aws

Guía para desarrolladores de SQL

Guía para desarrolladores de aplicaciones de Amazon Kinesis Data Analytics para SQL

Guía para desarrolladores de aplicaciones de Amazon Kinesis Data Analytics para SQL: Guía para desarrolladores de SQL

Table of Contents

Tras considerarlo detenidamente, hemos decidido dejar de utilizar Amazon Kinesis Data Analytics para aplicaciones SQL:

1. A partir del 1 de septiembre de 2025, no proporcionaremos ninguna corrección de errores para las aplicaciones de Amazon Kinesis Data Analytics for SQL porque tendremos un soporte limitado debido a la próxima discontinuación.

2. A partir del 15 de octubre de 2025, no podrá crear nuevas aplicaciones de Kinesis Data Analytics for SQL.

3. Eliminaremos sus aplicaciones a partir del 27 de enero de 2026. No podrá iniciar ni utilizar sus aplicaciones de Amazon Kinesis Data Analytics para SQL. A partir de ese momento, el servicio de soporte de Amazon Kinesis Data Analytics para SQL dejará de estar disponible. Para obtener más información, consulte Retirada de las aplicaciones de Amazon Kinesis Data Analytics para SQL.

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.

Retirada de las aplicaciones de Amazon Kinesis Data Analytics para SQL

Plan de retirada

Tras considerarlo detenidamente, hemos decidido retirar las aplicaciones de Amazon Kinesis Data Analytics para SQL. Para ayudarle a planificar y migrar aplicaciones de Amazon Kinesis Data Analytics para SQL, retiraremos la oferta gradualmente a lo largo de 15 meses. Estas son fechas importantes a tener en cuenta, el 1 de septiembre de 2025, el 15 de octubre de 2025 y el 27 de enero de 2026.

1. El 15 de octubre de 2025, suspenderemos sus aplicaciones y las pondremos en el estado `READY`. Podrá reiniciar sus aplicaciones en ese momento y seguir usándolas de forma normal con sujeción a los límites de servicio.
2. A partir del 15 de octubre de 2025, no podrá crear nuevas aplicaciones de Amazon Kinesis Data Analytics para SQL. Podrá ejecutar las aplicaciones existentes de forma normal con sujeción a los límites de servicio.
3. Eliminaremos sus aplicaciones a partir del 27 de enero de 2026. No podrá iniciar ni utilizar sus aplicaciones de Amazon Kinesis Data Analytics para SQL. A partir de ese momento, las aplicaciones de Amazon Kinesis Data Analytics para SQL dejarán de estar disponibles.

Le recomendamos que migre sus aplicaciones a Amazon Managed Service para Apache Flink o Amazon Managed Service para Apache Flink Studio antes del 15 de octubre de 2025. Para obtener recursos que le ayuden con la migración, consulte Ejemplos de migración a Managed Service para Apache Flink. Para obtener más información sobre Amazon Managed Service for Apache Flink o Amazon Managed Service para Apache Flink Studio, consulte la guía para desarrolladores de Amazon Managed Service para Apache Flink.

> **Note**
>
> Para ver una lista completa de sus aplicaciones de Amazon Kinesis Data Analytics para SQL, puede llamar a la API `ListApplications`. Para obtener más información, consulte ListApplications.

Migración a Amazon Managed Service para Apache Flink Studio

Para obtener más información sobre la migración de las aplicaciones y ver ejemplos de código y arquitectura, consulte Ejemplos de migración a Managed Service para Apache Flink.

Preguntas frecuentes

¿Por qué va a dejar de ofrecer aplicaciones de Amazon Kinesis Data Analytics para SQL?

Hemos descubierto que los clientes prefieren las ofertas de Amazon Managed Service para Apache Flink para las cargas de trabajo de procesamiento de flujo de datos en tiempo real. Amazon Managed Service para Apache Flink es un servicio de procesamiento de flujos en tiempo real sin servidor, de baja latencia, altamente escalable y disponible que utiliza Apache Flink, un motor de código abierto para procesar flujos de datos. Amazon Managed Service para Apache Flink ofrece funcionalidades como escalado nativo, semántica de procesamiento de una sola vez, compatibilidad con varios lenguajes (incluido SQL), más de 40 conectores de origen y destino, estado duradero de la aplicación y mucho más. Estas características ayudan a los clientes a crear canalizaciones de flujo de extremo a extremo y a garantizar la precisión y puntualidad de los datos.

¿Cuáles son las opciones actuales de los clientes?

Recomendamos a los clientes que actualicen sus aplicaciones de Amazon Kinesis Data Analytics para SQL a Amazon Managed Service para Apache Flink Studio o Amazon Managed Service para Apache Flink. En Amazon Managed Service para Apache Flink Studio, los clientes crean consultas mediante SQL, Python o Scala usando cuadernos interactivos. Para aplicaciones de ejecución prolongada en Amazon Kinesis Data Analytics for SQL, recomendamos Amazon Managed Service para Apache Flink, donde los clientes pueden crear aplicaciones con Java, Python, Scala y SQL integrado con todos los conectores, etc. de Apache Flink APIs.

¿Cuántos clientes utilizan Amazon Kinesis Data Analytics para SQL?

