámetros comunes.

La solicitud acepta los siguientes datos en JSON formato.

ResourceARN

El recurso Timestream del que se eliminarán las etiquetas. Este valor es un nombre de recurso de Amazon (ARN).

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. Longitud máxima de 1011.

Obligatorio: sí

TagKeys

Una lista de claves de etiquetas. Las etiquetas existentes del recurso cuyas claves son miembros de esta lista se eliminarán del recurso Timestream.

Tipo: matriz de cadenas

Miembros de la matriz: número mínimo de 0 artículos. La cantidad máxima es de 200 artículos.

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 128 caracteres.

Obligatorio: sí

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta de HTTP 200 puntos con el cuerpo vacíoHTTP.

Errores

Para obtener información acerca de los errores comunes a todas las acciones, consulte Errores comunes.

InvalidEndpointException

El punto final solicitado no era válido.

HTTPCódigo de estado: 400

ResourceNotFoundException

La operación intentó acceder a un recurso inexistente. Es posible que el recurso no esté especificado correctamente o que su estado no lo estéACTIVE.

HTTPCódigo de estado: 400

ServiceQuotaExceededException

Se ha superado la cuota de recursos de la instancia para esta cuenta.

HTTPCódigo de estado: 400

ThrottlingException

Un usuario realizó demasiadas solicitudes y estas superaron las cuotas de servicio. La solicitud se ha limitado.

HTTPCódigo de estado: 400

ValidationException

Solicitud no válida o mal formada.

HTTPCódigo de estado: 400

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- Interfaz de la línea de comandos de AWS
- AWS SDKpara. NET
- AWS SDKpara C++
- AWS SDKpara Go v2
- AWS SDKpara Java V2
- AWS SDKpara JavaScript V3
- AWS SDKpara PHP V3
- AWS SDKpara Python
- AWS SDKpara Ruby V3

UpdateDatabase
Servicio: Amazon Timestream Write

Modifica la AWS KMS clave de una base de datos existente. Al actualizar la base de datos, debe especificar el nombre de la base de datos y el identificador de la nueva AWS KMS clave que se va a utilizar (`KmsKeyId`). Si hay `UpdateDatabase` solicitudes simultáneas, gana el primer escritor.

Consulte el ejemplo de código para obtener más detalles.

Sintaxis de la solicitud

```
{
   "DatabaseName": "string",
   "KmsKeyId": "string"
}
```

Parámetros de la solicitud

Para obtener información sobre los parámetros comunes a todas las acciones, consulte Parámetros comunes.

La solicitud acepta los siguientes datos en JSON formato.

DatabaseName

El nombre de la base de datos.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 3. La longitud máxima es de 256 caracteres.

Obligatorio: sí

KmsKeyId

El identificador de la nueva AWS KMS clave (`KmsKeyId`) que se utilizará para cifrar los datos almacenados en la base de datos. Si la información `KmsKeyId` actualmente registrada `KmsKeyId` en la base de datos es la misma que la de la solicitud, no habrá ninguna actualización.

Puede especificarlo `KmsKeyId` mediante cualquiera de las siguientes opciones:

- ID de clave: `1234abcd-12ab-34cd-56ef-1234567890ab`
- ClaveARN: `arn:aws:kms:us-east-1:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab`

- Nombre de alias: `alias/ExampleAlias`
- AliasARN: `arn:aws:kms:us-east-1:111122223333:alias/ExampleAlias`

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Obligatorio: sí

Sintaxis de la respuesta

```
{
   "Database": {
      "Arn": "string",
      "CreationTime": number,
      "DatabaseName": "string",
      "KmsKeyId": "string",
      "LastUpdatedTime": number,
      "TableCount": number
   }
}
```

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta de HTTP 200.

El servicio devuelve los siguientes datos en JSON formato.

Database

Un contenedor de nivel superior para una tabla. Las bases de datos y las tablas son los conceptos de administración fundamentales de Amazon Timestream. Todas las tablas de una base de datos se cifran con la misma AWS KMS clave.

Tipo: objeto Database

Errores

Para obtener información acerca de los errores comunes a todas las acciones, consulte Errores comunes.

AccessDeniedException

No está autorizado a realizar esta acción.

HTTPCódigo de estado: 400

InternalServerException

Timestream no pudo procesar completamente esta solicitud debido a un error interno del servidor.

HTTPCódigo de estado: 500

InvalidEndpointException

El punto final solicitado no era válido.

HTTPCódigo de estado: 400

ResourceNotFoundException

La operación intentó acceder a un recurso inexistente. Es posible que el recurso no esté especificado correctamente o que su estado no lo estéACTIVE.

HTTPCódigo de estado: 400

ServiceQuotaExceededException

Se ha superado la cuota de recursos de la instancia para esta cuenta.

HTTPCódigo de estado: 400

ThrottlingException

Un usuario realizó demasiadas solicitudes y estas superaron las cuotas de servicio. La solicitud se ha limitado.

HTTPCódigo de estado: 400

ValidationException

Solicitud no válida o mal formada.

HTTPCódigo de estado: 400

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- Interfaz de la línea de comandos de AWS
- AWS SDKpara. NET
- AWS SDKpara C++
- AWS SDKpara Go v2
- AWS SDKpara Java V2
- AWS SDKpara JavaScript V3
- AWS SDKpara PHP V3
- AWS SDKpara Python
- AWS SDKpara Ruby V3

UpdateTable

Servicio: Amazon Timestream Write

Modifica la duración de retención del almacén de memoria y del almacén magnético de la tabla Timestream. Tenga en cuenta que el cambio en la duración de la retención se aplica inmediatamente. Por ejemplo, si el período de retención del almacén de memoria se estableció inicialmente en 2 horas y luego se cambió a 24 horas, el almacén de memoria podrá almacenar 24 horas de datos, pero se rellenará con 24 horas de datos 22 horas después de realizar este cambio. Timestream no recupera los datos del almacén magnético para rellenar el almacén de memoria.

Consulte el ejemplo de código para obtener más detalles.

Sintaxis de la solicitud

```
{
   "DatabaseName": "string",
   "MagneticStoreWriteProperties": {
      "EnableMagneticStoreWrites": boolean,
      "MagneticStoreRejectedDataLocation": {
         "S3Configuration": {
            "BucketName": "string",
            "EncryptionOption": "string",
            "KmsKeyId": "string",
            "ObjectKeyPrefix": "string"
         }
      }
   },
   "RetentionProperties": {
      "MagneticStoreRetentionPeriodInDays": number,
      "MemoryStoreRetentionPeriodInHours": number
   },
   "Schema": {
      "CompositePartitionKey": [
         {
            "EnforcementInRecord": "string",
            "Name": "string",
            "Type": "string"
         }
      ]
   },
   "TableName": "string"
}
```

Parámetros de la solicitud

Para obtener información sobre los parámetros comunes a todas las acciones, consulte Parámetros comunes.

La solicitud acepta los siguientes datos en formato. JSON

DatabaseName

Nombre de la base de datos de Timestream.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 3. La longitud máxima es de 256 caracteres.

Obligatorio: sí

MagneticStoreWriteProperties

Contiene las propiedades que se tienen que establecer en la tabla cuando se habilitan las escrituras en el almacén magnético.

Tipo: objeto MagneticStoreWriteProperties

Obligatorio: no

RetentionProperties

El tiempo de retención del almacén de memoria y del almacén magnético.

Tipo: objeto RetentionProperties

Obligatorio: no

Schema

El esquema de la tabla.

Tipo: objeto Schema

Obligatorio: no

TableName

Nombre de la tabla de Timestream.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 3. La longitud máxima es de 256 caracteres.

Obligatorio: sí

Sintaxis de la respuesta

```
{
   "Table": {
      "Arn": "string",
      "CreationTime": number,
      "DatabaseName": "string",
      "LastUpdatedTime": number,
      "MagneticStoreWriteProperties": {
         "EnableMagneticStoreWrites": boolean,
         "MagneticStoreRejectedDataLocation": {
            "S3Configuration": {
               "BucketName": "string",
               "EncryptionOption": "string",
               "KmsKeyId": "string",
               "ObjectKeyPrefix": "string"
            }
         }
      },
      "RetentionProperties": {
         "MagneticStoreRetentionPeriodInDays": number,
         "MemoryStoreRetentionPeriodInHours": number
      },
      "Schema": {
         "CompositePartitionKey": [
            {
               "EnforcementInRecord": "string",
               "Name": "string",
               "Type": "string"
            }
         ]
      },
      "TableName": "string",
      "TableStatus": "string"
   }
}
```

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta de HTTP 200.

El servicio devuelve los siguientes datos en JSON formato.

Table

La tabla Timestream actualizada.

Tipo: objeto Table

Errores

Para obtener información acerca de los errores comunes a todas las acciones, consulte Errores comunes.

AccessDeniedException

No está autorizado a realizar esta acción.

HTTPCódigo de estado: 400

InternalServerException

Timestream no pudo procesar completamente esta solicitud debido a un error interno del servidor.

HTTPCódigo de estado: 500

InvalidEndpointException

El punto final solicitado no era válido.

HTTPCódigo de estado: 400

ResourceNotFoundException

La operación intentó acceder a un recurso inexistente. Es posible que el recurso no esté especificado correctamente o que su estado no lo estéACTIVE.

HTTPCódigo de estado: 400

ThrottlingException

Un usuario realizó demasiadas solicitudes y estas superaron las cuotas de servicio. La solicitud se ha limitado.

HTTPCódigo de estado: 400

ValidationException

Solicitud no válida o mal formada.

HTTPCódigo de estado: 400

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- Interfaz de la línea de comandos de AWS
- AWS SDKpara. NET
- AWS SDKpara C++
- AWS SDKpara Go v2
- AWS SDKpara Java V2
- AWS SDKpara JavaScript V3
- AWS SDKpara PHP V3
- AWS SDKpara Python
- AWS SDKpara Ruby V3

WriteRecords
Servicio: Amazon Timestream Write

Le permite escribir sus datos de series temporales en Timestream. Puede especificar un único punto de datos o un lote de puntos de datos para insertarlos en el sistema. Timestream le ofrece un esquema flexible que detecta automáticamente los nombres de las columnas y los tipos de datos de las tablas de Timestream en función de los nombres de las dimensiones y los tipos de datos de los puntos de datos que especifique al invocar las escrituras en la base de datos.

Timestream admite una semántica de lectura coherente eventual. Esto significa que cuando consultas datos inmediatamente después de escribir un lote de datos en Timestream, es posible que los resultados de la consulta no reflejen los resultados de una operación de escritura completada recientemente. Los resultados también pueden incluir algunos datos obsoletos. Si repite la solicitud de consulta después de un breve período de tiempo, los resultados deberían arrojar los datos más recientes. Se aplican cuotas de servicio.

Consulte el ejemplo de código para obtener más detalles.

¿Molestas

Puede utilizar el `Version` parámetro en una `WriteRecords` solicitud para actualizar los puntos de datos. Timestream rastrea un número de versión con cada registro. `Version`el valor predeterminado es 1 cuando no está especificado para el registro en la solicitud. Timestream actualiza el valor de medición de un registro existente junto con el suyo `Version` cuando recibe una solicitud de escritura con un `Version` número superior para ese registro. Cuando recibe una solicitud de actualización en la que el valor de la medida es el mismo que el del registro existente, Timestream sigue `Version` actualizándose si es mayor que el valor existente de. `Version` Puede actualizar un punto de datos tantas veces como desee, siempre que el valor de aumente de `Version` forma continua.

Por ejemplo, supongamos que escribe un registro nuevo sin indicarlo `Version` en la solicitud. Timestream almacena este registro y lo establece `Version` en. 1 Ahora, supongamos que intenta actualizar este registro con una `WriteRecords` solicitud del mismo registro con un valor de medida diferente, pero, como antes, no lo proporciona. `Version` En este caso, Timestream rechazará esta actualización con una, `RejectedRecordsException` ya que la versión del registro actualizado no es superior al valor actual de la versión.

Sin embargo, si volvieras a enviar la solicitud de actualización con el valor `Version` establecido en2, Timestream lograría actualizar el valor del registro y se establecería en. `Version` 2 A continuación, supongamos que ha enviado una `WriteRecords` solicitud con el mismo registro y un valor de medida idéntico, pero con `Version` el valor establecido en. 3 En este caso, Timestream solo se

`Version` actualizaría a. `3` Cualquier actualización adicional tendría que enviar un número de versión superior a`3`, o las solicitudes de actualización recibirían un. `RejectedRecordsException`

Sintaxis de la solicitud

```
{
   "CommonAttributes": {
      "Dimensions": [
         {
            "DimensionValueType": "string",
            "Name": "string",
            "Value": "string"
         }
      ],
      "MeasureName": "string",
      "MeasureValue": "string",
      "MeasureValues": [
         {
            "Name": "string",
            "Type": "string",
            "Value": "string"
         }
      ],
      "MeasureValueType": "string",
      "Time": "string",
      "TimeUnit": "string",
      "Version": number
   },
   "DatabaseName": "string",
   "Records": [
      {
         "Dimensions": [
            {
               "DimensionValueType": "string",
               "Name": "string",
               "Value": "string"
            }
         ],
         "MeasureName": "string",
         "MeasureValue": "string",
         "MeasureValues": [
            {
               "Name": "string",
               "Type": "string",
```

```
                "Value": "string"
            }
        ],
        "MeasureValueType": "string",
        "Time": "string",
        "TimeUnit": "string",
        "Version": number
    }
  ],
  "TableName": "string"
}
```

Parámetros de la solicitud

Para obtener información sobre los parámetros comunes a todas las acciones, consulte Parámetros comunes.

La solicitud acepta los siguientes datos en JSON formato.

CommonAttributes

Un registro que contiene los atributos comunes de medida, dimensión, tiempo y versión que se comparten en todos los registros de la solicitud. Los atributos de medida y dimensión especificados se fusionarán con los atributos de medida y dimensión del objeto de registro cuando los datos se escriban en Timestream. Es posible que las dimensiones no se superpongan o `ValidationException` se generará una. En otras palabras, un registro debe contener dimensiones con nombres únicos.

Tipo: objeto Record

Obligatorio: no

DatabaseName

Nombre de la base de datos de Timestream.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 3. La longitud máxima es de 256 caracteres.

Obligatorio: sí

Records

Matriz de registros que contiene los atributos únicos de medida, dimensión, tiempo y versión de cada punto de datos de la serie temporal.

Tipo: matriz de objetos Record

Miembros de la matriz: número mínimo de 1 artículo. Número máximo de 100 artículos.

Obligatorio: sí

TableName

Nombre de la tabla de Timestream.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 3. La longitud máxima es de 256 caracteres.

Obligatorio: sí

Sintaxis de la respuesta

```
{
   "RecordsIngested": {
      "MagneticStore": number,
      "MemoryStore": number,
      "Total": number
   }
}
```

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta de HTTP 200.

El servicio devuelve los siguientes datos en JSON formato.

RecordsIngested

Información sobre los registros ingeridos por esta solicitud.

Tipo: objeto RecordsIngested

Errores

Para obtener información acerca de los errores comunes a todas las acciones, consulte Errores comunes.

AccessDeniedException

No está autorizado a realizar esta acción.

HTTPCódigo de estado: 400

InternalServerException

Timestream no pudo procesar completamente esta solicitud debido a un error interno del servidor.

HTTPCódigo de estado: 500

InvalidEndpointException

El punto final solicitado no era válido.

HTTPCódigo de estado: 400

RejectedRecordsException

WriteRecords generaría esta excepción en los siguientes casos:

- Registros con datos duplicados en los que hay varios registros con las mismas dimensiones, marcas de tiempo y nombres de medidas, pero:
  - Los valores de las medidas son diferentes
  - La versión no está presente en la solicitud o el valor de la versión en el nuevo registro es igual o inferior al valor existente

  En este caso, si Timestream rechaza los datos, el `ExistingVersion` campo de la `RejectedRecords` respuesta indicará la versión del registro actual. Para forzar una actualización, puedes volver a enviar la solicitud con una versión del registro establecida en un valor superior a. `ExistingVersion`
- Registros con marcas de tiempo que se encuentran fuera del período de retención del almacén de memoria.
- Registros con dimensiones o medidas que superan los límites definidos por Timestream.

Para obtener más información, consulte Cuotas en la Guía para desarrolladores de Amazon Timestream.

HTTPCódigo de estado: 400

ResourceNotFoundException

La operación intentó acceder a un recurso inexistente. Es posible que el recurso no esté especificado correctamente o que su estado no lo estéACTIVE.

HTTPCódigo de estado: 400

ThrottlingException

Un usuario realizó demasiadas solicitudes y estas superaron las cuotas de servicio. La solicitud se ha limitado.

HTTPCódigo de estado: 400

ValidationException

Solicitud no válida o mal formada.

HTTPCódigo de estado: 400

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- Interfaz de la línea de comandos de AWS
- AWS SDKpara. NET
- AWS SDKpara C++
- AWS SDKpara Go v2
- AWS SDKpara Java V2
- AWS SDKpara JavaScript V3
- AWS SDKpara PHP V3
- AWS SDKpara Python
- AWS SDKpara Ruby V3

Consulta de Amazon Timestream

Amazon Timestream Query admite las siguientes acciones:

- CancelQuery
- CreateScheduledQuery
- DeleteScheduledQuery
- DescribeAccountSettings
- DescribeEndpoints
- DescribeScheduledQuery
- ExecuteScheduledQuery
- ListScheduledQueries
- ListTagsForResource
- PrepareQuery
- Query
- TagResource
- UntagResource
- UpdateAccountSettings
- UpdateScheduledQuery

CancelQuery
Servicio: Amazon Timestream Query

Cancela una consulta que se ha emitido. La cancelación solo se proporciona si la consulta no se completó antes de que se emitiera la solicitud de cancelación. Como la cancelación es una operación idempotente, las solicitudes de cancelación posteriores devolverán un`CancellationMessage`, lo que indica que la consulta ya se ha cancelado. Consulte el ejemplo de código para obtener más detalles.

Sintaxis de la solicitud

```
{
   "QueryId": "string"
}
```

Parámetros de la solicitud

Para obtener información sobre los parámetros comunes a todas las acciones, consulte Parámetros comunes.

La solicitud acepta los siguientes datos en JSON formato.

QueryId

El identificador de la consulta que se debe cancelar. `QueryID`se devuelve como parte del resultado de la consulta.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 64.

Patrón: `[a-zA-Z0-9]+`

Obligatorio: sí

Sintaxis de la respuesta

```
{
   "CancellationMessage": "string"
}
```

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta de HTTP 200.

El servicio devuelve los siguientes datos en JSON formato.

CancellationMessage

A `CancellationMessage` se devuelve cuando ya se `QueryId` ha emitido una `CancelQuery` solicitud para la consulta especificada por.

Tipo: cadena

Errores

Para obtener información acerca de los errores comunes a todas las acciones, consulte Errores comunes.

AccessDeniedException

No está autorizado a realizar esta acción.

HTTPCódigo de estado: 400

InternalServerException

El servicio no pudo procesar completamente esta solicitud debido a un error interno del servidor.

HTTPCódigo de estado: 400

InvalidEndpointException

El punto final solicitado no era válido.

HTTPCódigo de estado: 400

ThrottlingException

La solicitud fue denegada debido a una limitación de la solicitud.

HTTPCódigo de estado: 400

ValidationException

Solicitud no válida o mal formada.

HTTPCódigo de estado: 400

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- Interfaz de la línea de comandos de AWS
- AWS SDKpara. NET
- AWS SDKpara C++
- AWS SDKpara Go v2
- AWS SDKpara Java V2
- AWS SDKpara JavaScript V3
- AWS SDKpara PHP V3
- AWS SDKpara Python
- AWS SDKpara Ruby V3

CreateScheduledQuery

Servicio: Amazon Timestream Query

Crear una consulta programada que se ejecutará en su nombre según la programación configurada. Timestream asume la función de ejecución proporcionada como parte del parámetro `ScheduledQueryExecutionRoleArn` para ejecutar la consulta. Puede usar el parámetro `NotificationConfiguration` para configurar la notificación de las operaciones de consulta programadas.

Sintaxis de la solicitud

```
{
   "ClientToken": "string",
   "ErrorReportConfiguration": {
      "S3Configuration": {
         "BucketName": "string",
         "EncryptionOption": "string",
         "ObjectKeyPrefix": "string"
      }
   },
   "KmsKeyId": "string",
   "Name": "string",
   "NotificationConfiguration": {
      "SnsConfiguration": {
         "TopicArn": "string"
      }
   },
   "QueryString": "string",
   "ScheduleConfiguration": {
      "ScheduleExpression": "string"
   },
   "ScheduledQueryExecutionRoleArn": "string",
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ],
   "TargetConfiguration": {
      "TimestreamConfiguration": {
         "DatabaseName": "string",
         "DimensionMappings": [
            {
```

```
                "DimensionValueType": "string",
                "Name": "string"
            }
         ],
         "MeasureNameColumn": "string",
         "MixedMeasureMappings": [
            {
               "MeasureName": "string",
               "MeasureValueType": "string",
               "MultiMeasureAttributeMappings": [
                  {
                     "MeasureValueType": "string",
                     "SourceColumn": "string",
                     "TargetMultiMeasureAttributeName": "string"
                  }
               ],
               "SourceColumn": "string",
               "TargetMeasureName": "string"
            }
         ],
         "MultiMeasureMappings": {
            "MultiMeasureAttributeMappings": [
               {
                  "MeasureValueType": "string",
                  "SourceColumn": "string",
                  "TargetMultiMeasureAttributeName": "string"
               }
            ],
            "TargetMultiMeasureName": "string"
         },
         "TableName": "string",
         "TimeColumn": "string"
      }
   }
}
```

Parámetros de la solicitud

Para obtener información sobre los parámetros comunes a todas las acciones, consulte Parámetros comunes.

La solicitud acepta los siguientes datos en JSON formato.

ClientToken

El uso de a ClientToken hace que la llamada sea CreateScheduledQuery idempotente, en otras palabras, si se hace la misma solicitud repetidamente, se obtendrá el mismo resultado. Hacer varias CreateScheduledQuery solicitudes idénticas tiene el mismo efecto que hacer una sola solicitud.

- Si CreateScheduledQuery se llama sin un`ClientToken`, la consulta SDK genera un `ClientToken` en tu nombre.
- Después de 8 horas, cualquier solicitud con el mismo `ClientToken` es tratada como una nueva solicitud.

Tipo: cadena

Restricciones de longitud: longitud mínima de 32. Longitud máxima de 128.

Obligatorio: no

ErrorReportConfiguration

Configuración para el informe de errores. Se generarán informes de errores cuando se encuentre un problema al escribir los resultados de la consulta.

Tipo: objeto ErrorReportConfiguration

Obligatorio: sí

KmsKeyId

La KMS clave de Amazon utilizada para cifrar el recurso de consulta programada, en reposo. Si no se especifica la KMS clave de Amazon, el recurso de consulta programado se cifrará con una clave de Amazon KMS propiedad de Timestream. Para especificar una KMS clave, usa el ID de clave, la claveARN, el nombre o el alias. ARN Cuando utilice un nombre del alias, anteponga el nombre con alias/

Si `SSE_KMS` se ErrorReportConfiguration utiliza como tipo de cifrado, KmsKeyId se utiliza lo mismo para cifrar el informe de errores en reposo.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Obligatorio: no

Name

Nombre de la consulta programada.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 64.

Patrón: `[a-zA-Z0-9|!\-_*'\(\)]([a-zA-Z0-9]|[!\-_*'\(\)\/.])+`

Obligatorio: sí

NotificationConfiguration

Configuración de notificaciones para la consulta programada. Timestream envía una notificación cuando finaliza la ejecución de una consulta, cuando se actualiza el estado o cuando se elimina.

Tipo: objeto NotificationConfiguration

Obligatorio: sí

QueryString

La cadena de consulta que se ejecutará. Los nombres de los parámetros se pueden especificar en la cadena de consulta `@` seguida de un identificador. El parámetro denominado `@scheduled_runtime` está reservado y se puede usar en la consulta para obtener la hora a la que está programada la ejecución de la consulta.

La marca de tiempo calculada en función del ScheduleConfiguration parámetro será el valor del `@scheduled_runtime` parámetro para cada consulta que se ejecute. Por ejemplo, considere una instancia de una consulta programada que se ejecuta el 01-12-2021 00:00:00. Para esta instancia, el parámetro `@scheduled_runtime` se inicializa en la marca de tiempo 01-12-2021 00:00:00 al invocar la consulta.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 262144 caracteres.

Obligatorio: sí

ScheduleConfiguration

La configuración del cronograma de la consulta.

Tipo: objeto ScheduleConfiguration

Obligatorio: sí

ScheduledQueryExecutionRoleArn

Es ARN para la IAM función que asumirá Timestream al ejecutar la consulta programada.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Obligatorio: sí

Tags

Lista de pares clave-valor para etiquetar la consulta programada.

Tipo: matriz de objetos Tag

Miembros de la matriz: número mínimo de 0 artículos. La cantidad máxima es de 200 artículos.

Obligatorio: no

TargetConfiguration

Configuración utilizada para escribir el resultado de una consulta.

Tipo: objeto TargetConfiguration

Obligatorio: no

Sintaxis de la respuesta

```
{
   "Arn": "string"
}
```

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta de HTTP 200.

El servicio devuelve los siguientes datos en JSON formato.

Arn

ARNpara la consulta programada creada.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Errores

Para obtener información acerca de los errores comunes a todas las acciones, consulte Errores comunes.

AccessDeniedException

No está autorizado a realizar esta acción.

HTTPCódigo de estado: 400

ConflictException

No se han podido sondear los resultados de una consulta cancelada.

HTTPCódigo de estado: 400

InternalServerException

El servicio no pudo procesar completamente esta solicitud debido a un error interno del servidor.

HTTPCódigo de estado: 400

InvalidEndpointException

El punto final solicitado no era válido.

HTTPCódigo de estado: 400

ServiceQuotaExceededException

Ha superado la cuota de servicio.

HTTPCódigo de estado: 400

ThrottlingException

La solicitud fue denegada debido a una limitación de la solicitud.

HTTPCódigo de estado: 400

ValidationException

Solicitud no válida o con formato incorrecto.

HTTPCódigo de estado: 400

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- Interfaz de la línea de comandos de AWS
- AWS SDKpara. NET
- AWS SDKpara C++
- AWS SDKpara Go v2
- AWS SDKpara Java V2
- AWS SDKpara JavaScript V3
- AWS SDKpara PHP V3
- AWS SDKpara Python
- AWS SDKpara Ruby V3

DeleteScheduledQuery
Servicio: Amazon Timestream Query

Elimina una consulta programada determinada. Se trata de una operación irreversible.

Sintaxis de la solicitud

```
{
   "ScheduledQueryArn": "string"
}
```

Parámetros de la solicitud

Para obtener información sobre los parámetros comunes a todas las acciones, consulte Parámetros comunes.

La solicitud acepta los siguientes datos en JSON formato.

ScheduledQueryArn

El ARN de la consulta programada.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Obligatorio: sí

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta de HTTP 200 puntos con HTTP el cuerpo vacío.

Errores

Para obtener información acerca de los errores comunes a todas las acciones, consulte Errores comunes.

AccessDeniedException

No está autorizado a realizar esta acción.

HTTPCódigo de estado: 400

InternalServerException

El servicio no pudo procesar completamente esta solicitud debido a un error interno del servidor.

HTTPCódigo de estado: 400

InvalidEndpointException

El punto final solicitado no era válido.

HTTPCódigo de estado: 400

ResourceNotFoundException

No se ha encontrado el recurso solicitado.

HTTPCódigo de estado: 400

ThrottlingException

La solicitud fue denegada debido a una limitación de la solicitud.

HTTPCódigo de estado: 400

ValidationException

Solicitud no válida o mal formada.

HTTPCódigo de estado: 400

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- Interfaz de la línea de comandos de AWS
- AWS SDKpara. NET
- AWS SDKpara C++
- AWS SDKpara Go v2
- AWS SDKpara Java V2
- AWS SDKpara JavaScript V3

- AWS SDKpara PHP V3
- AWS SDKpara Python
- AWS SDKpara Ruby V3

DescribeAccountSettings
Servicio: Amazon Timestream Query

Describe la configuración de su cuenta, que incluye el modelo de precios de las consultas y el máximo configurado que TCUs el servicio puede utilizar para la carga de trabajo de las consultas.

Solo se le cobrará por la duración de las unidades de cómputo utilizadas para sus cargas de trabajo.

Sintaxis de la respuesta

```
{
   "MaxQueryTCU": number,
   "QueryPricingModel": "string"
}
```

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta de HTTP 200.

El servicio devuelve los siguientes datos en JSON formato.

MaxQueryTCU

El número máximo de unidades de cálculo de Timestream (TCUs) que el servicio utilizará en cualquier momento para atender sus consultas.

Tipo: entero

QueryPricingModel

El modelo de precios para las consultas de su cuenta.

Tipo: cadena

Valores válidos: `BYTES_SCANNED | COMPUTE_UNITS`

Errores

Para obtener información acerca de los errores comunes a todas las acciones, consulte Errores comunes.

AccessDeniedException

No está autorizado a realizar esta acción.

HTTPCódigo de estado: 400

InternalServerException

El servicio no pudo procesar completamente esta solicitud debido a un error interno del servidor.

HTTPCódigo de estado: 400

InvalidEndpointException

El punto final solicitado no era válido.

HTTPCódigo de estado: 400

ThrottlingException

La solicitud fue denegada debido a una limitación de la solicitud.

HTTPCódigo de estado: 400

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- Interfaz de la línea de comandos de AWS
- AWS SDKpara. NET
- AWS SDKpara C++
- AWS SDKpara Go v2
- AWS SDKpara Java V2
- AWS SDKpara JavaScript V3
- AWS SDKpara PHP V3
- AWS SDKpara Python
- AWS SDKpara Ruby V3

DescribeEndpoints
Servicio: Amazon Timestream Query

DescribeEndpoints devuelve una lista de puntos finales disponibles para realizar llamadas de Timestream. API APIEstá disponible a través de Write y Query.

Dado que los Timestream SDKs están diseñados para funcionar de forma transparente con la arquitectura del servicio, incluida la administración y el mapeo de los puntos finales del servicio, no se recomienda utilizarlos a menos que: API

- Está utilizando VPCendpoints () con Timestream AWS PrivateLink
- Su aplicación utiliza un lenguaje de programación que aún no es compatible SDK
- Necesita un mejor control sobre la implementación del lado del cliente

Para obtener información detallada sobre cómo y cuándo usar e implementar DescribeEndpoints, consulte The Endpoint Discovery Pattern.

Sintaxis de la respuesta

```
{
   "Endpoints": [
      {
         "Address": "string",
         "CachePeriodInMinutes": number
      }
   ]
}
```

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta de HTTP 200.

El servicio devuelve los siguientes datos en JSON formato.

Endpoints

Se devuelve un `Endpoints` objeto cuando se realiza una `DescribeEndpoints` solicitud.

Tipo: matriz de objetos Endpoint

Errores

Para obtener información acerca de los errores comunes a todas las acciones, consulte Errores comunes.

InternalServerException

El servicio no pudo procesar completamente esta solicitud debido a un error interno del servidor.

HTTPCódigo de estado: 400

ThrottlingException

La solicitud fue denegada debido a una limitación de la solicitud.

HTTPCódigo de estado: 400

ValidationException

Solicitud no válida o mal formada.

HTTPCódigo de estado: 400

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- Interfaz de la línea de comandos de AWS
- AWS SDKpara. NET
- AWS SDKpara C++
- AWS SDKpara Go v2
- AWS SDKpara Java V2
- AWS SDKpara JavaScript V3
- AWS SDKpara PHP V3
- AWS SDKpara Python
- AWS SDKpara Ruby V3

DescribeScheduledQuery
Servicio: Amazon Timestream Query

Proporciona información detallada sobre una consulta programada.

Sintaxis de la solicitud

```
{
   "ScheduledQueryArn": "string"
}
```

Parámetros de la solicitud

Para obtener información sobre los parámetros comunes a todas las acciones, consulte Parámetros comunes.

La solicitud acepta los siguientes datos en JSON formato.

ScheduledQueryArn

El ARN de la consulta programada.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Obligatorio: sí

Sintaxis de la respuesta

```
{
   "ScheduledQuery": {
      "Arn": "string",
      "CreationTime": number,
      "ErrorReportConfiguration": {
         "S3Configuration": {
            "BucketName": "string",
            "EncryptionOption": "string",
            "ObjectKeyPrefix": "string"
         }
      },
      "KmsKeyId": "string",
      "LastRunSummary": {
```

```
         "ErrorReportLocation": {
            "S3ReportLocation": {
               "BucketName": "string",
               "ObjectKey": "string"
            }
         },
         "ExecutionStats": {
            "BytesMetered": number,
            "CumulativeBytesScanned": number,
            "DataWrites": number,
            "ExecutionTimeInMillis": number,
            "QueryResultRows": number,
            "RecordsIngested": number
         },
         "FailureReason": "string",
         "InvocationTime": number,
         "QueryInsightsResponse": {
            "OutputBytes": number,
            "OutputRows": number,
            "QuerySpatialCoverage": {
               "Max": {
                  "PartitionKey": [ "string" ],
                  "TableArn": "string",
                  "Value": number
               }
            },
            "QueryTableCount": number,
            "QueryTemporalRange": {
               "Max": {
                  "TableArn": "string",
                  "Value": number
               }
            }
         },
         "RunStatus": "string",
         "TriggerTime": number
      },
      "Name": "string",
      "NextInvocationTime": number,
      "NotificationConfiguration": {
         "SnsConfiguration": {
            "TopicArn": "string"
         }
      },
```

```
      "PreviousInvocationTime": number,
      "QueryString": "string",
      "RecentlyFailedRuns": [
         {
            "ErrorReportLocation": {
               "S3ReportLocation": {
                  "BucketName": "string",
                  "ObjectKey": "string"
               }
            },
            "ExecutionStats": {
               "BytesMetered": number,
               "CumulativeBytesScanned": number,
               "DataWrites": number,
               "ExecutionTimeInMillis": number,
               "QueryResultRows": number,
               "RecordsIngested": number
            },
            "FailureReason": "string",
            "InvocationTime": number,
            "QueryInsightsResponse": {
               "OutputBytes": number,
               "OutputRows": number,
               "QuerySpatialCoverage": {
                  "Max": {
                     "PartitionKey": [ "string" ],
                     "TableArn": "string",
                     "Value": number
                  }
               },
               "QueryTableCount": number,
               "QueryTemporalRange": {
                  "Max": {
                     "TableArn": "string",
                     "Value": number
                  }
               }
            },
            "RunStatus": "string",
            "TriggerTime": number
         }
      ],
      "ScheduleConfiguration": {
         "ScheduleExpression": "string"
```

```
      },
      "ScheduledQueryExecutionRoleArn": "string",
      "State": "string",
      "TargetConfiguration": {
         "TimestreamConfiguration": {
            "DatabaseName": "string",
            "DimensionMappings": [
               {
                  "DimensionValueType": "string",
                  "Name": "string"
               }
            ],
            "MeasureNameColumn": "string",
            "MixedMeasureMappings": [
               {
                  "MeasureName": "string",
                  "MeasureValueType": "string",
                  "MultiMeasureAttributeMappings": [
                     {
                        "MeasureValueType": "string",
                        "SourceColumn": "string",
                        "TargetMultiMeasureAttributeName": "string"
                     }
                  ],
                  "SourceColumn": "string",
                  "TargetMeasureName": "string"
               }
            ],
            "MultiMeasureMappings": {
               "MultiMeasureAttributeMappings": [
                  {
                     "MeasureValueType": "string",
                     "SourceColumn": "string",
                     "TargetMultiMeasureAttributeName": "string"
                  }
               ],
               "TargetMultiMeasureName": "string"
            },
            "TableName": "string",
            "TimeColumn": "string"
         }
      }
   }
```

```
}
```

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta de HTTP 200.

El servicio devuelve los siguientes datos en JSON formato.

ScheduledQuery

La consulta programada.

Tipo: objeto ScheduledQueryDescription

Errores

Para obtener información acerca de los errores comunes a todas las acciones, consulte Errores comunes.

AccessDeniedException

No está autorizado a realizar esta acción.

HTTPCódigo de estado: 400

InternalServerException

El servicio no pudo procesar completamente esta solicitud debido a un error interno del servidor.

HTTPCódigo de estado: 400

InvalidEndpointException

El punto final solicitado no era válido.

HTTPCódigo de estado: 400

ResourceNotFoundException

No se ha encontrado el recurso solicitado.

HTTPCódigo de estado: 400

ThrottlingException

La solicitud fue denegada debido a una limitación de la solicitud.

HTTPCódigo de estado: 400

ValidationException

Solicitud no válida o mal formada.

HTTPCódigo de estado: 400

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- Interfaz de la línea de comandos de AWS
- AWS SDKpara. NET
- AWS SDKpara C++
- AWS SDKpara Go v2
- AWS SDKpara Java V2
- AWS SDKpara JavaScript V3
- AWS SDKpara PHP V3
- AWS SDKpara Python
- AWS SDKpara Ruby V3

ExecuteScheduledQuery
Servicio: Amazon Timestream Query

Puede usarlo API para ejecutar una consulta programada de forma manual.

Si lo has activado`QueryInsights`, API también se muestran estadísticas y métricas relacionadas con la consulta que ejecutaste como parte de una SNS notificación de Amazon. `QueryInsights`sayuda a ajustar el rendimiento de la consulta. Para obtener más información`QueryInsights`, consulte Uso de la información de consultas para optimizar las consultas en Amazon Timestream.

Sintaxis de la solicitud

```
{
   "ClientToken": "string",
   "InvocationTime": number,
   "QueryInsights": {
      "Mode": "string"
   },
   "ScheduledQueryArn": "string"
}
```

Parámetros de la solicitud

Para obtener información sobre los parámetros comunes a todas las acciones, consulte Parámetros comunes.

La solicitud acepta los siguientes datos en JSON formato.

ClientToken

No se utiliza.

Tipo: cadena

Restricciones de longitud: longitud mínima de 32. Longitud máxima de 128.

Obligatorio: no

InvocationTime

La marca de tiempo en. UTC La consulta se ejecutará como si se hubiera invocado en esta marca de tiempo.

Tipo: marca temporal

Obligatorio: sí

QueryInsights

Encapsula la configuración para habilitarla. `QueryInsights`

La activación `QueryInsights` devuelve estadísticas y métricas como parte de la SNS notificación de Amazon para la consulta que ejecutaste. Puede utilizarlos `QueryInsights` para ajustar el rendimiento y el coste de sus consultas.

Tipo: objeto ScheduledQueryInsights

Obligatorio: no

ScheduledQueryArn

ARNde la consulta programada.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Obligatorio: sí

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta de HTTP 200 puntos con HTTP el cuerpo vacío.

Errores

Para obtener información acerca de los errores comunes a todas las acciones, consulte Errores comunes.

AccessDeniedException

No está autorizado a realizar esta acción.

HTTPCódigo de estado: 400

InternalServerException

El servicio no pudo procesar completamente esta solicitud debido a un error interno del servidor.

HTTPCódigo de estado: 400

InvalidEndpointException

El punto final solicitado no era válido.

HTTPCódigo de estado: 400

ResourceNotFoundException

No se ha encontrado el recurso solicitado.

HTTPCódigo de estado: 400

ThrottlingException

La solicitud fue denegada debido a una limitación de la solicitud.

HTTPCódigo de estado: 400

ValidationException

Solicitud no válida o mal formada.

HTTPCódigo de estado: 400

Ejemplos

Mensaje de notificación de consulta programada para el CONTROL modo ENABLED WITH _ RATE _ _

El siguiente ejemplo muestra un mensaje de notificación de consulta programado correctamente para el `ENABLED_WITH_RATE_CONTROL` modo del `QueryInsights` parámetro.

```
"SuccessNotificationMessage": {
    "type": "MANUAL_TRIGGER_SUCCESS",
    "arn": "arn:aws:timestream:<Region>:<Account>:scheduled-query/sq-test-49c6ed55-
c2e7-4cc2-9956-4a0ecea13420-80e05b035236a4c3",
    "scheduledQueryRunSummary": {
        "invocationEpochSecond": 1723710546,
        "triggerTimeMillis": 1723710547490,
        "runStatus": "MANUAL_TRIGGER_SUCCESS",
        "executionStats": {
            "executionTimeInMillis": 17343,
            "dataWrites": 1024,
```

```
            "bytesMetered": 0,
            "cumulativeBytesScanned": 600,
            "recordsIngested": 1,
            "queryResultRows": 1
        },
        "queryInsightsResponse": {
            "querySpatialCoverage": {
                "max": {
                    "value": 1.0,
                    "tableArn": "arn:aws:timestream:<Region>:<Account>:database/BaseDb/
table/BaseTable",
                    "partitionKey": [
                        "measure_name"
                    ]
                }
            },
            "queryTemporalRange": {
                "max": {
                    "value": 2399999999999,
                    "tableArn": "arn:aws:timestream:<Region>:<Account>:database/BaseDb/
table/BaseTable"
                }
            },
            "queryTableCount": 1,
            "outputRows": 1,
            "outputBytes": 59
        }
    }
}
```

Mensaje de notificación de consulta programado para el DISABLED modo

En el siguiente ejemplo, se muestra un mensaje de notificación de consulta programado correctamente para el `DISABLED` modo del `QueryInsights` parámetro.

```
"SuccessNotificationMessage": {
    "type": "MANUAL_TRIGGER_SUCCESS",
    "arn": "arn:aws:timestream:<Region>:<Account>:scheduled-query/sq-test-
fa109d9e-6528-4a0d-ac40-482fa05e657f-140faaeecdc5b2a7",
    "scheduledQueryRunSummary": {
        "invocationEpochSecond": 1723711401,
        "triggerTimeMillis": 1723711402144,
        "runStatus": "MANUAL_TRIGGER_SUCCESS",
```

```
        "executionStats": {
            "executionTimeInMillis": 17992,
            "dataWrites": 1024,
            "bytesMetered": 0,
            "cumulativeBytesScanned": 600,
            "recordsIngested": 1,
            "queryResultRows": 1
        }
    }
}
```

Mensaje de notificación de error para el CONTROL modo ENABLED WITH _ RATE _ _

El siguiente ejemplo muestra un mensaje de notificación de consulta programada fallido para el `ENABLED_WITH_RATE_CONTROL` modo del `QueryInsights` parámetro.

```
"FailureNotificationMessage": {
    "type": "MANUAL_TRIGGER_FAILURE",
    "arn": "arn:aws:timestream:<Region>:<Account>:scheduled-query/sq-test-
b261670d-790c-4116-9db5-0798071b18b1-b7e27a1d79be226d",
    "scheduledQueryRunSummary": {
        "invocationEpochSecond": 1727915513,
        "triggerTimeMillis": 1727915513894,
        "runStatus": "MANUAL_TRIGGER_FAILURE",
        "executionStats": {
            "executionTimeInMillis": 10777,
            "dataWrites": 0,
            "bytesMetered": 0,
            "cumulativeBytesScanned": 0,
            "recordsIngested": 0,
            "queryResultRows": 4
        },
        "errorReportLocation": {
            "s3ReportLocation": {
                "bucketName": "my-amzn-s3-demo-bucket",
                "objectKey": "4my-organization-f7a3c5d065a1a95e/1727915513/
MANUAL/1727915513894/5e14b3df-b147-49f4-9331-784f749b68ae"
            }
        },
        "failureReason": "Schedule encountered some errors and is incomplete. Please
 take a look at error report for further details"
    }
}
```

Mensaje de notificación de error para el DISABLED modo

El siguiente ejemplo muestra un mensaje de notificación de consulta programada fallido para el `DISABLED` modo del `QueryInsights` parámetro.

```
"FailureNotificationMessage": {
    "type": "MANUAL_TRIGGER_FAILURE",
    "arn": "arn:aws:timestream:<Region>:<Account>:scheduled-query/sq-test-
b261670d-790c-4116-9db5-0798071b18b1-b7e27a1d79be226d",
    "scheduledQueryRunSummary": {
        "invocationEpochSecond": 1727915194,
        "triggerTimeMillis": 1727915195119,
        "runStatus": "MANUAL_TRIGGER_FAILURE",
        "executionStats": {
            "executionTimeInMillis": 10777,
            "dataWrites": 0,
            "bytesMetered": 0,
            "cumulativeBytesScanned": 0,
            "recordsIngested": 0,
            "queryResultRows": 4
        },
        "errorReportLocation": {
            "s3ReportLocation": {
                "bucketName": "my-amzn-s3-demo-bucket",
                "objectKey": "4my-organization-b7e27a1d79be226d/1727915194/
MANUAL/1727915195119/08dea9f5-9a0a-4e63-a5f7-ded23247bb98"
            }
        },
        "failureReason": "Schedule encountered some errors and is incomplete. Please
 take a look at error report for further details"
    }
}
```

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- Interfaz de la línea de comandos de AWS
- AWS SDKpara. NET
- AWS SDKpara C++
- AWS SDKpara Go v2

- AWS SDKpara Java V2
- AWS SDKpara JavaScript V3
- AWS SDKpara PHP V3
- AWS SDKpara Python
- AWS SDKpara Ruby V3

ListScheduledQueries
Servicio: Amazon Timestream Query

Obtiene una lista de todas las consultas programadas en la cuenta de Amazon y la región de la persona que llama. `ListScheduledQueries`al final es coherente.

Sintaxis de la solicitud

```
{
   "MaxResults": number,
   "NextToken": "string"
}
```

Parámetros de la solicitud

Para obtener información sobre los parámetros comunes a todas las acciones, consulte Parámetros comunes.

La solicitud acepta los siguientes datos en JSON formato.

MaxResults

El número máximo de elementos que se devolverán en la salida. Si el número total de artículos disponibles es superior al valor especificado, `NextToken` se proporciona a en la salida. Para reanudar la paginación, proporcione el `NextToken` valor como argumento para la siguiente llamada a`ListScheduledQueriesRequest`.

Tipo: entero

Rango válido: valor mínimo de 1. Valor máximo de 1000.

Obligatorio: no

NextToken

Un token de paginación para reanudar la paginación.

Tipo: cadena

Requerido: no

Sintaxis de la respuesta

```
{
   "NextToken": "string",
   "ScheduledQueries": [
      {
         "Arn": "string",
         "CreationTime": number,
         "ErrorReportConfiguration": {
            "S3Configuration": {
               "BucketName": "string",
               "EncryptionOption": "string",
               "ObjectKeyPrefix": "string"
            }
         },
         "LastRunStatus": "string",
         "Name": "string",
         "NextInvocationTime": number,
         "PreviousInvocationTime": number,
         "State": "string",
         "TargetDestination": {
            "TimestreamDestination": {
               "DatabaseName": "string",
               "TableName": "string"
            }
         }
      }
   ]
}
```

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta de HTTP 200.

El servicio devuelve los siguientes datos en JSON formato.

NextToken

Un token destinado a especificar dónde iniciar la paginación. Se trata NextToken de una respuesta previamente truncada.

Tipo: cadena

ScheduledQueries

Una lista de consultas programadas.

Tipo: matriz de objetos ScheduledQuery

Errores

Para obtener información acerca de los errores comunes a todas las acciones, consulte Errores comunes.

AccessDeniedException

No está autorizado a realizar esta acción.

HTTPCódigo de estado: 400

InternalServerException

El servicio no pudo procesar completamente esta solicitud debido a un error interno del servidor.

HTTPCódigo de estado: 400

InvalidEndpointException

El punto final solicitado no era válido.

HTTPCódigo de estado: 400

ThrottlingException

La solicitud fue denegada debido a una limitación de la solicitud.

HTTPCódigo de estado: 400

ValidationException

Solicitud no válida o con formato incorrecto.

HTTPCódigo de estado: 400

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- Interfaz de la línea de comandos de AWS
- AWS SDKpara. NET
- AWS SDKpara C++
- AWS SDKpara Go v2
- AWS SDKpara Java V2
- AWS SDKpara JavaScript V3
- AWS SDKpara PHP V3
- AWS SDKpara Python
- AWS SDKpara Ruby V3

ListTagsForResource
Servicio: Amazon Timestream Query

Listar todas las etiquetas de un recurso de consulta de Timestream.

Sintaxis de la solicitud

```
{
   "MaxResults": number,
   "NextToken": "string",
   "ResourceARN": "string"
}
```

Parámetros de la solicitud

Para obtener información sobre los parámetros comunes a todas las acciones, consulte Parámetros comunes.

La solicitud acepta los siguientes datos en JSON formato.

MaxResults

El número máximo de etiquetas que se van a devolver.

Tipo: entero

Rango válido: valor mínimo de 1. Valor máximo de 200.

Obligatorio: no

NextToken

Un token de paginación para reanudar la paginación.

Tipo: cadena

Requerido: no

ResourceARN

El recurso Timestream con las etiquetas que se van a enumerar. Este valor es un nombre de recurso de Amazon (ARN).

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Obligatorio: sí

Sintaxis de la respuesta

```
{
   "NextToken": "string",
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ]
}
```

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta de HTTP 200.

El servicio devuelve los siguientes datos en JSON formato.

NextToken

Un token de paginación para reanudar la paginación con una llamada posterior a. `ListTagsForResourceResponse`

Tipo: cadena

Tags

Las etiquetas asociadas actualmente al recurso Timestream.

Tipo: matriz de objetos Tag

Miembros de la matriz: número mínimo de 0 artículos. La cantidad máxima es de 200 artículos.

Errores

Para obtener información acerca de los errores comunes a todas las acciones, consulte Errores comunes.

InvalidEndpointException

El punto final solicitado no era válido.

HTTPCódigo de estado: 400

ResourceNotFoundException

No se ha encontrado el recurso solicitado.

HTTPCódigo de estado: 400

ThrottlingException

La solicitud fue denegada debido a una limitación de la solicitud.

HTTPCódigo de estado: 400

ValidationException

Solicitud no válida o mal formada.

HTTPCódigo de estado: 400

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- Interfaz de la línea de comandos de AWS
- AWS SDKpara. NET
- AWS SDKpara C++
- AWS SDKpara Go v2
- AWS SDKpara Java V2
- AWS SDKpara JavaScript V3
- AWS SDKpara PHP V3
- AWS SDKpara Python
- AWS SDKpara Ruby V3

PrepareQuery
Servicio: Amazon Timestream Query

Operación sincrónica que permite enviar una consulta con parámetros para que Timestream los almacene para su posterior ejecución. Timestream solo admite el uso de esta operación si está establecido en. `ValidateOnly` `true`

Sintaxis de la solicitud

```
{
   "QueryString": "string",
   "ValidateOnly": boolean
}
```

Parámetros de la solicitud

Para obtener información sobre los parámetros comunes a todas las acciones, consulte Parámetros comunes.

La solicitud acepta los siguientes datos en JSON formato.

QueryString

La cadena de consulta de Timestream que desea utilizar como sentencia preparada. Los nombres de los parámetros se pueden especificar en la cadena de consulta @ seguida de un identificador.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 262144 caracteres.

Obligatorio: sí

ValidateOnly

Al establecer este valor en`true`, Timestream solo validará que la cadena de consulta sea una consulta Timestream válida y no almacenará la consulta preparada para su uso posterior.

Tipo: Booleano

Obligatorio: no

Sintaxis de la respuesta

```
{
   "Columns": [
      {
         "Aliased": boolean,
         "DatabaseName": "string",
         "Name": "string",
         "TableName": "string",
         "Type": {
            "ArrayColumnInfo": {
               "Name": "string",
               "Type": "Type"
            },
            "RowColumnInfo": [
               {
                  "Name": "string",
                  "Type": "Type"
               }
            ],
            "ScalarType": "string",
            "TimeSeriesMeasureValueColumnInfo": {
               "Name": "string",
               "Type": "Type"
            }
         }
      }
   ],
   "Parameters": [
      {
         "Name": "string",
         "Type": {
            "ArrayColumnInfo": {
               "Name": "string",
               "Type": "Type"
            },
            "RowColumnInfo": [
               {
                  "Name": "string",
                  "Type": "Type"
               }
            ],
            "ScalarType": "string",
            "TimeSeriesMeasureValueColumnInfo": {
```

```
                "Name": "string",
                "Type": "Type"
            }
        }
    }
  ],
  "QueryString": "string"
}
```

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta de 200. HTTP

El servicio devuelve los siguientes datos en JSON formato.

Columns

Una lista de columnas de SELECT cláusulas de la cadena de consulta enviada.

Tipo: matriz de objetos SelectColumn

Parameters

Una lista de los parámetros utilizados en la cadena de consulta enviada.

Tipo: matriz de objetos ParameterMapping

QueryString

La cadena de consulta que desea preparar.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 262144 caracteres.

Errores

Para obtener información acerca de los errores comunes a todas las acciones, consulte Errores comunes.

AccessDeniedException

No está autorizado a realizar esta acción.

HTTPCódigo de estado: 400

InternalServerException

El servicio no pudo procesar completamente esta solicitud debido a un error interno del servidor.

HTTPCódigo de estado: 400

InvalidEndpointException

El punto final solicitado no era válido.

HTTPCódigo de estado: 400

ThrottlingException

La solicitud fue denegada debido a una limitación de la solicitud.

HTTPCódigo de estado: 400

ValidationException

Solicitud no válida o con formato incorrecto.

HTTPCódigo de estado: 400

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- Interfaz de la línea de comandos de AWS
- AWS SDKpara. NET
- AWS SDKpara C++
- AWS SDKpara Go v2
- AWS SDKpara Java V2
- AWS SDKpara JavaScript V3
- AWS SDKpara PHP V3
- AWS SDKpara Python
- AWS SDKpara Ruby V3

Query
Servicio: Amazon Timestream Query

`Query`es una operación sincrónica que le permite ejecutar una consulta con sus datos de Amazon Timestream.

Si la habilitó`QueryInsights`, API también devuelve información y métricas relacionadas con la consulta que ejecutó. `QueryInsights`sayuda a ajustar el rendimiento de la consulta. Para obtener más información`QueryInsights`, consulte Uso de la información de consultas para optimizar las consultas en Amazon Timestream.

> **Note**
>
> El número máximo de `Query` API solicitudes que puede realizar con la `QueryInsights` opción habilitada es de 1 consulta por segundo ()QPS. Si superas esta tasa de consultas, es posible que se produzcan limitaciones.

`Query`se agotará el tiempo de espera después de 60 segundos. Debe actualizar el tiempo de espera predeterminado en el SDK para que admita un tiempo de espera de 60 segundos. Consulte el ejemplo de código para obtener más información.

La solicitud de consulta fallará en los siguientes casos:

- Si envías una `Query` solicitud con el mismo token de cliente fuera del período de idempotencia de 5 minutos.
- Si envías una `Query` solicitud con el mismo token de cliente, pero cambias otros parámetros, dentro del plazo de idempotencia de 5 minutos.
- Si el tamaño de la fila (incluidos los metadatos de la consulta) supera 1 MB, la consulta fallará y mostrará el siguiente mensaje de error:

  ```
  Query aborted as max page response size has been exceeded by the output result row
  ```

- Si el IAM principal del iniciador de la consulta y el lector de resultados no son iguales o el iniciador de la consulta y el lector de resultados no tienen la misma cadena de consulta en las solicitudes de consulta, la consulta fallará y se generará un `Invalid pagination token` error.

Sintaxis de la solicitud

```
{
   "ClientToken": "string",
   "MaxRows": number,
   "NextToken": "string",
   "QueryInsights": {
      "Mode": "string"
   },
   "QueryString": "string"
}
```

Parámetros de la solicitud

Para obtener información sobre los parámetros comunes a todas las acciones, consulte Parámetros comunes.

La solicitud acepta los siguientes datos en JSON formato.

ClientToken

Cadena única, que distingue entre mayúsculas y minúsculas, de hasta 64 ASCII caracteres que se especifica al realizar una `Query` solicitud. Si se proporciona a, la llamada se `ClientToken` vuelve `Query` idempotente. Esto significa que ejecutar la misma consulta repetidamente producirá el mismo resultado. En otras palabras, realizar varias `Query` solicitudes idénticas tiene el mismo efecto que realizar una sola solicitud. Cuando lo `ClientToken` utilices en una consulta, ten en cuenta lo siguiente:

- Si API se crea una instancia de la consulta sin un`ClientToken`, la consulta SDK genera un `ClientToken` en tu nombre.
- Si la `Query` invocación solo contiene la, `ClientToken` pero no la incluye`NextToken`, se supone que la invocación de `Query` es una nueva ejecución de consulta.
- Si la invocación contiene`NextToken`, se supone que esa invocación en particular es una invocación posterior de una llamada anterior a la consulta y se devuelve un API conjunto de resultados.
- Transcurridas 4 horas, cualquier solicitud con lo mismo `ClientToken` se trata como una nueva solicitud.

Tipo: cadena

Restricciones de longitud: longitud mínima de 32. Longitud máxima de 128.

Obligatorio: no

MaxRows

El número total de filas que se devolverán en la `Query` salida. La ejecución inicial `Query` con un `MaxRows` valor especificado devolverá el conjunto de resultados de la consulta en dos casos:

- El tamaño del resultado es inferior a`1MB`.
- El número de filas del conjunto de resultados es inferior al valor de`maxRows`.

De lo contrario, la invocación inicial de `Query` solo devuelve a`NextToken`, que luego se puede usar en llamadas posteriores para obtener el conjunto de resultados. Para reanudar la paginación, indique el `NextToken` valor en el comando siguiente.

Si el tamaño de la fila es grande (por ejemplo, una fila tiene muchas columnas), Timestream puede devolver menos filas para evitar que el tamaño de la respuesta supere el límite de 1 MB. Si no `MaxRows` se proporciona, Timestream enviará el número de filas necesario para cumplir con el límite de 1 MB.

Tipo: entero

Rango válido: valor mínimo de 1. Valor máximo de 1000.

Obligatorio: no

NextToken

Un token de paginación que se utiliza para devolver un conjunto de resultados. Cuando `Query` API se invoca utilizando`NextToken`, se supone que esa invocación en particular es una invocación posterior de una llamada anterior a`Query`, y se devuelve un conjunto de resultados. Sin embargo, si la `Query` invocación solo contiene el`ClientToken`, se supone que la invocación de `Query` es una nueva ejecución de consulta.

Tenga en cuenta lo siguiente cuando se utilice NextToken en una consulta:

- Un token de paginación se puede utilizar para un máximo de cinco `Query` invocaciones o durante un máximo de 1 hora, lo que ocurra primero.
- Si lo usas, `NextToken` obtendrás el mismo conjunto de registros. Para seguir paginando el conjunto de resultados, debe utilizar el más reciente. `nextToken`
- Supongamos que una `Query` invocación devuelve dos `NextToken` valores y. `TokenA` `TokenB` Si `TokenB` se usa en una `Query` invocación posterior, `TokenA` se invalida y no se puede reutilizar.

- Para solicitar un conjunto de resultados anterior de una consulta una vez iniciada la paginación, debe volver a invocar la consulta. API
- La última `NextToken` debe usarse para paginar hasta que se devuelva, momento en el que `null` se debe usar una nueva. `NextToken`
- Si el IAM principio del iniciador de consultas y el lector de resultados no son iguales o el iniciador de la consulta y el lector de resultados no tienen la misma cadena de consulta en las solicitudes de consulta, la consulta fallará y se generará un error. `Invalid pagination token`

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Obligatorio: no

QueryInsights

Encapsula la configuración para la activación. `QueryInsights`

La activación `QueryInsights` devuelve información y métricas además de los resultados de la consulta que ejecutó. Se puede utilizar `QueryInsights` para ajustar el rendimiento de las consultas.

Tipo: objeto QueryInsights

Obligatorio: no

QueryString

La consulta que ejecutará Timestream.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 262144 caracteres.

Obligatorio: sí

Sintaxis de la respuesta

```
{
   "ColumnInfo": [
      {
         "Name": "string",
```

```
         "Type": {
            "ArrayColumnInfo": "ColumnInfo",
            "RowColumnInfo": [
               "ColumnInfo"
            ],
            "ScalarType": "string",
            "TimeSeriesMeasureValueColumnInfo": "ColumnInfo"
         }
      }
   ],
   "NextToken": "string",
   "QueryId": "string",
   "QueryInsightsResponse": {
      "OutputBytes": number,
      "OutputRows": number,
      "QuerySpatialCoverage": {
         "Max": {
            "PartitionKey": [ "string" ],
            "TableArn": "string",
            "Value": number
         }
      },
      "QueryTableCount": number,
      "QueryTemporalRange": {
         "Max": {
            "TableArn": "string",
            "Value": number
         }
      },
      "UnloadPartitionCount": number,
      "UnloadWrittenBytes": number,
      "UnloadWrittenRows": number
   },
   "QueryStatus": {
      "CumulativeBytesMetered": number,
      "CumulativeBytesScanned": number,
      "ProgressPercentage": number
   },
   "Rows": [
      {
         "Data": [
            {
               "ArrayValue": [
                  "Datum"
```

```
                ],
                "NullValue": boolean,
                "RowValue": "Row",
                "ScalarValue": "string",
                "TimeSeriesValue": [
                   {
                      "Time": "string",
                      "Value": "Datum"
                   }
                ]
             }
          ]
       }
    ]
}
```

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta de HTTP 200.

El servicio devuelve los siguientes datos en JSON formato.

ColumnInfo

Los tipos de datos de las columnas del conjunto de resultados devuelto.

Tipo: matriz de objetos ColumnInfo

NextToken

Un token de paginación que se puede volver a utilizar en una `Query` llamada para obtener el siguiente conjunto de resultados.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

QueryId

Un identificador único para la consulta dada.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 64.

Patrón: `[a-zA-Z0-9]+`

QueryInsightsResponse

`QueryInsights`Encapsula información y métricas relacionadas con la consulta que ejecutaste.

Tipo: objeto QueryInsightsResponse

QueryStatus

Información sobre el estado de la consulta, incluidos el progreso y los bytes escaneados.

Tipo: objeto QueryStatus

Rows

El conjunto de filas de resultados devueltas por la consulta.

Tipo: matriz de objetos Row

Errores

Para obtener información acerca de los errores comunes a todas las acciones, consulte Errores comunes.

AccessDeniedException

No está autorizado a realizar esta acción.

HTTPCódigo de estado: 400

ConflictException

No se han podido sondear los resultados de una consulta cancelada.

HTTPCódigo de estado: 400

InternalServerException

El servicio no pudo procesar completamente esta solicitud debido a un error interno del servidor.

HTTPCódigo de estado: 400

InvalidEndpointException

El punto final solicitado no era válido.

HTTPCódigo de estado: 400

QueryExecutionException

Timestream no pudo ejecutar la consulta correctamente.

HTTPCódigo de estado: 400

ThrottlingException

La solicitud fue denegada debido a una limitación de la solicitud.

HTTPCódigo de estado: 400

ValidationException

Solicitud no válida o mal formada.

HTTPCódigo de estado: 400

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- Interfaz de la línea de comandos de AWS
- AWS SDKpara. NET
- AWS SDKpara C++
- AWS SDKpara Go v2
- AWS SDKpara Java V2
- AWS SDKpara JavaScript V3
- AWS SDKpara PHP V3
- AWS SDKpara Python
- AWS SDKpara Ruby V3

TagResource
Servicio: Amazon Timestream Query

Asocie un conjunto de etiquetas a un recurso de Timestream. A continuación, puede activar estas etiquetas definidas por el usuario para que aparezcan en la consola Billing and Cost Management para el seguimiento de la asignación de costes.

Sintaxis de la solicitud

```
{
   "ResourceARN": "string",
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ]
}
```

Parámetros de la solicitud

Para obtener información sobre los parámetros comunes a todas las acciones, consulte Parámetros comunes.

La solicitud acepta los siguientes datos en JSON formato.

ResourceARN

Identifica el recurso Timestream al que se deben agregar las etiquetas. Este valor es un nombre de recurso de Amazon (ARN).

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Obligatorio: sí

Tags

Las etiquetas que se van a asignar al recurso Timestream.

Tipo: matriz de objetos Tag

Miembros de la matriz: número mínimo de 0 artículos. La cantidad máxima es de 200 artículos.

Obligatorio: sí

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta de HTTP 200 puntos con el cuerpo vacíoHTTP.

Errores

Para obtener información acerca de los errores comunes a todas las acciones, consulte Errores comunes.

InvalidEndpointException

El punto final solicitado no era válido.

HTTPCódigo de estado: 400

ResourceNotFoundException

No se ha encontrado el recurso solicitado.

HTTPCódigo de estado: 400

ServiceQuotaExceededException

Ha superado la cuota de servicio.

HTTPCódigo de estado: 400

ThrottlingException

La solicitud fue denegada debido a una limitación de la solicitud.

HTTPCódigo de estado: 400

ValidationException

Solicitud no válida o mal formada.

HTTPCódigo de estado: 400

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- Interfaz de la línea de comandos de AWS
- AWS SDKpara. NET
- AWS SDKpara C++
- AWS SDKpara Go v2
- AWS SDKpara Java V2
- AWS SDKpara JavaScript V3
- AWS SDKpara PHP V3
- AWS SDKpara Python
- AWS SDKpara Ruby V3

UntagResource
Servicio: Amazon Timestream Query

Elimina la asociación de etiquetas de un recurso de consulta de Timestream.

Sintaxis de la solicitud

```
{
   "ResourceARN": "string",
   "TagKeys": [ "string" ]
}
```

Parámetros de la solicitud

Para obtener información sobre los parámetros comunes a todas las acciones, consulte Parámetros comunes.

La solicitud acepta los siguientes datos en JSON formato.

ResourceARN

El recurso Timestream del que se eliminarán las etiquetas. Este valor es un nombre de recurso de Amazon (ARN).

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Obligatorio: sí

TagKeys

Una lista de claves de etiquetas. Las etiquetas existentes del recurso cuyas claves son miembros de esta lista se eliminarán del recurso Timestream.

Tipo: matriz de cadenas

Miembros de la matriz: número mínimo de 0 artículos. La cantidad máxima es de 200 artículos.

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 128 caracteres.

Obligatorio: sí

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta de HTTP 200 puntos con el cuerpo vacíoHTTP.

Errores

Para obtener información acerca de los errores comunes a todas las acciones, consulte Errores comunes.

InvalidEndpointException

El punto final solicitado no era válido.

HTTPCódigo de estado: 400

ResourceNotFoundException

No se ha encontrado el recurso solicitado.

HTTPCódigo de estado: 400

ThrottlingException

La solicitud fue denegada debido a una limitación de la solicitud.

HTTPCódigo de estado: 400

ValidationException

Solicitud no válida o con formato incorrecto.

HTTPCódigo de estado: 400

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- Interfaz de la línea de comandos de AWS
- AWS SDKpara. NET
- AWS SDKpara C++
- AWS SDKpara Go v2

- AWS SDKpara Java V2
- AWS SDKpara JavaScript V3
- AWS SDKpara PHP V3
- AWS SDKpara Python
- AWS SDKpara Ruby V3

UpdateAccountSettings
Servicio: Amazon Timestream Query

Hace la transición de su cuenta TCUs para utilizarla en los precios de las consultas y modifica el número máximo de unidades de cálculo de consultas que ha configurado. Si reduce el valor de `MaxQueryTCU` a la configuración deseada, el nuevo valor puede tardar hasta 24 horas en hacerse efectivo.

> **Note**
>
> Una vez que haya hecho la transición de su cuenta TCUs para utilizarla en los precios de consulta, no podrá pasar a utilizar los bytes escaneados para utilizarla en los precios de consulta.

Sintaxis de la solicitud

```
{
   "MaxQueryTCU": number,
   "QueryPricingModel": "string"
}
```

Parámetros de la solicitud

Para obtener información sobre los parámetros comunes a todas las acciones, consulte Parámetros comunes.

La solicitud acepta los siguientes datos en JSON formato.

MaxQueryTCU

El número máximo de unidades de cómputo que el servicio utilizará en cualquier momento para atender sus consultas. Para ejecutar consultas, debe establecer una capacidad mínima de 4TCU. Puede establecer el número máximo de TCU múltiplos de 4, por ejemplo, 4, 8, 16, 32, etc.

El valor máximo admitido `MaxQueryTCU` es 1000. Para solicitar un aumento de este límite flexible, ponte en contacto con AWS Support. Para obtener información sobre la cuota predeterminada maxQueryTCU, consulte Cuotas predeterminadas.

Tipo: entero

Obligatorio: no

QueryPricingModel

El modelo de precios para las consultas en una cuenta.

> **Note**
>
> Varias operaciones de Timestream utilizan el `QueryPricingModel` parámetro; sin embargo, la `UpdateAccountSettings` API operación no reconoce ningún otro valor que no sea. `COMPUTE_UNITS`

Tipo: cadena

Valores válidos: `BYTES_SCANNED | COMPUTE_UNITS`

Obligatorio: no

Sintaxis de la respuesta

```
{
   "MaxQueryTCU": number,
   "QueryPricingModel": "string"
}
```

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta de HTTP 200.

El servicio devuelve los siguientes datos en JSON formato.

MaxQueryTCU

El número máximo configurado de unidades de cómputo que el servicio utilizará en cualquier momento para atender sus consultas.

Tipo: entero

QueryPricingModel

El modelo de precios de una cuenta.

Tipo: cadena

Valores válidos: `BYTES_SCANNED | COMPUTE_UNITS`

Errores

Para obtener información acerca de los errores comunes a todas las acciones, consulte Errores comunes.

AccessDeniedException

No está autorizado a realizar esta acción.

HTTPCódigo de estado: 400

InternalServerException

El servicio no pudo procesar completamente esta solicitud debido a un error interno del servidor.

HTTPCódigo de estado: 400

InvalidEndpointException

El punto final solicitado no era válido.

HTTPCódigo de estado: 400

ThrottlingException

La solicitud fue denegada debido a una limitación de la solicitud.

HTTPCódigo de estado: 400

ValidationException

Solicitud no válida o con formato incorrecto.

HTTPCódigo de estado: 400

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- Interfaz de la línea de comandos de AWS

- AWS SDKpara. NET
- AWS SDKpara C++
- AWS SDKpara Go v2
- AWS SDKpara Java V2
- AWS SDKpara JavaScript V3
- AWS SDKpara PHP V3
- AWS SDKpara Python
- AWS SDKpara Ruby V3

UpdateScheduledQuery
Servicio: Amazon Timestream Query

Actualizar una consulta programada.

Sintaxis de la solicitud

```
{
   "ScheduledQueryArn": "string",
   "State": "string"
}
```

Parámetros de la solicitud

Para obtener información sobre los parámetros comunes a todas las acciones, consulte Parámetros comunes.

La solicitud acepta los siguientes datos en JSON formato.

ScheduledQueryArn

ARNde la consulta programada.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Obligatorio: sí

State

Estado de la consulta programada.

Tipo: cadena

Valores válidos: `ENABLED | DISABLED`

Obligatorio: sí

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta de HTTP 200 puntos con HTTP el cuerpo vacío.

Errores

Para obtener información acerca de los errores comunes a todas las acciones, consulte Errores comunes.

AccessDeniedException

No está autorizado a realizar esta acción.

HTTPCódigo de estado: 400

InternalServerException

El servicio no pudo procesar completamente esta solicitud debido a un error interno del servidor.

HTTPCódigo de estado: 400

InvalidEndpointException

El punto final solicitado no era válido.

HTTPCódigo de estado: 400

ResourceNotFoundException

No se ha encontrado el recurso solicitado.

HTTPCódigo de estado: 400

ThrottlingException

La solicitud fue denegada debido a una limitación de la solicitud.

HTTPCódigo de estado: 400

ValidationException

Solicitud no válida o con formato incorrecto.

HTTPCódigo de estado: 400

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- Interfaz de la línea de comandos de AWS
- AWS SDKpara. NET
- AWS SDKpara C++
- AWS SDKpara Go v2
- AWS SDKpara Java V2
- AWS SDKpara JavaScript V3
- AWS SDKpara PHP V3
- AWS SDKpara Python
- AWS SDKpara Ruby V3

Data Types

Amazon Timestream Write admite los siguientes tipos de datos:

- BatchLoadProgressReport
- BatchLoadTask
- BatchLoadTaskDescription
- CsvConfiguration
- Database
- DataModel
- DataModelConfiguration
- DataModelS3Configuration
- DataSourceConfiguration
- DataSourceS3Configuration
- Dimension
- DimensionMapping
- Endpoint
- MagneticStoreRejectedDataLocation
- MagneticStoreWriteProperties
- MeasureValue
- MixedMeasureMapping
- MultiMeasureAttributeMapping

- MultiMeasureMappings
- PartitionKey
- Record
- RecordsIngested
- RejectedRecord
- ReportConfiguration
- ReportS3Configuration
- RetentionProperties
- S3Configuration
- Schema
- Table
- Tag

Amazon Timestream Query admite los siguientes tipos de datos:

- ColumnInfo
- Datum
- DimensionMapping
- Endpoint
- ErrorReportConfiguration
- ErrorReportLocation
- ExecutionStats
- MixedMeasureMapping
- MultiMeasureAttributeMapping
- MultiMeasureMappings
- NotificationConfiguration
- ParameterMapping
- QueryInsights
- QueryInsightsResponse
- QuerySpatialCoverage
- QuerySpatialCoverageMax

- QueryStatus
- QueryTemporalRange
- QueryTemporalRangeMax
- Row
- S3Configuration
- S3ReportLocation
- ScheduleConfiguration
- ScheduledQuery
- ScheduledQueryDescription
- ScheduledQueryInsights
- ScheduledQueryInsightsResponse
- ScheduledQueryRunSummary
- SelectColumn
- SnsConfiguration
- Tag
- TargetConfiguration
- TargetDestination
- TimeSeriesDataPoint
- TimestreamConfiguration
- TimestreamDestination
- Type

Amazon Timestream Write

Amazon Timestream Write admite los siguientes tipos de datos:

- BatchLoadProgressReport
- BatchLoadTask
- BatchLoadTaskDescription
- CsvConfiguration
- Database
- DataModel

- DataModelConfiguration
- DataModelS3Configuration
- DataSourceConfiguration
- DataSourceS3Configuration
- Dimension
- DimensionMapping
- Endpoint
- MagneticStoreRejectedDataLocation
- MagneticStoreWriteProperties
- MeasureValue
- MixedMeasureMapping
- MultiMeasureAttributeMapping
- MultiMeasureMappings
- PartitionKey
- Record
- RecordsIngested
- RejectedRecord
- ReportConfiguration
- ReportS3Configuration
- RetentionProperties
- S3Configuration
- Schema
- Table
- Tag

BatchLoadProgressReport
Servicio: Amazon Timestream Write

Detalles sobre el progreso de una tarea de carga por lotes.

Contenido

BytesMetered

Tipo: largo

Obligatorio: no

FileFailures

Tipo: largo

Obligatorio: no

ParseFailures

Tipo: largo

Obligatorio: no

RecordIngestionFailures

Tipo: largo

Obligatorio: no

RecordsIngested

Tipo: largo

Obligatorio: no

RecordsProcessed

Tipo: largo

Obligatorio: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

BatchLoadTask
Servicio: Amazon Timestream Write

Detalles sobre una tarea de carga por lotes.

Contenido

CreationTime

La hora en que se creó la tarea de carga por lotes de Timestream.

Tipo: marca temporal

Obligatorio: no

DatabaseName

Nombre de base de datos de la base de datos en la que se cargan los datos una tarea de carga por lotes.

Tipo: cadena

Requerido: no

LastUpdatedTime

Hora en la que se actualizó por última vez la tarea de carga por lotes de Timestream.

Tipo: marca temporal

Obligatorio: no

ResumableUntil

Tipo: marca temporal

Obligatorio: no

TableName

Nombre de la tabla en la que una tarea de carga por lotes carga los datos.

Tipo: cadena

Requerido: no

TaskId

El ID de la tarea de carga por lotes.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 3. La longitud máxima es de 32 caracteres.

Patrón: `[A-Z0-9]+`

Obligatorio: no

TaskStatus

Estado de la tarea de carga por lotes.

Tipo: cadena

Valores válidos: `CREATED | IN_PROGRESS | FAILED | SUCCEEDED | PROGRESS_STOPPED | PENDING_RESUME`

Obligatorio: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

BatchLoadTaskDescription
Servicio: Amazon Timestream Write

Detalles sobre una tarea de carga por lotes.

Contenido

CreationTime

La hora en que se creó la tarea de carga por lotes de Timestream.

Tipo: marca temporal

Obligatorio: no

DataModelConfiguration

Configuración del modelo de datos para una tarea de carga por lotes. Contiene detalles sobre dónde se almacena un modelo de datos para una tarea de carga por lotes.

Tipo: objeto DataModelConfiguration

Obligatorio: no

DataSourceConfiguration

Detalles de configuración sobre la fuente de datos de una tarea de carga por lotes.

Tipo: objeto DataSourceConfiguration

Obligatorio: no

ErrorMessage

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Obligatorio: no

LastUpdatedTime

La hora en que se actualizó por última vez la tarea de carga por lotes de Timestream.

Tipo: marca temporal

Obligatorio: no

ProgressReport

Tipo: objeto BatchLoadProgressReport

Obligatorio: no

RecordVersion

Tipo: largo

Obligatorio: no

ReportConfiguration

Informe de la configuración de una tarea de carga por lotes. Contiene detalles sobre dónde se almacenan los informes de errores.

Tipo: objeto ReportConfiguration

Obligatorio: no

ResumableUntil

Tipo: marca temporal

Obligatorio: no

TargetDatabaseName

Tipo: cadena

Requerido: no

TargetTableName

Tipo: cadena

Requerido: no

TaskId

El ID de la tarea de carga por lotes.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 3. La longitud máxima es de 32 caracteres.

Patrón: `[A-Z0-9]+`

Obligatorio: no

TaskStatus

Estado de la tarea de carga por lotes.

Tipo: cadena

Valores válidos: `CREATED | IN_PROGRESS | FAILED | SUCCEEDED | PROGRESS_STOPPED | PENDING_RESUME`

Obligatorio: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

CsvConfiguration
Servicio: Amazon Timestream Write

Formato de datos delimitados en el que el separador de columnas puede ser una coma y el separador de registros es un carácter de nueva línea.

Contenido

ColumnSeparator

El separador de columnas puede ser de tipo coma (','), barra vertical ('|), punto y coma (';'), tabulador ('/t') o espacio en blanco (").

Tipo: cadena

Restricciones de longitud: longitud fija de 1.

Obligatorio: no

EscapeChar

El personaje de escape puede ser uno de

Tipo: cadena

Restricciones de longitud: longitud fija de 1.

Obligatorio: no

NullValue

Puede ser un espacio en blanco (").

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 256 caracteres.

Obligatorio: no

QuoteChar

Puede ser una comilla simple (') o una comilla doble («).

Tipo: cadena

Restricciones de longitud: longitud fija de 1.

Obligatorio: no

TrimWhiteSpace

Especifica que se recorten los espacios en blanco iniciales y finales.

Tipo: Booleano

Obligatorio: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

Database

Servicio: Amazon Timestream Write

Un contenedor de nivel superior para una mesa. Las bases de datos y las tablas son los conceptos de administración fundamentales de Amazon Timestream. Todas las tablas de una base de datos se cifran con la misma AWS KMS clave.

Contenido

Arn

El nombre del recurso de Amazon que identifica de forma exclusiva a esta base de datos.

Tipo: cadena

Requerido: no

CreationTime

La hora en que se creó la base de datos, calculada a partir de la época de Unix.

Tipo: marca temporal

Obligatorio: no

DatabaseName

Nombre de la base de datos de Timestream.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 3. La longitud máxima es de 256 caracteres.

Obligatorio: no

KmsKeyId

El identificador de la AWS KMS clave utilizada para cifrar los datos almacenados en la base de datos.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Obligatorio: no

LastUpdatedTime

La última vez que se actualizó esta base de datos.

Tipo: marca temporal

Obligatorio: no

TableCount

El número total de tablas que se encuentran en una base de datos de Timestream.

Tipo: largo

Obligatorio: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

DataModel

Servicio: Amazon Timestream Write

Modelo de datos para una tarea de carga por lotes.

Contenido

DimensionMappings

Mapeos de dimensiones de origen a destino.

Tipo: matriz de objetos DimensionMapping

Miembros de la matriz: número mínimo de 1 artículo.

Obligatorio: sí

MeasureNameColumn

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 256 caracteres.

Obligatorio: no

MixedMeasureMappings

Mapeos de origen a destino para las medidas.

Tipo: matriz de objetos MixedMeasureMapping

Miembros de la matriz: número mínimo de 1 artículo.

Obligatorio: no

MultiMeasureMappings

Mapeos de origen a destino para registros de múltiples medidas.

Tipo: objeto MultiMeasureMappings

Obligatorio: no

TimeColumn

La columna de origen se asignará al tiempo.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 256 caracteres.

Obligatorio: no

TimeUnit

La granularidad de la unidad de marca de tiempo. Indica si el valor de tiempo está en segundos, milisegundos, nanosegundos u otros valores admitidos. El valor predeterminado es `MILLISECONDS`.

Tipo: cadena

Valores válidos: `MILLISECONDS | SECONDS | MICROSECONDS | NANOSECONDS`

Obligatorio: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

DataModelConfiguration
Servicio: Amazon Timestream Write

Contenido

DataModel

Tipo: objeto DataModel

Obligatorio: no

DataModelS3Configuration

Tipo: objeto DataModelS3Configuration

Obligatorio: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

DataModelS3Configuration
Servicio: Amazon Timestream Write

Contenido

BucketName

Tipo: cadena

Limitaciones de longitud: longitud mínima de 3. La longitud máxima es de 63 caracteres.

Patrón: `[a-z0-9][\.\-a-z0-9]{1,61}[a-z0-9]`

Obligatorio: no

ObjectKey

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 1024 caracteres.

Patrón: `[a-zA-Z0-9|!\-_*'\(\)]([a-zA-Z0-9]|[!\-_*'\(\)\/.])+`

Obligatorio: no

Véase también

Para obtener más información sobre su uso API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

DataSourceConfiguration
Servicio: Amazon Timestream Write

Define los detalles de configuración de la fuente de datos.

Contenido

DataFormat

Esto es actualmenteCSV.

Tipo: cadena

Valores válidos: `CSV`

Obligatorio: sí

DataSourceS3Configuration

Configuración de una ubicación en S3 para un archivo que contiene los datos que se van a cargar.

Tipo: objeto DataSourceS3Configuration

Obligatorio: sí

CsvConfiguration

Formato de datos delimitado en el que el separador de columnas puede ser una coma y el separador de registros es un carácter de nueva línea.

Tipo: objeto CsvConfiguration

Obligatorio: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

DataSourceS3Configuration
Servicio: Amazon Timestream Write

Contenido

BucketName

El nombre del bucket de S3 del cliente.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 3. La longitud máxima es de 63 caracteres.

Patrón: `[a-z0-9][\.\-a-z0-9]{1,61}[a-z0-9]`

Obligatorio: sí

ObjectKeyPrefix

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 1024 caracteres.

Patrón: `[a-zA-Z0-9|!\-_*'\(\)]([a-zA-Z0-9]|[!\-_*'\(\)\/.])+`

Obligatorio: no

Véase también

Para obtener más información sobre su uso API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

Dimension
Servicio: Amazon Timestream Write

Representa los atributos de los metadatos de la serie temporal. Por ejemplo, el nombre y la zona de disponibilidad de una EC2 instancia o el nombre del fabricante de un aerogenerador son dimensiones.

Contenido

Name

La dimensión representa los atributos de los metadatos de la serie temporal. Por ejemplo, el nombre y la zona de disponibilidad de una EC2 instancia o el nombre del fabricante de un aerogenerador son dimensiones.

Para conocer las restricciones en los nombres de las dimensiones, consulte Restricciones de nomenclatura.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. Longitud máxima de 60.

Obligatorio: sí

Value

El valor de la dimensión.

Tipo: cadena

Obligatorio: sí

DimensionValueType

El tipo de datos de la dimensión del punto de datos de la serie temporal.

Tipo: cadena

Valores válidos: `VARCHAR`

Obligatorio: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

DimensionMapping
Servicio: Amazon Timestream Write

Contenido

DestinationColumn

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1.

Obligatorio: no

SourceColumn

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1.

Obligatorio: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

Endpoint
Servicio: Amazon Timestream Write

Representa un punto final disponible contra el que realizar API llamadas, así como el punto final TTL para ese punto final.

Contenido

Address

Una dirección de punto final.

Tipo: cadena

Obligatorio: sí

CachePeriodInMinutes

La TTL del punto final, en minutos.

Tipo: largo

Obligatorio: sí

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

MagneticStoreRejectedDataLocation
Servicio: Amazon Timestream Write

La ubicación en donde se tienen que escribir los informes de errores para los registros rechazados de forma asíncrona durante las escrituras en el almacén magnético.

Contenido

S3Configuration

La configuración de una ubicación de S3 en donde se tienen que escribir los informes de errores para los registros rechazados de forma asíncrona durante las escrituras en el almacén magnético.

Tipo: objeto S3Configuration

Obligatorio: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

MagneticStoreWriteProperties
Servicio: Amazon Timestream Write

El conjunto de propiedades de una tabla para configurar las escrituras en el almacén magnético.

Contenido

EnableMagneticStoreWrites

Un indicador para habilitar las escrituras en el almacén magnético.

Tipo: Booleano

Obligatorio: sí

MagneticStoreRejectedDataLocation

La ubicación en donde se tienen que escribir los informes de errores para los registros rechazados de forma asíncrona durante las escrituras en el almacén magnético.

Tipo: objeto MagneticStoreRejectedDataLocation

Obligatorio: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

MeasureValue
Servicio: Amazon Timestream Write

Representa el atributo de datos de la serie temporal. Por ejemplo, la CPU utilización de una EC2 instancia o la RPM de un aerogenerador son medidas. MeasureValue tiene nombre y valor.

MeasureValue solo está permitido para el tipo`MULTI`. Con el `MULTI` tipo, puede pasar varios atributos de datos asociados a la misma serie temporal en un solo registro

Contenido

Name

El nombre del MeasureValue.

Para conocer las restricciones de MeasureValue nombres, consulte Restricciones de nomenclatura en la Guía para desarrolladores de Amazon Timestream.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1.

Obligatorio: sí

Type

Contiene el tipo de datos del punto de datos MeasureValue de la serie temporal.

Tipo: cadena

Valores válidos: `DOUBLE | BIGINT | VARCHAR | BOOLEAN | TIMESTAMP | MULTI`

Obligatorio: sí

Value

El valor del MeasureValue. Para obtener información, consulte Tipos de datos.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Obligatorio: sí

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

MixedMeasureMapping
Servicio: Amazon Timestream Write

Contenido

MeasureValueType

Tipo: cadena

Valores válidos: `DOUBLE | BIGINT | VARCHAR | BOOLEAN | TIMESTAMP | MULTI`

Obligatorio: sí

MeasureName

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1.

Obligatorio: no

MultiMeasureAttributeMappings

Tipo: matriz de objetos MultiMeasureAttributeMapping

Miembros de la matriz: número mínimo de 1 artículo.

Obligatorio: no

SourceColumn

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1.

Obligatorio: no

TargetMeasureName

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1.

Obligatorio: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

MultiMeasureAttributeMapping
Servicio: Amazon Timestream Write

Contenido

SourceColumn

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1.

Obligatorio: sí

MeasureValueType

Tipo: cadena

Valores válidos: `DOUBLE | BIGINT | BOOLEAN | VARCHAR | TIMESTAMP`

Obligatorio: no

TargetMultiMeasureAttributeName

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1.

Obligatorio: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

MultiMeasureMappings
Servicio: Amazon Timestream Write

Contenido

MultiMeasureAttributeMappings

Tipo: matriz de objetos MultiMeasureAttributeMapping

Miembros de la matriz: número mínimo de 1 artículo.

Obligatorio: sí

TargetMultiMeasureName

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1.

Obligatorio: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

PartitionKey
Servicio: Amazon Timestream Write

Atributo que se utiliza para particionar los datos de una tabla. Una clave de dimensión divide los datos usando los valores de la dimensión especificados en el nombre de la dimensión como clave de partición, mientras que una clave de medida divide los datos usando nombres de medidas (valores de la columna «measure_name»).

Contenido

Type

El tipo de clave de partición. Las opciones son DIMENSION (clave de dimensión) y MEASURE (clave de medida).

Tipo: cadena

Valores válidos: `DIMENSION | MEASURE`

Obligatorio: sí

EnforcementInRecord

El nivel de cumplimiento de la especificación de una clave de dimensión en los registros ingeridos. Las opciones son REQUIRED (se debe especificar la clave de dimensión) y OPTIONAL (no es necesario especificar la clave de dimensión).

Tipo: cadena

Valores válidos: `REQUIRED | OPTIONAL`

Obligatorio: no

Name

El nombre del atributo utilizado como clave de dimensión.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1.

Obligatorio: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

Record

Servicio: Amazon Timestream Write

Representa un punto de datos de una serie temporal que se escribe en Timestream. Cada registro contiene una matriz de dimensiones. Las dimensiones representan los atributos de los metadatos de un punto de datos de una serie temporal, como el nombre de la instancia o la zona de disponibilidad de una EC2 instancia. Un registro también contiene el nombre de la medida, que es el nombre de la medida que se recopila (por ejemplo, la CPU utilización de una EC2 instancia). Además, un registro contiene el valor de la medida y el tipo de valor, que es el tipo de datos del valor de la medida. Además, el registro contiene la marca de tiempo del momento en que se recopiló la medida y la unidad de la marca de tiempo, que representa la granularidad de la marca de tiempo.

Los registros tienen un `Version` campo, que es de 64 bits, que se puede usar para actualizar los `long` puntos de datos. La escritura de un registro duplicado con la misma dimensión, marca de tiempo y nombre de medida, pero con un valor de medida diferente, solo se realizará correctamente si el `Version` atributo del registro de la solicitud de escritura es superior al del registro existente. El valor predeterminado de Timestream es de o para los registros sin `Version` el campo`1`. `Version`

Contenido

Dimensions

Contiene la lista de dimensiones de los puntos de datos de series temporales.

Tipo: matriz de objetos Dimension

Miembros de la matriz: número máximo de 128 elementos.

Obligatorio: no

MeasureName

La medida representa el atributo de datos de la serie temporal. Por ejemplo, la CPU utilización de una EC2 instancia o la RPM de un aerogenerador son medidas.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 256 caracteres.

Obligatorio: no

MeasureValue

Contiene el valor de medición del punto de datos de la serie temporal.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Obligatorio: no

MeasureValues

Contiene la lista de cuatro puntos MeasureValue de datos de series temporales.

Esto solo está permitido para el tipo`MULTI`. Para valores escalares, utilice directamente el `MeasureValue` atributo del registro.

Tipo: matriz de objetos MeasureValue

Obligatorio: no

MeasureValueType

Contiene el tipo de datos del valor de medición del punto de datos de la serie temporal. El tipo predeterminado es`DOUBLE`. Para obtener más información, consulte Tipos de datos.

Tipo: cadena

Valores válidos: `DOUBLE | BIGINT | VARCHAR | BOOLEAN | TIMESTAMP | MULTI`

Obligatorio: no

Time

Contiene la hora en la que se recopiló el valor de medición del punto de datos. El valor de tiempo más la unidad proporciona el tiempo transcurrido desde la época. Por ejemplo, si el valor del tiempo es `12345` y la unidad es`ms`, entonces `12345 ms` han transcurrido desde la época.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 256 caracteres.

Obligatorio: no

TimeUnit

La granularidad de la unidad de marca de tiempo. Indica si el valor de tiempo está en segundos, milisegundos, nanosegundos u otros valores admitidos. El valor predeterminado es `MILLISECONDS`.

Tipo: cadena

Valores válidos: `MILLISECONDS | SECONDS | MICROSECONDS | NANOSECONDS`

Obligatorio: no

Version

Atributo de 64 bits utilizado para las actualizaciones de registros. Si se escriben solicitudes de datos duplicados con un número de versión superior, se actualizarán el valor de medida y la versión existentes. En los casos en que el valor de la medida sea el mismo, se `Version` seguirá actualizando. El valor predeterminado es 1.

> **Note**
>
> `Version`debe ser igual 1 o superior o recibirá un `ValidationException` error.

Tipo: largo

Obligatorio: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

RecordsIngested
Servicio: Amazon Timestream Write

Información sobre los registros ingeridos por esta solicitud.

Contenido

MagneticStore

Recuento de registros ingresados en el almacén magnético.

Tipo: entero

Obligatorio: no

MemoryStore

Recuento de registros ingeridos en el almacén de memoria.

Tipo: entero

Obligatorio: no

Total

Recuento total de registros ingeridos correctamente.

Tipo: entero

Obligatorio: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

RejectedRecord
Servicio: Amazon Timestream Write

Representa los registros que no se insertaron correctamente en Timestream debido a problemas de validación de datos que deben resolverse antes de volver a insertar los datos de series temporales en el sistema.

Contenido

ExistingVersion

La versión existente del registro. Este valor se rellena en situaciones en las que existe un registro idéntico con una versión superior a la versión de la solicitud de escritura.

Tipo: largo

Obligatorio: no

Reason

El motivo por el que un registro no se insertó correctamente en Timestream. Entre las posibles causas del error se incluyen las siguientes:

- Registros con datos duplicados en los que hay varios registros con las mismas dimensiones, marcas de tiempo y nombres de medidas, pero:
  - Los valores de las medidas son diferentes
  - La versión no está presente en la solicitud o el valor de la versión en el nuevo registro es igual o inferior al valor existente

  Si Timestream rechaza los datos en este caso, el `ExistingVersion` campo de la `RejectedRecords` respuesta indicará la versión del registro actual. Para forzar una actualización, puede volver a enviar la solicitud con una versión del registro establecida en un valor superior a. `ExistingVersion`
- Registros con marcas de tiempo que se encuentran fuera del período de retención del almacén de memoria.

  > **Note**
  >
  > Cuando se actualice la ventana de retención, recibirá una `RejectedRecords` excepción si intenta ingerir datos inmediatamente dentro de la nueva ventana. Para evitar una `RejectedRecords` excepción, espere hasta que finalice la nueva

ventana para ingerir nuevos datos. Para obtener más información, consulte Prácticas recomendadas para configurar Timestream y la explicación de cómo funciona el almacenamiento en Timestream.

- Registros con dimensiones o medidas que superan los límites definidos por Timestream.

Para obtener más información, consulte Access Management en la Guía del desarrollador de Timestream.

Tipo: cadena

Requerido: no

RecordIndex

El índice del registro en la solicitud de entrada de. WriteRecords Los índices comienzan por 0.

Tipo: entero

Obligatorio: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

ReportConfiguration
Servicio: Amazon Timestream Write

Configuración de informes para una tarea de carga por lotes. Contiene detalles sobre dónde se almacenan los informes de errores.

Contenido

ReportS3Configuration

Configuración de una ubicación de S3 para escribir informes de errores y eventos para una carga por lotes.

Tipo: objeto ReportS3Configuration

Obligatorio: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

ReportS3Configuration
Servicio: Amazon Timestream Write

Contenido

BucketName

Tipo: cadena

Limitaciones de longitud: longitud mínima de 3. La longitud máxima es de 63 caracteres.

Patrón: `[a-z0-9][\.\-a-z0-9]{1,61}[a-z0-9]`

Obligatorio: sí

EncryptionOption

Tipo: cadena

Valores válidos: `SSE_S3 | SSE_KMS`

Obligatorio: no

KmsKeyId

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Obligatorio: no

ObjectKeyPrefix

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. Longitud máxima de 928.

Patrón: `[a-zA-Z0-9|!\-_*'\(\)]([a-zA-Z0-9]|[!\-_*'\(\)\/.])+`

Obligatorio: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

RetentionProperties
Servicio: Amazon Timestream Write

Las propiedades de retención contienen el tiempo durante el cual los datos de serie temporal deben almacenarse en el almacén magnético y en el almacén de memoria.

Contenido

MagneticStoreRetentionPeriodInDays

El tiempo durante el que deben almacenarse los datos en el almacén magnético.

Tipo: largo

Rango válido: valor mínimo de 1. Valor máximo de 73000.

Obligatorio: sí

MemoryStoreRetentionPeriodInHours

El tiempo durante el que deben almacenarse los datos en el almacén de memoria.

Tipo: largo

Rango válido: valor mínimo de 1. Valor máximo de 8766.

Obligatorio: sí

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

S3Configuration

Servicio: Amazon Timestream Write

La configuración que especifica una ubicación de S3.

Contenido

BucketName

El nombre del bucket de S3 del cliente.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 3. La longitud máxima es de 63 caracteres.

Patrón: `[a-z0-9][\.\-a-z0-9]{1,61}[a-z0-9]`

Obligatorio: no

EncryptionOption

La opción de cifrado de la ubicación de S3 del cliente. Las opciones son el cifrado S3 del lado del servidor con una clave gestionada o AWS una clave gestionada de S3.

Tipo: cadena

Valores válidos: `SSE_S3 | SSE_KMS`

Obligatorio: no

KmsKeyId

El identificador de AWS KMS clave de la ubicación del cliente en S3 al cifrar con una AWS clave gestionada.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Obligatorio: no

ObjectKeyPrefix

La vista previa de la clave de objeto para la ubicación de S3 del cliente.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. Longitud máxima de 928.

Patrón: `[a-zA-Z0-9|!\-_*'\(\)]([a-zA-Z0-9]|[!\-_*'\(\)\/.])+`

Obligatorio: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

Schema

Servicio: Amazon Timestream Write

Un esquema especifica el modelo de datos esperado de la tabla.

Contenido

CompositePartitionKey

Una lista no vacía de claves de partición que definen los atributos utilizados para particionar los datos de la tabla. El orden de la lista determina la jerarquía de particiones. El nombre y el tipo de cada clave de partición, así como el orden de las claves de partición, no se pueden cambiar una vez creada la tabla. Sin embargo, se puede cambiar el nivel de cumplimiento de cada clave de partición.

Tipo: matriz de objetos PartitionKey

Miembros de la matriz: número mínimo de 1 artículo.

Obligatorio: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

Table

Servicio: Amazon Timestream Write

Representa una tabla de base de datos en Timestream. Las tablas contienen una o más series temporales relacionadas. Puede modificar la duración de retención del almacén de memoria y del almacén magnético de una tabla.

Contenido

Arn

El nombre del recurso de Amazon que identifica de forma exclusiva a esta tabla.

Tipo: cadena

Requerido: no

CreationTime

La hora en que se creó la tabla Timestream.

Tipo: marca temporal

Obligatorio: no

DatabaseName

Nombre de la base de datos de Timestream que contiene esta tabla.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 3. La longitud máxima es de 256 caracteres.

Obligatorio: no

LastUpdatedTime

Hora en la que se actualizó por última vez la tabla Timestream.

Tipo: marca temporal

Obligatorio: no

MagneticStoreWriteProperties

Contiene las propiedades que se tienen que establecer en la tabla cuando se habilitan las escrituras en el almacén magnético.

Tipo: objeto MagneticStoreWriteProperties

Obligatorio: no

RetentionProperties

Duración de la retención del almacén de memoria y el almacén magnético.

Tipo: objeto RetentionProperties

Obligatorio: no

Schema

El esquema de la tabla.

Tipo: objeto Schema

Obligatorio: no

TableName

Nombre de la tabla de Timestream.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 3. La longitud máxima es de 256 caracteres.

Obligatorio: no

TableStatus

El estado actual de la tabla:

- `DELETING`- Se está borrando la tabla.
- `ACTIVE`- La mesa está lista para su uso.

Tipo: cadena

Valores válidos: `ACTIVE | DELETING | RESTORING`

Obligatorio: no

Véase también

Para obtener más información sobre su uso API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

Tag
Servicio: Amazon Timestream Write

Una etiqueta es una etiqueta que se asigna a and/or table. Each tag consists of a key and an optional value, both of which you define. With tags, you can categorize databases and/or las tablas de una base de datos de Timestream, por ejemplo, por propósito, propietario o entorno.

Contenido

Key

La clave de la etiqueta. Las claves de etiqueta distinguen entre mayúsculas y minúsculas.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 128 caracteres.

Obligatorio: sí

Value

El valor de la etiqueta. Los valores de las etiquetas distinguen mayúsculas de minúsculas y pueden ser nulos.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 0. La longitud máxima es de 256 caracteres.

Obligatorio: sí

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

Consulta de Amazon Timestream

Amazon Timestream Query admite los siguientes tipos de datos:

- ColumnInfo
- Datum
- DimensionMapping
- Endpoint
- ErrorReportConfiguration
- ErrorReportLocation
- ExecutionStats
- MixedMeasureMapping
- MultiMeasureAttributeMapping
- MultiMeasureMappings
- NotificationConfiguration
- ParameterMapping
- QueryInsights
- QueryInsightsResponse
- QuerySpatialCoverage
- QuerySpatialCoverageMax
- QueryStatus
- QueryTemporalRange
- QueryTemporalRangeMax
- Row
- S3Configuration
- S3ReportLocation
- ScheduleConfiguration
- ScheduledQuery
- ScheduledQueryDescription
- ScheduledQueryInsights
- ScheduledQueryInsightsResponse
- ScheduledQueryRunSummary
- SelectColumn
- SnsConfiguration

- Tag
- TargetConfiguration
- TargetDestination
- TimeSeriesDataPoint
- TimestreamConfiguration
- TimestreamDestination
- Type

ColumnInfo
Servicio: Amazon Timestream Query

Contiene los metadatos de los resultados de la consulta, como los nombres de las columnas, los tipos de datos y otros atributos.

Contenido

Type

El tipo de datos de la columna del conjunto de resultados. El tipo de datos puede ser escalar o complejo. Los tipos de datos escalares son enteros, cadenas, dobles, booleanos y otros. Los tipos de datos complejos son tipos como matrices, filas y otros.

Tipo: objeto Type

Obligatorio: sí

Name

El nombre de la columna del conjunto de resultados. El nombre del conjunto de resultados está disponible para las columnas de todos los tipos de datos, excepto para las matrices.

Tipo: cadena

Requerido: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

Datum

Servicio: Amazon Timestream Query

El dato representa un único punto de datos en el resultado de una consulta.

Contenido

ArrayValue

Indica si el punto de datos es una matriz.

Tipo: matriz de objetos Datum

Obligatorio: no

NullValue

Indica si el punto de datos es nulo.

Tipo: Booleano

Obligatorio: no

RowValue

Indica si el punto de datos es una fila.

Tipo: objeto Row

Obligatorio: no

ScalarValue

Indica si el punto de datos es un valor escalar, como entero, cadena, doble o booleano.

Tipo: cadena

Requerido: no

TimeSeriesValue

Indica si el punto de datos es un tipo de datos de serie temporal.

Tipo: matriz de objetos TimeSeriesDataPoint

Obligatorio: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

DimensionMapping
Servicio: Amazon Timestream Query

Este tipo se utiliza para asignar columnas del resultado de la consulta a una dimensión de la tabla de destino.

Contenido

DimensionValueType

Tipo para la dimensión.

Tipo: cadena

Valores válidos: `VARCHAR`

Obligatorio: sí

Name

Nombre de la columna del resultado de la consulta.

Tipo: cadena

Obligatorio: sí

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

Endpoint
Servicio: Amazon Timestream Query

Representa un punto final disponible contra el que realizar API llamadas, así como el punto final TTL para ese punto final.

Contenido

Address

Una dirección de punto final.

Tipo: cadena

Obligatorio: sí

CachePeriodInMinutes

La TTL del punto final, en minutos.

Tipo: largo

Obligatorio: sí

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

ErrorReportConfiguration
Servicio: Amazon Timestream Query

Se requiere configuración para los informes de error.

Contenido

S3Configuration

La configuración del S3 para los informes de error.

Tipo: objeto S3Configuration

Obligatorio: sí

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

ErrorReportLocation
Servicio: Amazon Timestream Query

Contiene la ubicación del informe de errores de una única llamada de consulta programada.

Contenido

S3ReportLocation

La ubicación de S3 donde se escriben los informes de errores.

Tipo: objeto S3ReportLocation

Obligatorio: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

ExecutionStats
Servicio: Amazon Timestream Query

Estadísticas de una sola ejecución de consulta programada.

Contenido

BytesMetered

Bytes medidos para una sola ejecución de consulta programada.

Tipo: largo

Obligatorio: no

CumulativeBytesScanned

Bytes escaneados para una única ejecución de consulta programada.

Tipo: largo

Obligatorio: no

DataWrites

Las escrituras de datos se miden para los registros ingeridos en una sola ejecución de consulta programada.

Tipo: largo

Obligatorio: no

ExecutionTimeInMillis

Tiempo total, medido en milisegundos, que se necesitó para completar la ejecución de la consulta programada.

Tipo: largo

Obligatorio: no

QueryResultRows

Número de filas presentes en el resultado de ejecutar una consulta antes de transferirla a la fuente de datos de destino.

Tipo: largo

Obligatorio: no

RecordsIngested

El número de registros ingeridos para una sola ejecución de consulta programada.

Tipo: largo

Obligatorio: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

MixedMeasureMapping
Servicio: Amazon Timestream Query

MixedMeasureMappings son mapeos que se pueden usar para incorporar datos en una combinación de medidas limitadas y múltiples en la tabla derivada.

Contenido

MeasureValueType

Tipo del valor desde el que se va a leer. sourceColumn Si el mapeo es paraMULTI, utilice MeasureValueType. MULTI.

Tipo: cadena

Valores válidos: `BIGINT | BOOLEAN | DOUBLE | VARCHAR | MULTI`

Obligatorio: sí

MeasureName

Se refiere al valor de measure_name en una fila de resultados. Este campo es obligatorio si MeasureNameColumn se proporciona.

Tipo: cadena

Requerido: no

MultiMeasureAttributeMappings

Obligatorio cuando measureValueType esMULTI. Asignaciones de atributos para medidas de MULTI valor.

Tipo: matriz de objetos MultiMeasureAttributeMapping

Miembros de la matriz: número mínimo de 1 artículo.

Obligatorio: no

SourceColumn

Este campo se refiere a la columna de origen de la que debe leerse el valor de la medida para la materialización del resultado.

Tipo: cadena

Requerido: no

TargetMeasureName

Nombre de la medida objetivo que se va a utilizar. Si no se proporciona, el nombre de la medida objetivo de forma predeterminada sería measure-name si se proporciona o no. sourceColumn

Tipo: cadena

Requerido: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

MultiMeasureAttributeMapping
Servicio: Amazon Timestream Query

Mapeo de atributos para medidas de MULTI valor.

Contenido

MeasureValueType

Tipo de atributo que se va a leer de la columna de origen.

Tipo: cadena

Valores válidos: `BIGINT | BOOLEAN | DOUBLE | VARCHAR | TIMESTAMP`

Obligatorio: sí

SourceColumn

Columna de origen de la que se va a leer el valor del atributo.

Tipo: cadena

Obligatorio: sí

TargetMultiMeasureAttributeName

Nombre personalizado que se utilizará para el nombre del atributo en la tabla derivada. Si no se proporciona, se utilizará el nombre de la columna de origen.

Tipo: cadena

Requerido: no

Véase también

Para obtener más información sobre su uso API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

MultiMeasureMappings
Servicio: Amazon Timestream Query

Solo MultiMeasureMappings se proporcionará uno de MixedMeasureMappings ellos. MultiMeasureMappings se puede usar para ingerir datos como medidas múltiples en la tabla derivada.

Contenido

MultiMeasureAttributeMappings

Obligatorio. Mapeos de atributos que se utilizarán para asignar los resultados de la consulta a los datos de ingesta para los atributos multimedida.

Tipo: matriz de objetos MultiMeasureAttributeMapping

Miembros de la matriz: número mínimo de 1 artículo.

Obligatorio: sí

TargetMultiMeasureName

El nombre de la medida múltiple objetivo en la tabla derivada. Esta entrada es obligatoria cuando no measureNameColumn se proporciona. Si MeasureNameColumn se proporciona, el valor de esa columna se utilizará como nombre de varias medidas.

Tipo: cadena

Requerido: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

NotificationConfiguration
Servicio: Amazon Timestream Query

Configuración de notificaciones para una consulta programada. Timestream envía una notificación cuando se crea una consulta programada, se actualiza su estado o se elimina.

Contenido

SnsConfiguration

Detalles de la SNS configuración.

Tipo: objeto SnsConfiguration

Obligatorio: sí

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

ParameterMapping
Servicio: Amazon Timestream Query

Mapeo de parámetros con nombre.

Contenido

Name

Nombre del parámetro.

Tipo: cadena

Obligatorio: sí

Type

Contiene el tipo de datos de una columna del conjunto de resultados de una consulta. El tipo de datos puede ser escalar o complejo. Los tipos de datos escalares admitidos son enteros, booleanos, cadenas, dobles, marcas de tiempo, fecha, hora e intervalos. Los tipos de datos complejos admitidos son matrices, filas y series temporales.

Tipo: objeto Type

Obligatorio: sí

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

QueryInsights
Servicio: Amazon Timestream Query

`QueryInsights`es una función de ajuste del rendimiento que le ayuda a optimizar sus consultas, reducir los costes y mejorar el rendimiento. Con `QueryInsights` ella, puede evaluar la eficiencia de reducción de sus consultas e identificar las áreas de mejora para mejorar el rendimiento de las consultas. También puede analizar la eficacia de sus consultas en términos de reducción temporal y espacial, e identificar oportunidades para mejorar el rendimiento. `QueryInsights` En concreto, puede evaluar en qué medida sus consultas utilizan las estrategias de indexación basadas en el tiempo y en las claves de partición para optimizar la recuperación de datos. Para optimizar el rendimiento de las consultas, es esencial ajustar los parámetros temporales y espaciales que rigen la ejecución de las consultas.

Las métricas clave que proporciona `QueryInsights` son `QuerySpatialCoverage` y. `QueryTemporalRange` `QuerySpatialCoverage`indica qué parte del eje espacial explora la consulta; los valores más bajos son más eficientes. `QueryTemporalRange`muestra el intervalo de tiempo escaneado, mientras que los intervalos más estrechos ofrecen un mejor rendimiento.

Ventajas de QueryInsights

Los principales beneficios de su uso son los siguientes`QueryInsights`:

- Identificación de consultas ineficientes: `QueryInsights` proporciona información sobre cómo depurar las tablas a las que accede la consulta en función del tiempo y de los atributos. Esta información le ayuda a identificar las tablas a las que no se accede de forma óptima.
- Optimización del modelo de datos y el particionamiento: puede utilizar la `QueryInsights` información para acceder a su modelo de datos y su estrategia de particionamiento y ajustarlos.
- Ajustar las consultas: `QueryInsights` destaca las oportunidades de utilizar los índices de forma más eficaz.

> **Note**
>
> El número máximo de `Query` API solicitudes que puedes realizar con la `QueryInsights` opción activada es de 1 consulta por segundo (QPS). Si superas esta tasa de consultas, es posible que se produzcan limitaciones.

Contenido

Mode

Proporciona los siguientes modos de activación: `QueryInsights`

- `ENABLED_WITH_RATE_CONTROL`— Habilita `QueryInsights` el procesamiento de las consultas. Este modo también incluye un mecanismo de control de velocidad, que limita la `QueryInsights` función a 1 consulta por segundo (QPS).
- `DISABLED`— Deshabilita. `QueryInsights`

Tipo: cadena

Valores válidos: `ENABLED_WITH_RATE_CONTROL | DISABLED`

Obligatorio: sí

Véase también

Para obtener más información sobre su uso API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

QueryInsightsResponse
Servicio: Amazon Timestream Query

Proporciona varios datos y métricas relacionados con la consulta que ejecutó.

Contenido

OutputBytes

Indica el tamaño del conjunto de resultados de la consulta en bytes. Puede utilizar estos datos para validar si el conjunto de resultados ha cambiado como parte del ejercicio de ajuste de la consulta.

Tipo: largo

Obligatorio: no

OutputRows

Indica el número total de filas devueltas como parte del conjunto de resultados de la consulta. Puede utilizar estos datos para validar si el número de filas del conjunto de resultados ha cambiado como parte del ejercicio de ajuste de la consulta.

Tipo: largo

Obligatorio: no

QuerySpatialCoverage

Proporciona información sobre la cobertura espacial de la consulta, incluida la tabla con un ajuste espacial subóptimo (máximo). Esta información puede ayudarlo a identificar las áreas que deben mejorarse en su estrategia de partición para mejorar la reducción espacial.

Tipo: objeto QuerySpatialCoverage

Obligatorio: no

QueryTableCount

Indica el número de tablas de la consulta.

Tipo: largo

Obligatorio: no

QueryTemporalRange

Proporciona información sobre el rango temporal de la consulta, incluida la tabla con el rango de tiempo más grande (máximo). Las siguientes son algunas de las posibles opciones para optimizar la poda en función del tiempo:

- Añada los predicados temporales que faltan.
- Elimine las funciones en torno a los predicados de tiempo.
- Agregue predicados de tiempo a todas las subconsultas.

Tipo: objeto QueryTemporalRange

Obligatorio: no

UnloadPartitionCount

Indica las particiones creadas por la `Unload` operación.

Tipo: largo

Obligatorio: no

UnloadWrittenBytes

Indica el tamaño, en bytes, escrito por la `Unload` operación.

Tipo: largo

Obligatorio: no

UnloadWrittenRows

Indica las filas escritas por la `Unload` consulta.

Tipo: largo

Obligatorio: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++

- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

QuerySpatialCoverage
Servicio: Amazon Timestream Query

Proporciona información sobre la cobertura espacial de la consulta, incluida la tabla con una reducción espacial subóptima (máxima). Esta información puede ayudarlo a identificar las áreas que deben mejorarse en su estrategia de partición para mejorar la reducción espacial

Por ejemplo, puede hacer lo siguiente con la `QuerySpatialCoverage` información:

- Añada measure_name o utilice los predicados de clave de [partición () definidos por el cliente](). CDPK
- Si ya ha realizado la acción anterior, elimine las funciones que las rodeen o las cláusulas, como. `LIKE`

Contenido

Max

Proporciona información sobre la cobertura espacial de la consulta ejecutada y de la tabla con la depuración espacial más ineficiente.

- `Value`— La relación máxima de cobertura espacial.
- `TableArn`— El nombre del recurso de Amazon (ARN) de la tabla con una reducción espacial subóptima.
- `PartitionKey`— La clave de partición utilizada para la partición, que puede ser una predeterminada `measure_name` o una. CDPK

Tipo: objeto QuerySpatialCoverageMax

Obligatorio: no

Véase también

Para obtener más información sobre su uso API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

QuerySpatialCoverageMax
Servicio: Amazon Timestream Query

Proporciona información sobre la tabla con el rango espacial menos óptimo analizado por la consulta.

Contenido

PartitionKey

La clave de partición utilizada para la partición, que puede ser una clave de partición predeterminada `measure_name` o definida por el cliente.

Tipo: matriz de cadenas

Obligatorio: no

TableArn

El nombre del recurso de Amazon (ARN) de la tabla con la reducción espacial menos óptima.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Obligatorio: no

Value

La relación máxima de cobertura espacial.

Tipo: Doble

Obligatorio: no

Véase también

Para obtener más información sobre su uso API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

QueryStatus
Servicio: Amazon Timestream Query

Información sobre el estado de la consulta, incluidos el progreso y los bytes escaneados.

Contenido

CumulativeBytesMetered

La cantidad de datos escaneados por la consulta en bytes que se le cobrará. Se trata de una suma acumulada y representa la cantidad total de datos que se le cobrará desde que se inició la consulta. El cargo se aplica solo una vez y se aplica cuando la consulta termina de ejecutarse o cuando se cancela.

Tipo: largo

Obligatorio: no

CumulativeBytesScanned

La cantidad de datos escaneados por la consulta en bytes. Se trata de una suma acumulativa y representa la cantidad total de bytes escaneados desde que se inició la consulta.

Tipo: largo

Obligatorio: no

ProgressPercentage

El progreso de la consulta, expresado como porcentaje.

Tipo: Doble

Obligatorio: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

QueryTemporalRange
Servicio: Amazon Timestream Query

Proporciona información sobre el rango temporal de la consulta, incluida la tabla con el rango de tiempo más grande (máximo).

Contenido

Max

Encapsula las siguientes propiedades que proporcionan información sobre la tabla de rendimiento menos óptima del eje temporal:

- `Value`— La duración máxima en nanosegundos entre el inicio y el final de la consulta.
- `TableArn`— El nombre del recurso de Amazon (ARN) de la tabla que se consulta con el intervalo de tiempo más amplio.

Tipo: objeto QueryTemporalRangeMax

Obligatorio: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

QueryTemporalRangeMax
Servicio: Amazon Timestream Query

Proporciona información sobre la tabla con la reducción temporal menos óptima analizada por la consulta.

Contenido

TableArn

El nombre del recurso de Amazon (ARN) de la tabla que se consulta con el intervalo de tiempo más amplio.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Obligatorio: no

Value

La duración máxima en nanosegundos entre el inicio y el final de la consulta.

Tipo: largo

Obligatorio: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

Row
Servicio: Amazon Timestream Query

Representa una sola fila en los resultados de la consulta.

Contenido

Data

Lista de puntos de datos en una sola fila del conjunto de resultados.

Tipo: matriz de objetos Datum

Obligatorio: sí

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

S3Configuration
Servicio: Amazon Timestream Query

Detalles sobre la ubicación de S3 para los informes de error que resultan de la puesta en marcha de una consulta.

Contenido

BucketName

Nombre del bucket de S3 en el que se crearán los informes de error.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 3. La longitud máxima es de 63 caracteres.

Patrón: `[a-z0-9][\.\-a-z0-9]{1,61}[a-z0-9]`

Obligatorio: sí

EncryptionOption

Opciones de cifrado en reposo para los informes de error. Si no se especifica ninguna opción de cifrado, Timestream elegirá SSE _S3 de forma predeterminada.

Tipo: cadena

Valores válidos: `SSE_S3 | SSE_KMS`

Obligatorio: no

ObjectKeyPrefix

Prefijo de la clave del informe de error. Timestream añade de forma predeterminada el siguiente prefijo a la ruta del informe de errores.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. Longitud máxima de 896.

Patrón: `[a-zA-Z0-9|!\-_*'\(\)]([a-zA-Z0-9]|[!\-_*'\(\)\/.])+`

Obligatorio: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

S3ReportLocation
Servicio: Amazon Timestream Query

Ubicación del informe S3 para la ejecución de la consulta programada.

Contenido

BucketName

Nombre del bucket de S3.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 3. La longitud máxima es de 63 caracteres.

Patrón: `[a-z0-9][\.\-a-z0-9]{1,61}[a-z0-9]`

Obligatorio: no

ObjectKey

Clave S3.

Tipo: cadena

Requerido: no

Véase también

Para obtener más información sobre su uso API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

ScheduleConfiguration
Servicio: Amazon Timestream Query

Configuración de la programación de la consulta.

Contenido

ScheduleExpression

Una expresión que denota cuándo activar la puesta en marcha de la consulta programada. Puede ser una expresión de cron o una expresión rate.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 256 caracteres.

Obligatorio: sí

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

ScheduledQuery
Servicio: Amazon Timestream Query

Consulta programada

Contenido

Arn

El nombre del recurso de Amazon.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Obligatorio: sí

Name

El nombre de la consulta programada.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 64.

Patrón: `[a-zA-Z0-9|!\-_*'\(\)]([a-zA-Z0-9]|[!\-_*'\(\)\/.])+`

Obligatorio: sí

State

Estado de la consulta programada.

Tipo: cadena

Valores válidos: `ENABLED | DISABLED`

Obligatorio: sí

CreationTime

Hora de creación de la consulta programada.

Tipo: marca temporal

Obligatorio: no

ErrorReportConfiguration

Configuración para la notificación de errores de consultas programadas.

Tipo: objeto ErrorReportConfiguration

Obligatorio: no

LastRunStatus

Estado de la última ejecución de la consulta programada.

Tipo: cadena

Valores válidos: `AUTO_TRIGGER_SUCCESS | AUTO_TRIGGER_FAILURE | MANUAL_TRIGGER_SUCCESS | MANUAL_TRIGGER_FAILURE`

Obligatorio: no

NextInvocationTime

La próxima vez que se ejecute la consulta programada.

Tipo: marca temporal

Obligatorio: no

PreviousInvocationTime

La última vez que se ejecutó la consulta programada.

Tipo: marca temporal

Obligatorio: no

TargetDestination

Fuente de datos de destino en la que se escribirá el resultado final de la consulta programada.

Tipo: objeto TargetDestination

Obligatorio: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

ScheduledQueryDescription
Servicio: Amazon Timestream Query

Estructura que describe la consulta programada.

Contenido

Arn

Consulta programadaARN.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Obligatorio: sí

Name

Nombre de la consulta programada.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 64.

Patrón: `[a-zA-Z0-9|!\-_*'\(\)]([a-zA-Z0-9]|[!\-_*'\(\)\/.])+`

Obligatorio: sí

NotificationConfiguration

Configuración de notificaciones.

Tipo: objeto NotificationConfiguration

Obligatorio: sí

QueryString

La consulta que se va a ejecutar.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 262144 caracteres.

Obligatorio: sí

ScheduleConfiguration

Programar la configuración.

Tipo: objeto ScheduleConfiguration

Obligatorio: sí

State

Estado de la consulta programada.

Tipo: cadena

Valores válidos: `ENABLED | DISABLED`

Obligatorio: sí

CreationTime

Hora de creación de la consulta programada.

Tipo: marca temporal

Obligatorio: no

ErrorReportConfiguration

Configuración de notificación de errores para la consulta programada.

Tipo: objeto ErrorReportConfiguration

Obligatorio: no

KmsKeyId

Una KMS clave proporcionada por el cliente que se utiliza para cifrar el recurso de consulta programada.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Obligatorio: no

LastRunSummary

Resumen del tiempo de ejecución de la última consulta programada.

Tipo: objeto ScheduledQueryRunSummary

Obligatorio: no

NextInvocationTime

La próxima vez que esté programada la ejecución de la consulta programada.

Tipo: marca temporal

Obligatorio: no

PreviousInvocationTime

La última vez que se ejecutó la consulta.

Tipo: marca temporal

Obligatorio: no

RecentlyFailedRuns

Resumen del tiempo de ejecución de las últimas cinco ejecuciones de consultas programadas fallidas.

Tipo: matriz de objetos ScheduledQueryRunSummary

Obligatorio: no

ScheduledQueryExecutionRoleArn

IAMfunción que Timestream utiliza para ejecutar la consulta de programación.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Obligatorio: no

TargetConfiguration

Configuración del almacén de destino de la consulta programada.

Tipo: objeto TargetConfiguration

Obligatorio: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

ScheduledQueryInsights
Servicio: Amazon Timestream Query

Encapsula la configuración para habilitarla `QueryInsights` en un`ExecuteScheduledQueryRequest`.

Contenido

Mode

Proporciona los siguientes modos de activación`ScheduledQueryInsights`:

- `ENABLED_WITH_RATE_CONTROL`— Habilita `ScheduledQueryInsights` el procesamiento de las consultas. Este modo también incluye un mecanismo de control de velocidad, que limita la `QueryInsights` función a 1 consulta por segundo (QPS).
- `DISABLED`— Desactiva. `ScheduledQueryInsights`

Tipo: cadena

Valores válidos: `ENABLED_WITH_RATE_CONTROL | DISABLED`

Obligatorio: sí

Véase también

Para obtener más información sobre su uso API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

ScheduledQueryInsightsResponse
Servicio: Amazon Timestream Query

Proporciona varios datos y métricas relacionados con lo `ExecuteScheduledQueryRequest` que se ejecutó.

Contenido

OutputBytes

Indica el tamaño del conjunto de resultados de la consulta en bytes. Puede utilizar estos datos para validar si el conjunto de resultados ha cambiado como parte del ejercicio de ajuste de la consulta.

Tipo: largo

Obligatorio: no

OutputRows

Indica el número total de filas devueltas como parte del conjunto de resultados de la consulta. Puede utilizar estos datos para validar si el número de filas del conjunto de resultados ha cambiado como parte del ejercicio de ajuste de la consulta.

Tipo: largo

Obligatorio: no

QuerySpatialCoverage

Proporciona información sobre la cobertura espacial de la consulta, incluida la tabla con un ajuste espacial subóptimo (máximo). Esta información puede ayudarlo a identificar las áreas que deben mejorarse en su estrategia de partición para mejorar la reducción espacial.

Tipo: objeto QuerySpatialCoverage

Obligatorio: no

QueryTableCount

Indica el número de tablas de la consulta.

Tipo: largo

Obligatorio: no

QueryTemporalRange

Proporciona información sobre el rango temporal de la consulta, incluida la tabla con el rango de tiempo más grande (máximo). Las siguientes son algunas de las posibles opciones para optimizar la poda en función del tiempo:

- Añada los predicados temporales que faltan.
- Elimine las funciones relacionadas con los predicados de tiempo.
- Agregue predicados de tiempo a todas las subconsultas.

Tipo: objeto QueryTemporalRange

Obligatorio: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

ScheduledQueryRunSummary
Servicio: Amazon Timestream Query

Ejecute el resumen de la consulta programada

Contenido

ErrorReportLocation

Ubicación de S3 para el informe de errores.

Tipo: objeto ErrorReportLocation

Obligatorio: no

ExecutionStats

Estadísticas de tiempo de ejecución de una ejecución programada.

Tipo: objeto ExecutionStats

Obligatorio: no

FailureReason

Mensaje de error para la consulta programada en caso de error. Puede que tengas que consultar el informe de errores para obtener información más detallada sobre los motivos del error.

Tipo: cadena

Requerido: no

InvocationTime

InvocationTime para esta carrera. Esta es la hora a la que está programada la ejecución de la consulta. El parámetro se `@scheduled_runtime` puede usar en la consulta para obtener el valor.

Tipo: marca temporal

Obligatorio: no

QueryInsightsResponse

Proporciona varios datos y métricas relacionados con el resumen de ejecución de la consulta programada.

Tipo: objeto ScheduledQueryInsightsResponse

Obligatorio: no

RunStatus

El estado de la ejecución de una consulta programada.

Tipo: cadena

Valores válidos: `AUTO_TRIGGER_SUCCESS | AUTO_TRIGGER_FAILURE | MANUAL_TRIGGER_SUCCESS | MANUAL_TRIGGER_FAILURE`

Obligatorio: no

TriggerTime

La hora real en que se ejecutó la consulta.

Tipo: marca temporal

Obligatorio: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

SelectColumn
Servicio: Amazon Timestream Query

Detalles de la columna que devuelve la consulta.

Contenido

Aliased

Es cierto si la consulta puso un alias al nombre de la columna. De lo contrario, es falso.

Tipo: Booleano

Obligatorio: no

DatabaseName

Base de datos que contiene esta columna.

Tipo: cadena

Requerido: no

Name

El nombre de la columna.

Tipo: cadena

Requerido: no

TableName

Tabla de la base de datos que tiene esta columna.

Tipo: cadena

Requerido: no

Type

Contiene el tipo de datos de una columna del conjunto de resultados de una consulta. El tipo de datos puede ser escalar o complejo. Los tipos de datos escalares admitidos son enteros, booleanos, cadenas, dobles, marcas de tiempo, fecha, hora e intervalos. Los tipos de datos complejos admitidos son matrices, filas y series temporales.

Tipo: objeto Type

Obligatorio: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

SnsConfiguration
Servicio: Amazon Timestream Query

Se SNS requieren detalles al respecto para enviar la notificación.

Contenido

TopicArn

SNStema al ARN que se enviarán las notificaciones de estado de la consulta programada.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Obligatorio: sí

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

Tag

Servicio: Amazon Timestream Query

Una etiqueta es una etiqueta que se asigna a and/or table. Each tag consists of a key and an optional value, both of which you define. Tags enable you to categorize databases and/or las tablas de una base de datos de Timestream, por ejemplo, por propósito, propietario o entorno.

Contenido

Key

La clave de la etiqueta. Las claves de etiqueta distinguen entre mayúsculas y minúsculas.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 128 caracteres.

Obligatorio: sí

Value

El valor de la etiqueta. Los valores de etiquetas distinguen entre mayúsculas y minúsculas y pueden ser nulos.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 0. La longitud máxima es de 256 caracteres.

Obligatorio: sí

Véase también

Para obtener más información sobre su uso API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

TargetConfiguration
Servicio: Amazon Timestream Query

Configuración que se utiliza para escribir el resultado de una consulta.

Contenido

TimestreamConfiguration

Configuración necesaria para escribir datos en la base de datos y la tabla de Timestream.

Tipo: objeto TimestreamConfiguration

Obligatorio: sí

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

TargetDestination
Servicio: Amazon Timestream Query

Detalles de destino para escribir datos para una fuente de datos de destino. La fuente de datos compatible actual es Timestream.

Contenido

TimestreamDestination

Consulte los detalles del destino de los resultados para la fuente de datos de Timestream.

Tipo: objeto TimestreamDestination

Obligatorio: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

TimeSeriesDataPoint
Servicio: Amazon Timestream Query

El tipo de datos de serie temporal representa los valores de una medida a lo largo del tiempo. Una serie temporal es una matriz de filas de marcas temporales y valores de medidas, con las filas ordenadas en orden temporal ascendente. A TimeSeriesDataPoint es un único punto de datos de la serie temporal. Representa una tupla de (tiempo, valor de medición) en una serie temporal.

Contenido

Time

La marca temporal en la que se recopiló el valor de la medida.

Tipo: cadena

Obligatorio: sí

Value

El valor de medición del punto de datos.

Tipo: objeto Datum

Obligatorio: sí

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

TimestreamConfiguration
Servicio: Amazon Timestream Query

Configuración para escribir datos en la base de datos y la tabla de Timestream. Esta configuración permite al usuario asignar las columnas de selección del resultado de la consulta a las columnas de la tabla de destino.

Contenido

DatabaseName

Nombre de la base de datos Timestream en la que se escribirá el resultado de la consulta.

Tipo: cadena

Obligatorio: sí

DimensionMappings

Esto permite asignar columnas del resultado de la consulta a la dimensión en la tabla de destino.

Tipo: matriz de objetos DimensionMapping

Obligatorio: sí

TableName

Nombre de la tabla Timestream en la que se escribirá el resultado de la consulta. La tabla debe estar dentro de la misma base de datos que se proporciona en la configuración de Timestream.

Tipo: cadena

Obligatorio: sí

TimeColumn

Columna del resultado de la consulta que debe utilizarse como columna de tiempo en la tabla de destino. El tipo de columna para esto debería serTIMESTAMP.

Tipo: cadena

Obligatorio: sí

MeasureNameColumn

Nombre de la columna de medidas.

Tipo: cadena

Requerido: no

MixedMeasureMappings

Especifica cómo asignar medidas a registros de múltiples medidas.

Tipo: matriz de objetos MixedMeasureMapping

Miembros de la matriz: número mínimo de 1 artículo.

Obligatorio: no

MultiMeasureMappings

Asignaciones de varias medidas.

Tipo: objeto MultiMeasureMappings

Obligatorio: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

TimestreamDestination
Servicio: Amazon Timestream Query

Destino de la consulta programada.

Contenido

DatabaseName

Nombre de la base de datos Timestream.

Tipo: cadena

Requerido: no

TableName

Nombre de la tabla de transmisión temporal.

Tipo: cadena

Requerido: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

Type
Servicio: Amazon Timestream Query

Contiene el tipo de datos de una columna del conjunto de resultados de una consulta. El tipo de datos puede ser escalar o complejo. Los tipos de datos escalares admitidos son enteros, booleanos, cadenas, dobles, marcas de tiempo, fecha, hora e intervalos. Los tipos de datos complejos admitidos son matrices, filas y series temporales.

Contenido

ArrayColumnInfo

Indica si la columna es una matriz.

Tipo: objeto ColumnInfo

Obligatorio: no

RowColumnInfo

Indica si la columna es una fila.

Tipo: matriz de objetos ColumnInfo

Obligatorio: no

ScalarType

Indica si la columna es de tipo cadena, entero, booleano, doble, marca de tiempo, fecha o hora. Para obtener más información, consulte Tipos de datos compatibles.

Tipo: cadena

Valores válidos: `VARCHAR | BOOLEAN | BIGINT | DOUBLE | TIMESTAMP | DATE | TIME | INTERVAL_DAY_TO_SECOND | INTERVAL_YEAR_TO_MONTH | UNKNOWN | INTEGER`

Obligatorio: no

TimeSeriesMeasureValueColumnInfo

Indica si la columna es un tipo de datos de serie temporal.

Tipo: objeto ColumnInfo

Obligatorio: no

Véase también

Para obtener más información sobre cómo usarlo API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDKpara C++
- AWS SDKpara Java V2
- AWS SDKpara Ruby V3

Errores comunes

En esta sección se enumeran los errores comunes a las API acciones de todos los AWS servicios. Para ver los errores específicos de una API acción de este servicio, consulte el tema correspondiente a esa API acción.

AccessDeniedException

No tiene acceso suficiente para realizar esta acción.

HTTPCódigo de estado: 400

IncompleteSignature

La firma de la solicitud no se ajusta a AWS las normas.

HTTPCódigo de estado: 400

InternalFailure

El procesamiento de la solicitud ha devuelto un error debido a un error o una excepción desconocidos.

HTTPCódigo de estado: 500

InvalidAction

La acción u operación solicitada no es válida. Compruebe que la acción se ha escrito correctamente.

HTTPCódigo de estado: 400

InvalidClientTokenId

El identificador de clave de AWS acceso o certificado X.509 proporcionado no existe en nuestros registros.

HTTPCódigo de estado: 403

NotAuthorized

No tiene permiso para realizar esta acción.

HTTPCódigo de estado: 400

OptInRequired

La ID de la clave de AWS acceso necesita una suscripción al servicio.

HTTPCódigo de estado: 403

RequestExpired

La solicitud llegó al servicio más de 15 minutos después del sello de fecha de la solicitud o más de 15 minutos después de la fecha de caducidad de la solicitud (por ejemplo, en el caso de los prefirmadosURLs), o el sello de fecha de la solicitud es más de 15 minutos en el futuro.

HTTPCódigo de estado: 400

ServiceUnavailable

La solicitud no se ha ejecutado correctamente debido a un error temporal del servidor.

HTTPCódigo de estado: 503

ThrottlingException

La solicitud fue denegada debido a una limitación de la solicitud.

HTTPCódigo de estado: 400

ValidationError

La entrada no cumple con las restricciones especificadas por un AWS servicio.

HTTPCódigo de estado: 400

Parámetros comunes

La siguiente lista contiene los parámetros que utilizan todas las acciones para firmar solicitudes de Signature Version 4 con una cadena de consulta. Los parámetros específicos de acción se enumeran en el tema correspondiente a la acción. Para obtener más información sobre la versión 4 de Signature, consulte Firmar AWS API solicitudes en la Guía del IAM usuario.

Action

Las acciones que se van a realizar.

Tipo: cadena

Obligatorio: sí

Version

La API versión para la que está escrita la solicitud, expresada en el formato YYYY-MM-DD.

Tipo: cadena

Obligatorio: sí

X-Amz-Algorithm

El algoritmo de hash que utilizó para crear la solicitud de firma.

Condición: especifique este parámetro cuando incluya la información de autenticación en una cadena de consulta en lugar de en el encabezado de HTTP autorización.

Tipo: cadena

Valores válidos: `AWS4-HMAC-SHA256`

Obligatorio: condicional

X-Amz-Credential

El valor del ámbito de la credencial, que es una cadena que incluye la clave de acceso, la fecha, la región a la que se dirige, el servicio que solicita y una cadena de terminación ("aws4_request"). El valor se expresa en el siguiente formato: access_key//region YYYYMMDD/ service /aws4_request.

Para obtener más información, consulte Crear una solicitud firmada en la Guía del usuario. AWS API IAM

Condición: especifique este parámetro cuando incluya la información de autenticación en una cadena de consulta en lugar de en el encabezado de HTTP autorización.

Tipo: cadena

Obligatorio: condicional

X-Amz-Date

La fecha utilizada para crear la firma. El formato debe ser el formato básico ISO 8601 (YYYYMMDD'T' HHMMSS 'Z'). Por ejemplo, la siguiente fecha y hora es un X-Amz-Date valor válido:`20120325T120000Z`.

Condición: X-Amz-Date es opcional para todas las solicitudes; se puede usar para anular la fecha utilizada para firmar las solicitudes. Si el encabezado de fecha se especifica en el formato básico ISO 8601, no X-Amz-Date es obligatorio. Cuando X-Amz-Date se usa, siempre anula el valor del encabezado de fecha. Para obtener más información, consulte Elementos de la firma de una AWS API solicitud en la Guía del IAM usuario.

Tipo: cadena

Obligatorio: condicional

X-Amz-Security-Token

El token de seguridad temporal que se obtuvo mediante una llamada a AWS Security Token Service (AWS STS). Para obtener una lista de los servicios que admiten credenciales de seguridad temporales AWS STS, consulte la Guía del IAM usuario con IAM los Servicios de AWS que puede trabajar.

Condición: si utilizas credenciales de seguridad temporales de AWS STS, debes incluir el token de seguridad.

Tipo: cadena

Obligatorio: condicional

X-Amz-Signature

Especifica la firma codificada hexadecimal que se calculó a partir de la cadena que se va a firmar y la clave de firma derivada.

Condición: especifique este parámetro cuando incluya la información de autenticación en una cadena de consulta en lugar de en el encabezado de HTTP autorización.

Tipo: cadena

Obligatorio: condicional

X-Amz-SignedHeaders

Especifica todos HTTP los encabezados que se incluyeron como parte de la solicitud canónica. Para obtener más información sobre cómo especificar los encabezados firmados, consulte Crear una AWS API solicitud firmada en la Guía del usuario. IAM

Condición: especifique este parámetro cuando incluya la información de autenticación en una cadena de consulta en lugar de en el encabezado de HTTP autorización.

Tipo: cadena

Obligatorio: condicional

Historial del documento

Cambio	Descripción	Fecha
Actualización exclusiva de la documentación	Se actualizó el tema Cuotas para separar las cuotas predeterminadas y los límites del sistema.	22 de octubre de 2024
Amazon Timestream ahora admite información sobre consultas	Timestream ahora incluye compatibilidad con la función de información sobre consultas, que le ayuda a optimizar las consultas , mejorar su rendimiento y reducir los costos.	22 de octubre de 2024
Actualización de Amazon Timestream for InfluxDB a una política existente.	Amazon Timestream para InfluxDB ha añadido la acción a `ec2:DescribeRouteTables` la política gestionada a `AmazonTimestreamIn`	8 de octubre de 2024

	`fluxDBFullAccess` existente para describir las tablas de rutas.	
`AmazonTimestreamReadOnlyAccess` — Actualización de una política existente	Timestream for LiveAnalytics ha añadido el `DescribeAccountSettings` permiso a la política `AmazonTimestreamReadOnlyAccess` gestionada para describir Cuenta de AWS la configuración.	3 de junio de 2024
Amazon Timestream admite LiveAnalytics por ahora unidades de cómputo Timestream () TCUs	LiveAnalytics Por ahora, Amazon Timestream incluye soporte para Timestream Compute Units TCUs () para medir la capacidad informática asignada a las necesidades de sus consultas.	29 de abril de 2024

Se han agregado nuevas políticas	Amazon Timestream para InfluxDB agregó dos políticas nuevas: una que permite al servicio administrar las interfaces de red y los grupos de seguridad de su cuenta. Para obtener más información, consulte. AmazonTimestreamInfluxDBServiceRolePolicy Otra que proporciona acceso administrativo completo para crear, actualizar, eliminar y enumerar instancias de Amazon Timestream InfluxDB y crear y enumerar grupos de parámetros. Para obtener más información, consulte. AmazonTimestreamInfluxDBFullAccess	14 de marzo de 2024
Amazon Timestream para InfluxDB ya está disponible de forma general.	Esta documentación cubre la versión inicial de Amazon Timestream para InfluxDB.	14 de marzo de 2024

Los eventos de Amazon Timestream LiveAnalytics for Query están disponibles en AWS CloudTrail	Amazon Timestream publica LiveAnalytics por ahora los eventos de datos de API Query en. AWS CloudTrail Los clientes pueden auditar todas API las solicitudes de consulta realizadas en sus AWS cuentas y ver informaci ón como qué IAM usuario o rol realizó la solicitud, cuándo se realizó la solicitud, qué bases de datos y tablas se consultar on y el ID de consulta de la solicitud.	12 de septiembre de 2023
Amazon Timestream para LiveAnalytics UNLOAD	LiveAnalytics Por ahora, Amazon Timestream UNLOAD admite la exportación de los resultados de las consultas a S3.	12 de mayo de 2023
Amazon Timestream LiveAnalytics para actualizar una política existente.	Se han añadido permisos de carga por lotes a una política gestionada.	24 de febrero de 2023
Amazon Timestream LiveAnalytics para carga de lotes.	Amazon Timestream admite LiveAnalytics por ahora la funcionalidad de carga por lotes.	24 de febrero de 2023
Amazon Timestream LiveAnalytics por ahora es compatible. AWS Backup	Amazon Timestream LiveAnalytics por ahora es compatible. AWS Backup	14 de diciembre de 2022

Amazon Timestream LiveAnalytics para actualizaciones de las políticas gestionadas AWS	Nueva información sobre las políticas AWS gestionadas y Amazon Timestream LiveAnalytics for, incluidas las actualizaciones de las políticas gestionadas existentes.	29 de noviembre de 2021
Amazon Timestream LiveAnalytics for admite consultas programadas	LiveAnalytics Por ahora, Amazon Timestream permite ejecutar una consulta en su nombre, según un cronograma.	29 de noviembre de 2021
Amazon Timestream LiveAnalytics para soporte de almacenamiento magnético.	LiveAnalytics Por ahora, Amazon Timestream admite el uso de almacenamiento magnético para las escrituras en tablas.	29 de noviembre de 2021
Amazon Timestream LiveAnalytics para registros de múltiples medidas.	LiveAnalytics Por ahora, Amazon Timestream admite un formato más compacto para almacenar los datos de series temporales.	29 de noviembre de 2021
Amazon Timestream LiveAnalytics para actualizaciones de las políticas gestionadas AWS	Nueva información sobre las políticas AWS gestionadas y Amazon Timestream LiveAnalytics for, incluidas las actualizaciones de las políticas gestionadas existentes.	24 de mayo de 2021
Amazon Timestream LiveAnalytics for ya está disponible en la región de Europa (Fráncfort).	Amazon Timestream LiveAnalytics for ya está disponible de forma general en la región de Europa (Fráncfort) (). `eu-central-1`	23 de abril de 2021

Amazon Timestream LiveAnalytics for ya es VPC compatible con endpoints ().AWS PrivateLink	Amazon Timestream admite LiveAnalytics por ahora el uso de puntos VPC de enlace ().AWS PrivateLink	23 de marzo de 2021
Amazon Timestream ahora admite consultas entre tablas.	Puede usar Amazon Timestream LiveAnalytics para ejecutar consultas entre tablas.	10 de febrero de 2021
Amazon Timestream, LiveAnalytics por ahora, admite estadísticas de ejecución de consultas mejoradas.	LiveAnalytics Por ahora, Amazon Timestream admite estadísticas de ejecución de consultas mejoradas, como la cantidad de datos escaneado s.	10 de febrero de 2021
Amazon Timestream admite LiveAnalytics por ahora funciones avanzadas de series temporales.	Puede utilizar Amazon Timestream LiveAnalytics para SQL ejecutar consultas con funciones avanzadas de series temporales, como derivadas, integrales y correlaciones.	10 de febrero de 2021
Amazon Timestream HIPAA ya ISO está disponible y LiveAnalytics cumple con las normas. PCI	Ahora puede usar Amazon Timestream LiveAnalytics para cargas de trabajo que HIPAA requieren una infraestructura compatibleISO. PCI	27 de enero de 2021

Amazon Timestream admite LiveAnalytics por ahora telegraf y grafana de código abierto.	Ahora puede usar Telegraf, el agente de servidor de código abierto y basado en complementos para recopilar e informar métricas, y Grafana, la plataforma de análisis y monitoreo de bases de datos de código abierto, con Amazon Timestream para. LiveAnalytics	25 de noviembre de 2020
Amazon Timestream LiveAnalytics for ya está disponible de forma general.	Esta documentación cubre la versión inicial de Amazon LiveAnalytics Timestream para.	30 de septiembre de 2020

¿Qué es Timestream para InfluxDB?

Amazon Timestream para InfluxDB es un motor de base de datos de series temporales gestionado que facilita a los desarrolladores DevOps y equipos de aplicaciones la ejecución de bases de datos de InfluxDB para aplicaciones de series temporales en tiempo real mediante código abierto. AWS APIs Con Amazon Timestream para InfluxDB, es fácil configurar, operar y escalar cargas de trabajo de series temporales que pueden responder consultas con un tiempo de respuesta de consultas de milisegundos de un solo dígito.

Amazon Timestream para InfluxDB le da acceso a las funciones de la conocida versión de código abierto de InfluxDB en su rama 2.x. Esto significa que el código, las aplicaciones y las herramientas que ya utiliza en la actualidad con sus bases de datos de código abierto de InfluxDB existentes deberían funcionar a la perfección con Amazon Timestream para InfluxDB. Amazon Timestream para InfluxDB puede realizar automáticamente una copia de seguridad de la base de datos y mantener el software de la base de datos actualizado con la versión más reciente. Además, Amazon Timestream para InfluxDB facilita el uso de la replicación para mejorar la disponibilidad de la base de datos y mejorar la durabilidad de los datos. Como ocurre con todos los AWS servicios, no se requieren inversiones iniciales y solo paga por los recursos que utilice.

Instancias de base de datos

Una instancia de base de datos es un entorno de base de datos aislado que se ejecuta en la nube. Es el componente básico de Amazon Timestream para InfluxDB. Una instancia de base de datos puede contener varias bases de datos creadas por el usuario (u organizaciones y grupos en el caso de las bases de datos InfluxDb 2.x), y se puede acceder a ella mediante las mismas herramientas y aplicaciones de cliente que podría utilizar para acceder a una instancia de InfluxDB autogestionada e independiente. Las instancias de base de datos son fáciles de crear y modificar con las herramientas de línea de AWS comandos, las operaciones de Amazon Timestream API InfluxDB o. AWS Management Console

> **Note**
>
> Amazon Timestream para InfluxDB permite el acceso a las bases de datos mediante las operaciones de Influx y la API interfaz de usuario de Influx. Amazon Timestream para InfluxDB no permite el acceso directo al host.

Puede tener hasta 40 instancias de Amazon Timestream para InfluxDB.

Cada instancia de base de datos tiene un nombre de instancia de base de datos. Este nombre proporcionado por el cliente identifica de forma exclusiva la instancia de base de datos cuando interactúa con Amazon Timestream for InfluxDB y los comandos. API AWS CLI El nombre de la instancia de base de datos debe ser único para ese cliente en una región. AWS

El nombre de la instancia de base de datos forma parte del DNS nombre de host asignado a su instancia por Timestream para InfluxDB. Por ejemplo, si especifica influxdb1 como nombre de la instancia de base de datos, Timestream asignará automáticamente un punto final a su instancia. DNS Un ejemplo de punto final es`influxdb1-3ksj4dla5nfjhi.us-east-1.timestream-influxdb.amazonaws.com`, ¿dónde `influxdb1` está el nombre de la instancia?

En el punto final de ejemplo`influxdb1-3ksj4dla5nfjhi.us-east-1.timestream-influxdb.amazonaws.com`, la cadena `3ksj4dla5nfjhi` es un identificador de cuenta único generado por AWS. El identificador `3ksj4dla5nfjhi` del ejemplo no cambia para la cuenta especificada en una región determinada. Por lo tanto, todas las instancias de base de datos creadas por esta cuenta comparten el mismo identificador fijo. Tenga en cuenta las siguientes características del identificador fijo:

- Actualmente, Timestream for InfluxDB no admite el cambio de nombre de las instancias de base de datos.
- Si elimina y vuelve a crear una instancia de base de datos con el mismo identificador de instancia de base de datos, el punto de conexión es el mismo.
- Si utiliza la misma cuenta para crear una instancia de base de datos en una región diferente, el identificador generado internamente es diferente porque la región es diferente, como en `influxdb2.4a3j5du5ks7md2.us-west-1.timestream-influxdb.amazonaws.com`.

Cada instancia de base de datos admite solo un motor de base de datos Timestream for InfluxDB.

Al crear una instancia de base de datos, InfluxDB requiere que se especifique el nombre de una organización. Una instancia de base de datos puede alojar varias organizaciones y varios depósitos asociados a cada organización.

Amazon Timestream para InfluxDB le permite crear una cuenta de usuario maestra y una contraseña para su instancia de base de datos como parte del proceso de creación. Este usuario maestro tiene permisos para crear organizaciones y depósitos y realizar operaciones de lectura, escritura, eliminación y modificación de sus datos. También podrás acceder a InfluxUI y recuperar tu token de

operador al iniciar sesión por primera vez. Desde allí, también podrás gestionar todos tus tokens de acceso. Debe establecer la contraseña de usuario maestra al crear una instancia de base de datos, pero puede cambiarla en cualquier momento mediante InfluxAPI, Influx CLI o InfluxUI.

Clases de instancia de base de datos

La clase de instancia de base de datos determina la capacidad de cálculo y memoria de una instancia de base de datos de `fi UbfkyxDB` Amazon Timestream. La clase de instancia de base de datos que se necesite dependerá de la potencia de procesamiento y de los requisitos de memoria.

Una clase de instancia de base de datos determina tanto el tamaño como el tipo de clase de instancia de base de datos. Por ejemplo, `db.influx` es un tipo de clase de instancia de base de datos optimizada para la memoria adecuada para los requisitos de memoria de alto rendimiento relacionados con la ejecución de cargas de trabajo. InfluxDb Dentro del tipo de clase de `db.influx` instancia, `db.influx.2xlarge` hay una clase de instancia de base de datos. El tamaño de esta clase es 2xlarge.

Para obtener más información sobre los precios de las clases de instancia, consulte los precios de Amazon Timestream for InfluxDB.

Tipos de clase de instancia de base de datos

Amazon Timestream para InfluxDB admite clases de instancias de base de datos para los siguientes casos de uso optimizados para los casos de uso de InfluxDB.

- **`db.influx`**—Estas clases de instancias son ideales para ejecutar cargas de trabajo que consumen mucha memoria en bases de datos de InfluxDB de código abierto

Especificaciones de hardware para las clases de instancias de base de datos

La siguiente terminología describe las especificaciones de hardware de las clases de instancias de base de datos:

- v CPU

  El número de unidades centrales de procesamiento virtuales (CPUs). Una virtual CPU es una unidad de capacidad que se puede utilizar para comparar clases de instancias de base de datos.

- Memoria (GiB)

  LaRAM, en gibibytes, asignada a la instancia de base de datos. Suele haber una relación constante entre la memoria y v. CPU Como ejemplo, tomemos la clase de instancia db.flux, que tiene una CPU relación entre memoria y v similar a la clase de instancia EC2 r7g.

- Optimizado para Influx

  Una instancia de base de datos utiliza una pila de configuración optimizada y proporciona capacidad adicional y dedicada para las E/S. Esta optimización proporciona el mejor rendimiento, ya que reduce al mínimo la contención entre las E/S y otro tráfico procedente de la instancia.

- Ancho de banda de red

  La velocidad de red relativa a otras clases de instancia de base de datos. En la siguiente tabla, encontrarás detalles de hardware sobre las clases de instancias de Amazon Timestream for InfluxDB.

Clase de instancias	v CPU	Memoria (GiB)	Storage Type	Ancho de banda de red (Gbps)
db.influx.medium	1	8	Influencia incluida IOPS	10
db.influx.large	2	16	Influencia incluida IOPS	10
db.influx.xlarge	4	32	Influencia incluida IOPS	10
db.influx.2xlarge	8	64	Influencia incluida IOPS	10
db.influx.4xlarge	16	128	Influencia incluida IOPS	10
db.influx.8xlarge	32	256	Influencia incluida IOPS	12

Clase de instancias	v CPU	Memoria (GiB)	Storage Type	Ancho de banda de red (Gbps)
db.influx.12 x large	48	384	Influencia incluida IOPS	20
db.influx .16xlarge	64	512	Influencia incluida IOPS	25

Almacenamiento de instancias de InfluxDB

Las instancias de bases de datos de Amazon Timestream para InfluxDB utilizan los volúmenes incluidos de Influx para bases de datos y IOPS almacenamiento de registros.

En algunos casos, es posible que la carga de trabajo de la base de datos no pueda alcanzar el 100 por ciento de la IOPS que ha aprovisionado. Para obtener más información, consulte Factores que afectan al rendimiento del almacenamiento. Para obtener más información sobre los precios de almacenamiento de Timestream for InfluxDB, consulte los precios de Amazon Timestream.

Tipos de almacenamiento de Amazon Timestream para InfluxDB

Amazon Timestream para InfluxDB admite un tipo de almacenamiento, Influx incluido. IOPS Puede crear Timestream para instancias de InfluxDB con hasta 16 tebibytes (TiB) de almacenamiento.

Esta es una breve descripción del tipo de almacenamiento disponible:

- Almacenamiento incluido Influx IO: el rendimiento del almacenamiento es la combinación de las operaciones de E/S por segundo (IOPS) y la rapidez con la que el volumen de almacenamiento puede realizar lecturas y escrituras (rendimiento del almacenamiento). En los volúmenes de almacenamiento IOPS incluidos en Influx, Amazon Timestream para InfluxDB ofrece 3 niveles de almacenamiento preconfigurados con el rendimiento IOPS óptimo necesario para diferentes tipos de cargas de trabajo.

Tamaño de las instancias de InfluxDB

La configuración óptima de una instancia de Timestream para InfluxDB depende de muchos factores, entre los que se incluyen la tasa de ingesta, el tamaño de los lotes, la cardinalidad de las

series temporales y las consultas simultáneas y los tipos de consultas. En un esfuerzo por ofrecer recomendaciones de tamaño, nos centramos en una carga de trabajo ejemplar con las siguientes características:

- Los datos los recopila y escribe una flota de agentes de Telegraf que recopilan el sistema, la memoriaCPU, el disco, las E/S, etc. de un centro de datos.

  Cada solicitud de escritura contiene 5000 líneas.
- El tipo de consultas que se ejecutan en el sistema se clasifican como consultas de «complejidad moderada». Esta categoría de consultas presenta las siguientes características:
  - Tienen varias funciones y una o dos expresiones regulares
  - También se puede agrupar por cláusulas o muestrear un intervalo de tiempo de varias semanas.
  - Normalmente tarda de unos cientos de milisegundos a un par de miles de milisegundos en ejecutarse.
  - CPUfavorece principalmente el rendimiento de las consultas.

Clase de instancia	Storage Type	Escribe (líneas por segundo)	Lecturas (consultas por segundo)
db.influx.large	Influx IO incluye 3K	~50 000	<10
db.influx.2xlarge	Influence IO incluido, 3K	~150.000	<25
db.influx.4xlarge	Influence IO incluido (3K)	~200.000	~25
db.influx.4xlarge	Influence IO incluido (12 K)	~250.000	~35
db.influx.8x large	Influence IO incluido: 12 K	~ 500 000	~50
db.influx. 12 x grande	Entrada IP incluida (12 K)	<750.000	<100

AWS Regiones y zonas de disponibilidad

Los recursos de informática en la nube de Amazon están alojados en varias ubicaciones de todo el mundo. Estas ubicaciones se componen de AWS regiones y zonas de disponibilidad. Cada AWS región es un área geográfica independiente. Cada AWS región tiene varias ubicaciones aisladas conocidas como zonas de disponibilidad.

> **Note**
>
> Para obtener información sobre cómo encontrar las zonas de disponibilidad de una AWS región, consulta Regiones y zonas en la Guía del EC2 usuario de Amazon.

Amazon Timestream para InfluxDB le permite colocar recursos, como instancias de bases de datos y datos, en varias ubicaciones.

Amazon opera centros state-of-the-art de datos de alta disponibilidad. Aunque es infrecuente, puede suceder que se produzcan errores que afecten a la disponibilidad de las instancias de bases de datos que están en la misma ubicación. Si aloja todas sus instancias de base de datos en una ubicación que se ve afectada por dicho error, ninguna de sus instancias de base de datos estará disponible.

Es importante recordar que cada AWS región es completamente independiente. Cualquier actividad de Amazon Timestream for InfluxDB que inicie (por ejemplo, crear instancias de bases de datos o enumerar las instancias de bases de datos disponibles) se ejecuta solo en su región predeterminada actual. AWS La Región de AWS por defecto se puede cambiar en la consola, o estableciendo la variable de entorno `AWS_DEFAULT_REGION`. O bien, se puede anular mediante el uso del parámetro con la `--region` tecla (). AWS Command Line Interface AWS CLI Para obtener más información, consulte Configuración de AWS Command Line Interface, específicamente, las secciones sobre variables de entorno y opciones de línea de comandos.

Para crear o trabajar con una instancia de base de datos de Amazon Timestream for InfluxDB en una región AWS específica, utilice el punto de enlace del servicio regional correspondiente.

AWS Disponibilidad regional

Para obtener más información sobre AWS las regiones en las que Amazon Timestream para InfluxDB está disponible actualmente y el punto de enlace de cada región, consulte los puntos de enlace y las cuotas de Amazon Timestream.

AWS Diseño de regiones

Cada AWS región está diseñada para estar aislada de las demás AWS regiones. Este diseño logra la mayor tolerancia a errores y estabilidad posibles.

Cuando consulta sus recursos, solo ve los recursos que están vinculados a la AWS región que especificó. Esto se debe a que AWS las regiones están aisladas unas de otras y no replicamos automáticamente los recursos entre AWS ellas.

AWS Zonas de disponibilidad

Al crear una instancia de base de datos, Amazon Timestream for InfluxDB elige una de forma aleatoria en función de la configuración de subred. Una zona de disponibilidad se representa mediante un código de AWS región seguido de una letra identificadora (por ejemplo,). `us-east-1a`

Use el EC2 comando `describe-availability` -zones Amazon de la siguiente manera para describir las zonas de disponibilidad de la región especificada que están habilitadas para su cuenta.

```
aws ec2 describe-availability-zones --region region-name
```

Por ejemplo, para describir las zonas de disponibilidad de la región EE.UU. Este (Virginia del Norte) (us-east-1) que están habilitadas para su cuenta, ejecute el siguiente comando:

```
aws ec2 describe-availability-zones --region us-east-1
```

No puede elegir las zonas de disponibilidad para las instancias de base de datos principal y secundaria en una implementación de base de datos Multi-AZ. Amazon Timesteram para InfluxDB los elige aleatoriamente. Para obtener más información sobre las implementaciones Multi-AZ, consulte.. Configuración y administración de una implementación Multi-AZ

Facturación de instancias de base de datos para Amazon Timestream para InfluxDB

Las instancias de Amazon Timestream for InfluxDB se facturan en función de los siguientes componentes:

- Horas de la instancia de base de datos (por hora): según la clase de instancia de base de datos de la instancia de base de datos, por ejemplo, db.influx.large. Los precios se muestran por hora,

pero las facturas se ajustan hasta el segundo y muestran las horas en formato decimal. El uso de Amazon Timestream para InfluxDB se factura en incrementos de 1 segundo, con un mínimo de 10 minutos. Para obtener más información, consulte las clases de instancias de base de datos. Clases de instancia de base de datos

- Almacenamiento (por GiB al mes): capacidad de almacenamiento que ha aprovisionado para su instancia de base de datos. Para obtener más información, consulte Almacenamiento de instancias de InfluxDB.
- Transferencia de datos (por GB): transferencia de datos de entrada y salida de la instancia de base de datos desde o hacia Internet y otras AWS regiones.

Para obtener información sobre los precios de Amazon Timestream for InfluxDB, consulte la página de precios de Amazon Timestream for InfluxDB.

Configuración de Amazon Timestream para InfluxDB

Antes de usar Amazon Timestream para InfluxDB por primera vez, complete las siguientes tareas:

Si ya tiene una AWS cuenta, conozca sus requisitos de Amazon Timestream for InfluxDB y prefiera usar los valores predeterminados para IAM VPC Cómo empezar con Timestream para InfluxDB Primeros pasos con Amazon Timestream for InfluxDB.

Regístrese para obtener una cuenta AWS

Si no tiene una AWS cuenta, complete los siguientes pasos para crear una.

Para crear una AWS cuenta

- Ve a la página de inicio de AWS sesión.
- Selecciona Crear una cuenta nueva y sigue las instrucciones.

> **Note**
>
> Parte del procedimiento de registro consiste en recibir una llamada telefónica e indicar un código de verificación en el teclado del teléfono.

Al crear una AWS cuenta, se crea un usuario raíz de la AWS cuenta. El usuario raíz tiene acceso a todos los AWS servicios y recursos de la cuenta. Como práctica recomendada de seguridad, asigne

acceso administrativo a un usuario administrativo y utilice únicamente el usuario raíz para realizar tareas que requieran acceso de usuario raíz.

AWS le envía un correo electrónico de confirmación una vez finalizado el proceso de registro. En cualquier momento, puede ver la actividad de su cuenta actual y administrarla accediendo a https://aws.amazon.com/y seleccionando Mi cuenta.

Administración de usuarios

Crear un usuario administrativo

Creación de un usuario administrativo

Después de crear una AWS cuenta, cree un usuario administrativo para no utilizar el usuario root en las tareas diarias.

Proteja el usuario raíz de su AWS cuenta

Inicie sesión en la consola AWS de administración como propietario de la cuenta; para ello, seleccione el usuario raíz e introduzca la dirección de correo electrónico de su AWS cuenta. En la siguiente página, escriba su contraseña. Para obtener ayuda para iniciar sesión con un usuario root, consulte Iniciar sesión como usuario root en la Guía del usuario de AWS inicio de sesión

Activa la autenticación multifactorial (MFA) para tu usuario root. Para obtener instrucciones, consulte Habilitar un MFA dispositivo virtual para el usuario raíz de su AWS cuenta (consola) en la Guía del IAM usuario.

Otorgue acceso programático

Los usuarios necesitan acceso programático si quieren interactuar con personas AWS ajenas a. AWS Management Console La forma de conceder el acceso programático depende del tipo de usuario que acceda a AWS.

Para conceder a los usuarios acceso mediante programación, elija una de las siguientes opciones:

¿Qué usuario necesita acceso programático?	Para	Mediante
Identidad de la fuerza laboral (usuarios gestionados en IAM Identity Center)	Utilice credenciales temporales para firmar las solicitudes programáticas dirigidas al	Siga las instrucciones de la interfaz que desee utilizar.* Para ello, consulte AWS CLI

¿Qué usuario necesita acceso programático?	Para	Mediante
	AWS CLI AWS SDKs, o AWS APIs.	Configuración del Identity AWS CLI Center para usar AWS IAM en el Guía del usuario de la interfaz de la línea de comandos de AWS .* Para obtener AWS SDKs información sobre las herramientas y AWS APIs, consulte IAMAutenticación en el Centro en el AWSSDKsy la Guía de referencia de herramientas.
IAM	Utilice credenciales temporales para firmar las solicitudes programáticas dirigidas a AWS CLISDKs, yAPIs.	Siga las instrucciones de Uso de credenciales temporales con AWS recursos de la Guía del IAM usuario.

¿Qué usuario necesita acceso programático?	Para	Mediante
IAM	(No se recomienda) Utilice credenciales de larga duración para firmar las solicitudes programáticas dirigidas a AWS CLISDKs, yAPIs.	Siga las instrucciones de la interfaz que desee utilizar. Para ello, consulte AWS CLI Autenticación mediante credenciales de usuario IAM en el AWS Guía del usuario de la interfaz de línea de comandos .Para obtener informaci ón AWS SDKs sobre las herramientas, consulte Autenticarse con credenciales de larga duración en el AWS SDKsy Guía de referenci a de herramientas .Para ver AWS APIs, consulte Administrar las claves de acceso de los usuarios IAM en el IAMGuía del usuario.

Determinar las necesidades

El componente básico de Amazon Timestream for Influx es la instancia de base de datos. En una instancia de base de datos, usted crea sus buckets. Una instancia de base de datos proporciona una dirección de red llamada punto de enlace. Sus aplicaciones usarán este punto de enlace para conectarse a su instancia de base de datos. También accederá a su InfluxUI utilizando este mismo punto final desde su navegador. Al crear una instancia de base de datos, debe especificar detalles

como el almacenamiento, la memoria, el motor y la versión de la base de datos, la configuración de la red y la seguridad. Controla el acceso de red a una instancia de base de datos mediante un grupo de seguridad.

Antes de crear una instancia de base de datos y un grupo de seguridad, debe conocer su instancia de base de datos y las necesidades de la red. Aquí se indican algunos aspectos importantes que se deben tener en cuenta:

- Requisitos de recursos: ¿Cuáles son los requisitos de memoria y procesador para su aplicación o servicio? Use esta configuración como ayuda para determinar la clase de instancia de base de datos que se usará. Para ver las especificaciones sobre las clases de instancias de base de datos, consulte Clases de instancias de base de datos.
- VPCy grupo de seguridad: lo más probable es que su instancia de base de datos esté en una nube privada virtual (VPC). Para conectarse a su instancia de base de datos, debe configurar reglas de grupo de seguridad. Estas reglas se configuran de forma diferente en función del tipo de VPC dispositivo que utilice y de cómo lo utilice. Por ejemplo, puede usar: una predeterminada VPC o una definida por el usuarioVPC.

  En la siguiente lista se describen las reglas de cada VPC opción:

  - VPCPredeterminada: si su AWS cuenta tiene una configuración predeterminada VPC en la AWS región actual, VPC está configurada para admitir instancias de base de datos. Si especifica el valor predeterminado VPC al crear la instancia de base de datos, asegúrese de crear un grupo de VPC seguridad que autorice las conexiones desde la aplicación o el servicio a la instancia de base de datos Amazon Timestream for InfluxDB. Utilice la opción Grupo de seguridad de la VPC consola o para crear grupos de AWS CLI seguridad. VPC Para obtener más información, consulte el paso 3: Crear un grupo VPC de seguridad.
- Definida por el usuario VPC: si desea especificar una definida por el usuario VPC al crear una instancia de base de datos, tenga en cuenta lo siguiente:
  - Asegúrese de crear un grupo de VPC seguridad que autorice las conexiones desde la aplicación o el servicio a la instancia de base de datos Amazon Timestream for InfluxDB. Utilice la opción Grupo de seguridad de la VPC consola o para crear grupos de AWS CLI seguridad. VPC Para obtener más información, consulte el paso 3: Crear un grupo VPC de seguridad.
  - VPCDeben cumplir ciertos requisitos para alojar instancias de base de datos, como tener al menos dos subredes, cada una en una zona de disponibilidad independiente. Para obtener más información, consulte Amazon VPC VPCs y Amazon Timestream para InfluxDB.

- Alta disponibilidad: ¿necesita soporte de conmutación por error? En Amazon Timestream para InfluxDB, una implementación Multi-AZ crea una instancia de base de datos principal y una instancia de base de datos secundaria en espera en otra zona de disponibilidad para admitir la conmutación por error. Es recomendable usar implementaciones Multi-AZ para las cargas de trabajo de producción con el objeto de mantener una alta disponibilidad. Para fines de desarrollo y de pruebas, puede utilizar una implementación no Multi-AZ. Para obtener más información, consulte Implementaciones de instancias de base de datos Multi-AZ.
- IAMpolíticas: ¿Tiene su AWS cuenta políticas que otorgan los permisos necesarios para realizar las operaciones de Amazon Timestream for InfluxDB? Si se conecta AWS mediante IAM credenciales, su IAM cuenta debe tener IAM políticas que concedan los permisos necesarios para realizar las operaciones del plano de control de Amazon Timestream for InfluxDB. Para obtener más información, consulte Identity and Access Management para Amazon Timestream para InfluxDB.
- Puertos abiertos: ¿qué TCP puerto/IP escucha su base de datos? Los firewall de algunas empresas podrían bloquear las conexiones al puerto predeterminado para el motor de base de datos. El valor predeterminado de Timestream para InfluxDB es 8086.
- AWS Región: ¿en qué AWS región desea que aparezca su base de datos? Tener la base de datos cerca de la aplicación o el servicio web puede reducir la latencia de la red. Para obtener más información, consulte AWS Regiones y zonas de disponibilidad.
- Subsistema de disco de base de datos: ¿cuáles son sus requisitos de almacenamiento? Amazon Timestream para InfluxDB proporciona tres configuraciones para su tipo de almacenamiento Influx IOPS Included:
  - Influx Io incluye 3k () IOPS SSD
  - Influx Io incluye 12k IOPS () SSD
  - Influx Io incluido (25k) IOPS SSD

  Para obtener más información sobre Amazon Timestream para el almacenamiento de InfluxDB, consulte Amazon Timestream para el almacenamiento de instancias de base de datos InfluxDB. Cuando tenga la información que necesita para crear el grupo de seguridad y la instancia de base de datos, vaya al siguiente paso.

Proporcione acceso a su instancia de base de datos mediante la creación de un grupo de seguridad VPC

VPClos grupos de seguridad proporcionan acceso a las instancias de base de datos en unVPC. Actúan como firewall para la instancia de base de datos asociada, controlan el tráfico entrante y saliente a nivel de instancia de base de datos. Las instancias de base de datos se crean de manera predeterminada con un firewall y un grupo de seguridad predeterminado que protege la instancia de base de datos.

Para poder conectarse a la instancia de base de datos, debe agregar reglas a un grupo de seguridad que permitan conectarse. Use la información de red y de configuración para crear reglas que permitan el acceso a la instancia de base de datos.

Por ejemplo, supongamos que tiene una aplicación que accede a una base de datos de su instancia de base de datos en unVPC. En este caso, debe añadir una TCP regla personalizada que especifique el rango de puertos y las direcciones IP que utiliza la aplicación para acceder a la base de datos. Si tienes una aplicación en una EC2 instancia de Amazon, puedes usar el grupo de seguridad que configuraste para la EC2 instancia de Amazon.

Crear un grupo de seguridad para el VPC acceso

Para crear un grupo de VPC seguridad, inicie sesión en AWS Management Console y seleccione VPC.

> **Note**
>
> Asegúrese de estar en la VPC consola, no en la consola Amazon Timesteam for InfluxDB.

- En la esquina superior derecha de AWS Management Console, elija la AWSregión en la que desee crear el grupo de seguridad y la instancia de base de datosVPC. En la lista de VPC recursos de Amazon para esa AWS región, deberías ver al menos una VPC o varias subredes. Si no lo tiene, no tiene un valor predeterminado VPC en esa AWS región. .
- En el panel de navegación, elija Security Groups.
- Seleccione Crear grupo de seguridad.
- En la sección Detalles básicos de la página del grupo de seguridad, introduzca el nombre y la descripción del grupo de seguridad. Para ello VPC, elija la instancia de base de datos en la VPC que desee crear su instancia de base de datos.

- En Inbound rules (Reglas de entrada), elija Add rule (Agregar regla).
  - En Tipo, elija Personalizado TCP.
  - En Fuente, elija un nombre de grupo de seguridad o introduzca el intervalo de direcciones IP (CIDRvalor) desde el que accede a la instancia de base de datos. Si elige My IP (Mi IP), esto permite el acceso a la instancia de base de datos desde la dirección IP detectada en su navegador.

  En Source, elija un nombre de grupo de seguridad o escriba el intervalo de direcciones IP (CIDRvalor) desde el que accede a la instancia de base de datos. Si elige My IP (Mi IP), esto permite el acceso a la instancia de base de datos desde la dirección IP detectada en su navegador.
- (Opcional) En Outbound rules (Reglas de salida), agregue reglas para el tráfico saliente. De forma predeterminada, se permite todo el tráfico de salida.
- Elija Create Security Group (Crear grupo de seguridad).

Puede usar este grupo VPC de seguridad como grupo de seguridad para su instancia de base de datos al crearla.

> **Note**
>
> Si usa uno predeterminado VPC, se creará automáticamente un grupo de subredes predeterminado que abarque todas las subredes VPC de la misma. Al crear una instancia de base de datos, puede elegir el eiifccntf predeterminado VPC y elegir el predeterminado para el grupo de subredes de base de datos.

Una vez que haya completado los requisitos de configuración, puede crear una instancia de base de datos con sus requisitos y grupo de seguridad. Para ello, siga las instrucciones que se indican en Creación de una instancia de base de datos.

Cómo empezar con Timestream para InfluxDB

En los ejemplos siguientes, puede obtener información sobre cómo crear una instancia de base de datos y conectarse a ella mediante Amazon Timestream for InfluxDB Service.

> **Note**
>
> Debe completar las tareas que aparecen en Configuración de Amazon Timestream para InfluxDB antes de crear una instancia de base de datos o conectarse a ella.

Temas

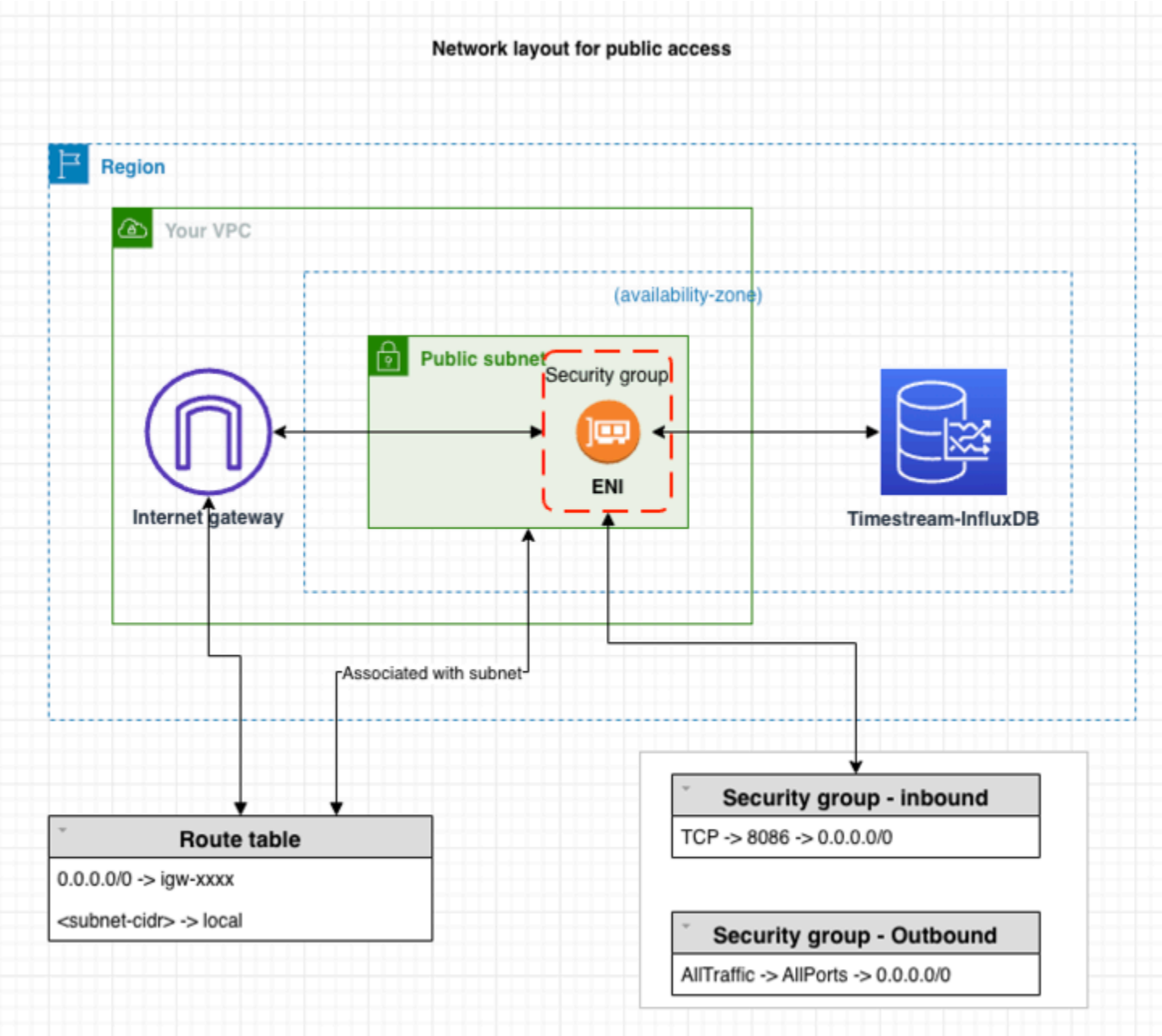
Crear una instancia de Timestream for InfluxDB y conectarse a ella

Este tutorial crea una instancia de Amazon y una EC2 instancia de base de datos Amazon Timestream for InfluxDB. El tutorial muestra cómo escribir datos en la instancia de base de datos desde la EC2 instancia mediante el cliente Telegraf. Como práctica recomendada, en este tutorial se crea una instancia de base de datos privada en una nube privada virtual ()VPC. En la mayoría de los casos, otros recursos de la misma instanciaVPC, como EC2 las instancias, pueden acceder a la instancia de base de datos, pero los recursos ajenos a ella no VPC pueden acceder a ella.

Tras completar el tutorial, habrá una subred pública y una privada en cada zona de disponibilidad de la suya. VPC En una zona de disponibilidad, la EC2 instancia está en la subred pública y la instancia de base de datos en la subred privada.

> **Note**
>
> La creación de una AWS cuenta no conlleva ningún cargo. Sin embargo, al completar este tutorial, es posible que incurra en costos por los AWS recursos que utilice. Puede eliminar estos recursos después de completar el tutorial si ya no son necesarios.

El siguiente diagrama muestra la configuración cuando la accesibilidad es pública.



Warning

No recomendamos usar 0.0.0.0/0 para el HTTP acceso, ya que permite que todas las direcciones IP accedan a su instancia pública de InfluxDB a través de ella. HTTP Este enfoque ni siquiera es aceptable durante un breve período de tiempo en un entorno de prueba. Autorice solo una dirección IP específica o un rango de direcciones para acceder a sus instancias de InfluxDB HTTP utilizando la WebUI o el acceso. API

En este tutorial se crea una instancia de base de datos que ejecuta InfluxDB con. AWS Management Console Nos centraremos únicamente en el tamaño de la instancia de base de datos y en el identificador de la instancia de base de datos. Usaremos la configuración predeterminada para las demás opciones de configuración. La instancia de base de datos creada con este ejemplo será privada.

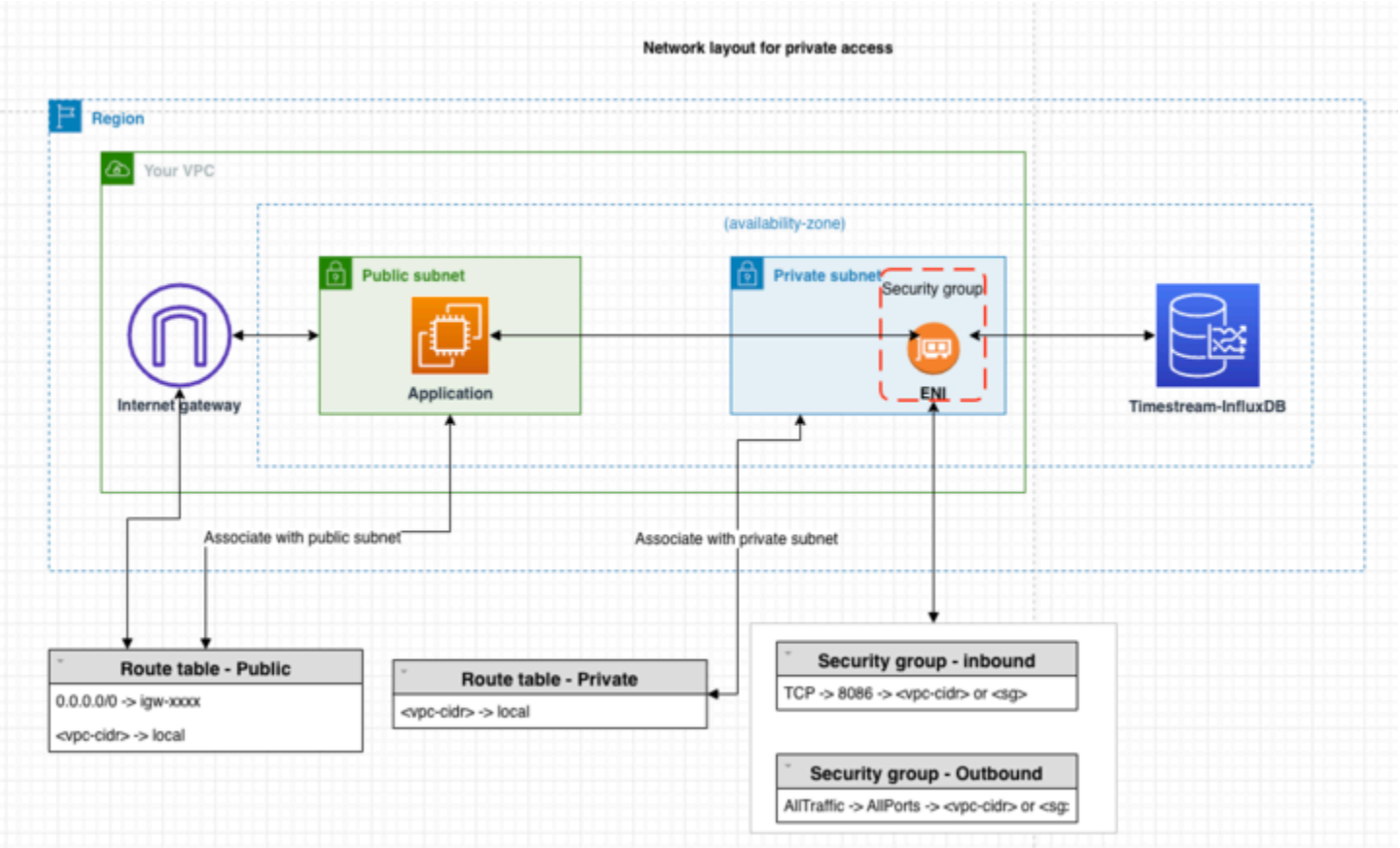
Otros ajustes que puede configurar incluyen la disponibilidad, la seguridad y el registro. Para crear una instancia de base de datos pública, debe elegir que su instancia sea «de acceso público» en la sección de configuración de conectividad. Para obtener información sobre la creación de instancias de base de datos, consulteCreación de una instancia de base de datos..

Si su instancia no es de acceso público, haga lo siguiente:

- Cree un host en la instancia a través VPC del cual pueda canalizar el tráfico.
- Configura la tunelización ssh hacia la instancia. Para obtener más información, consulte Reenvío de puertos de EC2 instancias de Amazon con AWS Systems Manager
- Para que el certificado funcione, agregue la siguiente línea al `/etc/hosts` archivo de su máquina cliente:`127.0.0.1`. Esta es la DNS dirección de su instancia.
- Conéctate a tu instancia con el nombre de dominio completo, por ejemplo, https: //< DNS >:8086.

> **Note**
>
> Localhost no puede validar el certificado porque localhost no forma parte del certificado. SAN

El siguiente diagrama muestra la configuración cuando la accesibilidad es privada:



Requisitos previos

Antes de empezar, complete los pasos de las siguientes secciones:

- Regístrese para obtener una AWS cuenta.
- Crear un usuario administrativo.

Paso 1: Crear una EC2 instancia de Amazon

Crea una EC2 instancia de Amazon que utilizarás para conectarte a tu base de datos.

1. Inicia sesión en la EC2 consola de Amazon AWS Management Console y ábrela en https://console.aws.amazon.com/ec2/.
2. En la esquina superior derecha de AWS Management Console, selecciona la AWS región en la que quieres crear la EC2 instancia.
3. Selecciona EC2Dashboard y, a continuación, selecciona Launch instance.

4. Cuando se abra la página Iniciar una instancia, seleccione los siguientes ajustes en la página Iniciar una instancia.

   a. En Nombre y etiquetas, en Nombre, escriba ec2-database-connect.

   b. En Imágenes de aplicaciones y sistemas operativos (Amazon Machine Image), selecciona Amazon Linux y, a continuación, Amazon Linux 2023AMI. Mantenga los valores predeterminados para las demás opciones.

   c. En Instance type (Tipo de instancia), elija t2.micro.

   d. En Key pair (login) [Par de claves (inicio)], elija Key pair name (Nombre de par de claves) para utilizar un par de claves existente. Para crear un nuevo par de claves para la EC2 instancia de Amazon, selecciona Crear nuevo par de claves y, a continuación, utiliza la ventana Crear par de claves para crearlo. Para obtener más información sobre la creación de un nuevo par de claves, consulte Crear un par de claves en la Guía del EC2 usuario de Amazon para instancias de Linux.

   e. En Permitir SSH el tráfico en la configuración de red, selecciona el origen de SSH las conexiones a la EC2 instancia. Puede elegir Mi IP si la dirección IP que se muestra es correcta para SSH las conexiones. De lo contrario, puede determinar la dirección IP que se utilizará para conectarse a EC2 las instancias si VPC utiliza Secure Shell (SSH). Para determinar su dirección IP pública, en otra ventana o pestaña del navegador, puede utilizar el servicio en https://checkip.amazonaws.com. Un ejemplo de dirección IP es 192.0.2.1/32. En muchos casos, puedes conectarte a través de un proveedor de servicios de Internet (ISP) o desde detrás del firewall sin una dirección IP estática. Si es así, asegúrese de identificar el rango de direcciones IP que utilizan los equipos cliente.

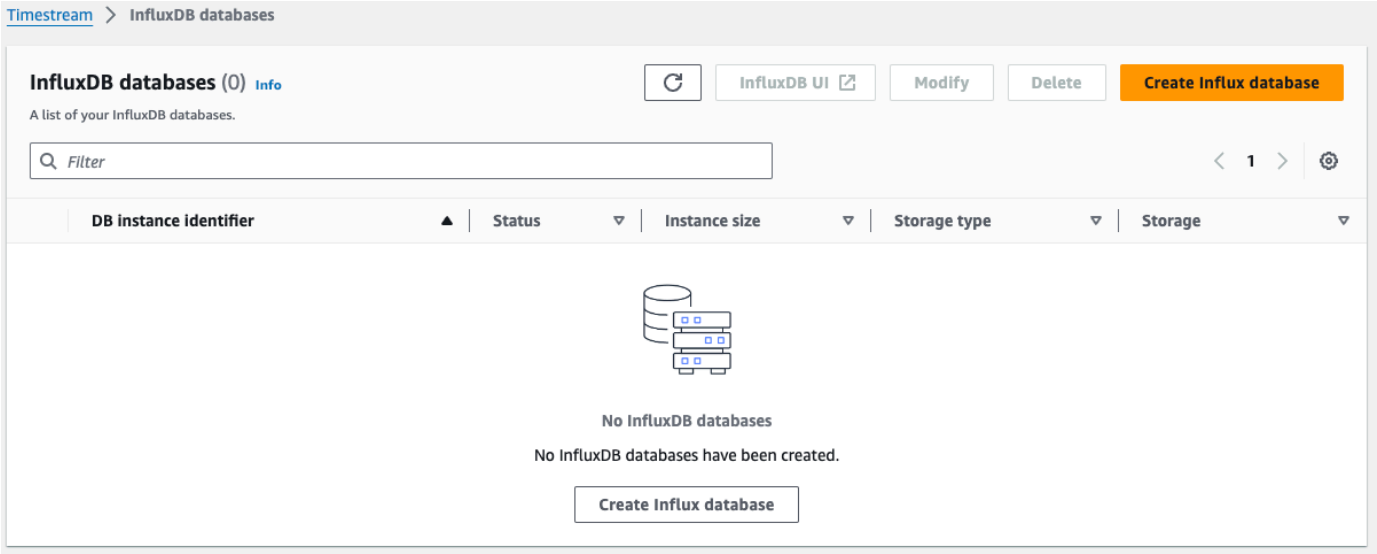
      > **Warning**
      >
      > No recomendamos usar 0.0.0.0/0 para el SSH acceso, ya que permite que todas las direcciones IP accedan a sus instancias públicas mediante. EC2 SSH Este enfoque ni siquiera es aceptable durante un breve período de tiempo en un entorno de prueba; autorice solo una dirección IP específica o un rango de direcciones para acceder a sus instancias. EC2 SSH

Paso 2: Cree una instancia de base de datos de InfluxDB

El componente básico de Amazon Timestream para InfluxDB es la instancia de base de datos. En este entorno se ejecutan las bases de datos de InfluxDB.

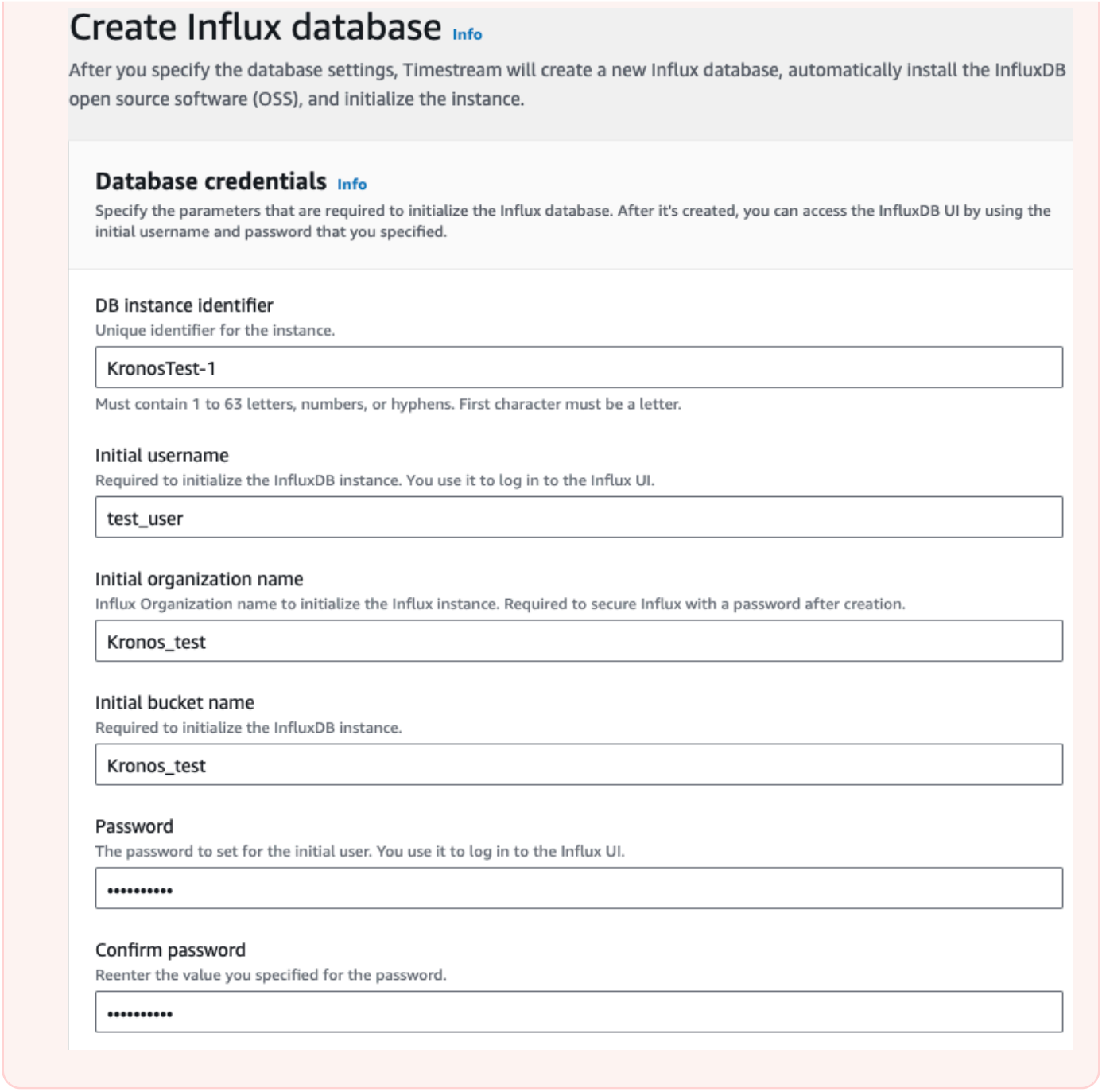
En este ejemplo, creará una instancia de base de datos que ejecute el motor de base de datos InfluxDB con una clase de instancia de base de datos db.influx.large.

1. Inicie sesión en la consola Amazon Timestream for InfluxDB AWS Management Console y ábrala en. https://console.aws.amazon.com/timestream/
2. En la esquina superior derecha de la consola Amazon Timestream for InfluxDB, elija la región en la que desee crear AWS la instancia de base de datos.
3. En el panel de navegación, elija InfluxDB Databases.
4. Seleccione Crear base de datos de Influx.

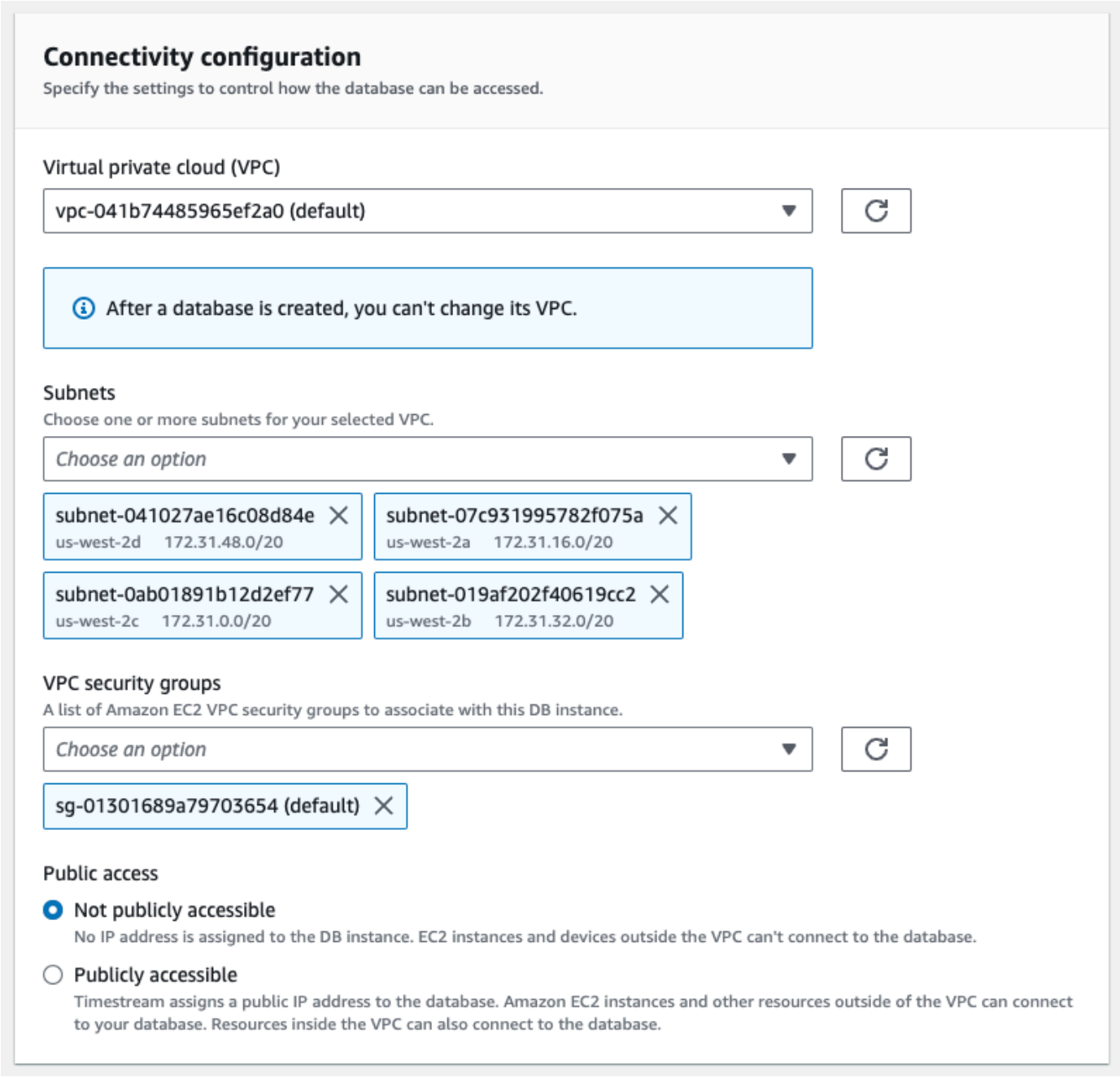


5. En Identificador de instancia de base de datos, introduzca KronosTest -1.
6. Proporcione los parámetros de configuración básicos de InfluxDB: nombre de usuario, organización, nombre del bucket y contraseña.

> **Important**
>
> No podrá volver a ver la contraseña de usuario. No podrás acceder a tu instancia ni obtener un token de operador sin tu contraseña. Si no la registra, es posible que tenga que cambiarla. Consulte Crear un nuevo token de operador para su instancia de InfluxDB.
>
> Si necesita cambiar la contraseña de usuario una vez que la instancia de base de datos esté disponible, puede modificarla para hacerlo. Para obtener más información acerca de la modificación de una instancia de base de datos, consulte Actualización de instancias de base de datos.



7. En Clase de instancia de base de datos, seleccione db.influx.large.
8. Para la clase de almacenamiento de base de datos, seleccione flux Included 3K. IOPS
9. Configure sus registros. Para obtener más información, consulte Configuración para ver los registros de InfluxDB en las instancias de Timestream Influxdb.
10. En la sección de configuración de conectividad, asegúrate de que tu instancia de InfluxDB esté en la misma subred que la instancia recién creada. EC2



11. Elija Crear base de datos de Influx.
12. En la lista de bases de datos, elija el nombre de la nueva instancia de InfluxDB para mostrar sus detalles. La instancia de base de datos tiene el estado de Creación hasta que esté lista para usarse.

Puede conectarse a la instancia de base de datos cuando el estado cambie a Disponible. Dependiendo de la clase de instancia de la base de datos y de la cantidad de almacenamiento, es posible que la nueva instancia tarde hasta 20 minutos en estar disponible.

> **Important**
>
> En este momento, no puede modificar la configuración de procesamiento (tipos de instancia) ni de almacenamiento (tipos de almacenamiento) de las instancias existentes.

Paso 3: envíe los datos de Telegraf a su instancia de InfluxDB

Ahora puede empezar a enviar datos de telemetría a su instancia de base de datos de InfluxDB mediante el agente de Telegraf. En este ejemplo, instalará y configurará un agente de Telegraf para enviar las métricas de rendimiento a su instancia de base de datos de InfluxDB.

1. Busque el punto final (DNSnombre) y el número de puerto de su instancia de base de datos.

   a. Inicie sesión en la consola AWS de administración y abra la consola de Amazon Timestream en. https://console.aws.amazon.com/timestream/

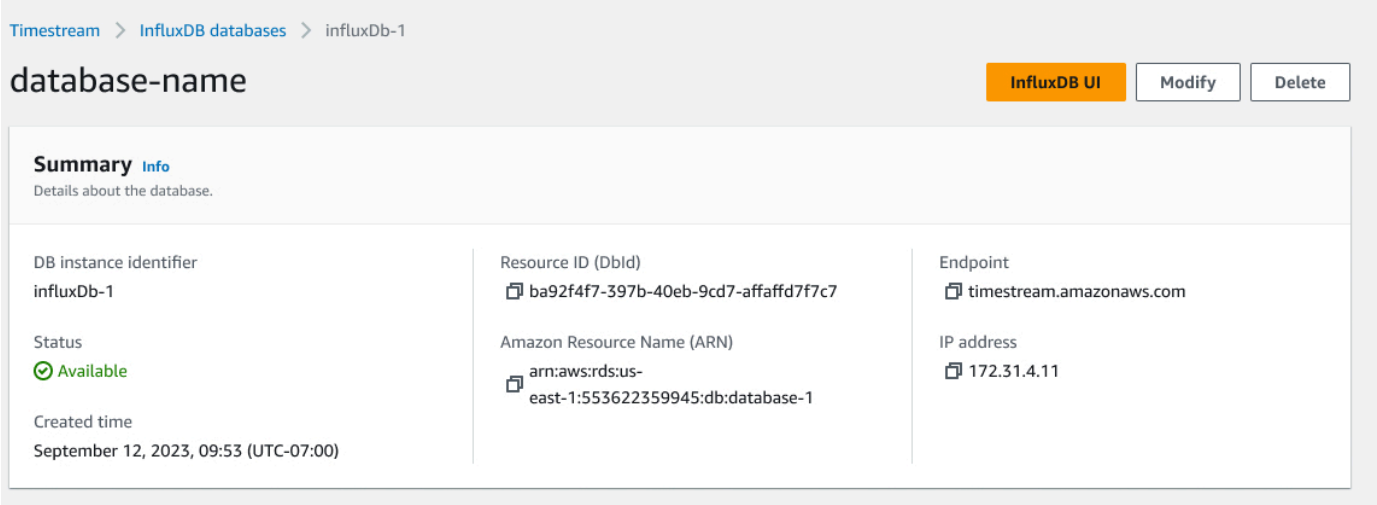
   b. En la esquina superior derecha de la consola de Amazon Timestream, elija la región de AWS la instancia de base de datos.

   c. En el panel de navegación, elija InfluxDB Databases.

   d. Elija el nombre de la instancia de base de datos de InfluxDB para mostrar sus detalles.

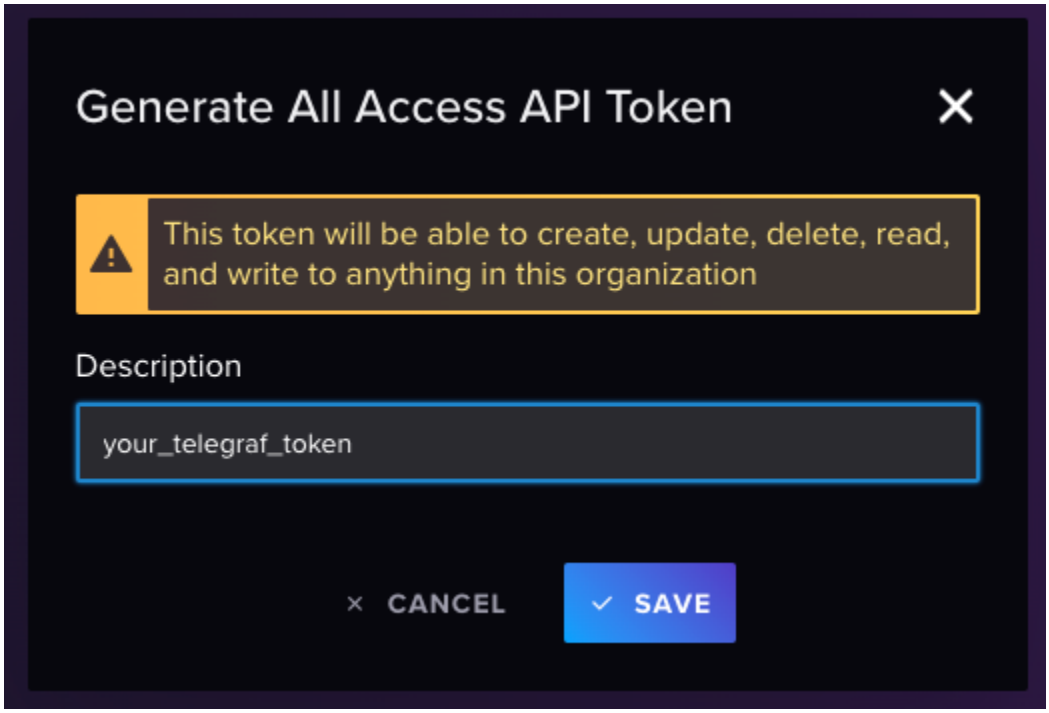
   e. En la sección Resumen, copie el punto final. También anote el número de puerto. Necesita tanto el punto final como el número de puerto para conectarse a la instancia de base de datos (el número de puerto predeterminado para InfluxDB es 8086).

2. A continuación, seleccione InfluxDB UI.



3. Esto abrirá una nueva ventana del navegador en la que debería ver un mensaje de inicio de sesión. Introduzca las credenciales que utilizó anteriormente para crear su instancia de base de datos de InfluxDB.
4. En el panel de navegación, haga clic en la flecha y seleccione API Tokens.
5. Para esta prueba, genere un token de acceso total.

> **Note**
>
> Para escenarios de producción, recomendamos crear tokens con acceso específico a los depósitos necesarios y que estén diseñados para las necesidades específicas de Telegraf.



6. Tu ficha aparecerá en la pantalla.

> **Important**
>
> Asegúrate de copiar y guardar el token, ya que no podrás volver a mostrarlo.

7. Conéctate a la EC2 instancia que creaste anteriormente siguiendo los pasos de Conéctate a tu instancia de Linux de la Guía del EC2 usuario de Amazon para instancias de Linux.

Te recomendamos que te conectes a tu EC2 instancia medianteSSH. Si la utilidad de SSH cliente está instalada en Windows, Linux o Mac, puedes conectarte a la instancia mediante el siguiente formato de comando:

```
ssh -i location_of_pem_file ec2-user@ec2-instance-public-dns-name
```

Por ejemplo, supongamos que `ec2-database-connect-key-pair.pem` está almacenado `/dir1` en Linux y que el público IPv4 DNS de la EC2 instancia sí lo está`ec2-12-345-678-90.compute-1.amazonaws.com`. El SSH comando tendría el siguiente aspecto:

```
ssh -i /dir1/ec2-database-connect-key-pair.pem ec2-
user@ec2-12-345-678-90.compute-1.amazonaws.com
```

8. Instale la última versión de telegraf en su instancia. Para ello, usa el siguiente comando:

```
cat <<EOF | sudo tee /etc/yum.repos.d/influxdata.repo
[influxdata]
name = InfluxData Repository - Stable
baseurl = https://repos.influxdata.com/stable/\$basearch/main
enabled = 1
gpgcheck = 1
gpgkey = https://repos.influxdata.com/influxdata-archive_compat.key
EOF

sudo yum install telegraf
```

9. Configura tu instancia de Telegraf.

> **Note**
>
> Si telegraf.conf no existe o contiene una `timestream` sección, puedes generar una con:
>
> ```
> telegraf –section-filter agent:inputs:outputs –input-filter cpu:mem –output-
> filter timestream config > telegraf.conf
> ```

a. Edite el archivo de configuración que normalmente se encuentra en. `/etc/telegraf`

```
sudo nano /etc/telegraf/telegraf.conf
```

b. Configure las entradas básicas paraCPU, MEM yDISK.

```
[[inputs.cpu]]
  percpu = true
  totalcpu = true
  collect_cpu_time = false
  report_active = false

[[inputs.mem]]

[[inputs.disk]]
  ignore_fs = ["tmpfs", "devtmpfs", "devfs"]
```

c. Configure el complemento de salida para enviar datos a su instancia de base de datos de InfluxDB y guardar los cambios.

```
[[outputs.influxdb_v2]]
   urls = ["https://us-west-2-1.aws.cloud2.influxdata.com"]
   token = "<your_telegraf_token"
   organization = "your_org"
   bucket = "your_bucket"
   timeout = "5s"
```

d. Configure el objetivo de Timestream.

```
# Configuration for sending metrics to Amazon Timestream.
[[outputs.timestream]]

  ## Amazon Region and credentials
  region = "us-east-1"
  access_key = "<AWS key here>"
  secret_key = "<AWS secret key here>"
  database_name = "<timestream database name>" # needs to exist

  ## Specifies if the plugin should describe t start.
  describe_database_on_start = false
  mapping_mode = "multi-table" # allows multible tables for each input metrics

  create_table_if_not_exists = true
  create_table_magnetic_store_retention_period_in_days = 365
```

```
  create_table_memory_store_retention_period_in_hours = 24

  use_multi_measure_records = true # Important to use multi-measure records
  measure_name_for_multi_measure_records = "telegraf_measure"
  max_write_go_routines = 25
```

10. Habilite e inicie el servicio Telegraf.

```
$ sudo systemctl enable telegraf
$ sudo systemctl start telegraf
```

Paso 4: Eliminar la instancia de Amazon y la EC2 instancia de base de datos de InfluxDB

Después de explorar los datos generados por Telegraph utilizando su instancia de base de datos de InfluxDB con la interfaz de usuario de InfluxUI, elimine tanto su EC2 instancia de base de datos como la de InfluxDB para que no se le cobre más por ellas.

Para eliminar la instancia: EC2

1. Inicia sesión en la EC2 consola de Amazon AWS Management Console y ábrela en https://console.aws.amazon.com/ec2/.
2. En el panel de navegación, seleccione Instances (Instancia[s]).
3. Selecciona la EC2 instancia, elige el estado de la instancia y finaliza la instancia.
4. Cuando se le indique que confirme, elija Terminar.

Para obtener más información sobre cómo eliminar una EC2 instancia, consulta Terminar tu instancia en la Guía del EC2 usuario de Amazon.

Para eliminar la instancia de base de datos sin una instantánea de base de datos final:

1. Inicie sesión en la consola Amazon Timestream for InfluxDB AWS Management Console y ábrala en. https://console.aws.amazon.com/timestream/
2. En el panel de navegación, elija InfluxDB Databases.
3. Elija la instancia de base de datos que desea eliminar.
4. En Actions (Acciones), elija Delete (Eliminar).
5. Complete la confirmación y seleccione Eliminar.

(Opcional) Conéctese a su instancia de base de datos mediante Amazon Managed Grafana

Puede utilizar Amazon Managed Grafana para crear cuadros de mando y supervisar el rendimiento de las EC2 instancias mediante Amazon Timestream para InfluxDB. Amazon Managed Grafana es un servicio totalmente gestionado para Grafana, una popular plataforma de análisis de código abierto que le permite consultar, visualizar y emitir alertas sobre sus métricas, registros y seguimientos.

Crear un nuevo token de operador para su instancia de InfluxDB

Si necesita obtener el token de operador para su nueva instancia de InfluxDB, lleve a cabo los siguientes pasos:

1. Para cambiar tu token de operador, te recomendamos que utilices Influx. CLI Para obtener instrucciones, consulta: Instalar y usar el flux CLI.
2. Configure CLI el `--username-password` suyo para poder crear el operador:

```
influx config create --config-name CONFIG_NAME1  --host-url "https://
yourinstanceid.eu-central-1.timestream-influxdb.amazonaws.com:8086" --org [YOURORG]
  --username-password [YOURUSERNAME] --active
```

3. Crea tu nuevo token de operador. Se te pedirá tu contraseña para confirmar este paso.

```
influx auth create --org [YOURORG] --operator
```

> **Important**
>
> Una vez que se haya creado un nuevo token de operador, tendrás que actualizar cualquier cliente que esté utilizando el anterior.

Migración de datos de InfluxDB autogestionado a Timestream para InfluxDB

El script de migración de Influx es un script de Python que migra datos entre OSS instancias de InfluxDB, independientemente de que esas instancias estén administradas por AWS o no.

InfluxDB es una base de datos de series temporales. InfluxDB contiene puntos, que contienen varios pares clave-valor y una marca de tiempo. Cuando los puntos se agrupan por pares clave-valor,

forman una serie. Una serie se agrupa mediante un identificador de cadena denominado medición. InfluxDB se utiliza a menudo para la supervisión, IOT los datos y el análisis de las operaciones. Un depósito es un tipo de contenedor dentro de InfluxDB para almacenar datos. AWS-managed InfluxDB es InfluxDB dentro del ecosistema. AWS InfluxDB proporciona InfluxDB v2 API para acceder a los datos y realizar cambios en la base de datos. El InfluxDB v2 API es lo que utiliza el script de migración de Influx para migrar datos.

- El script de migración de Influx puede migrar los buckets y sus metadatos, migrar todos los buckets de todas las organizaciones o realizar una migración completa, que reemplaza todos los datos de la instancia de destino.
- El script hace copias de seguridad de los datos de la instancia de origen de forma local, sea cual sea el sistema que ejecute el script, y, a continuación, restaura los datos en la instancia de destino. Los datos se guardan en los directorios code>influxdb-backup-<timestamp></timestamp>, uno para cada migración.
- El script ofrece una serie de opciones y configuraciones, entre las que se incluyen el montaje de depósitos S3 para limitar el uso del almacenamiento local durante la migración y la elección de las organizaciones que se van a utilizar durante la migración.

Temas

Preparación

La migración de datos para InfluxDB se realiza con un script de Python que utiliza las CLI funciones de InfluxDB y InfluxDB v2. API La ejecución del script de migración requiere la siguiente configuración de entorno:

- Versiones compatibles: se admite una versión mínima de 2.3 de InfluxDB e InfluxCLI.
- Variables de entorno simbólicas
  - Cree la variable de entorno `INFLUX_SRC_TOKEN` que contenga el token de su instancia de InfluxDB de origen.
  - Cree la variable de entorno `INFLUX_DEST_TOKEN` que contenga el token de la instancia de InfluxDB de destino.

- Python 3
  - Compruebe la instalación:`python3 --version`.
  - Si no está instalado, instálelo desde el sitio web de Python. Se requiere la versión 3.7 como mínimo. En Windows, el alias predeterminado de Python 3 es simplemente python.
  - Se requieren las solicitudes del módulo Python. Instálelo con: `shell python3 -m pip install requests`
  - TTheSe requiere el módulo de Python influxdb_client. Instálelo con: `shell python3 -m pip install influxdb_client`
- InfluxDB CLI
  - Confirme la instalación:. `influx version`
  - Si no está instalado, siga la guía de instalación en la documentación de InfluxDB.

    Añada afluencia a sus $. PATH
- Herramientas de montaje S3 (opcionales)

  Cuando se utiliza el montaje S3, todos los archivos de respaldo se almacenan en un bucket S3 definido por el usuario. El montaje en S3 puede resultar útil para ahorrar espacio en la máquina de ejecución o cuando es necesario compartir los archivos de copia de seguridad. Si no se utiliza el montaje en S3, si se omite la `--s3-bucket` opción, se creará un `influxdb-backup-<millisecond timestamp>` directorio local para almacenar los archivos de copia de seguridad en el mismo directorio en el que se ejecutó el script.

  Para Linux: mountpoint-s3.

  Para Windows: rclone (se necesita una configuración previa de rclone).
- Espacio en disco
  - El proceso de migración crea automáticamente directorios únicos para almacenar conjuntos de archivos de respaldo y conserva estos directorios de respaldo en S3 o en el sistema de archivos local, según los argumentos del programa proporcionados.
  - Asegúrese de que haya suficiente espacio en disco para la copia de seguridad de la base de datos; lo ideal sería duplicar el tamaño de la base de datos InfluxDB existente si opta por omitir la `--s3-bucket` opción y utilizar el almacenamiento local para la copia de seguridad y la restauración.
  - Compruebe el espacio con `df -h (UNIX/Linux)` o comprobando las propiedades de la unidad en Windows.

- Conexión directa

  Asegúrese de que exista una conexión de red directa entre el sistema que ejecuta el script de migración y los sistemas de origen y destino. `influx ping --host <host>`es una forma de comprobar una conexión directa.

Cómo usar el script

Un ejemplo sencillo de ejecución del script es el comando:

```
python3 influx_migration.py --src-host <source host> --src-bucket <source bucket> --
dest-host <destination host>
```

El cual migra un único depósito.

Para ver todas las opciones, ejecute:

```
python3 influx_migration.py -h
```

Uso

```
shell   influx_migration.py [-h] [--src-bucket SRC_BUCKET] [--dest-bucket DEST_BUCKET]
 [--src-host SRC_HOST] --dest-host DEST_HOST [--full] [--confirm-full] [--src-org
 SRC_ORG] [--dest-org DEST_ORG] [--csv] [--retry-restore-dir RETRY_RESTORE_DIR] [--dir-
name DIR_NAME] [--log-level LOG_LEVEL] [--skip-verify] [--s3-bucket S3_BUCKET]
```

Opciones

- -confirm-full (opcional): si se utiliza `--full` without, `--csv` se sustituirán todos los símbolos, usuarios, grupos, paneles y cualquier otro dato de valores clave de la base de datos de destino por los símbolos, usuarios, grupos, paneles y cualquier otro dato de valores clave de la base de datos de origen. `--full`con `--csv` solo migra todos los metadatos del bucket y del bucket, incluidas las organizaciones del bucket. Esta opción (`--confirm-full`) confirmará una migración completa y continuará sin intervención del usuario. Si no se proporciona esta opción y se `--full` ha proporcionado y `--csv` no se ha proporcionado, el script se detendrá para ejecutarse y esperará a que el usuario lo confirme. Se trata de una acción crítica, proceda con cautela. El valor predeterminado es falso.
- -csv (opcional): si se deben utilizar archivos csv para realizar copias de seguridad y restaurarlas. Si también `--full` se aprueba, se migrarán todos los grupos definidos por el usuario en todas las

organizaciones, no los grupos del sistema, los usuarios, los tokens o los paneles. Si se desea una organización única para todos los depósitos del servidor de destino en lugar de las organizaciones de origen ya existentes, utilice. `--dest-org`

- -dest-bucket DEST _ BUCKET (opcional): el nombre del depósito de InfluxDB en el servidor de destino no debe ser un depósito ya existente. El valor predeterminado es o si no se proporciona. `--src-bucket` `None` `--src-bucket`
- -dest-host DEST _ HOST: el host del servidor de destino. Ejemplo: http://localhost:8086.
- -dest-org DEST _ ORG (opcional): el nombre de la organización en la que se van a restaurar los depósitos en el servidor de destino. Si se omite, todos los depósitos migrados desde el servidor de origen conservarán su organización original y es posible que los cubos migrados no estén visibles en el servidor de destino sin crear y cambiar de organización. Este valor se utilizará en todas las formas de restauración, ya sea en un solo depósito, en una migración completa o en cualquier migración que utilice archivos csv para la copia de seguridad y la restauración.
- -dir-name DIR _ NAME (opcional): el nombre del directorio de respaldo que se va a crear. El valor predeterminado es `influxdb-backup-<timestamp>`. No debe existir ya.
- -completo (opcional): si se debe realizar una restauración completa, sustituyendo todos los datos del servidor de destino por todos los datos del servidor de origen de todas las organizaciones, incluidos todos los datos con valores clave, como fichas, paneles, usuarios, etc. Anula y. `--src-bucket` `--dest-bucket` Si se usa con`--csv`, solo migra los datos y metadatos de los buckets. El valor predeterminado es falso.
- h, --help: muestra el mensaje de ayuda y lo cierra.
- -log-level LOG _ LEVEL (opcional): el nivel de registro que se utilizará durante la ejecución. Las opciones son depuración, error e información. El valor predeterminado es info.
- - retry-restore-dir RETRY _ RESTORE _ DIR (opcional): el directorio que se usará para la restauración cuando haya fallado una restauración anterior, omitirá la creación de la copia de seguridad y el directorio, fallará si el directorio no existe, puede ser un directorio dentro de un bucket de S3. Si se produce un error en la restauración, la ruta del directorio de respaldo que se puede usar para la restauración se indicará en relación con el directorio actual. Los buckets de S3 tendrán el siguiente formulario`influxdb-backups/<s3 bucket>/<backup directory>`. El nombre predeterminado del directorio de respaldo es`influxdb-backup-<timestamp>`.
- -s3-bucket S3_ BUCKET (opcional): el nombre del bucket de S3 que se usará para almacenar los archivos de respaldo. En Linux, se trata simplemente del nombre del depósito de S3, por ejemplo, si se han `my-bucket` establecido o existen `AWS_ACCESS_KEY_ID` variables de `AWS_SECRET_ACCESS_KEY` entorno determinadas. `${HOME}/.aws/credentials` En Windows, este es el nombre del depósito y el control remotos `rclone` configurados, por ejemplo`my-`

`remote:my-bucket`. Tras la migración a un `influxdb-backups-<timestamp>` directorio creado, todos los archivos de copia de seguridad permanecerán en el depósito de S3. Se creará un directorio de montaje temporal denominado `influx-backups` en el directorio desde el que se ejecuta este script. Si no se proporciona, todos los archivos de copia de seguridad se almacenarán localmente en un `influxdb-backups-<timestamp>` directorio creado desde el que se ejecute este script.

- -skip-verify (opcional): omite TLS la verificación del certificado.
- -src-bucket SRC _ BUCKET (opcional): el nombre del depósito de InfluxDB en el servidor de origen. Si no se proporciona, debe proporcionarse. `--full`
- -src-host SRC _ HOST (opcional): el host del servidor de origen. El valor predeterminado es http://localhost:8086.

Como se indicó anteriormente, `mountpoint-s3` `rclone` son necesarios si `--s3-bucket` se van a utilizar, pero se pueden ignorar si el usuario no proporciona un valor para`--s3-bucket`, en cuyo caso, los archivos de copia de seguridad se almacenarán en un directorio único a nivel local.

Información general sobre la migración

Tras cumplir los requisitos previos:

1. Ejecute el script de migración: con una aplicación de terminal de su elección, ejecute el script de Python para transferir datos de la instancia de InfluxDB de origen a la instancia de InfluxDB de destino.
2. Proporcione credenciales: proporcione direcciones de host y puertos como opciones. CLI
3. Verifique los datos: asegúrese de que los datos se transfieran correctamente de la siguiente manera:

    a. Uso de la interfaz de usuario de InfluxDB e inspección de los cubos.

    b. Listar los cubos con. `influx bucket list -t <destination token> --host <destination host address> --skip-verify`

    c. Se utiliza `influx v1 shell -t <destination token> --host <destination host address> --skip-verify` y ejecuta `SELECT * FROM <migrated bucket>.<retention period>.<measurement name> LIMIT 100 to view contents of a bucket or SELECT COUNT(*) FROM <migrated bucket>.<retention period>.<measurment name>` para comprobar que se ha migrado el número correcto de registros.

Example Ejemplo de ejecución

1. Abre la aplicación de terminal que prefieras y asegúrate de que los requisitos previos necesarios estén correctamente instalados:

```
~ > python3 --version
Python 3.11.5
~ > influx version
Influx CLI 2.7.3 (git: 8b962c7e75) build_date: 2023-04-28T14:22:49Z
~ > s3fs --version
Amazon Simple Storage Service File System V1.92 (commit:unknown) with GnuTLS(gcrypt)
Copyright (C) 2010 Randy Rizun <rrizun@gmail.com>
License GPL2: GNU GPL version 2 <https://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
~ >
```

2. Navegue hasta el script de migración:

```
~ > cd sample-code/influxdb-sample/migration/influxdb
~/sample-code/influxdb-sample/migration/influxdb > ls *.py
influx_migration.py
~/sample-code/influxdb-sample/migration/influxdb >
```

3. Prepare la siguiente información:

   a. Nombre del depósito de origen que se va a migrar.

   b. (Opcional) Elija un nombre nuevo para el depósito migrado en el servidor de destino.

   c. Token raíz para las instancias de afluencia de origen y destino.

   d. Dirección de host de las instancias de afluencia de origen y destino.

   e. (Opcional) Nombre y credenciales del bucket de S3; AWS Command Line Interface las credenciales deben configurarse en las variables de entorno del sistema operativo.

```
# AWS credentials (for timestream testing)
    export AWS_ACCESS_KEY_ID="xxx"
    export AWS_SECRET_ACCESS_KEY="xxx"
```

   f. Construya el comando de la siguiente manera:

```
python3 influx_migration.py --src-bucket [source-bucket-name]  --dest-bucket
 [dest-bucket-name] --src-host [source host] --dest-host [dest host] --s3-
bucket [s3 bucket name](optional) --log-level debug
```

g. Ejecute el script:

```
~/sample-code/influxdb-sample/migration/influxdb > python3 influx_migration.py --src-bucket primary-bucket --src-host $INFLUXDB_1_HOST --dest-host $KRONOS_HOST --dest-bucket new-bucket-name
```

h. Espere a que el script termine de ejecutarse.

i. Compruebe la integridad de los datos del depósito recién migrado,`performance.txt`. Este archivo, ubicado en el mismo directorio en el que se ejecutó el script, contiene información básica sobre la duración de cada paso.

Escenarios de migración

Example Ejemplo 1: migración sencilla mediante almacenamiento local

Desea migrar un único depósito, el depósito principal, del servidor de origen (`http://localhost:8086`) a un servidor de destino. (`http://dest-server-address:8086`)

Después de asegurarse de que tiene TCP acceso (para HTTP acceder) a las dos máquinas que alojan las instancias de InfluxDB en el puerto 8086 y de que dispone de los tokens de origen y de destino y los ha almacenado como variables de entorno `INFLUX_SRC_TOKEN` y`INFLUX_DEST_TOKEN`, respectivamente, para mayor seguridad:

```
python3 influx_migration.py --src-bucket primary-bucket --src-host http://
localhost:8086 --dest-host http://dest-server-address:8086
```

El resultado debería tener un aspecto similar al siguiente:

```
INFO: influx_migration.py: Backing up bucket data and metadata using the InfluxDB CLI
2023/10/26 10:47:15 INFO: Downloading metadata snapshot
2023/10/26 10:47:15 INFO: Backing up TSM for shard 1
2023/10/26 10:47:15 INFO: Backing up TSM for shard 8245
2023/10/26 10:47:15 INFO: Backing up TSM for shard 8263
[More shard backups . . .]
2023/10/26 10:47:20 INFO: Backing up TSM for shard 8240
2023/10/26 10:47:20 INFO: Backing up TSM for shard 8268
2023/10/26 10:47:20 INFO: Backing up TSM for shard 2
INFO: influx_migration.py: Restoring bucket data and metadata using the InfluxDB CLI
2023/10/26 10:47:20 INFO: Restoring bucket "96c11c8876b3c016" as "primary-bucket"
2023/10/26 10:47:21 INFO: Restoring TSM snapshot for shard 12772
2023/10/26 10:47:22 INFO: Restoring TSM snapshot for shard 12773
[More shard restores . . .]
2023/10/26 10:47:28 INFO: Restoring TSM snapshot for shard 12825
```

```
2023/10/26 10:47:28 INFO: Restoring TSM snapshot for shard 12826
INFO: influx_migration.py: Migration complete
```

El directorio se `influxdb-backup-<timestamp>` creará y almacenará en el directorio desde el que se ejecutó el script, que contiene los archivos de respaldo.

Example Ejemplo 2: migración completa mediante almacenamiento local y registro de depuración

Igual que en el caso anterior, excepto que desea migrar todos los depósitos, símbolos, usuarios y paneles, eliminar los depósitos del servidor de destino y continuar sin que el usuario confirme la migración completa de la base de datos mediante esta opción. `--confirm-full` También querrá ver cuáles son las medidas de rendimiento para poder habilitar el registro de depuración.

```
python3 influx_migration.py --full --confirm-full --src-host http://localhost:8086 --
dest-host http://dest-server-address:8086 --log-level debug
```

El resultado debería tener un aspecto similar al siguiente:

```
INFO: influx_migration.py: Backing up bucket data and metadata using the InfluxDB CLI
2023/10/26 10:55:27 INFO: Downloading metadata snapshot
2023/10/26 10:55:27 INFO: Backing up TSM for shard 6952
2023/10/26 10:55:27 INFO: Backing up TSM for shard 6953
[More shard backups . . .]
2023/10/26 10:55:36 INFO: Backing up TSM for shard 8268
2023/10/26 10:55:36 INFO: Backing up TSM for shard 2
DEBUG: influx_migration.py: backup started at 2023-10-26 10:55:27 and took 9.41 seconds
 to run.
INFO: influx_migration.py: Restoring bucket data and metadata using the InfluxDB CLI
2023/10/26 10:55:36 INFO: Restoring KV snapshot
2023/10/26 10:55:38 WARN: Restoring KV snapshot overwrote the operator token, ensure
 following commands use the correct token
2023/10/26 10:55:38 INFO: Restoring SQL snapshot
2023/10/26 10:55:39 INFO: Restoring TSM snapshot for shard 6952
2023/10/26 10:55:39 INFO: Restoring TSM snapshot for shard 6953
[More shard restores . . .]
2023/10/26 10:55:49 INFO: Restoring TSM snapshot for shard 8268
2023/10/26 10:55:49 INFO: Restoring TSM snapshot for shard 2
DEBUG: influx_migration.py: restore started at 2023-10-26 10:55:36 and took 13.51
 seconds to run.
INFO: influx_migration.py: Migration complete
```

Example Ejemplo 3: Migración completa con CSV la organización de destino y el bucket de S3

Igual que en el ejemplo anterior, pero con Linux o Mac y almacenando los archivos en el bucket de S3,`my-s3-bucket`. Esto evita que los archivos de respaldo sobrecarguen la capacidad de almacenamiento local.

```
python3 influx_migration.py --full --src-host http://localhost:8086 --dest-host http://
dest-server-address:8086 --csv --dest-org MyOrg --s3-bucket my-s3-bucket
```

El resultado debería tener un aspecto similar al siguiente:

```
INFO: influx_migration.py: Creating directory influxdb-backups
INFO: influx_migration.py: Mounting influxdb-migration-bucket
INFO: influx_migration.py: Creating directory influxdb-backups/my-s3-bucket/influxdb-
backup-1698352128323
INFO: influx_migration.py: Backing up bucket data and metadata using the InfluxDB v2
 API
INFO: influx_migration.py: Restoring bucket data and metadata from csv
INFO: influx_migration.py: Restoring bucket some-bucket
INFO: influx_migration.py: Restoring bucket another-bucket
INFO: influx_migration.py: Restoring bucket primary-bucket
INFO: influx_migration.py: Migration complete
INFO: influx_migration.py: Unmounting influxdb-backups
INFO: influx_migration.py: Removing temporary mount directory
```

Configuración de una instancia de base de datos

En esta sección se muestra cómo configurar su instancia de base de datos Amazon Timestream para InfluxDB. Antes de crear una instancia de base de datos, decida qué clase de instancia de base de datos ejecutará la instancia de base de datos. Además, decida dónde se ejecutará la instancia de base de datos seleccionando una región. AWS A continuación, cree la instancia de base de datos.

Puede configurar una instancia de base de datos con un grupo de parámetros de base de datos. Un grupo de parámetros de base de datos actúa como contenedor de los valores de configuración del motor que se aplican a una o más instancias de base de datos.

Los parámetros disponibles dependen del motor de base de datos y de la versión del motor de base de datos. Puede especificar un grupo de parámetros de base de datos al crear una instancia de base de datos. También puede modificar una instancia de base de datos para especificarlas.

> **Important**
>
> En este momento, no puede modificar la configuración de cómputo (tipos de instancia) ni de almacenamiento (tipos de almacenamiento) de las instancias existentes.

Creación de una instancia de base de datos

Mediante la consola

1. Inicie sesión en Amazon Timestream para InfluxDB AWS Management Console y ábrala.
2. En la esquina superior derecha de la consola Amazon Timestream for InfluxDB, elija la región en la que desee crear AWS la instancia de base de datos.
3. En el panel de navegación, elija InfluxDB Databases.
4. Seleccione Crear base de datos de Influx.
5. En el campo Identificador de instancia de base de datos, introduzca un nombre que identifique la instancia.
6. Proporcione los parámetros de configuración básicos de InfluxDB: nombre de usuario, organización, nombre del bucket y contraseña.

   > **Important**
   >
   > Tu nombre de usuario, organización, nombre de bucket y contraseña se guardarán en secreto en AWS Secrets Manager, que se creará para tu cuenta.

   Si necesita cambiar la contraseña de usuario una vez que la instancia de base de datos esté disponible, puede modificarla mediante Influx CLI.
7.
8. Para la clase de instancia de base de datos, seleccione el tamaño de instancia que mejor se adapte a sus necesidades de carga de trabajo.
9. Para la clase de almacenamiento de base de datos, seleccione una clase de almacenamiento que se adapte a sus necesidades. En todos los casos, solo necesitará configurar el almacenamiento asignado.

10. En la sección de configuración de conectividad, asegúrese de que su instancia de InfluxDB esté en la misma subred que los nuevos clientes que requieren conectividad con su instancia de base de datos de Timestream for InfluxDB. También puede optar por hacer que su instancia de base de datos esté disponible públicamente.
11. Elija Crear base de datos de Influx.
12. En la lista de bases de datos, elija el nombre de la nueva instancia de InfluxDB para mostrar sus detalles. La instancia de base de datos tiene el estado Creando hasta que esté lista para usarse.
13. Cuando el estado cambie a Available (Disponible), podrá conectarse a la instancia de la base de datos. Dependiendo de la clase de instancia de la base de datos y de la cantidad de almacenamiento, es posible que la nueva instancia tarde hasta 20 minutos en estar disponible.

Utilización del CLI

Para crear una instancia de base de datos mediante el AWS Command Line Interface, llame al `create-db-instance` comando con los siguientes parámetros:

```
--name
--vpc-subnet-ids
--vpc-security-group-ids
--db-instance-type
--db-storage-type
--username
--organization
--password
--allocated-storage
```

Para obtener más información acerca de cada configuración, consulte Configuración de instancias de base de datos.

Example Ejemplo: usar las configuraciones de motor predeterminadas

Para Linux, macOS o Unix:

```
aws timestream-influxdb create-db-instance \
    --name myinfluxDbinstance \
    --allocated-storage 400 \
    --db-instance-type db.influx.4xlarge \
    --vpc-subnet-ids subnetid1 subnetid2
    --vpc-security-group-ids mysecuritygroup \
    --username masterawsuser \
```

```
    --password \
    --db-storage-type InfluxIOIncludedT2
```

Para Windows:

```
aws timestream-influxdb create-db-instance \
    --name myinfluxDbinstance \
    --allocated-storage 400 \
    --db-instance-type db.influx.4xlarge \
    --vpc-subnet-ids subnetid1 subnetid2
    --vpc-security-group-ids mysecuritygroup \
    --username masterawsuser \
    --password \
    --db-storage-type InfluxIOIncludedT2
```

Utilización del API

Para crear una instancia de base de datos mediante el AWS Command Line Interface, llame al `CreateDBInstance` comando con los siguientes parámetros:

Para obtener más información acerca de cada configuración, consulte Configuración de instancias de base de datos.

> **Important**
>
> Como parte del objeto de DBInstance respuesta, recibirá un influxAuthParametersSecretArn. Esto guardará un ARN SecretsManager secreto en tu cuenta. Solo se rellenará cuando sus instancias de base de datos de InfluxDB estén disponibles. El secreto contiene los parámetros de autenticación de afluencia proporcionados durante el `CreateDbInstance` proceso. Se trata de una READONLY copia, ya updates/modifications/deletions que cualquier dato de este secreto no afecta a la instancia de base de datos creada. Si elimina este secreto, nuestra API respuesta seguirá haciendo referencia al secreto eliminadoARN.

Cuando haya terminado de crear su instancia de base de datos Timestream for InfluxDB, le recomendamos que descargue, instale y configure Influx. CLI

La afluencia CLI proporciona una forma sencilla de interactuar con InfluxDB desde una línea de comandos. Para obtener instrucciones detalladas de instalación y configuración, consulte Use the Influx. CLI

Configuración de instancias de base de datos

Puede crear una instancia de base de datos mediante la consola, el `create-db-instance` CLI comando o la operación `CreateDBInstance` Timestream for API InfluxDB.

En la siguiente tabla se proporcionan detalles sobre la configuración que se elige al crear una instancia de base de datos.

Configuración de la consola	Descripción	CLIopción y parámetro Timestream API
Allocated storage (Almacenamiento asignado)	La cantidad de almacenamiento que se debe asignar a la instancia de base de datos (en gibibytes). En algunos casos, asignar a la instancia de base de datos una cantidad de almacenamiento mayor que el tamaño de la base de datos puede mejorar el desempeño de E/S. Para obtener más información, consulte Almacenamiento de instancias de InfluxDB.	CLI: `allocated-storage` API: `allocatedstorage`
Nombre del bucket	Un nombre para el depósito para inicializar la instancia InfluxDb	CLI: `bucket` API: `bucket`
Tipo de instancia de base de datos	La configuración de su instancia de base de datos. Por ejemplo, una clase de instancia de base de datos db.influx.large tiene 16 GiB de memoria, 2 optimizados para memoria. vCPUs Si es posible, elija un tipo de instancia de base de datos lo suficientemente grande como para almacenar en memoria un conjunto de trabajo de consultas típico. Cuando los conjuntos de trabajo se albergan en la memoria, el sistema puede evitar escribir en el disco, lo que mejora su rendimiento.	CLI: `db-instance-type` API: `Dbinstancetype`

Configuración de la consola	Descripción	CLIopción y parámetro Timestream API
	Para obtener más información, consulte Tipos de clase de instancia de base de datos.	
DB Instance Identifier (Identificador de instancias de bases de datos)	Nombre de la instancia de base de datos. Asigne a sus instancias de base de datos el mismo nombre que a sus servidores en las instalaciones. El identificador de su instancia de base de datos puede contener hasta 63 caracteres alfanuméricos y debe ser único para su cuenta en la AWS región que elija.	CLI: `db-instance-identifier` API: `Dbinstanceidentifier`
DB Parameter Group (Grupo de parámetros de base de datos)	Grupo de parámetros para la instancia de base de datos. Puede elegir el grupo de parámetros predeterminado o crear un grupo de parámetros personalizado. Para obtener más información, consulte Trabajo con los grupos de parámetros de base de datos.	CLI: `db-parameter-group-name` API: `DBParameterGroupName`
Configuración de entrega de registros	El nombre del depósito de S3 donde se almacenarán los registros de InfluxDB.	CLI: `LogDeliveryConfiguration` API: `log-delivery-configuration`

Configuración de la consola	Descripción	CLIopción y parámetro Timestream API
Multi-AZ deployment (Implementación Multi-AZ)	Create a standby instance (Crear una instancia en espera) para crear una réplica secundaria pasiva de la instancia de base de datos en otra zona de disponibilidad para admitir el soporte si se produce algún error. Es recomendable usar varias zonas de disponibilidad Multi-AZ para las cargas de trabajo de producción con el objeto de mantener una alta disponibilidad. En el caso de desarrollo y pruebas, puede elegir Do not create a standby instance (No crear una instancia en espera). Para obtener más información, consulte Configuración y administración de una implementación Multi-AZ.	CLI: `MultiAz` API: `multi-az`
Password	Esta será la contraseña maestra que utilizará para inicializar su instancia de base de datos de InfluxDB. Utilizará esta contraseña para iniciar sesión en InfluxUI y obtener su token de operador.	CLI: `password` API: `password`

Configuración de la consola	Descripción	CLIopción y parámetro Timestream API
Acceso público	Sí, para dar a la instancia de base de datos una dirección IP pública, lo que significa que se puede acceder a ella desde fuera deVPC. Para que sea de acceso público, la instancia de base de datos también debe estar en una subred pública delVPC. No, para hacer que la instancia de base de datos solo sea accesible desde dentro deVPC. Para conectarse a una instancia de base de datos desde fuera de ellaVPC, la instancia de base de datos debe ser de acceso público. Además, el acceso debe concederse mediante las reglas entrantes del grupo de seguridad de la instancia de base de datos. Además, deben cumplirse otros requisitos.	CLI: `publicly-accessible` API: `PubliclyAccessible`
Storage Type	El tipo de almacenamiento de la instancia de base de datos Puede elegir entre tres tipos diferentes de almacenamiento aprovisionado e IOPS incluido de acuerdo con los requisitos de sus cargas de trabajo: * Influencia incluida: 3000 IOPS IOPS * Influencia IOPS incluida: 12.000 IOPS * INflux IOPS Incluye 16000 IOPS Para obtener más información, consulte Almacenamiento de instancias de InfluxDB.	CLI: `db-storage-type` API: `DbStorageType`

Configuración de la consola	Descripción	CLIopción y parámetro Timestream API
Nombre de usuario inicial	Este será el usuario maestro con el que inicializar la instancia de base de datos de InfluxDB. Utilizará este nombre de usuario para iniciar sesión en InfluxUI y obtener su token de operador.	CLI: `username` API: `Username`
Subredes	Una subred de vpc para asociarla a esta instancia de base de datos.	CLI: `vpc-subnet-ids` API: `VPCSubnetIds`
VPCGrupo de seguridad (firewall)	Los grupos de seguridad con los que asociar la instancia de base de datos.	CLI: `vpc-security-group-ids` API: `VPCSecurityGroupIds`

Conexión a una instancia de base de datos Amazon Timestream para InfluxDB

Antes de conectarse a una instancia de base de datos, debe crear la instancia de base de datos. Para obtener más información, consulte Creación de una instancia de base de datos. Una vez que Amazon Timestream aprovisione su instancia de base de datos, utilice API InfluxDB, CLI Influx o cualquier cliente o utilidad compatible para que InfluxDB se conecte a la instancia de base de datos.

Temas

- Búsqueda de la información de conexión de una instancia de base de datos de Amazon Timestream for InfluxDB
- Opciones de autenticación de bases de datos
- Trabajo con los grupos de parámetros

Búsqueda de la información de conexión de una instancia de base de datos de Amazon Timestream for InfluxDB

La información de conexión de una instancia de base de datos incluye su punto final, puerto, nombre de usuario, contraseña y un token de acceso válido, como el operador o el token de acceso total. Por ejemplo, en el caso de una instancia de base de datos de InfluxDB, supongamos que el valor del punto final es. `influxdb1-123456789.us-east-1.timestream-influxdb.amazonaws.com` En este caso, el valor del puerto es 8086 y el usuario de la base de datos es admin. Dada esta información, se especifican los siguientes valores en una cadena de conexión:

- Para el host o el nombre o DNS nombre del host, especifique`influxdb1-123456789.us-east-1.timestream-influxdb.amazonaws.com`.
- Para el puerto, especifique 8086.
- Para el usuario, especifique admin.
- Como contraseña, especifique la que proporcionó al crear la instancia de base de datos.

> **Important**
>
> Cuando creó su instancia de base de datos Timestream for InfluxDB, recibirá una parte del objeto de DBInstance respuesta. `influxAuthParametersSecretArn` Esto guardará un secreto en tu cuenta. SecretsManager Solo se rellenará cuando sus instancias de base de datos de InfluxDB estén disponibles. El secreto contiene los parámetros de autenticación de afluencia proporcionados durante el `CreateDbInstance` proceso. Se trata de una READONLY copia, ya updates/modifications/deletions que cualquier dato de este secreto no afecta a la instancia de base de datos creada. Si elimina este secreto, nuestra API respuesta seguirá haciendo referencia al secreto eliminado arn.

El punto de enlace es único para cada instancia de base de datos y los valores del puerto y del usuario pueden variar. Para conectarse a una instancia de base de datos, puede utilizar InfluxCLI, Influx API o cualquier otro cliente compatible con InfluxDB.

Para buscar la información de conexión de una instancia de base de datos, utilice la consola de AWS administración. También puede utilizar el AWS comando Command Line Interface (AWS CLI) `describe-db-instances` o la operación API `GetDBInstance` Timestream-InfluxDB.

Usando el AWS Management Console

1. Inicie sesión en la consola Amazon Timestream AWS Management Console y ábrala.
2. En el panel de navegación, elija InfluxDB Databases para ver una lista de sus instancias de base de datos.
3. Elija el nombre de la instancia de base de datos para ver sus detalles.
4. En la sección Resumen, copie el punto final. También anote el número de puerto. Necesita el punto de enlace y el número de puerto para conectarse a la instancia de base de datos.

Si necesita encontrar la información del nombre de usuario y la contraseña, seleccione la pestaña Detalles de configuración y elija la `influxAuthParametersSecretArn` para acceder a Secrets Manager.

Usando el CLI

- Para buscar la información de conexión de una instancia de base de datos de InfluxDB mediante el AWS CLI, ejecute el get-db-instance comando. En la llamada, busque el ID de la instancia de base de datos, el punto final, el puerto y influxAuthParameters el secreto.

  Para Linux, macOS o Unix:

  ```
  aws timestream-influxdb get-db-instance --identifier id \
   --query "[name,endpoint,influxAuthParametersSecretArn]"
  ```

  Para Windows:

  ```
  aws timestream-influxdb get-db-instance --identifier id \
   --query "[name,endpoint,influxAuthParametersSecretArn]"
  ```

  El resultado debería ser similar al siguiente. Para acceder a la información del nombre de usuario, tendrá que comprobar la`InfluxAuthParameterSecret`.

  ```
  [
      [
          "mydb",
          "mydb-123456789012.us-east-1.timestream-influxdb.amazonaws.com",
          8086,
      ]
  ```

```
]
```

Creación de tokens de acceso

Con esta información, podrá conectarse a su instancia para recuperar o crear sus tokens de acceso. Existen varias formas de lograrlo:

Usando el CLI

1. Si aún no lo ha hecho, descargue, instale y configure la afluencia CLI.
2. Al configurar su configuración de CLI Influx, utilice `--username-password` para autenticarse.

```
influx config create --config-name YOUR_CONFIG_NAME  --host-url "https://
yourinstance.timestream-influxdb.amazonaws.com:8086"  --org yourorg --username-
password admin --active
```

3. Usa el comando flux auth create para volver a crear tu token de operador. Tenga en cuenta que este proceso invalidará el token de operador anterior.

```
influx auth create --org kronos --operator
```

4. Una vez que tengas el token del operador, puedes usar el comando flux auth list para ver todos tus tokens. Puedes usar el comando flux auth create para crear un token de acceso total.

> **Important**
>
> Deberá realizar este paso para obtener primero su token de operador y, a continuación, poder crear nuevos tokens utilizando API InfluxDB o. CLI

Uso de la interfaz de usuario de Influx

1. Navegue hasta su instancia de Timestream for InfluxDB utilizando el punto final creado para iniciar sesión y acceder a la interfaz de usuario de InfluxDB. Deberá usar el nombre de usuario y la contraseña utilizados para crear su instancia de base de datos de InfluxDB. Puede recuperar esta información de la `influxAuthParametersSecretArn` que se especificó en el objeto de respuesta de. `CreateDbInstance`

Como alternativa, puede abrir InfluxUI desde la consola de administración de Timestream for InfluxDB:

a. Inicie sesión en la consola Amazon Timestream for InfluxDB AWS Management Console y ábrala en. https://console.aws.amazon.com/timestream/
b. En la esquina superior derecha de la consola Amazon Timestream for InfluxDB, elija la región en la que creó AWS la instancia de base de datos.
c. En la lista de bases de datos, elija el nombre de la instancia de InfluxDB para mostrar sus detalles. En la esquina superior derecha, selecciona Open Influx UI.

2. Una vez que hayas iniciado sesión en tu InfluxUI, ve a Cargar datos y luego a APITokens desde la barra de navegación izquierda.
3. Elija + GENERATE API TOKEN y seleccione All Access API Token.
4. Introduce una descripción para el API token y elige SAVE.
5. Copia el token generado y guárdalo de forma segura.

> **Important**
>
> Al crear fichas desde InfluxUI, las fichas recién creadas solo se mostrarán una vez. Asegúrate de copiarlos, ya que si no, tendrás que volver a crearlos.

Usando el InfluxDB API

- Envíe una solicitud al API `/api/v2/authorizations` punto final de InfluxDB mediante el método de solicitud. POST

  Incluya lo siguiente en su solicitud:

  a. Encabezados:

     i. Autorización: Token < INFLUX _ OPERATOR _ TOKEN >
     ii. Tipo de contenido: application/json

  b. Cuerpo de la solicitud: JSON cuerpo con las siguientes propiedades:

     i. estado: «activo»
     ii. descripción: descripción del API token

iii. OrgID: ID de organización de InfluxDB

iv. permisos: matriz de objetos en la que cada objeto representa los permisos para un tipo de recurso de InfluxDB o un recurso específico. Cada permiso contiene las siguientes propiedades:

A. acción: «leer» o «escribir»

B. recurso: JSON objeto que representa el recurso de InfluxDB al que se va a conceder el permiso. Cada recurso contiene al menos la siguiente propiedad: OrgID: ID de organización de InfluxDB

C. tipo: tipo de recurso. Para obtener información sobre los tipos de recursos de InfluxDB que existen, utilice the /api/v2/resources endpoint.

En el siguiente ejemplo, se utiliza `curl` y el InfluxDB API para generar un token de acceso total:

```
export INFLUX_HOST=https://influxdb1-123456789.us-east-1.timestream-
influxdb.amazonaws.com
export INFLUX_ORG_ID=<YOUR_INFLUXDB_ORG_ID>
export INFLUX_TOKEN=<YOUR_INFLUXDB_OPERATOR_TOKEN>

curl --request POST \
"$INFLUX_HOST/api/v2/authorizations" \
  --header "Authorization: Token $INFLUX_TOKEN" \
  --header "Content-Type: text/plain; charset=utf-8" \
  --data '{
    "status": "active",
    "description": "All access token for get started tutorial",
    "orgID": "'"$INFLUX_ORG_ID"'",
    "permissions": [
      {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
 "authorizations"}},
      {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
 "authorizations"}},
      {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
 "buckets"}},
      {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
 "buckets"}},
      {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
 "dashboards"}},
      {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
 "dashboards"}},
```

```
      {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type": "orgs"}},
      {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type": "orgs"}},
      {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
"sources"}},
      {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
"sources"}},
      {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type": "tasks"}},
      {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
"tasks"}},
      {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
"telegrafs"}},
      {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
"telegrafs"}},
      {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type": "users"}},
      {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
"users"}},
      {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
"variables"}},
      {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
"variables"}},
      {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
"scrapers"}},
      {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
"scrapers"}},
      {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
"secrets"}},
      {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
"secrets"}},
      {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
"labels"}},
      {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
"labels"}},
      {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type": "views"}},
      {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
"views"}},
      {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
"documents"}},
      {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
"documents"}},
      {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
"notificationRules"}},
      {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
"notificationRules"}},
```

```
      {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
 "notificationEndpoints"}},
      {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
 "notificationEndpoints"}},
      {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
 "checks"}},
      {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
 "checks"}},
      {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type": "dbrp"}},
      {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type": "dbrp"}},
      {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
 "notebooks"}},
      {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
 "notebooks"}},
      {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
 "annotations"}},
      {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
 "annotations"}},
      {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
 "remotes"}},
      {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
 "remotes"}},
      {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
 "replications"}},
      {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
 "replications"}}
    ]
  }
'
```

Opciones de autenticación de bases de datos

Amazon Timestream para InfluxDB admite las siguientes formas de autenticar a los usuarios de bases de datos:

- Con la autenticación de contraseña,– la instancia de base de datos realiza toda la administración de las cuentas de usuario. Puede crear usuarios, especificar contraseñas y administrar los tokens mediante InfluxUI, Influx o flux. CLI API
- Autenticación mediante token: su instancia de base de datos se encarga de toda la administración de las cuentas de usuario. Puede crear usuarios, especificar una contraseña y administrar los tokens con su token de operador mediante Influx CLI e InfluxAPI.

Conexiones cifradas

Puede utilizar Secure Socket Layer (SSL) o Transport Layer Security (TLS) desde su aplicación para cifrar una conexión a una instancia de base de datos. Los certificados necesarios para el TLS enlace entre InfluxDB y las aplicaciones creadas y administradas por el servicio de Kronos. Cuando se renueva el certificado, la instancia se actualiza automáticamente con la última versión sin que sea necesaria la intervención del usuario.

Trabajo con los grupos de parámetros

Parámetros de la base de datos especificar cómo está configurada la base de datos. Por ejemplo, los parámetros de la base de datos pueden especificar la cantidad de recursos, como la memoria, que se asignarán a una base de datos.

La configuración de la base de datos se gestiona asociando las instancias de base de datos a los grupos de parámetros. Amazon Timestream para InfluxDB define los grupos de parámetros con la configuración predeterminada. También puede definir sus propios grupos de parámetros con una configuración personalizada.

Descripción general de los grupos de parámetros

Un grupo de parámetros de base de datos sirve de contenedor para los valores de configuración del motor que se aplican a una o varias instancias de bases de datos.

Temas

- Grupos de parámetros predeterminados y personalizados
- Creación de un grupo de parámetros de base de datos
- Parámetros de instancias de base de datos estáticos y dinámicos
- Parámetros y valores de parámetros admitidos

Grupos de parámetros predeterminados y personalizados

Las instancias de base de datos utilizan grupos de parámetros de base de datos. En las secciones siguientes se describe cómo configurar y administrar los grupos de parámetros de instancia de base de datos.

Creación de un grupo de parámetros de base de datos

Puede crear un nuevo grupo de parámetros de base de datos mediante el AWS Management Console, el o el Timestream AWS Command Line Interface. API

Se aplican las siguientes limitaciones al nombre del grupo de parámetros de base de datos:

- Debe tener de 1 a 255 letras, números o guiones.
- Los nombres de los grupos de parámetros predeterminados pueden incluir un punto, como `default.InfluxDB.2.7`. Sin embargo, los nombres de grupos de parámetros personalizados no pueden incluir un punto.
- El primer carácter debe ser una letra.
- El nombre no puede empezar por «dbpg-»
- El nombre no puede incluir dos guiones consecutivos ni finalizar con guion.
- Si crea una instancia de base de datos sin especificar un grupo de parámetros de base de datos, la instancia de base de datos utiliza los valores predeterminados del motor InfluxDB.

La configuración de los parámetros de un grupo de parámetros predeterminado no se puede modificar. En su lugar, puede hacer lo siguiente:

1. Cree un nuevo grupo de parámetros.
2. Cambie la configuración de los parámetros que desee. No todos los parámetros del motor de base de datos pueden cambiarse en el grupo de parámetros.
3. Actualice la instancia de base de datos para usar el grupo de parámetros personalizado. Para obtener información sobre la actualización de una instancia de base de datos, consulteActualización de instancias de base de datos.

> **Note**
>
> Si ha modificado la instancia de base de datos para usar un grupo de parámetros personalizado e inicia la instancia de base de datos, Amazon Timestream for InfluxDB reinicia automáticamente la instancia de base de datos como parte del proceso de inicio. Actualmente, no podrá modificar los grupos de parámetros personalizados una vez que se hayan creado. Si necesita cambiar un parámetro, es necesario que cree un nuevo grupo de parámetros personalizado y lo asigne a las instancias que requieran este cambio de configuración. Si actualiza una instancia de base de datos existente para asignar un nuevo grupo de parámetros, siempre se aplicará inmediatamente y se reiniciará la instancia.

Parámetros de instancias de base de datos estáticos y dinámicos

Los parámetros de las instancias de base de datos de InfluxDB son siempre estáticos. Se comportan de la siguiente manera:

Al cambiar un parámetro estático, guardar el grupo de parámetros de base de datos y asignarlo a una instancia, el cambio de parámetro se aplica automáticamente después de reiniciar la instancia.

Al asociar un nuevo grupo de parámetros de base de datos a una instancia de base de datos, Timestream aplica los parámetros estáticos modificados solo después de reiniciar la instancia de base de datos. Actualmente, la única opción se aplica de forma inmediata.

Para obtener información sobre el cambio del grupo de parámetros de base de datos, consulte Actualización de instancias de base de datos.

Parámetros y valores de parámetros admitidos

Para determinar los parámetros compatibles con la instancia de base de datos, consulte los parámetros del grupo de parámetros de base de datos que utiliza la instancia de base de datos. Para obtener más información, consulte Visualización de los valores de los parámetros de un grupo de parámetros de base de datos.

Para obtener más información sobre todos los parámetros compatibles con la versión de código abierto de InfluxDB, consulte las opciones de configuración de InfluxDB. Actualmente, solo podrá modificar los siguientes parámetros de InfluxDB:

Parámetro	Descripción	Valor predeterminado	Valor	Rango válido	Nota
flux-log-enabled	Incluya la opción para mostrar los registros detallados de las consultas de Flux	FALSE	true, false	N/A	

Parámetro	Descripción	Valor predeterminado	Valor	Rango válido	Nota
a nivel de registro	Nivel de salida del registro. InfluxDB genera entradas de registro con niveles de gravedad superiores o iguales al nivel especificado.	info	depuración, información, error	N/A	
sin tareas	Número de consultas que se pueden ejecutar simultáneamente. Si se establece en 0, se permite un número ilimitado de consultas simultáneas.	FALSE	true, false	N/A	

Parámetro	Descripción	Valor predeterminado	Valor	Rango válido	Nota
concurrencia de consultas	Deshabilite el programador de tareas. Si las tareas problemáticas impiden que InfluxDB se inicie, utilice esta opción para iniciar InfluxDB sin programar ni ejecutar tareas.	1024		N/A	

Parámetro	Descripción	Valor predeterminado	Valor	Rango válido	Nota
query-queue-size	Número máximo de consultas permitidas en la cola de ejecución. Cuando se alcanza el límite de cola, se rechazan las consultas nuevas. Si se establece en 0, se permite un número ilimitado de consultas en la cola.	1024		N/A	
tipo de rastreo	Habilita el rastreo en InfluxDB y especifica el tipo de rastreo. El rastreo está desactivado de forma predeterminada.	""	registro, jaeger	N/A	

Parámetro	Descripción	Valor predeterminado	Valor	Rango válido	Nota
métricas deshabilitadas	Deshabilite el punto final HTTP /metrics que expone las métricas internas de InfluxDB.	FALSE		N/A	

Parámetro	Descripción	Valor predeterminado	Valor	Rango válido	Nota
http-idle-timeout	Durante el tiempo máximo que el servidor debe mantener activas las conexiones establecidas mientras espera nuevas solicitudes. Establézcalo `0` en No se agota el tiempo de espera.	30 ms	Duración con unidad`hours`,,,`seconds``milliseconds` Ejemplo: `durationType=minutes,value=10`	Horas: -Mínimo: 0 -Máximo: 256 205 Minutos: -Mínimo: 0 -Máximo: 15372286 Segundos: -Mínimo: 0 -Máximo: 922337 203 Milisegundos: -Mínimo: 0 -Máximo: 922337203 685	

Parámetro	Descripción	Valor predeterminado	Valor	Rango válido	Nota
http-read-header-timeout	Tiempo máximo durante el que el servidor debe intentar leer los encabezados de las nuevas solicitud esHTTP. Establézcalo en `0` «No hay tiempo de espera».	10 segundos	Duración con unidad`hours`,, `minutes``seconds` . Ejemplo: `durationType=minutes,value=10`	Horas: -Mínimo: 0 -Máximo: 256 205 Minutos: -Mínimo: 0 -Máximo: 15372286 Segundos: -Mínimo: 0 -Máximo: 922337 203 Milisegundos: -Mínimo: 0 -Máximo: 922337203 685	

Parámetro	Descripción	Valor predeterminado	Valor	Rango válido	Nota
http-read-timeout	Tiempo máximo durante el que el servidor debe intentar leer la totalidad de las nuevas solicitudes. 0Establézcalo en «No hay tiempo de espera».	0	Duración con unidadhours,minds . Ejemplo: durationType=minutes,value=10	Horas: -Mínimo: 0 -Máximo: 256 205 Minutos: -Mínimo: 0 -Máximo: 15372286 Segundos: -Mínimo: 0 -Máximo: 922337 203 Milisegundos: -Mínimo: 0 -Máximo: 922337203 685	

Parámetro	Descripción	Valor predeterminado	Valor	Rango válido	Nota
http-write-timeout	Tiempo máximo que el servidor debe dedicar a procesar y responder a las solicitudes de escritura. 0Establézcalo en «No hay tiempo de espera».	0	Duración con unidadhours,mnds . Ejemplo: durationType=minutes,value=10	Horas: -Mínimo: 0 -Máximo: 256 205 Minutos: -Mínimo: 0 -Máximo: 15372286 Segundos: -Mínimo: 0 -Máximo: 922337 203 Milisegundos: -Mínimo: 0 -Máximo: 922337203 685	

Parámetro	Descripción	Valor predeterminado	Valor	Rango válido	Nota
influxql-max-select-buckets	Número máximo de grupos por intervalos de tiempo que puede crear una sentencia . SELECT 0permite un número ilimitado de intervalos.	0	Largo	Mínimo: 0 Máximo: 9.223.372 .036.854. 775.807	

Parámetro	Descripción	Valor predeterminado	Valor	Rango válido	Nota
influxql-max-select-point	Número máximo de puntos que puede procesar una declaración. SELECT 0permite un número ilimitado de puntos. InfluxDB comprueba el recuento de puntos cada segundo (por lo que las consultas que superan el máximo no se cancelan inmediatamente).	0	Largo	Mínimo: 0 Máximo: 9.223.372.036.854.775.807	

Parámetro	Descripción	Valor predeterminado	Valor	Rango válido	Nota
influxql-max-select-series	Número máximo de series que puede devolver una declaración. `SELECT 0` permite un número ilimitado de series.	0	Largo	Mínimo: 0 Máximo: 9.223.372.036.854.775.807	
Prof-deshabilitado	Deshabilite el punto final. `/debug/pprof` HTTP Este punto final proporciona datos de creación de perfiles en tiempo de ejecución y puede resultar útil a la hora de depurar.	FALSE	Booleano	N/A	

Parámetro	Descripción	Valor predeterminado	Valor	Rango válido	Nota
query-initial-memory-bytes	Los bytes iniciales de memoria asignados a una consulta.	0	Largo	Mínimo: 0 Máximo: query-memory-bytes	
query-max-memory-bytes	Número máximo total de bytes de memoria permitido para las consultas.	0	Largo	Mínimo: 0 Máximo: 9.223.372.036.854.775.807	
query-memory-bytes	Especifica el tiempo de vida () en minutos de las sesiones de usuario recién creadas. TTL	0	Largo	Mínimo: 0 Máximo: 2.147.483.647	Debe ser mayor o igual a. query-initial-memory-bytes
duración de la sesión	Especifica el tiempo de vida (TTL) en minutos de las sesiones de usuario recién creadas.	60	Entero	Mínimo: 0 Máximo: 2880	

Parámetro	Descripción	Valor predeterminado	Valor	Rango válido	Nota
session-renew-disabled	Desactiva la extensión automática de la sesión de un usuario TTL en cada solicitud. De forma predeterminada, cada solicitud establece el tiempo de caducidad de la sesión en cinco minutos a partir de ahora. Cuando está deshabilitada, las sesiones caducan después de la duración de sesión especificada y el usuario es redirigido a la página de inicio de sesión, incluso si ha	FALSE	Booleano	N/A	

Parámetro	Descripción	Valor predeterminado	Valor	Rango válido	Nota
	estado activa recientemente.				
storage-cache-max-memory-tamaño	Tamaño máximo (en bytes) que puede alcanzar la caché de un fragmento antes de que comience a rechazar las escrituras.	1073741824	Largo	Mínimo: 0 Máximo: 549755813888	Debe ser inferior a la capacidad total de memoria de la instancia. Recomendamos configurarlo por debajo del 15% de la capacidad total de memoria.

Parámetro	Descripción	Valor predeterminado	Valor	Rango válido	Nota
storage-cache-snapshot-memory-tamaño	Tamaño (en bytes) con el que el motor de almacenamiento capturará la caché y la escribirá en un TSM archivo para disponer de más memoria.	26214400	Largo	Mínimo: 0 Máximo: 549755813888	Debe ser inferior a la talla «-size». storage-cache-max-memory

Parámetro	Descripción	Valor predeterminado	Valor	Rango válido	Nota
storage-cache-snapshot-write-duración en frío	Duración durante la cual el motor de almacenamiento tomará una instantánea de la memoria caché y la escribirá en un nuevo TSM archivo si el fragmento no ha recibido operaciones de escritura o borrado.	100 ms	Duración con unidadhours,,,seconds milliseconds Ejemplo: durationType=minutes,value=10	Horas: -Mínimo: 0 -Máximo: 256 205 Minutos: -Mínimo: 0 -Máximo: 15372286 Segundos: -Mínimo: 0 -Máximo: 922337 203 Milisegundos: -Mínimo: 0 -Máximo: 922337203 685	

Parámetro	Descripción	Valor predeterminado	Valor	Rango válido	Nota
storage-compact-full-write-duración en frío	Duración durante la cual el motor de almacenamiento compactará todos los TSM archivos de un fragmento si no ha recibido escrituras ni eliminaciones.	40h0m0s	Duración con unidad,,,. `hours` `minutes` `seconds` `milliseconds` Ejemplo: `durationType=minutes,value=10`	Horas: -Mínimo: 0 -Máximo: 256 205 Minutos: -Mínimo: 0 -Máximo: 15372286 Segundos: -Mínimo: 0 -Máximo: 922337 203 Milisegundos: -Mínimo: 0 -Máximo: 922337203 685	

Parámetro	Descripción	Valor predeterminado	Valor	Rango válido	Nota
storage-compact-throughput-burst	Límite de velocidad (en bytes por segundo) que las TSM compactaciones pueden escribir en el disco.	50331648	Largo	Mínimo: 0 Máximo: 9.223.372.036.854.775.807	

Parámetro	Descripción	Valor predeterminado	Valor	Rango válido	Nota
storage-max-concurrent-compactions	Número máximo de compactaciones completas y niveladas que se pueden ejecutar simultáneamente. Un valor de `0` da como resultado que `runtime.GOMAXPROCS(0)` se utilice el 50% en tiempo de ejecución. Cualquier número superior a cero limita las compactaciones a ese valor. Esta configuración no se aplica a las instantáneas en caché.	0	Entero	Mínimo: 0 Máximo: 64	

Parámetro	Descripción	Valor predeterminado	Valor	Rango válido	Nota
storage-max-index-log-tamaño de archivo	Tamaño (en bytes) en el que un archivo de registro de índice escrito con anticipación (WAL) se compactará en un archivo de índice. Los tamaños más bajos harán que los archivos de registro se compacten más rápidamente y, por lo tanto, se reducirá el uso de pilas a expensas del rendimiento de escritura.	1048576	Largo	Mínimo: 0 Máximo: 9.223.372.036.854.775.807	

Parámetro	Descripción	Valor predeterminado	Valor	Rango válido	Nota
storage-no-validate-field-tamaño	Omita la validación del tamaño del campo en las solicitudes de escritura entrantes.	FALSE	Booleano	N/A	
storage-retention-check-interval	Intervalo de comprobaciones de aplicación de la política de retención.	300 ms	Duración con unidad`hours`,,,`seconds` `milliseconds` Ejemplo: `durationType=minutes,value=10`	N/A	Horas: -Mínimo: 0 -Máximo: 256 205 Minutos: -Mínimo: 0 -Máximo: 15372286 Segundos: -Mínimo: 0 -Máximo: 922337 203 Milisegundos: -Mínimo: 0 -Máximo: 922337203 685

Parámetro	Descripción	Valor predeterminado	Valor	Rango válido	Nota
storage-series-file-max-concurrent-snapshot-compactions	Número máximo de compactaciones de instantáneas que se pueden ejecutar simultáneamente en todas las particiones de serie de una base de datos.	0	Entero	Mínimo: 0 Máximo: 64	

Parámetro	Descripción	Valor predeterminado	Valor	Rango válido	Nota
storage-series-id-set-tamaño de la memoria caché	Tamaño de la memoria caché interna utilizada en el TSI índice para almacenar los resultados de las series calculadas anteriormente. Los resultados en caché se devuelven rápidamente en lugar de tener que volver a calcularlos cuando se ejecuta una consulta posterior con el mismo predicado clave/valor de etiqueta. Si se establece este valor en, `0` se deshabilitará	100	Largo	Mínimo: 0 Máximo: 9.223.372.036.854.775.807	

Parámetro	Descripción	Valor predeterminado	Valor	Rango válido	Nota
	la memoria caché y es posible que se reduzca el rendimiento de la consulta.				
storage-wal-max-concurrent-escribe	Número máximo de escrituras en el WAL directorio que se pueden intentar al mismo tiempo.	0	Entero	Mínimo: 0 Máximo: 256	

Parámetro	Descripción	Valor predeterminado	Valor	Rango válido	Nota
storage-wal-max-write-retraso	Tiempo máximo que esperará una solicitud de escritura en el WAL directorio cuando se alcance el número máximo de escrituras activas simultáneas en el WAL directorio. Configúrelo `0` en para deshabilitar el tiempo de espera.	10 m	Duración con unidad`hours`,, `minutes``seconds` . Ejemplo: `durationType=minutes,value=10`	Horas: -Mínimo: 0 -Máximo: 256 205 Minutos: -Mínimo: 0 -Máximo: 15372286 Segundos: -Mínimo: 0 -Máximo: 922337 203 Milisegundos: -Mínimo: 0 -Máximo: 922337203 685	

Parámetro	Descripción	Valor predeterminado	Valor	Rango válido	Nota
Ui-deshabilitado	Deshabilite la interfaz de usuario (UI) de InfluxDB. La interfaz de usuario está habilitada de forma predeterminada.	FALSE	Booleano	N/A	

Si los parámetros de un grupo de parámetros se configuran de forma incorrecta, pueden producirse efectos adversos no deseados, como la degradación del rendimiento y la inestabilidad del sistema. Tenga siempre cuidado al modificar los parámetros de la base de datos. Pruebe los cambios en la configuración del grupo de parámetros en una instancia de base de datos de prueba antes de aplicar esos cambios en el grupo de parámetros a una instancia de base de datos de producción.

Trabajo con los grupos de parámetros de base de datos

Las instancias de base de datos utilizan grupos de parámetros de base de datos. En las secciones siguientes se describe cómo configurar y administrar los grupos de parámetros de instancia de base de datos.

Temas

- Creación de un grupo de parámetros de base de datos
- Asociación de un grupo de parámetros de base de datos con una instancia de base de datos
- Descripción de grupos de parámetros de base de datos
- Visualización de los valores de los parámetros de un grupo de parámetros de base de datos

Creación de un grupo de parámetros de base de datos

Uso del AWS Management Console

1. Inicie sesión en la consola Amazon Timestream for InfluxDB AWS Management Console y ábrala.
2. En el panel de navegación, seleccione Parameter groups (Grupos de parámetros).
3. Elija Create parameter group.
4. En el cuadro Group name (Nombre de grupo), escriba el nombre del nuevo grupo de parámetros de base de datos.
5. En el cuadro Description (Descripción), escriba una descripción para el nuevo grupo de parámetros de base de datos.
6. Elija los parámetros que desee modificar y aplique los valores deseados. Para obtener más información sobre los parámetros admitidos, consulteParámetros y valores de parámetros admitidos.
7. Seleccione Guardar.

Uso del AWS Command Line Interface

- Para crear un grupo de parámetros de base de datos mediante el AWS CLI, llame al `create-db-parameter-group` comando con los siguientes parámetros:

```
--db-parameter-group-name <value>
--description <value>
--endpoint_url <value>
--region <value>
--parameters (list) (string)
```

Example Ejemplo

Para obtener más información acerca de cada configuración, consulte Configuración de instancias de base de datos. En este ejemplo, se utilizan las configuraciones de motor predeterminadas.

```
aws timestream-influxdb create-db-parameter-group
    --db-parameter-group-name YOUR_PARAM_GROUP_NAME\
    --endpoint-url YOUR_ENDPOINT
    --region YOUR_REGION \
```

```
    --parameters
"InfluxDBv2={logLevel=debug,queryConcurrency=10,metricsDisabled=true}" \" \
    --debug
```

Asociación de un grupo de parámetros de base de datos con una instancia de base de datos

Puede crear sus propios grupos de parámetros de base de datos con configuraciones personalizadas. Puede asociar un grupo de parámetros de base de datos a una instancia de base de datos mediante la AWS Management Console, la o la AWS Command Line Interface API Timestream-InfluxDB. Puede hacerlo al crear o modificar una instancia de base de datos.

Para obtener información sobre la creación de un grupo de parámetros de base de datos, consulte Creación de un grupo de parámetros de base de datos. Para obtener información acerca de la creación de una instancia de base de datos, consulte Creación de una instancia de base de datos. Para obtener información sobre la modificación de una instancia de base de datos, consulte.. Actualización de instancias de base de datos

> **Note**
>
> Al asociar un nuevo grupo de parámetros de base de datos a una instancia de base de datos, los parámetros estáticos modificados se aplican solo después de reiniciar la instancia de base de datos. Actualmente, solo se admite la aplicación inmediata. Timestream para InfluxDB solo admite parámetros estáticos.

Usando el AWS Management Console

1. Inicie sesión en la consola Amazon Timestream for InfluxDB AWS Management Console y ábrala.
2. En el panel de navegación, elija InfluxDB Databases y, a continuación, elija la instancia de base de datos que desee modificar.
3. Elija Actualizar. Aparece la página Actualizar la instancia de base de datos.
4. Cambie la configuración del grupo de parámetros de base de datos.
5. Elija Continue y consulte el resumen de las modificaciones.
6. Actualmente, solo se admite la opción Aplicar de forma inmediata. Esta opción puede provocar una interrupción en algunos casos, ya que reiniciará la instancia de base de datos.

7. En la página de confirmación, revise los cambios. Si son correctos, elija Actualizar instancia de base de datos para guardar los cambios y aplicarlos. O bien, elija Back (Atrás) para editar los cambios o Cancel (Cancelar) para cancelarlos.

Utilización del AWS Command Line Interface

Para Linux, macOS o Unix:

```
aws timestream-influxdb update-db-instance
--identifier YOUR_DB_INSTANCE_ID \
--region YOUR_REGION \
--db-parameter-group-identifier YOUR_PARAM_GROUP_ID \
--log-delivery-configuration "{\"s3Configuration\": {\"bucketName\":
 \"${LOGGING_BUCKET}\", \"enabled\": false }}"
```

Para Windows:

```
aws timestream-influxdb update-db-instance
--identifier YOUR_DB_INSTANCE_ID ^
--region YOUR_REGION ^
--db-parameter-group-identifier YOUR_PARAM_GROUP_ID ^
--log-delivery-configuration "{\"s3Configuration\": {\"bucketName\":
 \"${LOGGING_BUCKET}\", \"enabled\": false }}"
```

Descripción de grupos de parámetros de base de datos

Puede enumerar los grupos de parámetros de base de datos que ha creado para su AWS cuenta.

Usando el AWS Management Console

1. Inicie sesión en la consola Amazon Timestream for InfluxDB AWS Management Console y ábrala.
2. En el panel de navegación, seleccione Parameter groups (Grupos de parámetros).
3. Los grupos de parámetros de base de datos aparecen en una lista.

Usando la AWS Command Line Interface

Para enumerar todos los grupos de parámetros de base de datos de una AWS cuenta, utilice el AWS Command Line Interface `list-db-parameter-groups` comando.

```
aws timestream-influxdb list-db-parameter-groups --region region
```

Para devolver un grupo de parámetros de base de datos específico para una AWS cuenta, utilice el AWS Command Line Interface `get-db-parameter-group` comando.

```
aws timestream-influxdb get-db-parameter-group --region region --identifier identifier
```

Visualización de los valores de los parámetros de un grupo de parámetros de base de datos

Es posible obtener una lista de todos los parámetros de un grupo de parámetros de base de datos y sus valores.

Usando el AWS Management Console

1. Inicie sesión en la consola Amazon Timestream for InfluxDB AWS Management Console y ábrala.
2. En el panel de navegación, seleccione Parameter groups (Grupos de parámetros).
3. Los grupos de parámetros de base de datos aparecen en una lista.
4. Seleccione el nombre del grupo de parámetros para ver su lista de parámetros.

Usando la AWS Command Line Interface

Para ver los valores de los parámetros de un grupo de parámetros de base de datos, utilice el AWS Command Line Interface `get-db-parameters` comando con el siguiente parámetro obligatorio.

```
--db-parameter-group-name
```

Usando el API

Para ver los valores de los parámetros de un grupo de parámetros de base de datos, utilice el API `GetDBParameters` comando Timestream con el siguiente parámetro obligatorio.

```
DBParameterGroupName
```

Administración de instancias de base de datos

En esta sección se describen varios aspectos de la administración de Timestream for InfluxDB para garantizar un rendimiento, una disponibilidad y unas capacidades de supervisión óptimos. Proporciona orientación sobre cómo actualizar las configuraciones de las instancias de bases de datos, gestionar las implementaciones en zonas de disponibilidad múltiples (Multi-AZ) y los procesos de conmutación por error. También explica cómo eliminar instancias de bases de datos y configurar la visualización de registros para sus instancias de InfluxDB.

Temas

Actualización de instancias de base de datos

Puede actualizar los siguientes parámetros de configuración de su instancia de Timestream for InfluxDB:

- Clase de instancia
- Tipo de implementación
- Grupo de parámetros
- Configuración de entrega de registros

> **Important**
>
> Le recomendamos que pruebe todos los cambios en una instancia de prueba antes de modificar la instancia de producción para comprender su impacto, especialmente al actualizar las versiones de la base de datos. Revise el impacto en la base de datos y las aplicaciones antes de actualizar la configuración. Algunas modificaciones requieren el reinicio de la instancia de base de datos, lo que provoca un tiempo de inactividad.

Uso del AWS Management Console

1. Inicie sesión en la consola Amazon Timestream for InfluxDB AWS Management Console y ábrala.
2. En el panel de navegación, elija InfluxDB Databases y, a continuación, elija la instancia de base de datos que desee modificar.
3. Elija Modificar.
4. En la página Modificar la instancia de base de datos, realice los cambios que desee.
5. Elija Continue (Continuar) y consulte el resumen de las modificaciones.
6. Elija Next (Siguiente).
7. Revise los cambios.
8. Seleccione Modificar instancia para aplicar los cambios.

> **Note**
>
> Estas modificaciones requieren el reinicio de la instancia de base de datos de Influx y, en algunos casos, pueden provocar una interrupción.

Utilización del AWS Command Line Interface

Para actualizar una instancia de base de datos mediante el AWS Command Line Interface, ejecute el `update-db-instance` comando. Especifique el identificador de instancias de bases de datos y los valores de las opciones que desea modificar. Para obtener más información acerca de cada opción, consulte Configuración de instancias de base de datos.

Example Ejemplo

El código siguiente modifica mydbinstance configurando un dbparametergroup diferente. Los cambios se aplican inmediatamente.

Para Linux, macOS o Unix:

```
aws timestream-influxdb update-db-instance \
    --identifier mydbinstance \
    --db-instance-type desired-instance-type \
```

```
    --deployment-type desired-deployment-type \
    --db-parameter-group-name newparamgroup \
    --port 8086
```

Para Windows:

```
aws timestream-influxdb update-db-instance ^
    --identifier mydbinstance ^
    --db-instance-type desired-instance-type ^
    --deployment-type desired-deployment-type ^
    --db-parameter-group-name newparamgroup
    --port 8086
```

Mantenimiento de una instancia de base de datos

Periódicamente, Amazon Timestream para InfluxDB realiza tareas de mantenimiento en los recursos de Amazon Timestream for InfluxDB. El mantenimiento suele implicar la actualización de los siguientes recursos de la instancia de base de datos:

- Hardware subyacente
- Sistema operativo (SO) subyacente
- Versión del motor de base de datos

Las actualizaciones del sistema operativo suelen deberse a motivos de seguridad.

Algunos elementos de mantenimiento requieren que Amazon Timestream for InfluxDB desconecte la instancia de base de datos durante un breve período de tiempo. Entre los elementos de mantenimiento que requieren que un recurso esté desconectado están el sistema operativo necesario o la aplicación de parches a la base de datos. Los parches obligatorios que tienen que ver con la seguridad y la fiabilidad de la instancia son los únicos que se programan automáticamente. Estos parches se producen con poca frecuencia, normalmente una vez cada pocos meses. Rara vez se requiere más de una fracción de su período de mantenimiento.

- Los períodos de mantenimiento están configurados para que se realicen todos los días entre las 12 a. m. y las 4 a. m., hora local, de la región en la que se aloja la instancia.
- Los recursos del cliente pueden actualizarse una vez a la semana en cualquiera de los siete períodos de mantenimiento de la semana.

Eliminación de una instancia de base de datos

La eliminación de una instancia de base de datos afecta a la capacidad de recuperación de la instancia y a la disponibilidad de las instantáneas. Tenga en cuenta lo siguiente:

- Si desea eliminar todos los recursos de Timestream for InfluxDB, tenga en cuenta que los recursos de las instancias de base de datos conllevan gastos de facturación.
- Cuando se elimina el estado de una instancia de base de datos, el valor de su certificado de CA no aparece en la consola de Timestream for InfluxDB ni en el resultado de los comandos o las operaciones de Timestream. AWS Command Line Interface API
- El tiempo necesario para eliminar una instancia de base de datos varía en función de la cantidad de datos que se eliminen y de si se ha tomado una instantánea final.

Puede eliminar una instancia de base de datos mediante el AWS Management Console AWS Command Line Interface, el o el TimestreamAPI. Debe proporcionar el nombre de la instancia de base de datos:

Usando el AWS Management Console

1. Inicie sesión en la consola Amazon Timestream for InfluxDB AWS Management Console y ábrala.
2. En el panel de navegación, elija InfluxDB Databases y, a continuación, elija la instancia de base de datos que desee eliminar.
3. Elija Eliminar.
4. Introduzca confirmar en el cuadro.
5. Elija Eliminar.

Usando el AWS Command Line Interface

Para buscar la instancia IDs de las instancias de base de datos de su cuenta, `list-db-instances` ejecute el comando:

```
aws timestream-influxdb list-db-instances \
--endpoint-url YOUR_ENDPOINT \
--region YOUR_REGION
```

Para eliminar una instancia de base de datos mediante el AWSCLI, `delete-db-instance` ejecute el comando con las siguientes opciones:

```
aws timestream-influxdb list-db-instances \
--identifier YOUR_DB_INSTANCE \
```

Example Ejemplo

Para Linux, macOS o Unix:

```
aws timestream-influxdb delete-db-instance \
    --identifier mydbinstance
```

Para Windows:

```
aws timestream-influxdb delete-db-instance ^
    --identifier mydbinstance
```

Implementaciones de instancias de base de datos Multi-AZ

Amazon Timestream para InfluxDB proporciona alta disponibilidad y soporte de conmutación por error para las instancias de base de datos que utilizan implementaciones Multi-AZ con una única instancia de base de datos en espera. Este tipo de implementación se denomina Implementación de instancia de base de datos Multi-AZ. Amazon Timestream para InfluxDB utiliza la tecnología de conmutación por error de Amazon.

En una implementación de instancia de base de datos Multi-AZ, Amazon Timestream aprovisiona y mantiene automáticamente una réplica en espera sincrónica en una zona de disponibilidad diferente. La instancia de base de datos principal se replica de forma síncrona en las zonas de disponibilidad en una réplica en espera para proporcionar redundancia de datos. La ejecución de una instancia de base de datos con alta disponibilidad puede mejorar la disponibilidad durante un fallo de la instancia de base de datos y una interrupción en la zona de disponibilidad. Para obtener más información acerca de las zonas de disponibilidad, consulte AWS Regiones y zonas de disponibilidad.

> **Note**
>
> La opción de alta disponibilidad no es una solución de escalado para escenarios de solo lectura. No puede usar una réplica en espera para servir tráfico de lectura.

Con la consola Amazon Timestream, puede crear un despliegue de instancias de base de datos Multi-AZ simplemente especificando la opción Crear una instancia en espera en la sección de configuración de disponibilidad y durabilidad al crear una instancia de base de datos. También puede especificar una implementación de instancia de base de datos Multi-AZ con Amazon AWS Command Line Interface Timestream. API Utilice el CLI comando `create-db-instance` o o la operación. `CreateDBInstance` API

Las instancias de base de datos que usan implementaciones de bases de datos Multi-AZ pueden tener una latencia de escritura y confirmación superior a la de una implementación Single-AZ. Esto puede ocurrir debido a la replicación de datos síncrona que se produce. Es posible que se produzca un cambio en la latencia si la implementación se conmuta por error a la réplica en espera, aunque AWS está diseñada con conectividad de red de baja latencia entre las zonas de disponibilidad. Para las cargas de trabajo de producción, le recomendamos que utilice el almacenamiento IOPS incluido de 12 000 o 16 000 000 IOPS para obtener un rendimiento rápido y uniforme. Para obtener más información sobre las clases de instancias de bases de datos, consulte Clases de instancia de base de datos.

Configuración y administración de una implementación Multi-AZ

El flujo temporal de las implementaciones Multi-AZ de InfluxDB solo puede tener una en espera. Cuando la implementación tiene una instancia de base de datos en espera, se denomina implementación de instancia de base de datos Multi-AZ. Una implementación de instancia de base de datos Multi-AZ tiene una instancia de base de datos en espera que proporciona compatibilidad con la conmutación por error, pero no sirve tráfico de lectura.

> **Important**
>
> La instancia debe tener al menos dos subredes asociadas para ejecutar actualizaciones de una zona de disponibilidad única a una zona de disponibilidad múltiple. Una vez creada la instancia, no puedes modificar su modo de implementación de Single-AZ a Multi-AZ.

Puede usarlo AWS Management Console para determinar si su instancia de base de datos es una implementación en una zona de disponibilidad única o en una zona de disponibilidad múltiple.

Uso del AWS Management Console

1. Inicie sesión en la consola Amazon Timestream for InfluxDB AWS Management Console y ábrala.

2. En el panel de navegación, elija InfluxDB Databases y, a continuación, elija el identificador de base de datos.

Una implementación de instancia de base de datos Multi-AZ tiene las siguientes características:

- Solo hay una fila para la instancia de base de datos.
- El valor de Role (Rol) es Instance (Instancia) o Primary (Principal).
- El valor de Multi-AZ es Yes (Sí).

Proceso de conmutación por error para Amazon Timestream

Si una interrupción planificada o imprevista de su instancia de base de datos se debe a un defecto de infraestructura, Amazon Timestream for InfluxDB cambia automáticamente a una réplica en espera en otra zona de disponibilidad si ha activado Multi-AZ. El tiempo requerido para completar la conmutación por error dependerá de la actividad de la base de datos y de otras condiciones existentes cuando la instancia de base de datos principal dejó de estar disponible. Los tiempos de conmutación por error suelen estar entre 60–120 segundos. Sin embargo, las transacciones grandes o un proceso de recuperación largo pueden aumentar el tiempo de conmutación por error. Cuando se complete la conmutación por error, la consola Timestream puede tardar más tiempo en reflejar la nueva zona de disponibilidad.

> **Note**
>
> Amazon Timestream gestiona las conmutaciones por error automáticamente para que pueda reanudar las operaciones de la base de datos lo más rápido posible sin intervención administrativa. La instancia de base de datos principal conmuta automáticamente a la réplica en espera si se da cualquiera de las condiciones descritas en la siguiente tabla.

Motivo de la conmutación por error	Descripción
El sistema operativo subyacente a la instancia de base de datos de Timestream está siendo parcheado en una operación fuera de línea.	Se ha desencadenado una conmutación por error durante la ventana de mantenimiento para un parche del sistema operativo o una actualización de seguridad.

Motivo de la conmutación por error	Descripción
El host principal de la instancia Multi-AZ de Timestream no funciona correctamente.	La implementación de una instancia de base de datos Multi-AZ detectó una instancia de base de datos principal deteriorada y se produjo una conmutación por error.
No se puede acceder al host principal de la instancia Multi-AZ de Timestream debido a la pérdida de conectividad de red.	La supervisión de Timestream detectó un error de acceso a la red en la instancia de base de datos principal y provocó una conmutación por error.
El cliente modificó la instancia de Timestream.	Una modificación de la instancia de base de datos de Timesteam for InfluxDB provocó una conmutación por error. Para obtener más información, consulte Actualización de instancias de base de datos.
La instancia principal Multi-AZ de Timestream está ocupada y no responde.	La instancia de base de datos principal no responde. Le recomendamos que haga lo siguiente: * Examine el evento para detectar un uso excesivo CPU de memoria o espacio de intercambio. * Evalúe su carga de trabajo para determinar si está utilizando la clase de instancia de base de datos adecuada. Para obtener más información, consulte Clases de instancia de base de datos.
Se produjo un error en el volumen de almacenamiento subyacente al host principal de la instancia Multi-AZ de Timestream.	La implementación de una instancia de base de datos Multi-AZ detectó un problema de almacenamiento en la instancia de base de datos principal y se produjo una conmutación por error.

Configurar el formulario para las búsquedas de nombres JVM TTL DNS

El mecanismo de conmutación por error cambia automáticamente el registro del sistema de nombres de dominio (DNS) de la instancia de base de datos para que apunte a la instancia de base de datos en espera. Como consecuencia, necesita restablecer las conexiones existentes a la instancia de base de datos. En un entorno de máquina virtual Java (JVM), debido al funcionamiento del mecanismo de almacenamiento en DNS caché de Java, es posible que deba volver JVM a configurar los ajustes.

El archivo almacena en JVM caché las búsquedas de nombresDNS. Cuando convierte JVM un nombre de host en una dirección IP, guarda en caché la dirección IP durante un período de tiempo específico, conocido como (). time-to-liveTTL

Como AWS los recursos utilizan entradas de DNS nombres que cambian de vez en cuando, le recomendamos que las configure JVM con un TTL valor no superior a 60 segundos. De este modo, se garantiza que, cuando la dirección IP de un recurso cambie, la aplicación pueda recibir y utilizar la nueva dirección IP del recurso, ya que requiere laDNS.

En algunas configuraciones de Java, el JVM valor predeterminado TTL está establecido para que nunca actualice DNS las entradas hasta que JVM se reinicie. Por lo tanto, si la dirección IP de un AWS recurso cambia mientras la aplicación aún está en ejecución, no podrá usar ese recurso hasta que se reinicie manualmente JVM y se actualice la información de IP almacenada en caché. En este caso, es fundamental configurar la s para TTL que actualice periódicamente la JVM información de IP almacenada en caché.

Puedes obtener el JVM valor predeterminado TTL recuperando el valor de la `networkaddress.cache.ttl` propiedad:

```
String ttl = java.security.Security.getProperty("networkaddress.cache.ttl");
```

> **Note**
>
> El valor predeterminado TTL puede variar en función de la versión que tenga instalada JVM y de si tiene instalado un administrador de seguridad. Muchos JVMs proporcionan un valor predeterminado de TTL menos de 60 segundos. Si utiliza uno de estos JVM y no utiliza un administrador de seguridad, puede ignorar el resto de este tema.
> Para modificarlosTTL, establezca el JVM valor de la propiedad networkaddress.cache.ttl. Utilice uno de los siguientes métodos, en función de sus necesidades:

- Para establecer el valor de la propiedad de forma global para todas las aplicaciones que utilizan el, establézcalo en el archivo. JVM `networkaddress.cache.ttl` `$JAVA_HOME/jre/lib/security/java.security`

```
networkaddress.cache.ttl=60
```

- Para establecer la propiedad localmente sólo para la aplicación, establezca `networkaddress.cache.ttl` en el código de inicialización de la aplicación antes de establecer las conexiones de red.

```
java.security.Security.setProperty("networkaddress.cache.ttl" , "60");
```

Configuración para ver los registros de InfluxDB en las instancias de Timestream Influxdb

De forma predeterminada, InfluxDB genera registros que van a stdout. Para obtener más información, consulte Administrar los registros de InfluxDB

Para ver los registros de InfluxDB generados a partir de la instancia que ha creado a través de Timestream InfluxDB, ofrecemos la posibilidad de proporcionar registros por hora. Estos registros irán a un depósito de S3 específico que debe crear antes de crear la instancia.

- Antes de crear la instancia, el bucket de Amazon S3 proporcionado también debe dar permiso a Timestream-InfluxDB para enviar registros a este bucket. Para ello, debe proporcionar una política de bucket con el director de servicio de Timestream InfluxDB de la siguiente manera (sustituya *{BUCKET_NAME}* con el nombre real de su bucket de Amazon S3:

```
{
    "Version": "2012-10-17",
    "Id": "PolicyForInfluxLogs",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "timestream-influxdb.amazonaws.com"
            },
            "Action": "s3:PutObject",
```

```
            "Resource": "arn:aws:s3:::{BUCKET_NAME}/InfluxLogs/*"
        }
    ]
}
```

- El bucket proporcionado debe estar en la misma cuenta y región de la instancia de Timestream InfluxDB creada

  Este es el comando al que puedes llamar para crear una instancia que reciba los registros de afluencia:

```
aws timestream-influxdb create-db-instance \
    --name myinfluxDbinstance \
    --allocated-storage 400 \
    --db-instance-type db.influx.4xlarge \
    --vpc-subnet-ids subnetid1 subnetid2
    --vpc-security-group-ids mysecuritygroup \
    --username masterawsuser \
    --password \
    --db-storage-type InfluxIOIncludedT2
```

  Este es el formato de este parámetro.

```
-- log-delivery-configuration
{
    "S3Configuration": {
      "BucketName": "string",
      "Enabled": true|false
    }
}
```

- Este campo no es obligatorio y el registro no está activado de forma predeterminada.
- No configurar este campo equivale a no tener habilitados los registros.
- Los registros se enviarán al depósito especificado con el prefijo de`InfluxLogs/`.
- Tras crear la instancia, puede modificar la configuración de entrega de registros con el `update-db-instance` API comando.

InfluxDB ofrece diferentes tipos de registros. Estos se pueden configurar configurando los parámetros de InfluxDB. Utilice los parámetros flux-log-enabled y a nivel de registro para configurar

el tipo de registros que se emiten desde la instancia. Para obtener más información, consulte Parámetros y valores de parámetros admitidos.

Agregar etiquetas a los recursos

Puede etiquetar Amazon Timestream para los recursos de InfluxDB mediante etiquetas. Las etiquetas le permiten categorizar sus recursos de diferentes maneras, por ejemplo, por finalidad, propietario, entorno u otros criterios. Las etiquetas pueden ayudar a hacer lo siguiente:

- Identificar rápidamente un recurso según las etiquetas que le haya asignado.
- Consulte las AWS facturas desglosadas por etiquetas.

El etiquetado es compatible con AWS servicios como Amazon Elastic Compute Cloud (AmazonEC2), Amazon Simple Storage Service (Amazon S3), Timestream for InfluxDB y más. Un etiquetado eficiente puede ofrecerle información detallada sobre los costos permitiendo crear informes sobre los servicios que llevan una etiqueta determinada.

Por último, es conveniente seguir estrategias de etiquetado óptimas. Para obtener más información, consulte Estrategias de etiquetado de AWS.

Restricciones de etiquetado

Cada etiqueta consta de una clave y un valor, ambos definidos por el usuario. Se aplican las siguientes restricciones:

- Cada instancia de base de datos Timestream for InfluxDB solo puede tener una etiqueta con la misma clave. Si intenta añadir una etiqueta existente, el valor de la etiqueta existente se actualiza al nuevo valor.
- Un valor actúa como descriptor dentro de una categoría de etiquetas. En Timestream para InfluxDB, el valor no puede estar vacío ni ser nulo.
- Las claves y los valores de las etiquetas distinguen entre mayúsculas y minúsculas.
- La longitud máxima de la clave es de 128 caracteres Unicode.
- La longitud máxima del valor es de 256 caracteres Unicode.
- Los caracteres permitidos son letras, espacios en blanco y números, además de los caracteres especiales siguientes: `+ - = . _ : /`
- El número máximo de etiquetas por recurso es 50.

- AWS-a los nombres y valores de las etiquetas asignadas se les asigna automáticamente el `aws:` prefijo, que no puede asignar. AWS-los nombres de etiqueta asignados no cuentan para el límite de 50 etiquetas. Los nombres de etiquetas asignados por el usuario presentan el prefijo `user:` en el informe de asignación de costos.
- No es posible antedatar la aplicación de una etiqueta.

Prácticas recomendadas de seguridad para Timestream for InfluxDB

Optimiza las escrituras en InfluxDB

Como cualquier otra base de datos de series temporales, InfluxDB está diseñada para poder ingerir y procesar datos en tiempo real. Para que el sistema funcione al máximo, recomendamos las siguientes optimizaciones al escribir datos en InfluxDB:

- Escrituras por lotes: al escribir datos en InfluxDB, escriba los datos en lotes para minimizar la sobrecarga de red relacionada con cada solicitud de escritura. El tamaño de lote óptimo es de 5000 líneas de protocolo de línea por solicitud de escritura. Para escribir varias líneas en una solicitud, cada línea del protocolo de línea debe estar delimitada por una línea nueva (\n).
- Ordenar etiquetas por clave: antes de escribir puntos de datos en InfluxDB, ordene las etiquetas por clave en orden lexicográfico.

```
measurement,tagC=therefore,tagE=am,tagA=i,tagD=i,tagB=think fieldKey=fieldValue
 1562020262

# Optimized line protocol example with tags sorted by key
measurement,tagA=i,tagB=think,tagC=therefore,tagD=i,tagE=am fieldKey=fieldValue
 1562020262
```

- Utilice la precisión de tiempo más aproximada posible: — InfluxDB escribe los datos con una precisión de nanosegundos; sin embargo, si los datos no se recopilan en nanosegundos, no es necesario escribir con esa precisión. Para obtener un mejor rendimiento, utilice la mayor precisión posible para las marcas de tiempo. Puede especificar la precisión de escritura cuando:
  - Al utilizar elSDK, puede especificarlo WritePrecision al configurar el atributo de tiempo de su punto. Para obtener más información sobre las bibliotecas cliente de InfluxDB, consulte la documentación de InfluxDB.

- Al utilizar Telegraf, se configura la precisión del tiempo en la configuración del agente de Telegraf. La precisión se especifica como un intervalo con una unidad entera + (por ejemplo, 0s,10ms,2us,4s). Las unidades de tiempo válidas son «ns», «us», «ms» y «s».

```
[agent]
 interval ="10s"
 metric_batch_size="5000"
 precision = "0s"
```

- Utilice la compresión gzip: — Utilice la compresión gzip para acelerar las escrituras en InfluxDB y reducir el ancho de banda de la red. Los puntos de referencia han demostrado una mejora de velocidad de hasta 5 veces cuando se comprimen los datos.
  - Cuando utilices Telegraf, en la configuración del plugin de salida InfluxDB_v2 de tu telegraf.conf, establece la opción content_encoding en gzip:

```
[[outputs.influxdb_v2]]
  urls = ["http://localhost:8086"]
  # ...
  content_encoding = "gzip"
```

  - Al utilizar bibliotecas cliente, cada biblioteca cliente de InfluxDB ofrece opciones para comprimir las solicitudes de escritura o aplica la compresión de forma predeterminada. El método para habilitar la compresión es diferente para cada biblioteca. Para obtener instrucciones específicas, consulte la documentación de InfluxDB
  - Cuando utilice el API `/api/v2/write` punto final de InfluxDB para escribir datos, comprima los datos con gzip y establezca el encabezado Content-Encoding en gzip.

Diseñe pensando en el rendimiento

Diseñe su esquema para consultas más sencillas y de mayor rendimiento. Las siguientes pautas garantizarán que su esquema sea fácil de consultar y maximizarán el rendimiento de las consultas:

- Diseñe para la consulta: elija medidas, claves de etiquetas y claves de campo que sean fáciles de consultar. Para lograr este objetivo, siga estos principios:
  - Utilice medidas que tengan un nombre sencillo y que describan el esquema con precisión.
  - Evite usar el mismo nombre para una clave de etiqueta y una clave de campo dentro del mismo esquema.

  - Evite utilizar palabras clave y caracteres especiales reservados a Flux en las claves de etiquetas y campos.
  - Las etiquetas almacenan metadatos que describen los campos y son comunes en muchos puntos de datos.
  - Los campos almacenan datos únicos o muy variables, normalmente puntos de datos numéricos.
  - Las medidas y las claves no deben contener datos, sino que deben usarse para agregar o describir datos. Los datos se almacenarán en valores de etiquetas y campos.
- Mantenga la cardinalidad de las series temporales bajo control La alta cardinalidad de las series temporales es una de las principales causas de la disminución del rendimiento de escritura y lectura en InfluxDB. En el contexto de InfluxDB, la alta cardinalidad se refiere a la presencia de una gran cantidad de valores de etiquetas únicos. Los valores de las etiquetas están indexados en InfluxDB, lo que significa que un número muy alto de valores únicos generará un índice mayor, lo que puede ralentizar la ingesta de datos y el rendimiento de las consultas.

  Para comprender y resolver mejor los posibles problemas relacionados con la alta cardinalidad, puede seguir estos pasos:

  - Comprende las causas de la alta cardinalidad
  - Mide la cardinalidad de tus cubos
  - Toma medidas para resolver la alta cardinalidad
- Causas de la alta cardinalidad de las series InfluxDB indexa los datos en función de las mediciones y las etiquetas para acelerar la lectura de los datos. Cada conjunto de elementos de datos indexados forma una clave de serie. Las etiquetas que contienen información muy variable, como cadenas únicasIDs, códigos hash y cadenas aleatorias, conducen a un gran número de series, lo que también se conoce como cardinalidad de series altas. La alta cardinalidad de las series es el principal impulsor del uso elevado de memoria en InfluxDB.
- Medición de la cardinalidad de la serie Si experimenta una ralentización del rendimiento o observa un uso de memoria cada vez mayor en su instancia de Timestream for InfluxDB, le recomendamos que mida la cardinalidad de serie de sus segmentos.

  InfluxDB proporciona funciones que permiten medir la cardinalidad de las series tanto en Flux como en InfluxQL.

  - En Flux, usa la función `influxdb.cardinality()`
  - En FluxQL usa el comando `SHOW SERIES CARDINALITY`

En ambos casos, el motor devolverá el número de claves de serie únicas de sus datos. Tenga en cuenta que no se recomienda tener más de 10 millones de claves de serie en ninguna de sus instancias de Timestream para InfluxDB.

- Causas de la alta cardinalidad de las series Si descubres que alguno de tus cubos tiene una cardinalidad alta, puedes tomar algunas medidas correctivas para solucionarlo:
  - Revisa tus etiquetas: asegúrate de que tus cargas de trabajo no generen casos en los que las etiquetas tengan valores únicos para la mayoría de las entradas. Esto puede suceder en los casos en que el número de valores de etiquetas únicos siempre aumente con el tiempo, o si los mensajes de tipo registro se escriben en la base de datos, donde cada mensaje tendría una combinación única de marcas de tiempo, etiquetas, etc. Puedes usar el siguiente código de Flux para ayudarte a determinar qué etiquetas son las que más contribuyen a tus problemas de alta cardinalidad:

```
// Count unique values for each tag in a bucketimport "influxdata/influxdb/schema"

cardinalityByTag = (bucket) => schema.tagKeys(bucket: bucket)
    |> map(
        fn: (r) => ({
            tag: r._value,
            _value: if contains(set: ["_stop", "_start"], value: r._value) then
                0
            else
                (schema.tagValues(bucket: bucket, tag: r._value)
                    |> count()
                    |> findRecord(fn: (key) => true, idx: 0))._value,
        }),
    )
    |> group(columns: ["tag"])
    |> sum()

cardinalityByTag(bucket: "example-bucket")
```

Si tienes una cardinalidad muy alta, es posible que se agote el tiempo de espera de la consulta anterior. Si se agota el tiempo de espera, ejecuta las siguientes consultas, de una en una.

Genera una lista de etiquetas:

```
// Generate a list of tagsimport "influxdata/influxdb/schema"
```

```
schema.tagKeys(bucket: "example-bucket")
```

Cuente los valores de etiqueta únicos para cada etiqueta:

```
// Run the following for each tag to count the number of unique tag valuesimport
 "influxdata/influxdb/schema"

tag = "example-tag-key"

schema.tagValues(bucket: "my-bucket", tag: tag)
    |> count()
```

Te recomendamos que los ejecutes en diferentes momentos para identificar qué etiqueta está creciendo más rápido.

- Mejore su esquema: siga las recomendaciones de modelado que se describen en nuestraPrácticas recomendadas de seguridad para Timestream for InfluxDB.
- Elimine o agregue datos antiguos para reducir la cardinalidad: considere si sus casos de uso necesitan o no todos los datos que están causando sus problemas de alta cardinalidad. Si estos datos ya no son necesarios o no se accede a ellos con frecuencia, puede agregarlos, eliminarlos o exportarlos a otro motor, como Timestream para Live Analytics, para almacenarlos y analizarlos a largo plazo.

Solución de problemas

Advertencia de que no se reconoce la versión «dev»

Es posible que aparezca la advertencia «WARNNo se pudo analizar la versión «dev» reportada por el servidor, suponiendo que APIs se admitan las últimas copias de seguridad o restauraciones» durante la migración. Esta advertencia puede ignorarse.

La migración falló durante la etapa de restauración

En caso de que se produzca un error en la migración durante la fase de restauración, los usuarios pueden utilizar el `--retry-restore-dir` indicador para volver a intentar la restauración. Utilice la `--retry-restore-dir` marca con una ruta a un directorio del que se haya hecho una copia de seguridad anteriormente para saltarse la fase de copia de seguridad y volver a intentar la fase de

restauración. El directorio de copia de seguridad creado y utilizado para la migración se indicará si la migración falla durante la restauración.

Entre los posibles motivos por los que se produce un error en la restauración se incluyen:

- Token de destino de InfluxDB no válido: un depósito que existe en la instancia de destino con el mismo nombre que en la instancia de origen. Para las migraciones de buckets individuales, usa la `--dest-bucket` opción de establecer un nombre único para el bucket migrado
- Fallo de conectividad, ya sea con los hosts de origen o destino o con un bucket S3 opcional.

Directrices operativas básicas de Amazon Timestream para InfluxDB

Las siguientes son pautas operativas básicas que todo el mundo debería seguir cuando trabaje con Amazon Timestream para InfluxDB. Tenga en cuenta que el acuerdo de nivel de servicio de Amazon Timestream para InfluxDB requiere que siga estas pautas:

- Utilice métricas para supervisar el uso de memoria y almacenamientoCPU. Puedes configurar Amazon CloudWatch para que te notifique cuando cambien los patrones de uso o cuando te acerques a la capacidad de tu implementación. De esta forma, puede mantener el rendimiento y la disponibilidad del sistema.
- Escale la instancia de base de datos cuando se esté acercando a los límites de la capacidad de almacenamiento. Debe tener búfer de almacenamiento y de memoria para asumir incrementos imprevistos de la demanda de las aplicaciones. Tenga en cuenta que, en este momento, necesitará crear una nueva instancia y migrar sus datos para lograrlo.
- Si la carga de trabajo de la base de datos requiere más E/S de la aprovisionada, la recuperación tras una conmutación por error o tras un error de la base de datos será lenta. Para incrementar la capacidad de E/S de una instancia de base de datos, lleve a cabo una de las acciones siguientes o todas ellas:
  - Migre a una instancia de base de datos diferente con mayor capacidad de E/S.
  - Si ya utiliza el almacenamiento de almacenamiento IOPS incluido en Influx, aprovisione un tipo de almacenamiento con una capacidad superior IOPS incluida.
- Si su aplicación cliente almacena en caché los datos del Servicio de nombres de dominio (DNS) de sus instancias de base de datos, establezca un valor time-to-live (TTL) inferior a 30 segundos. La dirección IP subyacente de una instancia de base de datos puede cambiar después de producirse una conmutación por error. Por lo tanto, almacenar los DNS datos en caché durante un tiempo

prolongado puede provocar fallos de conexión. Es posible que tu aplicación intente conectarse a una dirección IP que ya no esté en servicio.

Recomendaciones sobre instancias RAM de base de datos

Una buena práctica de rendimiento de Amazon Timestream para InfluxDB es asignar RAM lo suficiente para que el conjunto de trabajo resida casi por completo en la memoria. El conjunto de trabajo son los datos e índices que se usan con frecuencia en su instancia. Cuanto más use la instancia de base de datos, más crecerá el conjunto de trabajo.

Seguridad en Timestream para InfluxDB

La seguridad en la nube es la máxima prioridad. AWS Como AWS cliente, usted se beneficia de una arquitectura de centro de datos y red diseñada para cumplir con los requisitos de las organizaciones más sensibles a la seguridad.

La seguridad es una responsabilidad compartida entre usted AWS y usted. El modelo de responsabilidad compartida la describe como seguridad de la nube y seguridad en la nube:

- Seguridad de la nube: AWS es responsable de proteger la infraestructura que ejecuta AWS los servicios en la AWS nube. AWS también le proporciona servicios que puede utilizar de forma segura. Auditores externos prueban y verifican periódicamente la eficacia de nuestra seguridad en el marco de los programas de conformidad de AWS. Para obtener más información sobre los programas de cumplimiento que se aplican a Timestream para InfluxDB, consulte los AWS servicios incluidos en el ámbito de aplicación por programa de cumplimiento.
- Seguridad en la nube: su responsabilidad viene determinada por el AWS servicio que utilice. Usted también es responsable de otros factores incluida la confidencialidad de los datos, los requisitos de la empresa y la legislación y los reglamentos aplicables.

Esta documentación lo ayudará a comprender cómo aplicar el modelo de responsabilidad compartida al utilizar Timestream para InfluxDB. Los siguientes temas le muestran cómo configurar Timestream para InfluxDB a fin de cumplir sus objetivos de seguridad y conformidad. También aprenderá a usar otros AWS servicios que pueden ayudarlo a monitorear y proteger sus recursos de Timestream for InfluxDB.

Temas

- Información general

- Autenticación de bases de datos con Amazon Timestream para InfluxDB
- Cómo utiliza Amazon Timestream para InfluxDB los secretos
- Protección de datos en Timestream para InfluxDB
- Identity and Access Management para Amazon Timestream para InfluxDB
- Registro y monitoreo en Timestream para InfluxDB
- Validación de conformidad de Amazon Timestream para InfluxDB
- Resiliencia en Amazon Timestream para InfluxDB
- Seguridad de infraestructura en Amazon Timestream para InfluxDB
- Análisis de configuración y vulnerabilidad en Timestream para InfluxDB
- Respuesta a incidentes en Timestream para InfluxDB
- Amazon Timestream para API InfluxDB y puntos finales de interfaz () VPC AWS PrivateLink
- Mejores prácticas de seguridad para Timestream for InfluxDB

Información general

Esta documentación le ayuda a comprender cómo aplicar el modelo de responsabilidad compartida al utilizar Amazon Timestream para InfluxDB. En los temas siguientes, se muestra cómo configurar Amazon Timestream para InfluxDB para cumplir sus objetivos de seguridad y conformidad. También aprenderá a utilizar otros AWS servicios que le ayudan a supervisar y proteger sus recursos de Amazon Timestream for InfluxDB.

Puede administrar el acceso a sus recursos de Amazon Timestream for InfluxDB y a sus bases de datos en una instancia de base de datos. El método que utilice para administrar el acceso depende del tipo de tarea que el usuario deba realizar con Amazon Timestream para InfluxDB:

- Ejecute su instancia de base de datos en una nube privada virtual (VPC) basada en el VPC servicio de Amazon para el control de acceso a la red.
- Utilice las políticas de AWS Identity and Access Management (IAM) para asignar permisos que determinen quién puede gestionar los recursos de Amazon Timestream para InfluxDB. Por ejemplo, puede utilizarlas IAM para determinar quién puede crear, describir, modificar y eliminar instancias de base de datos, etiquetar recursos o modificar grupos de seguridad.
- Utilice grupos de seguridad para controlar qué direcciones IP o EC2 instancias de Amazon pueden conectarse a las bases de datos de una instancia de base de datos. Cuando crea una instancia de

base de datos por primera vez, solo se puede acceder a ella mediante reglas especificadas por un grupo de seguridad asociado.

- Utilice conexiones Secure Socket Layer (SSL) o Transport Layer Security (TLS) con sus instancias de base de datos.
- Utilice las funciones de seguridad de su motor InfluxDB para controlar quién puede iniciar sesión en las bases de datos de una instancia de base de datos. Estas características funcionas de igual forma que si la base de datos estuviera en su red local. Para obtener más información, consulte Seguridad en Timestream para InfluxDB.

> **Note**
>
> Solo tiene que configurar la seguridad para sus casos de uso. No tiene que configurar el acceso de seguridad para los procesos que administra Amazon Timestream for InfluxDB. Estos incluyen la creación de copias de seguridad, la replicación de datos entre una instancia de base de datos primaria y una réplica de lectura, y otros procesos.

Temas

Seguridad general

Temas

Permisos

A los usuarios de InfluxDB se les deben conceder los permisos de mínimo privilegio. Durante la migración, solo se deben usar los tokens otorgados a usuarios específicos, en lugar de los tokens de operador.

Timestream for InfluxDB usa IAM permisos para controlar los permisos de los usuarios. Recomendamos que los usuarios tengan acceso a las acciones y recursos específicos que necesiten. Para obtener más información, consulte Otorgar acceso con privilegios mínimos.

Acceso a la red

El script de migración de Influx puede funcionar localmente, migrando datos entre dos instancias de InfluxDB del mismo sistema, pero se supone que el caso de uso principal de las migraciones será la migración de datos a través de la red, ya sea una red local o pública. Esto conlleva consideraciones de seguridad. El script de migración de Influx verificará, de forma predeterminada, TLS los certificados de las instancias que TLS estén habilitadas: recomendamos a los usuarios que habiliten la opción TLS en sus instancias de InfluxDB y no usen la `--skip-verify` opción para el script.

Le recomendamos que utilice una lista de permitidos para impedir que el tráfico de red provenga de las fuentes esperadas. Para ello, limite el tráfico de red a las instancias de InfluxDB únicamente desde las conocidas. IPs

Dependencias

Deben usarse las últimas versiones principales de todas las dependencias, incluidas Influx, InfluxDBCLI, Python, el módulo Requests y las dependencias opcionales como y. `mountpoint-s3` `rclone`

Buckets de S3

Si los buckets de S3 se utilizan como almacenamiento temporal para la migración, recomendamos habilitarTLS, versionar y deshabilitar el acceso público.

Uso de cubos de S3 para la migración

1. Abre el AWS Management Console, navega hasta Amazon Simple Storage Service y, a continuación, selecciona Buckets.
2. Elige el depósito que deseas usar.
3. Elija la pestaña Permisos.
4. En Block public access (bucket settings) (Bloquear acceso público [configuración de bucket]), elija Edit (Editar).
5. Marca Bloquear todo el acceso público.
6. Elija Guardar cambios.

7. En Política de bucket, elija Editar.
8. Introduce lo siguiente y <example-bucket>sustitúyelo por el nombre de tu bucket para imponer el uso de la TLS versión 1.2 o posterior para las conexiones:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EnforceTLSv12orHigher",
            "Principal": {
                "AWS": "*"
            },
            "Action": [
                "s3:*"
            ],
            "Effect": "Deny",
            "Resource": [
                "arn:aws:s3:::<example bucket>/*",
                "arn:aws:s3:::<example bucket>"
            ],
            "Condition": {
                "NumericLessThan": {
                    "s3:TlsVersion": 1.2
                }
            }
        }
    ]
}
```

9. Elija Guardar cambios.
10. Elija la pestaña Properties (Propiedades).
11. En Bucket Versioning (Versiones del bucket), elija Edit (Editar).
12. Marca Activar.
13. Elija Guardar cambios.

Para obtener información sobre las prácticas de seguridad recomendadas para los buckets de Amazon S3, consulte Prácticas recomendadas de seguridad para Amazon Simple Storage Service.

Autenticación de bases de datos con Amazon Timestream para InfluxDB

Amazon Timestream para InfluxDB admite dos formas de autenticar a los usuarios de bases de datos.

La autenticación de la base de datos mediante contraseña y token de acceso utiliza diferentes métodos de autenticación en la base de datos. Por lo tanto, un usuario específico puede iniciar sesión en una base de datos mediante un solo método de autenticación. En ambos casos, InfluxDB realiza toda la administración de las cuentas de usuario y los tokens. API

Autenticación de contraseña

Durante el proceso de creación de la instancia de base de datos de InfluxDB, creó una organización, un usuario y una contraseña. El usuario tiene permisos para gestionar todo el contenido de su instancia de base de datos Timestream for InfluxDB. Con esta combinación de nombre de usuario y contraseña, podrá acceder a LogIn su instancia mediante InfluxUI y también utilizar Influx CLI para generar un token de operador.

Se necesita un token de operador para crear usuarios, eliminar grupos, organizaciones, etc. Para obtener más información, consulte Opciones de autenticación de bases de datos.

APIfichas

APILos tokens de InfluxDB garantizan una interacción segura entre InfluxDB y herramientas externas, como clientes o aplicaciones. Un API token pertenece a un usuario específico e identifica los permisos de InfluxDB dentro de la organización del usuario.

Hay tres tipos de API tokens en InfluxDB:

- Token de operador: otorga acceso completo de lectura y escritura a todas las organizaciones y todos los recursos de la organización en OSS InfluxDB 2.x. Algunas operaciones, por ejemplo, la recuperación de la configuración del servidor, requieren permisos de operador. Para crear un token de operador manualmente con la interfaz de usuario de InfluxDB `api/v2`API, o Influx una CLI vez finalizado el proceso de configuración, debe utilizar un token de operador existente o su nombre de usuario y contraseña. Para crear un nuevo token de operador sin usar uno existente, consulta la autenticación de recuperación de influxd. CLI

> **Important**
>
> Como los tokens de operador tienen acceso completo de lectura y escritura a todas las organizaciones de la base de datos, recomendamos crear un token de acceso total para cada organización y usarlo para administrar InfluxDB. Esto ayuda a evitar interacciones accidentales entre las organizaciones.

- APIToken de acceso total: otorga acceso completo de lectura y escritura a todos los recursos de una organización.
- Tokens de lectura/escritura: otorga acceso de lectura, de escritura o ambos a grupos específicos de una organización.

Todos los InfluxDb tokens tienen una larga vida útil y no tienen una fecha de caducidad establecida, por lo que no se recomienda utilizar los tokens de operador o todos los de acceso para enviar datos de monitorización desde sus clientes o agentes de Telegraf ni para integrarlos en las aplicaciones de su panel de control. Para estas aplicaciones, cree tokens de lectura/escritura con los permisos necesarios para realizar su trabajo. Para obtener más información sobre cómo crear un token de InfluxDB, consulte Crear un token.

Secretos

Los tokens de operador de InfluxDB se generan al configurar la instancia; se pueden crear otros tipos de tokens, como los de acceso total y de lectura/escritura, mediante las funciones de rotación multiusuario InfluxCLI, Influx v2 API o Timestream for InfluxDB. Consulte Administrar los API tokens para saber cómo generar, ver, asignar y eliminar los tokens.

Le recomendamos que rote el Timestream para los tokens de InfluxDB, utilizando AWS Secrets Manager y almacenando los tokens a través de variables de entorno. Consulte Uso de tokens para conocer el uso de los tokens en las variables de entorno y Rotación del secreto para saber cómo rotar el Timestream para los usuarios y los tokens de InfluxDB.

Véase también:

- Seguridad de infraestructura en Amazon Timestream para InfluxDB
- Mejores prácticas de seguridad para Timestream for InfluxDB

Cómo utiliza Amazon Timestream para InfluxDB los secretos

Timestream para InfluxDB admite la autenticación de nombres de usuario y contraseñas a través de la interfaz de usuario y las credenciales simbólicas para las conexiones de clientes y aplicaciones con privilegios mínimos. Los usuarios de Timestream for InfluxDB tienen `allAccess` permisos dentro de su organización, mientras que los tokens pueden tener cualquier conjunto de permisos. Siguiendo las mejores prácticas para una gestión segura de los API tokens, los usuarios deberían estar preparados para gestionar los tokens y así poder acceder a ellos de forma más precisa dentro de una organización. Puede encontrar información adicional sobre las mejores prácticas de administración con Timestream para InfluxDB en la documentación de Influxdata.

AWS Secrets Manager es un servicio de almacenamiento secreto que puede utilizar para proteger las credenciales, API claves y otra información secreta de la base de datos. Luego, en tu código, puedes reemplazar las credenciales codificadas por una API llamada a Secrets Manager. Esto ayuda a garantizar que el secreto no se vea comprometido por alguien que examine tu código, ya que el secreto no está ahí. Para obtener una descripción general de Secrets Manager, consulte Qué es AWS Secrets Manager.

Al crear una instancia de base de datos, Timestream para InfluxDB crea automáticamente un secreto de administrador para que lo utilice con la función de rotación multiusuario. AWS Lambda Para rotar el Timestream para los usuarios y los tokens de InfluxDB, debes crear manualmente un nuevo secreto para cada usuario o token que desees rotar. Cada secreto se puede configurar para que gire según un cronograma con el uso de una función Lambda. El proceso para configurar un nuevo secreto rotativo consiste en cargar el código de la función Lambda, configurar el rol de Lambda, definir el nuevo secreto y configurar el programa de rotación secreto.

Qué hay en el secreto

Cuando almacene las credenciales de usuario de Timestream for InfluxDB en el secreto, utilice el siguiente formato.

Usuario único:

```
{
  "engine": "<required: must be set to 'timestream-influxdb'>",
  "username": "<required: username>",
  "password": "<required: password>",
  "dbIdentifier": "<required: DB identifier>"
}
```

Al crear una instancia de Timestream for InfluxDB, se almacena automáticamente un secreto de administrador en Secrets Manager con las credenciales que se utilizarán con la función Lambda multiusuario. Establézcalo en el `Authentication Properties Secret Manager ARN` valor `adminSecretArn` que aparece en la página de resumen de la instancia de base de datos o en el de un secreto de administrador. ARN Para crear un nuevo secreto de administrador, debe disponer ya de las credenciales asociadas y las credenciales deben tener privilegios de administrador.

Cuando almacene las credenciales del token de Timestream for InfluxDB en el secreto, utilice el siguiente formato.

Multiusuario:

```
{
  "engine": "<required: must be set to 'timestream-influxdb'>",
  "org": "<required: organization to associate token with>",
  "adminSecretArn": "<required: ARN of the admin secret>",
  "type": "<required: allAccess or operator or custom>",
  "dbIdentifier": "<required: DB identifier>",
  "token": "<required unless generating a new token: token being rotated>",
  "writeBuckets": "<optional: list of bucketIDs for custom type token, must be input
 within plaintext panel, for example ['id1','id2']>",
  "readBuckets": "<optional: list of bucketIDs for custom type token, must be input
 within plaintext panel, for example ['id1','id2']>",
  "permissions": "<optional: list of permissions for custom type token, must be input
 within plaintext panel, for example ['write-tasks','read-tasks']>"
}
```

Cuando almacene las credenciales de administrador de Timestream for InfluxDB en secreto, utilice el siguiente formato:

Secreto de administrador:

```
{
  "engine": "<required: must be set to 'timestream-influxdb'>",
  "username": "<required: username>",
  "password": "<required: password>",
  "dbIdentifier": "<required: DB identifier>",
  "organization": "<optional: initial organization>",
  "bucket": "<optional: initial bucket>"
}
```

Para activar la rotación automática del secreto, este debe estar en la JSON estructura correcta. Consulta Rotación del secreto cómo rotar Timestream para ver los secretos de InfluxDB.

Modificación de un secreto

Las credenciales generadas durante el proceso de creación de instancias de Timestream for InfluxDB se almacenan en un secreto de Secrets Manager en su cuenta. El objeto de GetDbInstancerespuesta contiene un `influxAuthParametersSecretArn` que mantiene el nombre del recurso de Amazon (ARN) en ese secreto. El secreto solo se rellenará cuando su instancia de Timestream for InfluxDB esté disponible. Se trata de una READONLY copia, ya que cualquier dato de este secreto no afecta updates/modifications/deletions a la instancia de base de datos creada. Si elimina este secreto, la APIrespuesta seguirá haciendo referencia al secreto eliminadoARN.

Para crear un nuevo token en la instancia de Timestream for InfluxDB en lugar de almacenar las credenciales de token existentes, puede crear tokens que no sean operadores dejando el `token` valor en blanco en el secreto y utilizando la función de rotación multiusuario con la variable de entorno `AUTHENTICATION_CREATION_ENABLED` Lambda establecida en. `true` Si crea un token nuevo, los permisos definidos en el secreto se asignan al token y no se pueden modificar tras la primera rotación correcta. Para obtener más información sobre la rotación de secretos, consulte Rotating AWS Secrets Manager Secrets.

Si se elimina un secreto, no se eliminará el usuario o token asociado en la instancia de Timestream for InfluxDB.

Rotación del secreto

Utilice las funciones Lambda de rotación de uno o varios usuarios de Timestream for InfluxDB para rotar Timestream para las credenciales de usuario y token de InfluxDB. Utilice la función Lambda de usuario único para rotar las credenciales de usuario de su instancia de Timestream for InfluxDB y utilice la función Lambda multiusuario para rotar las credenciales de token de su instancia de Timestream for InfluxDB.

La rotación de usuarios y tokens con las funciones Lambda para uno y varios usuarios es opcional. Las credenciales Timestream for InfluxDB nunca caducan y cualquier credencial expuesta supone el riesgo de que se cometan acciones malintencionadas contra su instancia de base de datos. La ventaja de rotar Timestream para las credenciales de InfluxDB con Secrets Manager es una capa de seguridad adicional que limita el vector de ataque de las credenciales expuestas al período de tiempo hasta el siguiente ciclo de rotación. Si su instancia de base de datos no cuenta

con un mecanismo de rotación, las credenciales expuestas serán válidas hasta que se eliminen manualmente.

Puede configurar Secrets Manager para rotar el secreto automáticamente de acuerdo con la programación que especifique. Esto le permite reemplazar secretos a largo plazo con secretos a corto plazo, lo que contribuye a reducir significativamente el riesgo de peligro. Para obtener más información sobre cómo rotar secretos con Secrets Manager, consulte Rotate AWS Secrets Manager Secrets.

Usuarios rotativos

Al rotar usuarios con la función Lambda de usuario único, se asignará una nueva contraseña aleatoria al usuario después de cada rotación. Para obtener más información sobre cómo habilitar la rotación automática, consulte Configurar la rotación automática para secretos ajenos a las bases de datos AWS Secrets Manager.

Rotación de los secretos de administrador

Para rotar un secreto de administrador, utiliza la función de rotación de un solo usuario. Debe añadir los `dbIdentifier` valores `engine` y al secreto, ya que esos valores no se rellenan automáticamente al inicializar la base de datos. Consulte Qué hay en el secreto la plantilla secreta completa.

Para encontrar un secreto de administrador para una instancia de Timestream for InfluxDB, utilice el secreto de administrador de la página de resumen de la instancia ARN de Timestream for InfluxDB. Se recomienda rotar todos los secretos de administrador de Timestream for InfluxDB, ya que los usuarios administradores tienen permisos elevados para la instancia de Timestream for InfluxDB.

Función de rotación de Lambda

Puede rotar un Timestream para un usuario de InfluxDB con la función de rotación para un solo usuario utilizando la función con un nuevo secreto y añadiendo los campos obligatorios para su usuario de Timestream for InfluxDB. Qué hay en el secreto Para obtener más información sobre las funciones Lambda de rotación secretas, consulte Rotación por función Lambda.

La función de rotación de un solo usuario se autentica con la instancia de base de datos Timestream for InfluxDB utilizando las credenciales definidas en el secreto, luego genera una nueva contraseña aleatoria y establece la nueva contraseña para el usuario. Para obtener más información sobre las funciones Lambda de rotación secretas, consulte Rotación por función Lambda.

Permisos de función de ejecución de funciones Lambda

Utilice la siguiente IAM política como rol para la función Lambda de usuario único. La política otorga a la función Lambda los permisos necesarios para realizar una rotación secreta para los usuarios de Timestream for InfluxDB.

Sustituya todos los elementos que figuran a continuación en la IAM política por los valores de su cuenta: AWS

- {rotating_secret_arn} — El ARN nombre del secreto que se está rotando se encuentra en los detalles secretos de Secrets Manager.
- {db_instance_arn}: la secuencia temporal de la instancia de InfluxDB se encuentra en la página de resumen de la secuencia temporal de la instancia de InfluxDB. ARN

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "secretsmanager:UpdateSecretVersionStage"
            ],
     "Resource": "{rotating_secret_arn}"
        },
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetRandomPassword"
            ],
            "Resource": "*"
        },
        {
            "Action": [
                "timestream-influxdb:GetDbInstance"
            ],
     "Resource": "{db_instance_arn}",
            "Effect": "Allow"
        }
```

```
    ]
}
```

Tokens rotativos

Puedes rotar un token de Timestream for InfluxDB con la función de rotación multiusuario utilizando la opción Qué hay en el secreto con un nuevo secreto y añadiendo los campos obligatorios para tu token de Timestream for InfluxDB. Para obtener más información sobre las funciones Lambda de rotación secretas, consulte Rotación por función Lambda.

Puede rotar un token Timestream for InfluxDB mediante la función Lambda multiusuario Timestream for InfluxDB. Defina la variable de `AUTHENTICATION_CREATION_ENABLED` entorno `true` en la configuración de Lambda para permitir la creación de tokens. Para crear un nuevo token, utilice el Qué hay en el secreto como valor secreto. Omita el par `token` clave-valor en el nuevo secreto y establézcalo en`allAccess`, o `type` defina los permisos específicos y defina el tipo en. `custom` La función de rotación creará un nuevo token durante el primer ciclo de rotación. No puede cambiar los permisos del token editando el secreto después de la rotación y, en cualquier rotación posterior, se utilizarán los permisos establecidos en la instancia de base de datos.

Función de rotación de Lambda

La función de rotación multiusuario rota las credenciales del token al crear un nuevo token idéntico al permiso utilizando las credenciales de administrador del secreto de administración. La función Lambda valida el valor del token en el secreto antes de crear el token de reemplazo, almacena el nuevo valor del token en el secreto y elimina el token anterior. Si la función Lambda está creando un nuevo token, primero validará que la variable de `AUTHENTICATION_CREATION_ENABLED` entorno esté establecida en`true`, que no haya ningún valor de token en el secreto y que el tipo de token no sea un operador de tipo.

Permisos de función de ejecución de funciones Lambda

Utilice la siguiente IAM política como rol para la función Lambda multiusuario. La política otorga a la función Lambda los permisos necesarios para realizar una rotación secreta para los tokens Timestream for InfluxDB.

Sustituya todos los elementos que figuran a continuación en la IAM política por los valores de su cuenta: AWS

- {rotating_secret_arn} — El ARN nombre del secreto que se está rotando se encuentra en los detalles secretos de Secrets Manager.

- {authentication_properties_admin_secret_arn}: el secreto de administrador de Timestream for InfluxDB se encuentra en la página de resumen de la instancia de Timestream for InfluxDB. ARN
- {db_instance_arn}: la secuencia temporal de la instancia de InfluxDB se encuentra en la página de resumen de la secuencia temporal de la instancia de InfluxDB. ARN

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "secretsmanager:UpdateSecretVersionStage"
            ],
     "Resource": "{rotating_secret_arn}"
        },
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue"
            ],
     "Resource": "{authentication_properties_admin_secret_arn}"
        },
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetRandomPassword"
            ],
            "Resource": "*"
        },
        {
            "Action": [
                "timestream-influxdb:GetDbInstance"
            ],
     "Resource": "{db_instance_arn}",
            "Effect": "Allow"
        }
    ]
}
```

Protección de datos en Timestream para InfluxDB

El modelo de se aplica a protección de datos en Amazon Timestream for InfluxDB. Como se describe en este modelo, AWS es responsable de proteger la infraestructura global en la que se ejecutan todos los. Nube de AWS Usted es responsable de mantener el control sobre el contenido alojado en esta infraestructura. Usted también es responsable de las tareas de administración y configuración de seguridad para los Servicios de AWS que utiliza. Para obtener más información sobre la privacidad de los datos, consulte la sección Privacidad de datos FAQ. Para obtener información sobre la protección de datos en Europa, consulte el modelo de responsabilidad AWS compartida y la entrada del GDPR blog sobre AWS seguridad.

Con fines de protección de datos, le recomendamos que proteja Cuenta de AWS las credenciales y configure los usuarios individuales con AWS IAM Identity Center o AWS Identity and Access Management (IAM). De esta manera, solo se otorgan a cada usuario los permisos necesarios para cumplir sus obligaciones laborales. También recomendamos proteger sus datos de la siguiente manera:

- Utilice la autenticación multifactorial (MFA) con cada cuenta.
- UseSSL/TLSpara comunicarse con AWS los recursos. Necesitamos TLS 1.2 y recomendamos TLS 1.3.
- Configure API y registre la actividad del usuario con AWS CloudTrail. Para obtener información sobre el uso de CloudTrail senderos para capturar AWS actividades, consulte Cómo trabajar con CloudTrail senderos en la Guía del AWS CloudTrail usuario.
- Utilice soluciones de AWS cifrado, junto con todos los controles de seguridad predeterminados Servicios de AWS.
- Utilice servicios de seguridad administrados avanzados, como Amazon Macie, que lo ayuden a detectar y proteger los datos confidenciales almacenados en Amazon S3.
- Si necesita entre FIPS 140 y 3 módulos criptográficos validados para acceder a AWS través de una interfaz de línea de comandos o unaAPI, utilice un FIPS terminal. Para obtener más información sobre los FIPS puntos finales disponibles, consulte la Norma federal de procesamiento de información () FIPS 140-3.

Se recomienda encarecidamente no introducir nunca información confidencial o sensible, como, por ejemplo, direcciones de correo electrónico de clientes, en etiquetas o campos de formato libre, tales como el campo Nombre. Esto incluye cuando trabaja con Timestream para InfluxDB u otro tipo de Servicios de AWS uso de la consola,, o. API AWS CLI AWS SDKs Cualquier dato que ingrese en

etiquetas o campos de formato libre utilizados para nombres se puede emplear para los registros de facturación o diagnóstico. Si proporciona una URL a un servidor externo, le recomendamos encarecidamente que no incluya información sobre las credenciales URL para validar su solicitud a ese servidor.

Para obtener información más detallada sobre los temas de protección de datos de Timestream for InfluxDB, como el cifrado en reposo y la administración de claves, seleccione cualquiera de los temas disponibles a continuación.

Temas

Cifrado en reposo

El cifrado en reposo de Timestream para InfluxDB proporciona una seguridad mejorada al cifrar todos sus datos en reposo mediante claves de cifrado almacenadas en ().AWS Key Management ServiceAWS KMS Esta funcionalidad ayuda a reducir la carga y la complejidad operativas que conlleva la protección de información confidencial. Con el cifrado en reposo, puede crear aplicaciones sensibles a la seguridad que necesitan cumplimiento estricto de cifrado y requisitos normativos.

- El cifrado está activado de forma predeterminada en la instancia de base de datos Timestream for InfluxDB y no se puede desactivar. El algoritmo de cifrado AES -256 estándar del sector es el algoritmo de cifrado predeterminado que se utiliza.
- AWS KMS se utiliza para el cifrado en reposo en Timestream para InfluxDB.
- No necesita modificar las aplicaciones cliente de la instancia de base de datos para utilizar el cifrado.

Cifrado en tránsito

Todos sus datos de Timestream for InfluxDB están cifrados en tránsito. De forma predeterminada, todas las comunicaciones hacia y desde Timestream for InfluxDB están protegidas mediante el cifrado Transport Layer Security (). TLS

El tráfico hacia y desde Amazon Timestream para InfluxDB está protegido mediante TLS las versiones compatibles 1.2 o 1.3.

Identity and Access Management para Amazon Timestream para InfluxDB

AWS Identity and Access Management (IAM) es una Servicio de AWS que ayuda al administrador a controlar de forma segura el acceso a los recursos. AWS IAMlos administradores controlan quién puede autenticarse (iniciar sesión) y quién está autorizado (tiene permisos) para usar Timestream para los recursos de InfluxDB. IAMes una Servicio de AWS que puede utilizar sin coste adicional.

Temas

Autenticación con identidades

La autenticación es la forma en que inicias sesión AWS con tus credenciales de identidad. Debe estar autenticado (con quien haya iniciado sesión AWS) como IAM usuario o asumiendo un IAM rol. Usuario raíz de la cuenta de AWS

Puede iniciar sesión AWS como una identidad federada mediante las credenciales proporcionadas a través de una fuente de identidad. AWS IAM Identity Center Los usuarios (IAMIdentity Center), la autenticación de inicio de sesión único de su empresa y sus credenciales de Google o Facebook son ejemplos de identidades federadas. Al iniciar sesión como una identidad federada, el administrador configuró previamente la federación de identidades mediante roles. IAM Cuando accede AWS mediante la federación, asume indirectamente un rol.

Según el tipo de usuario que sea, puede iniciar sesión en el portal AWS Management Console o en el de AWS acceso. Para obtener más información sobre cómo iniciar sesión AWS, consulte Cómo iniciar sesión Cuenta de AWS en su Guía del AWS Sign-In usuario.

Si accede AWS mediante programación, AWS incluye un kit de desarrollo de software (SDK) y una interfaz de línea de comandos (CLI) para firmar criptográficamente sus solicitudes con sus credenciales. Si no utilizas AWS herramientas, debes firmar las solicitudes tú mismo. Para obtener más información sobre cómo usar el método recomendado para firmar las solicitudes usted mismo, consulte la versión 4 de la AWS firma para ver API las solicitudes en la Guía del IAM usuario.

Independientemente del método de autenticación que use, es posible que deba proporcionar información de seguridad adicional. Por ejemplo, le AWS recomienda que utilice la autenticación multifactorial (MFA) para aumentar la seguridad de su cuenta. Para obtener más información, consulte Autenticación multifactorial en la Guía del AWS IAM Identity Center usuario y Autenticación AWS multifactorial IAM en la Guía del IAM usuario.

Usuarios y grupos de IAM

Un IAMusuario es una identidad propia Cuenta de AWS que tiene permisos específicos para una sola persona o aplicación. Siempre que sea posible, recomendamos utilizar credenciales temporales en lugar de crear IAM usuarios con credenciales de larga duración, como contraseñas y claves de acceso. Sin embargo, si tiene casos de uso específicos que requieren credenciales a largo plazo con IAM los usuarios, le recomendamos que rote las claves de acceso. Para obtener más información, consulte Rotar las claves de acceso con regularidad para los casos de uso que requieran credenciales de larga duración en la Guía del IAM usuario.

Un IAMgrupo es una identidad que especifica un conjunto de IAM usuarios. No puede iniciar sesión como grupo. Puede usar los grupos para especificar permisos para varios usuarios a la vez. Los grupos facilitan la administración de los permisos para grandes conjuntos de usuarios. Por ejemplo, puede asignar un nombre a un grupo IAMAdminsy concederle permisos para administrar IAM los recursos.

Los usuarios son diferentes de los roles. Un usuario se asocia exclusivamente a una persona o aplicación, pero la intención es que cualquier usuario pueda asumir un rol que necesite. Los usuarios tienen credenciales de larga duración permanentes; no obstante, los roles proporcionan credenciales temporales. Para obtener más información, consulte Casos de uso para IAM usuarios en la Guía del IAM usuario.

IAMroles

Un IAMrol es una identidad dentro de ti Cuenta de AWS que tiene permisos específicos. Es similar a un IAM usuario, pero no está asociado a una persona específica. Para asumir temporalmente un IAM rol en la AWS Management Console, puede cambiar de un IAM rol de usuario a uno (consola).

Puede asumir un rol llamando a una AWS API operación AWS CLI o o utilizando una operación personalizadaURL. Para obtener más información sobre los métodos de uso de los roles, consulte Métodos para asumir un rol en la Guía del IAM usuario.

IAMlos roles con credenciales temporales son útiles en las siguientes situaciones:

- Acceso de usuario federado: para asignar permisos a una identidad federada, puede crear un rol y definir sus permisos. Cuando se autentica una identidad federada, se asocia la identidad al rol y se le conceden los permisos define el rol. Para obtener información sobre los roles para la federación, consulte Creación de un rol para un proveedor de identidad externo en la Guía del IAM usuario. Si usa IAM Identity Center, configura un conjunto de permisos. Para controlar a qué pueden acceder sus identidades después de autenticarse, IAM Identity Center correlaciona el conjunto de permisos con un rol en. IAM Para obtener información acerca de los conjuntos de permisos, consulte Conjuntos de permisos en la Guía del usuario de AWS IAM Identity Center .
- Permisos IAM de usuario temporales: un IAM usuario o rol puede asumir un IAM rol para asumir temporalmente diferentes permisos para una tarea específica.
- Acceso multicuenta: puedes usar un IAM rol para permitir que alguien (un responsable de confianza) de una cuenta diferente acceda a los recursos de tu cuenta. Los roles son la forma principal de conceder acceso entre cuentas. Sin embargo, con algunos Servicios de AWS, puedes adjuntar una política directamente a un recurso (en lugar de usar un rol como proxy). Para conocer la diferencia entre las funciones y las políticas basadas en recursos para el acceso multicuenta, consulta el tema sobre el acceso a los recursos entre cuentas IAM en la Guía del IAM usuario.
- Acceso entre servicios: algunos Servicios de AWS utilizan funciones en otros. Servicios de AWS Por ejemplo, cuando realizas una llamada en un servicio, es habitual que ese servicio ejecute aplicaciones en Amazon EC2 o almacene objetos en Amazon S3. Es posible que un servicio haga esto usando los permisos de la entidad principal, usando un rol de servicio o usando un rol vinculado al servicio.
  - Sesiones de acceso directo (FAS): cuando utilizas un IAM usuario o un rol para realizar acciones en AWS ellas, se te considera director. Cuando utiliza algunos servicios, es posible que realice una acción que desencadene otra acción en un servicio diferente. FASutiliza los permisos del principal que llama a an Servicio de AWS, junto con los que solicitan, Servicio de AWS para realizar solicitudes a los servicios descendentes. FASlas solicitudes solo se realizan cuando un servicio recibe una solicitud que requiere interacciones con otros recursos Servicios de AWS o para completarse. En este caso, debe tener permisos para realizar ambas acciones. Para obtener detalles sobre la política a la hora de realizar FAS solicitudes, consulte Reenviar sesiones de acceso.

- Función de servicio: una función de servicio es una IAMfunción que un servicio asume para realizar acciones en su nombre. Un IAM administrador puede crear, modificar y eliminar un rol de servicio desde dentroIAM. Para obtener más información, consulte Crear un rol para delegar permisos Servicio de AWS en un rol en el IAMManual del usuario.
- Función vinculada a un servicio: una función vinculada a un servicio es un tipo de función de servicio que está vinculada a un. Servicio de AWS El servicio puede asumir el rol para realizar una acción en su nombre. Los roles vinculados al servicio aparecen en usted Cuenta de AWS y son propiedad del servicio. Un IAM administrador puede ver los permisos de los roles vinculados al servicio, pero no editarlos.

- Aplicaciones que se ejecutan en Amazon EC2: puedes usar un IAM rol para administrar las credenciales temporales de las aplicaciones que se ejecutan en una EC2 instancia y que realizan AWS CLI o AWS API solicitan. Esto es preferible a almacenar las claves de acceso en la EC2 instancia. Para asignar un AWS rol a una EC2 instancia y ponerlo a disposición de todas sus aplicaciones, debe crear un perfil de instancia adjunto a la instancia. Un perfil de instancia contiene el rol y permite que los programas que se ejecutan en la EC2 instancia obtengan credenciales temporales. Para obtener más información, consulte Uso de un IAM rol para conceder permisos a aplicaciones que se ejecutan en EC2 instancias de Amazon en la Guía del IAM usuario.

Administración de acceso mediante políticas

El acceso se controla AWS creando políticas y adjuntándolas a AWS identidades o recursos. Una política es un objeto AWS que, cuando se asocia a una identidad o un recurso, define sus permisos. AWS evalúa estas políticas cuando un director (usuario, usuario raíz o sesión de rol) realiza una solicitud. Los permisos en las políticas determinan si la solicitud se permite o se deniega. La mayoría de las políticas se almacenan AWS como JSON documentos. Para obtener más información sobre la estructura y el contenido de los documentos de JSON políticas, consulte Descripción general de JSON las políticas en la Guía del IAM usuario.

Los administradores pueden usar AWS JSON las políticas para especificar quién tiene acceso a qué. Es decir, qué entidad principal puede realizar acciones en qué recursos y en qué condiciones.

De forma predeterminada, los usuarios y los roles no tienen permisos. Para conceder a los usuarios permiso para realizar acciones en los recursos que necesitan, un IAM administrador puede crear IAM políticas. A continuación, el administrador puede añadir las IAM políticas a las funciones y los usuarios pueden asumir las funciones.

IAMlas políticas definen los permisos para una acción independientemente del método que se utilice para realizar la operación. Por ejemplo, suponga que dispone de una política que permite la acción `iam:GetRole`. Un usuario con esa política puede obtener información sobre el rol de AWS Management Console AWS CLI, el o el AWS API.

Políticas basadas en identidad

Las políticas basadas en la identidad son documentos de política de JSON permisos que se pueden adjuntar a una identidad, como un IAM usuario, un grupo de usuarios o un rol. Estas políticas controlan qué acciones pueden realizar los usuarios y los roles, en qué recursos y en qué condiciones. Para obtener información sobre cómo crear una política basada en la identidad, consulte Definir IAM permisos personalizados con políticas administradas por el cliente en la Guía del usuario. IAM

Las políticas basadas en identidades pueden clasificarse además como políticas insertadas o políticas administradas. Las políticas insertadas se integran directamente en un único usuario, grupo o rol. Las políticas administradas son políticas independientes que puede adjuntar a varios usuarios, grupos y funciones de su empresa. Cuenta de AWS Las políticas administradas incluyen políticas AWS administradas y políticas administradas por el cliente. Para saber cómo elegir entre una política gestionada o una política integrada, consulte Elegir entre políticas gestionadas y políticas integradas en la Guía del IAM usuario.

Políticas basadas en recursos

Las políticas basadas en recursos son documentos de JSON política que se adjuntan a un recurso. Algunos ejemplos de políticas basadas en recursos son las políticas de confianza de IAM roles y las políticas de bucket de Amazon S3. En los servicios que admiten políticas basadas en recursos, los administradores de servicios pueden utilizarlos para controlar el acceso a un recurso específico. Para el recurso al que se asocia la política, la política define qué acciones puede realizar una entidad principal especificada en ese recurso y en qué condiciones. Debe especificar una entidad principal en una política en función de recursos. Los principales pueden incluir cuentas, usuarios, roles, usuarios federados o. Servicios de AWS

Las políticas basadas en recursos son políticas insertadas que se encuentran en ese servicio. No puede usar políticas AWS administradas desde una política IAM basada en recursos.

Listas de control de acceso () ACLs

Las listas de control de acceso (ACLs) controlan qué responsables (miembros de la cuenta, usuarios o roles) tienen permisos para acceder a un recurso. ACLsson similares a las políticas basadas en recursos, aunque no utilizan el formato de documento de JSON políticas.

Amazon S3 AWS WAF y Amazon VPC son ejemplos de servicios compatiblesACLs. Para obtener más informaciónACLs, consulte la descripción general de la lista de control de acceso (ACL) en la Guía para desarrolladores de Amazon Simple Storage Service.

Otros tipos de políticas

AWS admite tipos de políticas adicionales y menos comunes. Estos tipos de políticas pueden establecer el máximo de permisos que los tipos de políticas más frecuentes le conceden.

- Límites de permisos: un límite de permisos es una función avanzada en la que se establecen los permisos máximos que una política basada en la identidad puede conceder a una IAM entidad (IAMusuario o rol). Puede establecer un límite de permisos para una entidad. Los permisos resultantes son la intersección de las políticas basadas en la identidad de la entidad y los límites de permisos. Las políticas basadas en recursos que especifiquen el usuario o rol en el campo `Principal` no estarán restringidas por el límite de permisos. Una denegación explícita en cualquiera de estas políticas anulará el permiso. Para obtener más información sobre los límites de los permisos, consulte los límites de los permisos para IAM las entidades en la Guía del IAMusuario.
- Políticas de control de servicios (SCPs): SCPs son JSON políticas que especifican los permisos máximos para una organización o unidad organizativa (OU) AWS Organizations. AWS Organizations es un servicio para agrupar y administrar de forma centralizada varios de los Cuentas de AWS que son propiedad de su empresa. Si habilitas todas las funciones de una organización, puedes aplicar políticas de control de servicios (SCPs) a una o a todas tus cuentas. SCPLimita los permisos de las entidades en las cuentas de los miembros, incluidas las de cada una Usuario raíz de la cuenta de AWS. Para obtener más información sobre OrganizationsSCPs, consulte las políticas de control de servicios en la Guía del AWS Organizations usuario.
- Políticas de sesión: las políticas de sesión son políticas avanzadas que se pasan como parámetro cuando se crea una sesión temporal mediante programación para un rol o un usuario federado. Los permisos de la sesión resultantes son la intersección de las políticas basadas en identidades del rol y las políticas de la sesión. Los permisos también pueden proceder de una política en función de recursos. Una denegación explícita en cualquiera de estas políticas anulará el permiso. Para obtener más información, consulte las políticas de sesión en la Guía del IAM usuario.

Varios tipos de políticas

Cuando se aplican varios tipos de políticas a una solicitud, los permisos resultantes son más complicados de entender. Para saber cómo se AWS determina si se debe permitir una solicitud cuando se trata de varios tipos de políticas, consulte la lógica de evaluación de políticas en la Guía del IAM usuario.

Cómo funciona Amazon Timestream para InfluxDB con IAM

IAMfunciones que puede utilizar con Amazon Timestream para InfluxDB

IAMcaracterística	Timestream para el soporte de InfluxDB
Políticas basadas en identidades	Sí
Políticas basadas en recursos	No
Acciones de políticas	Sí
Recursos de políticas	Sí
Claves de condición de política	No
ACLs	No
ABAC(etiquetas en las políticas)	Sí
Credenciales temporales	Sí
Permisos de entidades principales	Sí
Roles de servicio	No
Roles vinculados al servicio	Sí

Para obtener una visión general de cómo Timestream for InfluxDB y otros AWS servicios funcionan con la mayoría de las IAM funciones, consulte los AWS servicios con los que funcionan IAM en la Guía del usuario. IAM

Políticas basadas en la identidad para Timestream for InfluxDB

Compatibilidad con las políticas basadas en identidad: sí

Las políticas basadas en la identidad son documentos de política de JSON permisos que puede adjuntar a una identidad, como un IAM usuario, un grupo de usuarios o un rol. Estas políticas controlan qué acciones pueden realizar los usuarios y los roles, en qué recursos y en qué condiciones. Para obtener información sobre cómo crear una política basada en la identidad, consulte Definir IAM permisos personalizados con políticas administradas por el cliente en la Guía del usuario. IAM

Con las políticas IAM basadas en la identidad, puede especificar las acciones y los recursos permitidos o denegados, así como las condiciones en las que se permiten o deniegan las acciones. No es posible especificar la entidad principal en una política basada en identidad porque se aplica al usuario o rol al que está adjunto. Para obtener más información sobre todos los elementos que puede utilizar en una JSON política, consulte la referencia sobre los elementos de la IAM JSON política en la Guía del IAMusuario.

Ejemplos de políticas basadas en la identidad para Timestream for InfluxDB

Para ver ejemplos de políticas basadas en la identidad de Timestream for InfluxDB, consulte.. Ejemplos de políticas basadas en identidad para Amazon Timestream para InfluxDB

Políticas basadas en recursos dentro de Timestream para InfluxDB

Admite políticas basadas en recursos: no

Las políticas basadas en recursos son documentos de políticas que se adjuntan JSON a un recurso. Algunos ejemplos de políticas basadas en recursos son las políticas de confianza de IAM roles y las políticas de bucket de Amazon S3. En los servicios que admiten políticas basadas en recursos, los administradores de servicios pueden utilizarlos para controlar el acceso a un recurso específico. Para el recurso al que se asocia la política, la política define qué acciones puede realizar una entidad principal especificada en ese recurso y en qué condiciones. Debe especificar una entidad principal en una política en función de recursos. Los principales pueden incluir cuentas, usuarios, roles, usuarios federados o. Servicios de AWS

Para habilitar el acceso entre cuentas, puede especificar una cuenta completa o IAM entidades de otra cuenta como principales en una política basada en recursos. Añadir a una política en función de recursos una entidad principal entre cuentas es solo una parte del establecimiento de una relación de confianza. Cuando el principal y el recurso son diferentes Cuentas de AWS, el IAM administrador

de la cuenta de confianza también debe conceder permiso a la entidad principal (usuario o rol) para acceder al recurso. Para conceder el permiso, adjunte la entidad a una política basada en identidad. Sin embargo, si la política en función de recursos concede el acceso a una entidad principal de la misma cuenta, no es necesaria una política basada en identidad adicional. Para obtener más información, consulte Acceso a recursos entre cuentas IAM en la Guía del IAM usuario.

Acciones políticas para Timestream for InfluxDB

Compatibilidad con las acciones de política: sí

Los administradores pueden usar AWS JSON políticas para especificar quién tiene acceso a qué. Es decir, qué entidad principal puede realizar acciones en qué recursos y en qué condiciones.

El `Action` elemento de una JSON política describe las acciones que puede utilizar para permitir o denegar el acceso en una política. Las acciones de política suelen tener el mismo nombre que la AWS API operación asociada. Hay algunas excepciones, como las acciones que solo permiten permisos y que no tienen una operación coincidente. API También hay algunas operaciones que requieren varias acciones en una política. Estas acciones adicionales se denominan acciones dependientes.

Incluya acciones en una política para conceder permisos y así llevar a cabo la operación asociada.

Para ver una lista de las acciones de Timestream para InfluxDB, consulte Acciones definidas por Amazon Timestream para InfluxDB en la Referencia de autorización de servicio.

Las acciones políticas de Timestream para InfluxDB utilizan el siguiente prefijo antes de la acción:

```
timestream-influxdb
```

Para especificar varias acciones en una única instrucción, sepárelas con comas.

```
"Action": [
      "timestream-influxdb:action1",
      "timestream-influxdb:action2"
         ]
```

Puede utilizar caracteres comodín (*) para especificar varias acciones . Por ejemplo, para especificar todas las acciones que comiencen con la palabra `Describe`, incluya la siguiente acción:

```
"Action": "timestream-influxdb:Describe*"
```

Recursos de políticas para Timestream for InfluxDB

Compatibilidad con los recursos de políticas: sí

Los administradores pueden usar AWS JSON políticas para especificar quién tiene acceso a qué. Es decir, qué entidad principal puede realizar acciones en qué recursos y en qué condiciones.

El elemento `Resource` JSON de política especifica el objeto o los objetos a los que se aplica la acción. Las instrucciones deben contener un elemento `Resource` o `NotResource`. Como práctica recomendada, especifique un recurso mediante su nombre de recurso de Amazon (ARN). Puede hacerlo para acciones que admitan un tipo de recurso específico, conocido como permisos de nivel de recurso.

Para las acciones que no admiten permisos de nivel de recurso, como las operaciones de descripción, utilice un carácter comodín (*) para indicar que la instrucción se aplica a todos los recursos.

```
"Resource": "*"
```

Para ver una lista de Timestream para los tipos de recursos de InfluxDB y sus tipos de recursosARNs, consulte Recursos definidos por Amazon Timestream para InfluxDB en la Referencia de autorización de servicios. Para saber con qué acciones puede especificar cada recurso, consulte Acciones definidas por Amazon Timestream para InfluxDB. ARN

Claves de condición de la política de Timestream for InfluxDB

Admite claves de condición de política específicas del servicio: No

Los administradores pueden usar AWS JSON políticas para especificar quién tiene acceso a qué. Es decir, qué entidad principal puede realizar acciones en qué recursos y en qué condiciones.

El elemento `Condition` (o bloque de `Condition`) permite especificar condiciones en las que entra en vigor una instrucción. El elemento `Condition` es opcional. Puede crear expresiones condicionales que utilicen operadores de condición, tales como igual o menor que, para que la condición de la política coincida con los valores de la solicitud.

Si especifica varios elementos de `Condition` en una instrucción o varias claves en un único elemento de `Condition`, AWS las evalúa mediante una operación `AND` lógica. Si especifica varios valores para una única clave de condición, AWS evalúa la condición mediante una `OR` operación lógica. Se deben cumplir todas las condiciones antes de que se concedan los permisos de la instrucción.

También puede utilizar variables de marcador de posición al especificar condiciones. Por ejemplo, puede conceder a un IAM usuario permiso para acceder a un recurso solo si está etiquetado con su nombre de IAM usuario. Para obtener más información, consulte los elementos IAM de la política: variables y etiquetas en la Guía del IAM usuario.

AWS admite claves de condición globales y claves de condición específicas del servicio. Para ver todas las claves de condición AWS globales, consulte las claves de contexto de condición AWS globales en la Guía del IAMusuario.

Listas de control de acceso (ACLs) en Timestream para InfluxDB

ACLsSoportes: No

Las listas de control de acceso (ACLs) controlan qué directores (miembros de la cuenta, usuarios o roles) tienen permisos para acceder a un recurso. ACLsson similares a las políticas basadas en recursos, aunque no utilizan el formato de documento de JSON políticas.

Control de acceso basado en atributos (ABAC) con Timestream para InfluxDB

Soportes ABAC (etiquetas en las políticas): Sí

El control de acceso basado en atributos (ABAC) es una estrategia de autorización que define los permisos en función de los atributos. En AWS, estos atributos se denominan etiquetas. Puede adjuntar etiquetas a IAM entidades (usuarios o roles) y a muchos AWS recursos. Etiquetar entidades y recursos es el primer paso deABAC. Luego, diseñe ABAC políticas para permitir las operaciones cuando la etiqueta del principal coincida con la etiqueta del recurso al que está intentando acceder.

ABACes útil en entornos de rápido crecimiento y ayuda en situaciones en las que la administración de políticas se vuelve engorrosa.

Para controlar el acceso en función de etiquetas, debe proporcionar información de las etiquetas en el elemento de condición de una política utilizando las claves de condición `aws:ResourceTag/`*`key-name`*, `aws:RequestTag/`*`key-name`* o `aws:TagKeys`.

Si un servicio admite las tres claves de condición para cada tipo de recurso, el valor es Sí para el servicio. Si un servicio admite las tres claves de condición solo para algunos tipos de recursos, el valor es Parcial.

Para obtener más informaciónABAC, consulte Definir permisos con ABAC autorización en la Guía del IAM usuario. Para ver un tutorial con los pasos de configuraciónABAC, consulte Usar el control de acceso basado en atributos (ABAC) en la Guía del IAMusuario.

Uso de credenciales temporales con Timestream para InfluxDB

Compatibilidad con credenciales temporales: sí

Algunas Servicios de AWS no funcionan cuando inicias sesión con credenciales temporales. Para obtener información adicional, incluida la información sobre cuáles Servicios de AWS funcionan con credenciales temporales, consulta la sección Servicios de AWS Cómo trabajar con credenciales temporales IAM en la Guía del IAM usuario.

Está utilizando credenciales temporales si inicia sesión en ellas AWS Management Console mediante cualquier método excepto un nombre de usuario y una contraseña. Por ejemplo, cuando accedes AWS mediante el enlace de inicio de sesión único (SSO) de tu empresa, ese proceso crea automáticamente credenciales temporales. También crea credenciales temporales de forma automática cuando inicia sesión en la consola como usuario y luego cambia de rol. Para obtener más información sobre el cambio de rol, consulte Cambiar de un rol de usuario a un IAM rol (consola) en la Guía del IAMusuario.

Puede crear credenciales temporales manualmente con la tecla AWS CLI o AWS API. A continuación, puede utilizar esas credenciales temporales para acceder AWS. AWS recomienda generar credenciales temporales de forma dinámica en lugar de utilizar claves de acceso a largo plazo. Para obtener más información, consulte Credenciales de seguridad temporales en IAM.

Permisos principales entre servicios para Timestream for InfluxDB

Admite sesiones de acceso directo (): Sí FAS

Cuando utilizas un IAM usuario o un rol para realizar acciones en AWSél, se te considera director. Cuando utiliza algunos servicios, es posible que realice una acción que desencadene otra acción en un servicio diferente. FASutiliza los permisos del principal que llama a una Servicio de AWS, junto con los que solicita, Servicio de AWS para realizar solicitudes a los servicios descendentes. FASlas solicitudes solo se realizan cuando un servicio recibe una solicitud que requiere interacciones con otros recursos Servicios de AWS o para completarse. En este caso, debe tener permisos para

realizar ambas acciones. Para obtener detalles sobre la política a la hora de realizar FAS solicitudes, consulte Reenviar sesiones de acceso.

Funciones de servicio para Timestream for InfluxDB

Compatible con roles de servicio: No

Una función de servicio es una IAMfunción que asume un servicio para realizar acciones en su nombre. Un IAM administrador puede crear, modificar y eliminar un rol de servicio desde dentroIAM. Para obtener más información, consulte Crear un rol para delegar permisos Servicio de AWS en un rol en el IAMManual del usuario.

> **Warning**
>
> Cambiar los permisos de un rol de servicio podría interrumpir la funcionalidad de Timestream for InfluxDB. Edite las funciones de servicio solo cuando Timestream for InfluxDB proporcione instrucciones para hacerlo.

Funciones vinculadas al servicio para Timestream for InfluxDB

Admite roles vinculados al servicio: sí

Un rol vinculado a un servicio es un tipo de rol de servicio que está vinculado a un. Servicio de AWS El servicio puede asumir el rol para realizar una acción en su nombre. Los roles vinculados al servicio aparecen en usted Cuenta de AWS y son propiedad del servicio. Un IAM administrador puede ver los permisos de los roles vinculados al servicio, pero no editarlos.

Para obtener más información sobre cómo crear o administrar funciones vinculadas a un servicio, consulte los AWS servicios que funcionan con. IAM Busque un servicio en la tabla que incluya `Yes` en la columna Rol vinculado a un servicio. Seleccione el vínculo Sí para ver la documentación acerca del rol vinculado a servicios para ese servicio.

Ejemplos de políticas basadas en identidad para Amazon Timestream para InfluxDB

De forma predeterminada, los usuarios y los roles no tienen permiso para crear o modificar Timestream para los recursos de InfluxDB. Tampoco pueden realizar tareas con AWS Management Console, AWS Command Line Interface ()AWS CLI o. AWS API Para conceder a los usuarios permiso para realizar acciones en los recursos que necesitan, un IAM administrador puede crear IAM políticas. A continuación, el administrador puede añadir las IAM políticas a las funciones y los usuarios pueden asumir las funciones.

Para obtener información sobre cómo crear una política IAM basada en la identidad mediante estos documentos de JSON política de ejemplo, consulte Crear IAM políticas (consola) en la Guía del IAMusuario.

Para obtener más información sobre las acciones y los tipos de recursos definidos por Timestream for InfluxDB, incluido el formato de cada uno de los tipos de recursos, consulte Acciones, recursos y claves de condición de Amazon Timestream ARNs for InfluxDB en la Referencia de autorización de servicios.

Temas

Prácticas recomendadas sobre las políticas

Las políticas basadas en la identidad determinan si alguien puede crear los recursos de Timestream for InfluxDB de su cuenta, acceder a ellos o eliminarlos. Estas acciones pueden generar costes adicionales para su Cuenta de AWS. Siga estas directrices y recomendaciones al crear o editar políticas basadas en identidades:

- Comience con las políticas AWS administradas y avance hacia los permisos con privilegios mínimos: para empezar a conceder permisos a sus usuarios y cargas de trabajo, utilice las políticas AWS administradas que otorgan permisos para muchos casos de uso comunes. Están disponibles en su. Cuenta de AWS Le recomendamos que reduzca aún más los permisos definiendo políticas administradas por el AWS cliente que sean específicas para sus casos de uso. Para obtener más información, consulte las políticas AWS gestionadas o las políticas AWS gestionadas para las funciones laborales en la Guía del IAM usuario.
- Aplique permisos con privilegios mínimos: cuando establezca permisos con IAM políticas, conceda solo los permisos necesarios para realizar una tarea. Para ello, debe definir las acciones que se pueden llevar a cabo en determinados recursos en condiciones específicas, también conocidos como permisos de privilegios mínimos. Para obtener más información sobre cómo IAM aplicar permisos, consulte Políticas y permisos IAM en la IAM Guía del usuario.

- Utilice las condiciones en IAM las políticas para restringir aún más el acceso: puede añadir una condición a sus políticas para limitar el acceso a las acciones y los recursos. Por ejemplo, puede escribir una condición de política para especificar que todas las solicitudes deben enviarse medianteSSL. También puedes usar condiciones para conceder el acceso a las acciones del servicio si se utilizan a través de una acción específica Servicio de AWS, como AWS CloudFormation. Para obtener más información, consulte los elementos IAM JSON de la política: Condición en la Guía del IAM usuario.
- Utilice IAM Access Analyzer para validar sus IAM políticas y garantizar permisos seguros y funcionales: IAM Access Analyzer valida las políticas nuevas y existentes para que se ajusten al lenguaje de las políticas (JSON) y IAM a las IAM mejores prácticas. IAMAccess Analyzer proporciona más de 100 comprobaciones de políticas y recomendaciones prácticas para ayudarle a crear políticas seguras y funcionales. Para obtener más información, consulte Validar políticas con IAM Access Analyzer en la Guía del IAM usuario.
- Requerir autenticación multifactorial (MFA): si se encuentra en una situación en la que se requieren IAM usuarios o un usuario raíz Cuenta de AWS, actívela MFA para aumentar la seguridad. Para solicitarlo MFA cuando se convoque a API las operaciones, añada MFA condiciones a sus políticas. Para obtener más información, consulte APIAcceso seguro con MFA en la Guía del IAM usuario.

Para obtener más información sobre las prácticas recomendadasIAM, consulte las prácticas recomendadas de seguridad IAM en la Guía del IAM usuario.

Uso de la consola Timestream para InfluxDB

Para acceder a la consola Amazon Timestream for InfluxDB, debe tener un conjunto mínimo de permisos. Estos permisos deben permitirle enumerar y ver detalles sobre los recursos de Timestream for InfluxDB que tiene en su cuenta. Cuenta de AWS Si crea una política basada en identidades que sea más restrictiva que el mínimo de permisos necesarios, la consola no funcionará del modo esperado para las entidades (usuarios o roles) que tengan esa política.

No es necesario que concedas permisos mínimos de consola a los usuarios que solo realicen llamadas al o al. AWS CLI AWS API En su lugar, permita el acceso únicamente a las acciones que coincidan con la API operación que están intentando realizar.

Para garantizar que los usuarios y los roles puedan seguir utilizando la consola de Timestream for InfluxDB, adjunte también la versión Timestream for `ConsoleAccess` InfluxDB o la política gestionada a las entidades. `ReadOnly` AWS Para obtener más información, consulte Añadir permisos a un usuario en la Guía del usuario. IAM

Cómo permitir a los usuarios consultar sus propios permisos

En este ejemplo se muestra cómo se puede crear una política que permita a IAM los usuarios ver las políticas integradas y administradas asociadas a su identidad de usuario. Esta política incluye permisos para completar esta acción en la consola o mediante programación mediante la tecla o. AWS CLI AWS API

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

Acceso a un bucket de Amazon S3

En este ejemplo, quiere conceder a un IAM usuario de su AWS cuenta acceso a uno de sus buckets de Amazon S3,`examplebucket`. También desea permitir al usuario añadir, actualizar o eliminar objetos.

Además de conceder los permisos `s3:PutObject`, `s3:GetObject` y `s3:DeleteObject` al usuario, la política también concede los permisos `s3:ListAllMyBuckets`, `s3:GetBucketLocation` y `s3:ListBucket`. Estos son los permisos adicionales que requiere la consola. Las acciones `s3:PutObjectAcl` y `s3:GetObjectAcl` también son necesarias para poder copiar, cortar y pegar objetos en la consola. Para ver un tutorial de ejemplo en el que se conceden permisos a los usuarios y se prueban con la consola, consulte Tutorial de ejemplo: uso de las políticas del usuario para controlar el acceso al bucket.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"ListBucketsInConsole",
         "Effect":"Allow",
         "Action":[
            "s3:ListAllMyBuckets"
         ],
         "Resource":"arn:aws:s3:::*"
      },
      {
         "Sid":"ViewSpecificBucketInfo",
         "Effect":"Allow",
         "Action":[
            "s3:ListBucket",
            "s3:GetBucketLocation"
         ],
         "Resource":"arn:aws:s3:::examplebucket"
      },
      {
         "Sid":"ManageBucketContents",
         "Effect":"Allow",
         "Action":[
            "s3:PutObject",
            "s3:PutObjectAcl",
            "s3:GetObject",
            "s3:GetObjectAcl",
```

```
            "s3:DeleteObject"
         ],
         "Resource":"arn:aws:s3:::examplebucket/*"
      }
   ]
}
```

Permitiendo todas las operaciones

El siguiente es un ejemplo de política que permite todas las operaciones en Timestream para InfluxDB.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "timestream-influxdb:*"
            ],
            "Resource": "*"
        }
    ]
}
```

Cree, describa, elimine y actualice una instancia de base de datos

El siguiente ejemplo de política permite a un usuario crear, describir, eliminar y actualizar una instancia de base de datos`sampleDB`:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "timestream-influxdb:CreateDbInstance",
                "timestream-influxdb:GetDbInstance",
                "timestream-influxdb:DeleteDbInstance",
                "timestream-influxdb:UpdateDbInstance"
            ],
            "Resource": "arn:aws:timestream-influxdb:us-east-1:<account_ID>:dbinstance/
sampleDB"
```

```
        }
    ]
}
```

Solución de problemas de Amazon Timestream para la identidad y el acceso a InfluxDB

Utilice la siguiente información como ayuda para diagnosticar y solucionar los problemas más comunes que pueden surgir al trabajar con Timestream para InfluxDB y. IAM

Temas

- No estoy autorizado a realizar ninguna acción en Timestream para InfluxDB
- Quiero permitir que personas ajenas a mi AWS cuenta accedan a mis recursos de Timestream for InfluxDB

No estoy autorizado a realizar ninguna acción en Timestream para InfluxDB

Si AWS Management Console le indica que no está autorizado a realizar una acción, debe ponerse en contacto con su administrador para obtener ayuda. Su administrador es la persona que le facilitó su nombre de usuario y contraseña.

En el siguiente ejemplo, el error se produce cuando el usuario `mateojackson` intenta utilizar la consola para consultar los detalles acerca de un recurso ficticio *`my-example-widget`*, pero no tiene los permisos ficticios `timestream-influxdb:`*`GetWidget`*.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 timestream-influxdb:GetWidget on resource: my-example-widget
```

En este caso, Mateo pide a su administrador que actualice sus políticas de forma que pueda obtener acceso al recurso *`my-example-widget`* mediante la acción `timestream-influxdb:`*`GetWidget`*.

Quiero permitir que personas ajenas a mi AWS cuenta accedan a mis recursos de Timestream for InfluxDB

Puede crear un rol que los usuarios de otras cuentas o las personas externas a la organización puedan utilizar para acceder a sus recursos. Puede especificar una persona de confianza para

que asuma el rol. En el caso de los servicios que admiten políticas basadas en recursos o listas de control de acceso (ACLs), puede utilizar esas políticas para permitir que las personas accedan a sus recursos.

Para más información, consulte lo siguiente:

- Controlar el acceso a una instancia de base de datos en un VPC
- Para saber si Timestream for InfluxDB admite estas funciones, consulte Cómo funciona Amazon Timestream for InfluxDB con. IAM
- Para obtener información sobre cómo proporcionar acceso a sus recursos en todas AWS las cuentas de su propiedad, consulte Proporcionar acceso a un IAM usuario de otra AWS cuenta de su propiedad en la Guía del usuario. IAM
- Para obtener información sobre cómo proporcionar acceso a tus recursos a AWS cuentas de terceros, consulta Cómo proporcionar acceso a AWS cuentas propiedad de terceros en la Guía del IAM usuario.
- Para obtener información sobre cómo proporcionar acceso mediante la federación de identidades, consulte Proporcionar acceso a usuarios autenticados externamente (federación de identidades) en la Guía del IAM usuario.
- Para saber la diferencia entre el uso de roles y políticas basadas en recursos para el acceso entre cuentas, consulte En qué se diferencian los IAM roles de las políticas basadas en recursos en la Guía del usuario. IAM

Controlar el acceso a una instancia de base de datos en un VPC

Con Amazon Virtual Private Cloud (AmazonVPC), puede lanzar AWS recursos, como Amazon Timestream for InfluxDB, en una nube privada virtual (). VPC Cuando utilizas AmazonVPC, tienes el control de tu entorno de red virtual. Puede elegir su propio rango de direcciones IP, crear subredes y configurar listas de enrutamiento y control de acceso.

Un grupo VPC de seguridad controla el acceso a las instancias de base de datos dentro de unVPC. Cada regla VPC de grupo de seguridad permite VPC que una fuente específica acceda a una instancia de base de datos asociada a ese grupo VPC de seguridad. El origen puede ser un rango de direcciones (por ejemplo, 203.0.113.0/24) u otro grupo de seguridad. VPC Al especificar un grupo VPC de seguridad como origen, se permite el tráfico entrante desde todas las instancias (normalmente servidores de aplicaciones) que utilizan el grupo de seguridad de origen. VPC Antes de intentar conectarse a su instancia de base de datos, VPC configúrela para su caso de uso. Los siguientes son escenarios comunes para acceder a una instancia de base de datos en unVPC:

Una instancia de base de datos en una EC2 instancia de Amazon a la que VPC accede en la misma VPC

Un uso común de una instancia de base de datos en una VPC es compartir datos con un servidor de aplicaciones que se ejecuta en una EC2 instancia de la mismaVPC. La EC2 instancia puede ejecutar un servidor web con una aplicación que interactúa con la instancia de base de datos.

Una instancia de base de datos es una instancia VPC a la que accede una EC2 instancia de otra VPC

En algunos casos, la instancia de base de datos se encuentra en una EC2 instancia VPC diferente a la que está utilizando para acceder a ella. Si es así, puede utilizar la interconexión para acceder VPC a la instancia de base de datos.

Una instancia de base de datos en una aplicación cliente a la que VPC accede a través de Internet

Para acceder a una instancia de base de datos VPC desde una aplicación cliente a través de Internet, debe configurar una VPC con una única subred pública y utilizar las subredes públicas para crear la instancia de base de datos. También puede configurar una puerta de enlace a Internet en el VPC para permitir la comunicación a través de Internet. Para conectarse a una instancia de base de datos desde fuera de ellaVPC, la instancia de base de datos debe ser de acceso público. Además, el acceso debe concederse mediante las reglas entrantes del grupo de seguridad de la instancia de base de datos y deben cumplirse otros requisitos.

Para obtener más información sobre los grupos VPC de seguridad, consulte los grupos de seguridad en la Guía del usuario de Amazon Virtual Private Cloud.

Para obtener más información sobre cómo conectarse a una instancia de base de datos Timestream for InfluxDB, consulte. Conexión a una instancia de base de datos Amazon Timestream para InfluxDB

Escenario de grupos de seguridad

Un uso común de una instancia de base de datos en una VPC es compartir datos con un servidor de aplicaciones que se ejecuta en una EC2 instancia de Amazon en la mismaVPC, a la que accede una aplicación cliente externa aVPC. En este caso, utilice el Timestream para InfluxDB y VPC las páginas de InfluxDB AWS Management Console o el Timestream para InfluxDB y las EC2 API operaciones para crear las instancias y los grupos de seguridad necesarios:

1. Cree un grupo VPC de seguridad (por ejemplo`sg-0123ec2example`) y defina reglas de entrada que utilicen las direcciones IP de la aplicación cliente como fuente. Este grupo de seguridad

permite que la aplicación cliente se conecte a EC2 instancias de un grupo VPC que utilice este grupo de seguridad.

2. Cree una EC2 instancia para la aplicación y agréguela al grupo de VPC seguridad (`sg-0123ec2example`) que creó en el paso anterior. EC2
3. Cree un segundo grupo de VPC seguridad (por ejemplo,`sg-6789rdsexample`) y cree una nueva regla especificando el grupo de VPC seguridad que creó en el paso 1 (`sg-0123ec2example`) como origen.
4. Cree una nueva instancia de base de datos y agréguela al grupo de VPC seguridad (`sg-6789rdsexample`) que creó en el paso anterior. Al crear la base de datos, utilice el mismo número de puerto que el especificado para la regla de grupo de VPC seguridad (`sg-6789rdsexample`) que creó en el paso 3.

Crear un grupo VPC de seguridad

Puede crear un grupo VPC de seguridad para una instancia de base de datos mediante la VPC consola. Para obtener información sobre la creación de un grupo de seguridad, consulte Grupos de seguridad en la Guía del usuario de Amazon Virtual Private Cloud.

Asociación de un grupo de seguridad con una instancia de base de datos

Puede asociar un grupo de seguridad a una instancia de base de datos mediante Update en la consola Timestream for InfluxDB, `UpdateDBInstance` Timestream for InfluxDB o el comando. API `update-db-instance` AWS CLI

El siguiente CLI ejemplo asocia un grupo de VPC seguridad específico y elimina los grupos de seguridad de base de datos de la instancia de base de datos

```
aws timestream-influxdb update-db-instance --identifier dbName --vpc-security-group-
ids sg-ID
```

Para obtener más información sobre la modificación de una instancia de base de datos, consulte Actualización de instancias de base de datos.

Uso de roles vinculados a servicios para Amazon Timestream para InfluxDB

Amazon Timestream para InfluxDB AWS Identity and Access Management utiliza funciones vinculadas a servicios (). IAM Un rol vinculado a un servicio es un tipo único de IAM rol que está vinculado directamente a un AWS servicio, como Amazon Timestream para InfluxDB. Amazon Timestream for InfluxDB predefine las funciones vinculadas al servicio de Amazon Timestream for

InfluxDB. Incluyen todos los permisos que el servicio requiere para llamar a los servicios en nombre de sus instancias de base de datos. AWS

Un rol vinculado a un servicio facilita la configuración de Amazon Timestream para InfluxDB, ya que no es necesario añadir manualmente los permisos necesarios. Los roles ya existen en su AWS cuenta, pero están vinculados a los casos de uso de Amazon Timestream for InfluxDB y tienen permisos predefinidos. Solo Amazon Timestream para InfluxDB puede asumir estas funciones y solo estas funciones pueden usar la política de permisos predefinida. Las funciones se pueden eliminar únicamente después de eliminar primero sus recursos relacionados. Esto protege sus recursos de Amazon Timestream for InfluxDB porque no puede eliminar inadvertidamente los permisos necesarios para acceder a los recursos.

Para obtener información sobre otros servicios que admiten funciones vinculadas a servicios, consulte Servicios con los que funcionan IAM y busque AWS los servicios que tengan la palabra Sí en la columna Función vinculada a servicios. Elija una opción Sí con un enlace para ver la documentación acerca del rol vinculado a servicios en cuestión.

Contenido

Permisos de roles vinculados a servicios para Amazon Timestream para InfluxDB

Amazon Timestream para InfluxDB usa el rol vinculado al servicio AmazonTimestreamInfluxDBServiceRolePolicydenominado: Esta política permite a Timestream for

InfluxDB administrar los recursos en su nombre según sea necesario para administrar sus clústeres. AWS

La política de permisos de roles AmazonTimestreamInflux DBServiceRolePolicy vinculados al servicio permite a Amazon Timestream for InfluxDB realizar las siguientes acciones en los recursos especificados:

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Sid": "DescribeNetworkStatement",
   "Effect": "Allow",
   "Action": [
    "ec2:DescribeSubnets",
    "ec2:DescribeVpcs",
    "ec2:DescribeNetworkInterfaces"
   ],
   "Resource": "*"
  },
  {
   "Sid": "CreateEniInSubnetStatement",
   "Effect": "Allow",
   "Action": [
    "ec2:CreateNetworkInterface"
   ],
   "Resource": [
    "arn:aws:ec2:*:*:subnet/*",
    "arn:aws:ec2:*:*:security-group/*"
   ]
  },
  {
   "Sid": "CreateEniStatement",
   "Effect": "Allow",
   "Action": [
    "ec2:CreateNetworkInterface"
   ],
   "Resource": "arn:aws:ec2:*:*:network-interface/*",
   "Condition": {
    "Null": {
     "aws:RequestTag/AmazonTimestreamInfluxDBManaged": "false"
    }
   }
```

```
  },
  {
   "Sid": "CreateTagWithEniStatement",
   "Effect": "Allow",
   "Action": [
    "ec2:CreateTags"
   ],
   "Resource": "arn:aws:ec2:*:*:network-interface/*",
   "Condition": {
    "Null": {
     "aws:RequestTag/AmazonTimestreamInfluxDBManaged": "false"
    },
    "StringEquals": {
     "ec2:CreateAction": [
      "CreateNetworkInterface"
     ]
    }
   }
  },
  {
   "Sid": "ManageEniStatement",
   "Effect": "Allow",
   "Action": [
    "ec2:CreateNetworkInterfacePermission",
    "ec2:DeleteNetworkInterface"
   ],
   "Resource": "arn:aws:ec2:*:*:network-interface/*",
   "Condition": {
    "Null": {
     "aws:ResourceTag/AmazonTimestreamInfluxDBManaged": "false"
    }
   }
  },
  {
   "Sid": "PutCloudWatchMetricsStatement",
   "Effect": "Allow",
   "Action": [
    "cloudwatch:PutMetricData"
   ],
   "Condition": {
    "StringEquals": {
     "cloudwatch:namespace": [
      "AWS/Timestream/InfluxDB",
      "AWS/Usage"
```

```
     ]
    }
   },
   "Resource": [
    "*"
   ]
  },
  {
   "Sid": "ManageSecretStatement",
   "Effect": "Allow",
   "Action": [
    "secretsmanager:CreateSecret",
    "secretsmanager:DeleteSecret"
   ],
   "Resource": [
    "arn:aws:secretsmanager:*:*:secret:READONLY-InfluxDB-auth-parameters-*"
   ],
   "Condition": {
    "StringEquals": {
     "aws:ResourceAccount": "${aws:PrincipalAccount}"
    }
   }
  }
 ]
}
```

Para permitir que una entidad cree roles vinculados a un servicio IAM AmazonTimestreamInflux DBServiceRolePolicy

Agregue la siguiente declaración de política a los permisos de esa IAM entidad:

```
{
    "Effect": "Allow",
    "Action": [
        "iam:CreateServiceLinkedRole",
        "iam:PutRolePolicy"
    ],
    "Resource": "arn:aws:iam::*:role/aws-service-role/
timestreamforinfluxdb.amazonaws.com/AmazonTimestreamInfluxDBServiceRolePolicy*",
    "Condition": {"StringLike": {"iam:AWS ServiceName":
 "timestreamforinfluxdb.amazonaws.com"}}
}
```

Para permitir que una IAM entidad elimine funciones vinculadas a un AmazonTimestreamInflux DBServiceRolePolicy servicio

Agregue la siguiente declaración de política a los permisos de esa IAM entidad:

```
{
    "Effect": "Allow",
    "Action": [
        "iam:DeleteServiceLinkedRole",
        "iam:GetServiceLinkedRoleDeletionStatus"
    ],
    "Resource": "arn:aws:iam::*:role/aws-service-role/
timestreamforinfluxdb.amazonaws.com/AmazonTimestreamInfluxDBServiceRolePolicy*",
    "Condition": {"StringLike": {"iam:AWS ServiceName":
 "timestreamforinfluxdb.amazonaws.com"}}
}
```

Como alternativa, puede utilizar una política AWS gestionada para proporcionar acceso completo a Amazon Timestream for InfluxDB.

Creación de un rol vinculado a un servicio () IAM

No necesita crear manualmente un rol vinculado a servicios. Al crear una instancia de base de datos, Amazon Timestream for InfluxDB crea automáticamente el rol vinculado al servicio.

Si elimina este rol vinculado a servicios y necesita crearlo de nuevo, puede utilizar el mismo proceso para volver a crear el rol en su cuenta. Al crear una instancia de base de datos, Amazon Timestream for InfluxDB vuelve a crear el rol vinculado al servicio automáticamente.

Edición de la descripción de un rol vinculado a un servicio para Amazon Timestream para InfluxDB

Amazon Timestream para InfluxDB no le permite editar el rol vinculado al servicio. AmazonTimestreamInflux DBServiceRolePolicy Después de crear un rol vinculado a un servicio, no podrá cambiar el nombre del rol, ya que varias entidades podrían hacer referencia al mismo. Sin embargo, puede editar la descripción del rol utilizando. IAM

Edición de la descripción de un rol vinculado a un servicio (consola) IAM

Puede utilizar la IAM consola para editar la descripción de un rol vinculado a un servicio.

Para editar la descripción de un rol vinculado a un servicio (consola)

1. En el panel de navegación izquierdo de la IAM consola, selecciona Roles.

2. Seleccione el nombre del rol que desea modificar.
3. En el extremo derecho de Role description, seleccione Edit.
4. Ingrese una descripción nueva en el cuadro Save (Guardar).

Edición de la descripción de un rol vinculado a un servicio () IAM CLI

Puede utilizar IAM las operaciones del AWS Command Line Interface para editar la descripción de un rol vinculado a un servicio.

Para cambiar la descripción de un rol vinculado a un servicio () CLI

1. (Opcional) Para ver la descripción actual de un rol, utilice la operación AWS CLI forIAM. `get-role`

   Example

   ```
   $ aws iam get-role --role-name AmazonTimestreamInfluxDBServiceRolePolicy
   ```

   Utilice el nombre del rol, no elARN, para hacer referencia a los roles con las CLI operaciones. Por ejemplo, si un rol tiene lo siguienteARN:`arn:aws:iam::123456789012:role/myrole`, refiérase al rol como**`myrole`**.
2. Para actualizar la descripción de un rol vinculado a un servicio, utilice la operación AWS CLI forIAM. `update-role-description`

   Linux y macOS

   ```
   $ aws iam update-role-description \
       --role-name AmazonTimestreamInfluxDBServiceRolePolicy \
       --description "new description"
   ```

   Windows

   ```
   $ aws iam update-role-description ^
       --role-name AmazonTimestreamInfluxDBServiceRolePolicy ^
       --description "new description"
   ```

Edición de la descripción de un rol vinculado a un servicio () IAM API

Puede utilizar el IAM API para editar la descripción de un rol vinculado a un servicio.

Para cambiar la descripción de un rol vinculado a un servicio () API

1. (Opcional) Para ver la descripción actual de un rol, utilice la operación IAM API GetRole.

   Example

   ```
   https://iam.amazonaws.com/
      ?Action=GetRole
      &RoleName=AmazonTimestreamInfluxDBServiceRolePolicy
      &Version=2010-05-08
      &AUTHPARAMS
   ```

2. Para actualizar la descripción de un rol, utilice la IAM API operación UpdateRoleDescription.

   Example

   ```
   https://iam.amazonaws.com/
      ?Action=UpdateRoleDescription
      &RoleName=AmazonTimestreamInfluxDBServiceRolePolicy
      &Version=2010-05-08
      &Description="New description"
   ```

Eliminar un rol vinculado a un servicio para Amazon Timestream for InfluxDB

Si ya no necesita usar una característica o servicio que requieran un rol vinculado a un servicio, le recomendamos que elimine dicho rol. De esta forma no tiene una entidad no utilizada que no se monitoree ni mantenga de forma activa. Sin embargo, debe limpiar el rol vinculado al servicio antes de eliminarlo.

Amazon Timestream para InfluxDB no elimina el rol vinculado al servicio por usted.

Limpiar un rol vinculado a un servicio

Antes de poder eliminar un rol vinculado IAM a un servicio, confirme primero que el rol no tiene recursos (clústeres) asociados.

Para comprobar si el rol vinculado al servicio tiene una sesión activa en la consola IAM

1. Inicie sesión en AWS Management Console y abra la IAM consola en. https://console.aws.amazon.com/iam/
2. En el panel de navegación izquierdo de la IAM consola, selecciona Roles. A continuación, elija el nombre (no la casilla de verificación) del AmazonTimestreamInflux DBServiceRolePolicy rol.
3. En la página Resumen del rol seleccionado, seleccione la pestaña Asesor de acceso.
4. En la pestaña Asesor de acceso, revise la actividad reciente del rol vinculado a servicios.

Eliminar un rol vinculado a un servicio (consola) IAM

Puede usar la IAM consola para eliminar un rol vinculado a un servicio.

Para eliminar un rol vinculado a un servicio (consola)

1. Inicie sesión en AWS Management Console y abra la IAM consola en. https://console.aws.amazon.com/iam/
2. En el panel de navegación izquierdo de la IAM consola, selecciona Roles. A continuación, seleccione la casilla junto al nombre del rol que desea eliminar, no el nombre ni la fila.
3. En Role actions (Acciones de rol) en la parte superior de la página, elija Delete role (Eliminar rol).
4. En la página de confirmación, revise los datos del servicio al que se accedió por última vez, que muestran cuándo accedió por última vez a un AWS servicio cada uno de los roles seleccionados. Esto lo ayuda a confirmar si el rol está actualmente activo. Si desea continuar, seleccione Yes, Delete para enviar la solicitud de eliminación del rol vinculado al servicio.
5. Observe las notificaciones de la IAM consola para supervisar el progreso de la eliminación del rol vinculado al servicio. Como la eliminación del rol IAM vinculado al servicio es asincrónica, una vez enviado el rol para su eliminación, la tarea de eliminación puede realizarse correctamente o fallar. Si la tarea no se realiza correctamente, puede seleccionar View details (Ver detalles) o View Resources (Ver recursos) desde las notificaciones para obtener información sobre el motivo por el que no se pudo eliminar el rol.

Eliminar un rol vinculado a un servicio () IAM CLI

Puede utilizar IAM las operaciones de AWS Command Line Interface para eliminar un rol vinculado a un servicio.

Para eliminar un rol vinculado a un servicio () CLI

1. Si no conoce el nombre del rol vinculado a servicios que desea eliminar, ingrese el siguiente comando. Este comando muestra las funciones y sus nombres de recursos de Amazon (ARNs) en su cuenta.

   ```
   $ aws iam get-role --role-name role-name
   ```

   Utilice el nombre del rol, no elARN, para hacer referencia a los roles relacionados con las CLI operaciones. Por ejemplo, si un rol tiene el ARN`arn:aws:iam::123456789012:role/myrole`, se hace referencia al rol como**`myrole`**.

2. Como los roles vinculados a servicios no se puede eliminar si están en uso o tienen recursos asociados, debe enviar una solicitud de eliminación. Esta solicitud puede denegarse si no se cumplen estas condiciones. Debe apuntar el valor `deletion-task-id` de la respuesta para comprobar el estado de la tarea de eliminación. Ingrese lo siguiente para enviar una solicitud de eliminación de un rol vinculado a servicios.

   ```
   $ aws iam delete-service-linked-role --role-name role-name
   ```

3. Ingrese lo siguiente para verificar el estado de la tarea de eliminación.

   ```
   $ aws iam get-service-linked-role-deletion-status --deletion-task-id deletion-task-id
   ```

   El estado de la tarea de eliminación puede ser `NOT_STARTED`, `IN_PROGRESS`, `SUCCEEDED` o `FAILED`. Si ocurre un error durante la eliminación, la llamada devuelve el motivo del error para que pueda resolver el problema.

Eliminar un rol vinculado a un servicio () IAM API

Puede utilizar el para eliminar un IAM API rol vinculado a un servicio.

Para eliminar un rol vinculado a un servicio () API

1. Para enviar una solicitud de eliminación de un rol vinculado a un servicio, realice una llamada a DeleteServiceLinkedRole. En la solicitud, especifique un nombre de función.

Como los roles vinculados a servicios no se puede eliminar si están en uso o tienen recursos asociados, debe enviar una solicitud de eliminación. Esta solicitud puede denegarse si no se cumplen estas condiciones. Debe apuntar el valor `DeletionTaskId` de la respuesta para comprobar el estado de la tarea de eliminación.

2. Para comprobar el estado de la tarea de eliminación, realice una llamada a GetServiceLinkedRoleDeletionStatus. En la solicitud, especifique el`DeletionTaskId`.

   El estado de la tarea de eliminación puede ser `NOT_STARTED`, `IN_PROGRESS`, `SUCCEEDED` o `FAILED`. Si ocurre un error durante la eliminación, la llamada devuelve el motivo del error para que pueda resolver el problema.

Regiones compatibles con Amazon Timestream para funciones vinculadas al servicio de InfluxDB

Amazon Timestream para InfluxDB admite el uso de funciones vinculadas a servicios en todas las regiones en las que el servicio esté disponible. Para obtener más información, consulte puntos de conexión de servicio de AWS.

AWS políticas gestionadas para Amazon Timestream para InfluxDB

Para añadir permisos a usuarios, grupos y roles, es más fácil usar políticas AWS administradas que escribirlas tú mismo. Crear políticas gestionadas por los IAM clientes que proporcionen a tu equipo solo los permisos que necesita requiere tiempo y experiencia. Para empezar rápidamente, puedes usar nuestras políticas AWS gestionadas. Estas políticas cubren casos de uso comunes y están disponibles en tu AWS cuenta. Para obtener más información sobre las políticas AWS administradas, consulte las políticas AWS administradas en la Guía del IAM usuario.

AWS los servicios mantienen y AWS actualizan las políticas administradas. No puede cambiar los permisos en las políticas AWS gestionadas. En ocasiones, los servicios agregan permisos adicionales a una política administrada por AWS para admitir características nuevas. Este tipo de actualización afecta a todas las identidades (usuarios, grupos y roles) donde se asocia la política. Es más probable que los servicios actualicen una política administrada por AWS cuando se lanza una nueva característica o cuando se ponen a disposición nuevas operaciones. Los servicios no eliminan los permisos de una política AWS administrada, por lo que las actualizaciones de la política no afectarán a los permisos existentes.

Además, AWS admite políticas administradas para funciones laborales que abarcan varios servicios. Por ejemplo, la política ReadOnlyAccess AWS gestionada proporciona acceso de solo lectura a todos los AWS servicios y recursos. Cuando un servicio lanza una nueva función, AWS agrega permisos de solo lectura para nuevas operaciones y recursos. Para obtener una lista y una descripción de las políticas de funciones laborales, consulte las políticas AWS gestionadas para las funciones laborales en la Guía del IAMusuario.

AWS política gestionada: AmazonTimestreamInflux DBServiceRolePolicy

No puede adjuntar la política AmazonTimestreamInflux DBServiceRolePolicy AWS gestionada a las identidades de su cuenta. Esta política forma parte de la función vinculada al servicio de AWS TimestreamforInflux base de datos. Este rol permite al servicio administrar las interfaces de red y los grupos de seguridad de su cuenta.

Timestream for InfluxDB utiliza los permisos de esta política para administrar EC2 los grupos de seguridad y las interfaces de red. Esto es necesario para administrar Timestream para las instancias de base de datos de InfluxDB.

Para revisar el formato de esta política, consulte. JSON AmazonTimestreamInfluxDBServiceRolePolicy

AWS-políticas gestionadas para Amazon Timestream para InfluxDB

AWS aborda muchos casos de uso comunes al proporcionar IAM políticas independientes que son creadas y administradas por. AWS Las políticas administradas conceden los permisos necesarios para casos de uso comunes, lo que le evita tener que investigar los permisos que se necesitan. Para obtener más información, consulte Políticas AWS administradas en la Guía del IAM usuario.

Las siguientes políticas AWS gestionadas, que puede adjuntar a los usuarios de su cuenta, son específicas de Timestream for InfluxDB:

AmazonTimestreamInfluxDBFullAccess

Puede adjuntar la `AmazonTimestreamInfluxDBFullAccess` política a sus identidades. IAM Esta política otorga permisos administrativos que permiten el acceso total a todos los recursos de Timestream for InfluxDB.

También puede crear sus propias IAM políticas personalizadas para permitir permisos para las acciones de Amazon Timestream for InfluxDB. API Puede adjuntar estas políticas personalizadas a los IAM usuarios o grupos que requieran esos permisos.

Para revisar esta política en JSON formato, consulte AmazonTimestreamInfluxDBFullAccess.

Cronología de las actualizaciones de las políticas gestionadas por InfluxDB AWS

Vea los detalles sobre las actualizaciones de las políticas AWS administradas de Timestream for InfluxDB desde que este servicio comenzó a rastrear estos cambios. Para recibir alertas automáticas sobre los cambios en esta página, suscríbase al RSS feed de la página del historial de documentos de Timestream for InfluxDB.

Cambio	Descripción	Fecha
AmazonTimestreamInfluxDBFullAccess: actualización de una política actual	Se agregó la `ec2:DescribeRouteTables` acción a la política gestionada existente . `AmazonTimestreamInfluxDBFullAccess` Esta acción se utiliza para describir las tablas de rutas	8 de octubre de 2024
AWS política gestionada: AmazonTimestreamInfluxDBServiceRolePolicy: política nueva	Amazon Timestream para InfluxDB agregó una nueva política que permite al servicio administrar las interfaces de red y los grupos de seguridad de su cuenta.	14/03/2024

Cambio	Descripción	Fecha
AmazonTimestreamInfluxDBFullAccess: política nueva	Amazon Timestream para InfluxDB agregó una nueva política para proporcionar acceso administrativo completo para crear, actualizar, eliminar y enumerar instancias de Amazon Timestream InfluxDB y crear y enumerar grupos de parámetros.	14/03/2024

Conexión a Timestream para InfluxDB a través de un punto final VPC

Puede conectarse directamente a Timestream para InfluxDB a través de un punto final de interfaz privada en su nube privada virtual (). VPC Cuando utiliza un VPC punto final de interfaz, la comunicación entre usted VPC y Timestream for InfluxDB se lleva a cabo íntegramente dentro de la red. AWS

Timestream for InfluxDB es compatible con los puntos de conexión de Amazon Virtual Private Cloud (AmazonVPC) con la tecnología de. AWS PrivateLink Cada VPC punto final está representado por una o más interfaces de red elásticas (ENIs) con direcciones IP privadas en las subredes. VPC

El VPC punto final de la interfaz lo conecta VPC directamente a Timestream para InfluxDB sin necesidad de una pasarela de Internet, NAT dispositivo, VPN conexión o conexión. AWS Direct Connect Sus instancias VPC no necesitan direcciones IP públicas para comunicarse con Timestream for InfluxDB.

Regiones

Timestream for InfluxDB admite puntos finales y políticas de VPC puntos finales en todos los casos en los que Timestream for VPC InfluxDB es compatible. Regiones de AWS

Temas

- Consideraciones sobre la transmisión temporal VPC de los puntos finales de InfluxDB
- Crear un VPC punto final para Timestream for InfluxDB

- Conexión a un punto final de Timestream for InfluxDB VPC
- Controlar el acceso a un VPC punto final
- Uso de un VPC punto final en una declaración de política
- Registrar tu VPC punto final

Consideraciones sobre la transmisión temporal VPC de los puntos finales de InfluxDB

Antes de configurar un VPC punto final de interfaz para Timestream for InfluxDB, consulte el tema sobre las propiedades y limitaciones del punto final de la interfaz en la guía.AWS PrivateLink

El soporte de Timestream for InfluxDB para un punto final incluye lo siguiente. VPC

- Puede usar su VPC terminal para llamar a todas las operaciones de Timestream for InfluxDB desde su. API VPC
- Puede usar AWS CloudTrail los registros para auditar su uso de Timestream para los recursos de InfluxDB a través del punto final. VPC Para obtener más información, consulte Registrar tu VPC punto final.

Crear un VPC punto final para Timestream for InfluxDB

Puede crear un VPC punto final para Timestream for InfluxDB mediante la consola de Amazon VPC o Amazon. VPC API Para obtener más información, consulte Creación de un punto de conexión de interfaz en la Guía de AWS PrivateLink .

- Para crear un VPC punto final para Timestream for InfluxDB, utilice el siguiente nombre de servicio:

```
com.amazonaws.region.timestream-influxdb
```

  Por ejemplo, en la Región EE. UU. Oeste (Oregón) (`us-west-2`), el nombre del servicio sería:

```
com.amazonaws.us-west-2.timestream-influxdb
```

Para facilitar el uso del VPC punto final, puede habilitar un DNSnombre privado para su punto final. VPC Si selecciona la opción Habilitar DNS nombre, el flujo temporal estándar para el nombre de DNS host de InfluxDB se transferirá a su punto final. VPC Por ejemplo, se `https://timestream-`

`influxdb.us-west-2.amazonaws.com` resolvería en un VPC punto final conectado al nombre del servicio. `com.amazonaws.us-west-2.timestream-influxdb`

Esta opción facilita el uso del VPC punto final. De forma predeterminada, AWS CLI utilizan el DNS nombre de host Timestream estándar para InfluxDB, por lo que no es necesario especificar el VPC punto final URL en las aplicaciones y comandos. AWS SDKs

Para obtener más información, consulte Acceso a un servicio a través de un punto de conexión de interfaz en la Guía de AWS PrivateLink .

Conexión a un punto final de Timestream for InfluxDB VPC

Puede conectarse a Timestream para InfluxDB a través del VPC punto final utilizando una AWS SDK, la o. AWS CLI AWS Tools for PowerShell Para especificar el VPC punto final, utilice su nombre. DNS

Si habilitó los nombres de host privados al crear el VPC punto final, no necesita especificarlo en los VPC CLI comandos o URL en la configuración de la aplicación. El flujo temporal estándar del nombre de DNS host de InfluxDB se dirige a su punto final. VPC AWS CLI Y SDKs usa este nombre de host de forma predeterminada para que puedas empezar a usar el VPC punto final para conectarte a un punto final regional de Timestream for InfluxDB sin cambiar nada en tus scripts y aplicaciones.

Para utilizar nombres de host privados, sus `enableDnsSupport` atributos `enableDnsHostnames` y atributos deben estar configurados en. VPC `true` Para establecer estos atributos, utilice la ModifyVpcAttributeoperación. Para obtener más información, consulta Ver y actualizar tus DNS atributos VPC en la Guía del VPC usuario de Amazon.

Controlar el acceso a un VPC punto final

Para controlar el acceso a su VPC punto final para Timestream for InfluxDB, adjunte una política de punto final a su VPC punto final. VPC La política de puntos finales determina si los directores pueden usar el punto final para llamar a Timestream para las operaciones VPC de InfluxDB en los recursos de Timestream for InfluxDB.

Puede crear una política de puntos finales al crear su VPC punto final y puede cambiarla en cualquier momento. VPC Utilice la consola VPC de administración CreateVpcEndpointo las ModifyVpcEndpointoperaciones. También puede crear y cambiar una política VPC de puntos finales mediante una AWS CloudFormation plantilla. Para obtener ayuda sobre el uso de la consola de VPC administración, consulte Crear un punto final de interfaz y Modificar un punto final de interfaz en la AWS PrivateLink guía.

> **Note**
>
> Timestream for InfluxDB admite las políticas de VPC puntos finales a partir de julio de 2020. VPCLos puntos finales de Timestream for InfluxDB que se crearon antes de esa fecha tienen la política de puntos VPC finales predeterminada, pero puede cambiarla en cualquier momento.

Temas

- Acerca VPC de las políticas de puntos finales
- Política de VPC puntos finales predeterminada
- Crear una política de puntos finales VPC
- Visualización de una política VPC de puntos finales

Acerca VPC de las políticas de puntos finales

Para que una solicitud de Timestream for InfluxDB que utilice un VPC punto final sea exitosa, el director requiere permisos de dos fuentes:

- Una IAMpolítica debe dar permiso al principal para llamar a la operación en el recurso.
- Una política de VPC punto final debe otorgar al principal permiso para usar el punto final a fin de realizar la solicitud.

Política de VPC puntos finales predeterminada

Cada VPC punto final tiene una política de VPC punto final, pero no es necesario que especifique la política. Si no especifica una política, la política de punto de conexión predeterminada permite todas las operaciones de todas las entidades principales en todos los recursos del punto de conexión.

Sin embargo, en el caso de los recursos de Timestream for InfluxDB, el director también debe tener permiso para llamar a la operación desde una IAMpolítica. Por lo tanto, en la práctica, la política predeterminada establece que si un director tiene permiso para llamar a una operación en un recurso, también puede llamarla mediante el punto final.

```
{
  "Statement": [
```

```
    {
      "Action": "*",
      "Effect": "Allow",
      "Principal": "*",
      "Resource": "*"
    }
  ]
}
```

Para permitir que los principales utilicen el VPC punto final solo para un subconjunto de sus operaciones permitidas, cree o actualice la política de puntos finales. VPC

Crear una política de puntos finales VPC

Una política de VPC punto final determina si un principal tiene permiso para usar el VPC punto final para realizar operaciones en un recurso. En el caso de los recursos de Timestream for InfluxDB, el director también debe tener permiso para realizar las operaciones desde una política, IAM

Cada declaración de política VPC de puntos finales requiere los siguientes elementos:

- La entidad principal que puede realizar acciones
- Las acciones que se pueden realizar
- Los recursos en los que se pueden llevar a cabo las acciones

La declaración de política no especifica el VPC punto final. En cambio, se aplica a cualquier VPC punto final al que esté asociada la política. Para obtener más información, consulta Cómo controlar el acceso a los servicios con VPC puntos de conexión en la Guía del VPC usuario de Amazon.

AWS CloudTrail registra todas las operaciones que utilizan el VPC punto final.

Visualización de una política VPC de puntos finales

Para ver la política de VPC puntos finales de un punto final, utilice la consola VPC de administración o la DescribeVpcEndpointsoperación.

El siguiente AWS CLI comando obtiene la política del punto final con el ID de VPC punto final especificado.

Antes de ejecutar este comando, reemplace el ID de punto de conexión de ejemplo por uno válido de su cuenta.

```
$ aws ec2 describe-vpc-endpoints \
--query 'VpcEndpoints[?VpcEndpointId==`vpc-endpoint-id`].[PolicyDocument]'
--output text
```

Uso de un VPC punto final en una declaración de política

Puede controlar el acceso a Timestream para los recursos y las operaciones de InfluxDB cuando la solicitud proviene de un punto final VPC o lo utiliza. VPC Para ello, utilice una de las siguientes claves de condición globales en una política. IAM

- Utilice la clave de `aws:sourceVpce` condición para conceder o restringir el acceso en función del VPC punto final.
- Utilice la clave de `aws:sourceVpc` condición para conceder o restringir el acceso en función del dispositivo VPC que aloja el punto final privado.

> **Note**
>
> Tenga cuidado al crear políticas clave y IAM políticas basadas en su VPC punto final. Si una declaración de política exige que las solicitudes provengan de un VPC punto final VPC o concreto, las solicitudes de AWS los servicios integrados que utilizan un recurso de Timestream for InfluxDB en su nombre podrían fallar.
> Además, la clave de `aws:sourceIP` condición no entra en vigor cuando la solicitud proviene de un VPCpunto de conexión de Amazon. Para restringir las solicitudes a un VPC punto final, utilice las claves de `aws:sourceVpc` condición `aws:sourceVpce` o. Para obtener más información, consulte la sección Gestión de identidad y acceso para VPC puntos finales y servicios VPC de puntos finales en la AWS PrivateLink Guía.

Puede usar estas claves de condición globales para controlar el acceso a operaciones como las CreateDbInstanceque no dependen de ningún recurso en particular.

Registrar tu VPC punto final

AWS CloudTrail registra todas las operaciones que utilizan el VPC punto final. Cuando una solicitud a Timestream para InfluxDB utiliza un VPC punto final, el ID del punto VPC final aparece en la entrada de AWS CloudTrail registro que registra la solicitud. Puede usar el ID del punto final para auditar el uso de su punto final de Timestream for InfluxDB. VPC

Sin embargo, tus CloudTrail registros no incluyen las operaciones solicitadas por los directores de otras cuentas ni las solicitudes de Timestream para las operaciones de InfluxDB en Timestream para los recursos y alias de InfluxDB de otras cuentas. Además, para protegerlo, no se registran VPC las solicitudes que una política de VPC puntos finales deniega, pero que de otro modo se habrían permitido. AWS CloudTrail

Registro y monitoreo en Timestream para InfluxDB

La supervisión es una parte importante del mantenimiento de la confiabilidad, la disponibilidad y el rendimiento de Timestream para InfluxDB y sus soluciones. AWS Debe recopilar datos de monitoreo de todas las partes de su AWS solución para poder depurar más fácilmente una falla multipunto en caso de que se produzca. Sin embargo, antes de comenzar a monitorear Timestream para InfluxDB, debe crear un plan de monitoreo que incluya respuestas a las siguientes preguntas:

- ¿Cuáles son los objetivos de la supervisión?
- ¿Qué recursos va a supervisar?
- ¿Con qué frecuencia va a supervisar estos recursos?
- ¿Qué herramientas de monitoreo va a utilizar?
- ¿Quién se encargará de realizar las tareas de monitoreo?
- ¿Quién debería recibir una notificación cuando surjan problemas?

El siguiente paso es establecer una línea base para el flujo de tiempo normal del rendimiento de InfluxDB en su entorno, midiendo el rendimiento en varios momentos y bajo diferentes condiciones de carga. Mientras monitorea Timestream para InfluxDB, almacene los datos de monitoreo históricos para poder compararlos con los datos de rendimiento actuales, identificar los patrones de rendimiento normales y las anomalías de rendimiento, y diseñar métodos para abordar los problemas.

Para establecer un punto de referencia debe, como mínimo, monitorizar los elementos siguientes:

- Errores del sistema, para que pueda determinar si alguna solicitud ha provocado un error.

Temas

- Herramientas de monitoreo
- Registrar la transmisión temporal de las llamadas de InfluxDB con API AWS CloudTrail

Herramientas de monitoreo

AWS proporciona varias herramientas que puede usar para monitorear Timestream para InfluxDB. Puede configurar algunas de estas herramientas para que monitoricen por usted, pero otras herramientas requieren intervención manual. Le recomendamos que automatice las tareas de monitorización en la medida de lo posible.

Temas

Herramientas de monitoreo automatizadas

Puede utilizar las siguientes herramientas de monitoreo automatizadas para ver Timestream en InfluxDB e informar cuando algo vaya mal:

- Amazon CloudWatch Alarms: observe una sola métrica durante un período de tiempo que especifique y realice una o más acciones en función del valor de la métrica en relación con un umbral determinado durante varios períodos de tiempo. La acción es una notificación enviada a un tema de Amazon Simple Notification Service (AmazonSNS) o a una política de Amazon EC2 Auto Scaling. CloudWatch las alarmas no invocan acciones simplemente porque se encuentran en un estado determinado; el estado debe haber cambiado y se ha mantenido durante un número específico de períodos. Para obtener más información, consulte Monitorización con Amazon CloudWatch.

Herramientas de monitoreo manuales

Otra parte importante de la supervisión de Timestream para InfluxDB consiste en monitorear manualmente los elementos que las CloudWatch alarmas no cubren. El Timestream de InfluxDB y otros AWS Management Console paneles proporcionan una Trusted Advisor visión del estado de su entorno. CloudWatch at-a-glance AWS

- La página de CloudWatch inicio muestra lo siguiente:
  - Alarmas y estado actual
  - Gráficos de alarmas y recursos
  - Estado de los servicios

Además, se puede utilizar CloudWatch para hacer lo siguiente:

- Crear paneles personalizados para monitorizar los servicios que le interesan
- Realizar un gráfico con los datos de las métricas para resolver problemas y descubrir tendencias
- Busque y explore todas sus métricas AWS de recursos
- Crear y editar las alarmas de notificación de problemas

Registrar la transmisión temporal de las llamadas de InfluxDB con API AWS CloudTrail

Timestream for InfluxDB está integrado con un servicio que proporciona un registro de las acciones realizadas por un usuario AWS CloudTrail, un rol o un servicio en Timestream for InfluxDB. AWS CloudTrail captura el lenguaje de definición de datos (DDL) API y utiliza Timestream para InfluxDB como eventos. Las llamadas que se capturan incluyen las llamadas desde la consola Timestream para InfluxDB y las llamadas en código a Timestream para las operaciones de InfluxDB. API Si crea un registro, puede habilitar la entrega continua de CloudTrail eventos a un bucket de Amazon Simple Storage Service (Amazon S3), incluidos los eventos de Timestream for InfluxDB. Si no configura una ruta, podrá ver los eventos más recientes en la CloudTrail consola, en el historial de eventos. Con la información recopilada por InfluxDB CloudTrail, puede determinar la solicitud que se realizó a Timestream, la dirección IP desde la que se realizó la solicitud, quién la hizo, cuándo se realizó y detalles adicionales.

Para obtener más información CloudTrail, consulte la Guía del usuario.AWS CloudTrail

Secuencia temporal de la información de InfluxDB en CloudTrail

CloudTrail está activado en su AWS cuenta al crear la cuenta. Cuando se produce una actividad en Timestream para InfluxDB, esa actividad se registra en un CloudTrail evento junto con otros eventos de AWS servicio en el historial de eventos. Puede ver, buscar y descargar los últimos eventos de la cuenta de AWS . Para obtener más información, consulte Visualización de eventos con el historial de eventos. CloudTrail

Para obtener un registro continuo de los eventos de su AWS cuenta, incluidos los eventos de Timestream para InfluxDB, cree un registro. Un rastro permite CloudTrail entregar archivos de registro a un bucket de Amazon S3. De forma predeterminada, cuando crea una ruta en la consola, la ruta se aplica a todas AWS las regiones. La ruta registra los eventos de todas las regiones de la AWS partición y envía los archivos de registro al bucket de Amazon S3 que especifique. Además,

puede configurar otros AWS servicios para analizar más a fondo los datos de eventos recopilados en los CloudTrail registros y actuar en función de ellos.

Para obtener más información, consulte los siguientes temas en la Guía del usuario de AWS CloudTrail :

- Introducción a la creación de registros de seguimiento
- CloudTrail Integraciones y servicios compatibles
- Configuración de Amazon SNS Notifications para CloudTrail
- Recibir archivos de CloudTrail registro de varias regiones
- Recibir archivos de CloudTrail registro de varias cuentas
- Registrar eventos de datos

Cada entrada de registro o evento contiene información sobre quién generó la solicitud. La información de identidad del usuario lo ayuda a determinar lo siguiente:

- Si la solicitud se realizó con credenciales de usuario root o AWS Identity and Access Management (IAM)
- si la solicitud se realizó con credenciales de seguridad temporales de un rol o fue un usuario federado
- Si la solicitud la realizó otro AWS servicio

Para obtener más información, consulte el CloudTrail userIdentityElemento.

Validación de conformidad de Amazon Timestream para InfluxDB

Los auditores externos evalúan la seguridad y el cumplimiento de Amazon Timestream for InfluxDB como parte de varios programas de cumplimiento. AWS Estos incluyen los siguientes:

- GDPR
- HIPAA
- PCI
- SOC

Para saber si un programa de cumplimiento Servicio de AWS está dentro del ámbito de aplicación de programas de cumplimiento específicos, consulte Servicios de AWS Alcance by Compliance

Servicios de AWS de cumplimiento que le interese. Para obtener información general, consulte Programas de AWS cumplimiento > Programas AWS .

Puede descargar informes de auditoría de terceros utilizando AWS Artifact. Para obtener más información, consulte Descarga de informes en AWS Artifact .

Su responsabilidad de cumplimiento al Servicios de AWS utilizarlos viene determinada por la confidencialidad de sus datos, los objetivos de cumplimiento de su empresa y las leyes y reglamentos aplicables. AWS proporciona los siguientes recursos para ayudar con el cumplimiento:

- Guías de inicio rápido sobre seguridad y cumplimiento: estas guías de implementación analizan las consideraciones arquitectónicas y proporcionan los pasos para implementar entornos básicos centrados en AWS la seguridad y el cumplimiento.
- Diseñando una arquitectura basada en la HIPAA seguridad y el cumplimiento en Amazon Web Services: en este documento técnico se describe cómo pueden utilizar las empresas AWS para crear HIPAA aplicaciones aptas.

  > **Note**
  >
  > No todos son aptos. Servicios de AWS HIPAA Para obtener más información, consulta la Referencia de servicios HIPAA aptos.

- AWS Recursos de de cumplimiento: esta colección de libros de trabajo y guías puede aplicarse a su industria y ubicación.
- AWS Guías de cumplimiento para clientes: comprenda el modelo de responsabilidad compartida desde el punto de vista del cumplimiento. En las guías se resumen las mejores prácticas para garantizar la seguridad Servicios de AWS y se orientan a los controles de seguridad en varios marcos (incluidos el Instituto Nacional de Estándares y Tecnología (NIST), el Consejo de Normas de Seguridad del Sector de Tarjetas de Pago (PCI) y la Organización Internacional de Normalización (ISO)).
- Evaluación de los recursos con reglas en la guía para AWS Config desarrolladores: el AWS Config servicio evalúa en qué medida las configuraciones de los recursos cumplen con las prácticas internas, las directrices del sector y las normas.
- AWS Security Hub— Esto Servicio de AWS proporciona una visión completa del estado de su seguridad interior AWS. Security Hub utiliza controles de seguridad para evaluar sus recursos de AWS y comprobar su cumplimiento con los estándares y las prácticas recomendadas del

sector de la seguridad. Para obtener una lista de los servicios y controles compatibles, consulte la Referencia de controles de Security Hub.

- Amazon GuardDuty: Servicio de AWS detecta posibles amenazas para sus cargas de trabajo Cuentas de AWS, contenedores y datos mediante la supervisión de su entorno para detectar actividades sospechosas y maliciosas. GuardDuty puede ayudarlo a cumplir con varios requisitos de conformidad, por ejemplo PCIDSS, cumpliendo con los requisitos de detección de intrusiones exigidos por ciertos marcos de cumplimiento.
- AWS Audit Manager— Esto le Servicio de AWS ayuda a auditar continuamente su AWS consumo para simplificar la gestión del riesgo y el cumplimiento de las normativas y los estándares del sector.

Resiliencia en Amazon Timestream para InfluxDB

La infraestructura AWS global se basa en regiones y zonas de disponibilidad. AWS AWS Las regiones proporcionan varias zonas de disponibilidad aisladas y separadas físicamente, que están conectadas mediante redes de baja latencia, alto rendimiento y alta redundancia. Con las zonas de disponibilidad, puede diseñar y utilizar aplicaciones y bases de datos que realizan una conmutación por error automática entre las zonas sin interrupciones. Las zonas de disponibilidad tienen una mayor disponibilidad, tolerancia a errores y escalabilidad que las infraestructuras tradicionales de uno o varios centros de datos.

Para obtener más información sobre AWS las regiones y las zonas de disponibilidad, consulte Infraestructura global.AWS

Amazon Timestream para InfluxDB realiza copias de seguridad internas de forma periódica y las conserva durante 24 horas para garantizar la disponibilidad y la durabilidad. Las instantáneas se toman durante las eliminaciones y se conservan durante 30 días para permitir las restauraciones. Para acceder a ellas o utilizarlas, presenta un ticket en AWS el servicio de asistencia.

Puede crear su instancia con capacidades de recuperación en zonas de disponibilidad múltiples (Multi-AZ). Para obtener más información, consulte Implementaciones de instancias de base de datos Multi-AZ.

Seguridad de infraestructura en Amazon Timestream para InfluxDB

Como servicio gestionado, Amazon Timestream para InfluxDB está protegido por los procedimientos de seguridad de red global que se describen en AWS el documento técnico Amazon Web Services: Overview of Security Processes.

Utilice las API llamadas del plano de control AWS publicadas para acceder a Timestream for InfluxDB a través de la red. Para obtener más información, consulte Planos de control y planos de datos. Los clientes deben ser compatibles con Transport Layer Security (TLS) 1.2 o una versión posterior. Recomendamos la versión TLS 1.2 o la 1.3. Los clientes también deben admitir conjuntos de cifrado con total confidencialidad (PFS), como Ephemeral Diffie-Hellman () o Elliptic Curve Ephemeral Diffie-Hellman (DHE). ECDHE La mayoría de los sistemas modernos como Java 7 y posteriores son compatibles con estos modos.

Además, las solicitudes deben firmarse con un identificador de clave de acceso y una clave de acceso secreta que esté asociada a un director. IAM También puede utilizar AWS Security Token Service (AWS STS) para generar credenciales de seguridad temporales para firmar solicitudes.

Timestream for InfluxDB está diseñado de manera que el tráfico esté aislado en la AWS región específica en la que reside la instancia de Timestream for InfluxDB.

Grupos de seguridad

Los grupos de seguridad controlan el acceso del tráfico entrante y saliente a una instancia de base de datos. De forma predeterminada, el acceso de red está deshabilitado para una instancia de base de datos. Puede especificar reglas en un grupo de seguridad que permitan el acceso desde un rango de direcciones IP, un puerto o un grupo de seguridad. Una vez configuradas las reglas de entrada, se aplican las mismas reglas a todas las instancias de base de datos que están asociadas a ese grupo de seguridad.

Para obtener más información, consulte Controlar el acceso a una instancia de base de datos en un VPC.

Análisis de configuración y vulnerabilidad en Timestream para InfluxDB

La configuración y los controles de TI son una responsabilidad compartida entre AWS usted y usted, nuestro cliente. Para obtener más información, consulte el modelo de responsabilidad AWS compartida. Además del modelo de responsabilidad compartida, los usuarios de Timestream for InfluxDB deben tener en cuenta lo siguiente:

- Es responsabilidad del cliente colocar parches a sus aplicaciones cliente con las dependencias relevantes del lado del cliente.
- Los clientes deberían considerar la posibilidad de realizar pruebas de penetración, si procede (consulte las pruebas de penetración/.) https://aws.amazon.com/security/

Respuesta a incidentes en Timestream para InfluxDB

Los incidentes del servicio Amazon Timestream for InfluxDB se notifican en el Personal Health Dashboard. Puede obtener más información sobre el panel de control aquí. AWS Health

Timestream for InfluxDB admite el uso de informes. AWS CloudTrail Para obtener más información, consulte Registrar la transmisión temporal de las llamadas de InfluxDB con API AWS CloudTrail.

Amazon Timestream para API InfluxDB y puntos finales de interfaz () VPC AWS PrivateLink

Puede establecer una conexión privada entre sus puntos finales del API plano de control VPC y Amazon Timestream for InfluxDB mediante la creación de un punto final de interfaz. VPC Los puntos finales de la interfaz funcionan con. AWS PrivateLink AWS PrivateLink le permite acceder de forma privada a Amazon Timestream para las operaciones de API InfluxDB sin una puerta de enlace a InternetNAT, dispositivo, conexión o VPN conexión Direct Connect. AWS

Las instancias VPC que tenga no necesitan direcciones IP públicas para comunicarse con los puntos de enlace de Amazon Timestream for InfluxDB. API Sus instancias tampoco necesitan direcciones IP públicas para utilizar ninguno de los Timestream disponibles para las operaciones de InfluxDB. API El tráfico entre usted VPC y Amazon Timestream para InfluxDB no sale de la red de Amazon. Cada punto de conexión de la interfaz está representado por una o más interfaces de redes elásticas en las subredes. Para obtener más información sobre las interfaces de red elásticas, consulte las interfaces de red elásticas en la Guía del EC2 usuario de Amazon.

- Para obtener más información sobre los VPC puntos de enlace, consulte los VPCpuntos de enlace de la interfaz (AWS PrivateLink) en la Guía VPCdel usuario de Amazon.
- Para obtener más información sobre las operaciones de Timestream for InfluxDB, consulte Timestream for InfluxDB API operations. API

Después de crear un punto final de interfaz, si habilita los DNS nombres de host privados para el VPC punto final, será el Timestream predeterminado para el punto final de InfluxDB (https://timestream-influxb).*Region*.amazonaws.com) se dirige a tu punto final. VPC Si no habilitas DNS los nombres de servidor privados, Amazon VPC proporciona un nombre de DNS punto final que puedes usar en el siguiente formato:

```
VPC_Endpoint_ID.timestream-influxb.Region.vpce.amazonaws.com
```

Para obtener más información, consulte Interface VPC Endpoints (AWS PrivateLink) en la Guía del VPC usuario de Amazon. Timestream for InfluxDB permite realizar llamadas a todas sus acciones dentro de la tuya. API VPC

> **Note**
>
> DNSLos nombres de host privados solo se pueden habilitar para un punto final del. VPC VPC Si desea crear un VPC punto final adicional, el DNS nombre de host privado debe estar deshabilitado para él.

Consideraciones sobre los puntos finales VPC

Antes de configurar un VPC punto final de interfaz para los puntos de enlace de Amazon Timestream for API InfluxDB, asegúrese de revisar las propiedades y limitaciones de los puntos de enlace de interfaz en la Guía del usuario de Amazon. VPC Puede utilizar todas las API operaciones de Timestream for InfluxDB relevantes para la administración de los recursos de Amazon Timestream for InfluxDB. VPC AWS PrivateLink VPCLas políticas de puntos de enlace de Timestream para InfluxDB son compatibles. API De forma predeterminada, se permite el acceso total a Timestream para las operaciones de API InfluxDB a través del punto final. Para obtener más información, consulta Cómo controlar el acceso a los servicios con VPC puntos de conexión en la Guía del VPC usuario de Amazon.

Creación de un VPC punto final de interfaz para el Timestream de InfluxDB API

Puede crear un VPC punto final para Amazon Timestream for API InfluxDB mediante la consola de Amazon o laVPC. AWS CLI Para obtener más información, consulte Creación de un punto final de interfaz en la Guía del VPC usuario de Amazon.

Después de crear un VPC punto final de interfaz, puede habilitar los nombres de DNS host privados para el punto final. Cuando lo hagas, el punto final predeterminado de Amazon Timestream para InfluxDB (https://timestream-influxb).*Region*.amazonaws.com) se resuelve en su punto final. VPC Para obtener más información, consulta Acceder a un servicio a través de un punto final de interfaz en la Guía del VPC usuario de Amazon.

Creación de una política VPC de puntos finales para Amazon Timestream for InfluxDB API

Puede adjuntar una política de puntos finales a su VPC punto final que controle el acceso a Timestream para InfluxDB. API La política especifica lo siguiente:

- La entidad principal que puede realizar acciones.
- Las acciones que se pueden realizar.
- Los recursos en los que se pueden llevar a cabo las acciones.

Para obtener más información, consulta Cómo controlar el acceso a los servicios con VPC puntos de conexión en la Guía del VPC usuario de Amazon.

Example VPCpolítica de puntos finales para las acciones de Timestream for InfluxDB API

El siguiente es un ejemplo de una política de punto final para Timestream for InfluxDB. API Cuando se adjunta a un punto final, esta política otorga acceso a la secuencia temporal indicada para las acciones de InfluxDB API a todos los directores de todos los recursos.

```
{
 "Statement": [{
  "Principal": "*",
  "Effect": "Allow",
  "Action": [
   "timestream-influxb:CreateDbInstance",
   "timestream-influxb:UpdateDbInstance"
  ],
  "Resource": "*"
 }]
}
```

Example VPCpolítica de punto final que deniega todo acceso desde una cuenta específica AWS

La siguiente política de VPC puntos finales deniega la AWS cuenta *123456789012* todos los accesos a los recursos mediante el punto final. La política permite todas las acciones de otras cuentas.

```
{
 "Statement": [{
   "Action": "*",
   "Effect": "Allow",
   "Resource": "*",
   "Principal": "*"
  },
  {
   "Action": "*",
   "Effect": "Deny",
```

```
    "Resource": "*",
    "Principal": {
     "AWS": [
      "123456789012"
     ]
    }
   }
  ]
}
```

Mejores prácticas de seguridad para Timestream for InfluxDB

Amazon Timestream para InfluxDB proporciona una serie de características de seguridad que debe tener en cuenta a la hora de desarrollar e implementar sus propias políticas de seguridad. Las siguientes prácticas recomendadas son directrices generales y no constituyen una solución de seguridad completa. Puesto que es posible que estas prácticas recomendadas no sean adecuadas o suficientes para el entorno, considérelas como consideraciones útiles en lugar de como normas.

Implementación del acceso a los privilegios mínimos

Al conceder los permisos, usted decide quién obtiene qué permisos y qué recursos de Timestream for InfluxDB. Habilite las acciones específicas que desea permitir en dichos recursos. Por lo tanto, debe conceder únicamente los permisos obligatorios para realizar una tarea. La implementación del acceso con privilegios mínimos es esencial a la hora de reducir los riesgos de seguridad y el impacto que podrían causar los errores o los intentos malintencionados.

Usa roles IAM

Las aplicaciones de productor y cliente deben tener credenciales válidas para acceder a las instancias de base de datos de Timestream for InfluxDB. No debe almacenar AWS las credenciales directamente en una aplicación cliente o en un bucket de Amazon S3. Estas son las credenciales a largo plazo que no rotan automáticamente y que podrían tener un impacto empresarial significativo si se comprometen.

En su lugar, debe utilizar un IAM rol para administrar las credenciales temporales de sus aplicaciones de productor y cliente a fin de acceder a las instancias de base de datos de Timestream for InfluxDB. Al utilizar un rol, no tiene que utilizar credenciales a largo plazo (como un nombre de usuario y una contraseña o claves de acceso) para acceder a otros recursos.

Para obtener más información, consulte los siguientes temas de la Guía del usuario: IAM

- IAMFunciones
- Situaciones habituales con los roles: usuarios, aplicaciones y servicios

Utilice las cuentas AWS Identity and Access Management (IAM) para controlar el acceso a Amazon Timestream para las operaciones de API InfluxDB, especialmente las operaciones que crean, modifican o eliminan los recursos de Amazon Timestream for InfluxDB. Estos recursos incluyen instancias de bases de datos, grupos de seguridad y grupos de parámetros.

- Cree un usuario individual para cada persona que administre los recursos de Amazon Timestream for InfluxDB, incluido usted. No utilice credenciales AWS raíz para administrar los recursos de Amazon Timestream for InfluxDB.
- Asigne a cada usuario el conjunto mínimo de permisos requerido para realizar sus tareas.
- Utilice IAM grupos para administrar eficazmente los permisos de varios usuarios.
- Rote con regularidad sus credenciales de IAM.
- Configure AWS Secrets Manager para rotar automáticamente los secretos de Amazon Timestream for InfluxDB. Para obtener más información, consulte Rotación de AWS los secretos de Secrets Manager en la Guía del usuario de AWS Secrets Manager. También puede recuperar la credencial de AWS Secrets Manager mediante programación. Para obtener más información, consulte Recuperación del valor secreto en la Guía del usuario de AWS Secrets Manager.
- Proteja su flujo temporal para los tokens de afluencia API de InfluxDB utilizando el. APIfichas

Implementación del cifrado en el servidor en recursos dependientes

Los datos en reposo y los datos en tránsito se pueden cifrar en Timestream para InfluxDB. Para obtener más información, consulte Cifrado en tránsito.

Se usa para monitorear las llamadas CloudTrail API

Timestream for InfluxDB está integrado con AWS CloudTrail un servicio que proporciona un registro de las acciones realizadas por un usuario, un rol o un AWS servicio en Timestream for InfluxDB.

Con la información recopilada por InfluxDB CloudTrail, puede determinar la solicitud de InfluxDB que se realizó a Timestream, la dirección IP desde la que se realizó la solicitud, quién la hizo, cuándo se realizó y detalles adicionales.

Para obtener más información, consulte the section called “Registrar la transmisión temporal de las llamadas con LiveAnalytics API AWS CloudTrail”.

Amazon Timestream para InfluxDB admite eventos del plano de control, pero no del CloudTrail plano de datos. Para obtener más información, consulte Planos de control y planos de datos.

Public accessibility (Accesibilidad pública)

Al lanzar una instancia de base de datos dentro de una nube privada virtual (VPC) basada en el VPC servicio de Amazon, puede activar o desactivar la accesibilidad pública de esa instancia de base de datos. Para indicar si la instancia de base de datos que cree tiene un DNS nombre que se resuelve en una dirección IP pública, utilice el parámetro de accesibilidad pública. Con este parámetro, puede designar si hay acceso público a la instancia de base de datos

Si su instancia de base de datos está en una, VPC pero no es de acceso público, también puede usar una AWS Site-to-Site VPN conexión o una conexión AWS Direct Connect para acceder a ella desde una red privada.

Si su instancia de base de datos es de acceso público, asegúrese de tomar medidas para prevenir o ayudar a mitigar las amenazas relacionadas con la denegación de servicio. Para obtener más información, consulte Introducción a los ataques de denegación de servicio y Protección de redes.

APIreferencia

Para obtener una lista completa y detalles de Amazon Timestream para InfluxDBAPIs, consulte Amazon Timestream para InfluxDB. APIs

Para ver los códigos de error comunes a todos los AWS servicios, consulte la sección AWS Support.

Historial del documento

Cambio	Descripción	Fecha
Actualización exclusiva de la documentación	Se actualizó el tema Cuotas para separar las cuotas predeterminadas y los límites del sistema.	22 de octubre de 2024
Amazon Timestream ahora admite información sobre consultas	Timestream ahora incluye compatibilidad con la función de información sobre	22 de octubre de 2024

	consultas, que le ayuda a optimizar las consultas , mejorar su rendimiento y reducir los costos.	
Actualización de Amazon Timestream for InfluxDB a una política existente.	Amazon Timestream para InfluxDB ha añadido la acción a `ec2:DescribeRouteTables` la política gestionada a `AmazonTimestreamInfluxDBFullAccess` existente para describir las tablas de rutas.	8 de octubre de 2024
`AmazonTimestreamReadOnlyAccess` — Actualización de una política existente	Timestream for LiveAnalytics ha añadido el `DescribeAccountSettings` permiso a la política `AmazonTimestreamReadOnlyAccess` gestionada para describir Cuenta de AWS la configuración.	3 de junio de 2024
Amazon Timestream admite LiveAnalytics por ahora unidades de cómputo Timestream () TCUs	LiveAnalytics Por ahora, Amazon Timestream incluye soporte para Timestream Compute Units TCUs () para medir la capacidad informática asignada a las necesidades de sus consultas.	29 de abril de 2024

Se han agregado nuevas políticas	Amazon Timestream para InfluxDB agregó dos políticas nuevas: una que permite al servicio administrar las interfaces de red y los grupos de seguridad de su cuenta. Para obtener más información, consulte. AmazonTimestreamInfluxDBServiceRolePolicy Otra que proporciona acceso administrativo completo para crear, actualizar, eliminar y enumerar instancias de Amazon Timestream InfluxDB y crear y enumerar grupos de parámetros. Para obtener más información, consulte. AmazonTimestreamInfluxDBFullAccess	14 de marzo de 2024
Amazon Timestream para InfluxDB ya está disponible de forma general.	Esta documentación cubre la versión inicial de Amazon Timestream para InfluxDB.	14 de marzo de 2024

Los eventos de Amazon Timestream LiveAnalytics for Query están disponibles en AWS CloudTrail	Amazon Timestream publica LiveAnalytics por ahora los eventos de datos de API Query en. AWS CloudTrail Los clientes pueden auditar todas API las solicitudes de consulta realizadas en sus AWS cuentas y ver informaci ón como qué IAM usuario o rol realizó la solicitud, cuándo se realizó la solicitud, qué bases de datos y tablas se consultar on y el ID de consulta de la solicitud.	12 de septiembre de 2023
Amazon Timestream para LiveAnalytics UNLOAD	LiveAnalytics Por ahora, Amazon Timestream UNLOAD admite la exportación de los resultados de las consultas a S3.	12 de mayo de 2023
Amazon Timestream LiveAnalytics para actualizar una política existente.	Se han añadido permisos de carga por lotes a una política gestionada.	24 de febrero de 2023
Amazon Timestream LiveAnalytics para carga de lotes.	Amazon Timestream admite LiveAnalytics por ahora la funcionalidad de carga por lotes.	24 de febrero de 2023
Amazon Timestream LiveAnalytics por ahora es compatible. AWS Backup	Amazon Timestream LiveAnalytics por ahora es compatible. AWS Backup	14 de diciembre de 2022

Amazon Timestream LiveAnalytics para actualizaciones de las políticas gestionadas AWS	Nueva información sobre las políticas AWS gestionadas y Amazon Timestream LiveAnalytics for, incluidas las actualizaciones de las políticas gestionadas existentes.	29 de noviembre de 2021
Amazon Timestream LiveAnalytics for admite consultas programadas	LiveAnalytics Por ahora, Amazon Timestream permite ejecutar una consulta en su nombre, según un cronograma.	29 de noviembre de 2021
Amazon Timestream LiveAnalytics para soporte de almacenamiento magnético.	LiveAnalytics Por ahora, Amazon Timestream admite el uso de almacenamiento magnético para las escrituras en tablas.	29 de noviembre de 2021
Amazon Timestream LiveAnalytics para registros de múltiples medidas.	LiveAnalytics Por ahora, Amazon Timestream admite un formato más compacto para almacenar los datos de series temporales.	29 de noviembre de 2021
Amazon Timestream LiveAnalytics para actualizaciones de las políticas gestionadas AWS	Nueva información sobre las políticas AWS gestionadas y Amazon Timestream LiveAnalytics for, incluidas las actualizaciones de las políticas gestionadas existentes.	24 de mayo de 2021
Amazon Timestream LiveAnalytics for ya está disponible en la región de Europa (Fráncfort).	Amazon Timestream LiveAnalytics for ya está disponible de forma general en la región de Europa (Fráncfort) (). `eu-central-1`	23 de abril de 2021

Amazon Timestream LiveAnalytics for ya es VPC compatible con endpoints ().AWS PrivateLink	Amazon Timestream admite LiveAnalytics por ahora el uso de puntos VPC de enlace ().AWS PrivateLink	23 de marzo de 2021
Amazon Timestream ahora admite consultas entre tablas.	Puede usar Amazon Timestream LiveAnalytics para ejecutar consultas entre tablas.	10 de febrero de 2021
Amazon Timestream, LiveAnalytics por ahora, admite estadísticas de ejecución de consultas mejoradas.	LiveAnalytics Por ahora, Amazon Timestream admite estadísticas de ejecución de consultas mejoradas, como la cantidad de datos escaneado s.	10 de febrero de 2021
Amazon Timestream admite LiveAnalytics por ahora funciones avanzadas de series temporales.	Puede utilizar Amazon Timestream LiveAnalytics para SQL ejecutar consultas con funciones avanzadas de series temporales, como derivadas, integrales y correlaciones.	10 de febrero de 2021
Amazon Timestream HIPAA ya ISO está disponible y LiveAnalytics cumple con las normas. PCI	Ahora puede usar Amazon Timestream LiveAnalytics para cargas de trabajo que HIPAA requieren una infraestructura compatibleISO. PCI	27 de enero de 2021

Amazon Timestream admite LiveAnalytics por ahora telegraf y grafana de código abierto.	Ahora puede usar Telegraf, el agente de servidor de código abierto y basado en complementos para recopilar e informar métricas, y Grafana, la plataforma de análisis y monitoreo de bases de datos de código abierto, con Amazon Timestream para. LiveAnalytics	25 de noviembre de 2020
Amazon Timestream LiveAnalytics for ya está disponible de forma general.	Esta documentación cubre la versión inicial de Amazon LiveAnalytics Timestream para.	30 de septiembre de 2020

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.