No podemos divulgar los detalles de la información del cliente.

¿Cómo se actualizan las aplicaciones de Amazon Kinesis Data Analytics para SQL a una oferta de Amazon Managed Service para Apache Flink?

Para actualizar a Amazon Managed Service para Apache Flink o Amazon Managed Service para Apache Flink Studio, los clientes deben volver a crear su aplicación. Para ayudarlo, hemos proporcionado una biblioteca de consultas SQL habituales e instrucciones sobre cómo reescribirla en

Amazon Managed Service para Apache Flink Studio. Consulte Replicación de consultas de Kinesis Data Analytics para SQL en un servicio gestionado para Apache Flink Studio. También hemos proporcionado arquitecturas de patrones habituales que los clientes pueden seguir si van a crear aplicaciones de larga duración o utilizando machine learning en Amazon Managed Service para Apache Flink. Consulte Sustitución de Kinesis Data Firehose como origen por Kinesis Data Streams

Para obtener más información sobre Amazon Managed Service para Apache Flink, consulte Amazon Managed Service para Apache Flink.

Para obtener orientación sobre la migración, consulte Ejemplos de migración a Managed Service para Apache Flink.

¿Amazon Managed Service para Apache Flink será compatible con las características actuales de aplicaciones de Amazon Kinesis Data Analytics para SQL?

Amazon Managed Service para Apache Flink admite muchos de los conceptos disponibles en las aplicaciones de Amazon Kinesis Data Analytics para SQL, como conectores y ventanas, así como características que no estaban disponibles en las aplicaciones de Amazon Kinesis Data Analytics para SQL, como el escalado nativo, la semántica de procesamiento de un solo paso, la compatibilidad con varios lenguajes (incluido SQL), más de 40 conectores de origen y destino, un estado de aplicación duradero y mucho más.

Ejemplos de migración a Managed Service para Apache Flink

Tras considerarlo detenidamente, hemos decidido retirar las aplicaciones de Amazon Kinesis Data Analytics para SQL. Para ayudarle a planificar y migrar aplicaciones de Amazon Kinesis Data Analytics para SQL, retiraremos la oferta gradualmente a lo largo de 15 meses. Estas son fechas importantes a tener en cuenta, el 1 de septiembre de 2025, el 15 de octubre de 2025 y el 27 de enero de 2026.

1. A partir del 1 de septiembre de 2025, no proporcionaremos ninguna corrección de errores para las aplicaciones de Amazon Kinesis Data Analytics for SQL porque tendremos un soporte limitado debido a la próxima discontinuación.
2. A partir del 15 de octubre de 2025, no podrá crear nuevas aplicaciones de Amazon Kinesis Data Analytics para SQL.
3. Eliminaremos sus aplicaciones a partir del 27 de enero de 2026. No podrá iniciar ni utilizar sus aplicaciones de Amazon Kinesis Data Analytics para SQL. A partir de ese momento, las aplicaciones de Amazon Kinesis Data Analytics para SQL dejarán de estar disponibles. Para obtener más información, consulte Retirada de las aplicaciones de Amazon Kinesis Data Analytics para SQL.

Le recomendamos que utilice Amazon Managed Service para Apache Flink. Combina la facilidad de uso con capacidades analíticas avanzadas, lo que le permite crear aplicaciones de procesamiento de flujos en cuestión de minutos.

Esta sección proporciona ejemplos de código y arquitectura para ayudarle a migrar las cargas de trabajo de las aplicaciones de Amazon Kinesis Data Analytics para SQL a Managed Service para Apache Flink.

Para obtener más información, consulte también esta AWS entrada de blog: Migrate from Amazon Kinesis Data Analytics for SQL Applications to Amazon Managed Service for Apache Flink Studio.

Replicación de consultas de Kinesis Data Analytics para SQL en un servicio gestionado para Apache Flink Studio

Para migrar sus cargas de trabajo a Managed Service para Apache Flink Studio o Managed Service para Apache Flink, en esta sección se proporcionan traducciones de consultas que puede utilizar en casos de uso habituales.

Antes de explorar estos ejemplos, le recomendamos que consulte Uso de un cuaderno de Studio con Managed Service para Apache Flink.

Recreación de consultas de Kinesis Data Analytics para SQL en Managed Service para Apache Flink Studio

Las siguientes opciones proporcionan traducciones de consultas comunes de aplicaciones de Kinesis Data Analytics basadas en SQL a Managed Service para Apache Flink Studio.

Aplicación en varios pasos

SQL-based Kinesis Data Analytics application

```
CREATE
OR REPLACE STREAM "IN_APP_STREAM_001" (
   ingest_time TIMESTAMP,
   ticker_symbol VARCHAR(4),
   sector VARCHAR(16), price REAL, change REAL);
CREATE
OR REPLACE PUMP "STREAM_PUMP_001" AS
INSERT INTO
   "IN_APP_STREAM_001"
   SELECT
      STREAM APPROXIMATE_ARRIVAL_TIME,
      ticker_symbol,
      sector,
      price,
      change FROM "SOURCE_SQL_STREAM_001";
-- Second in-app stream and pump
CREATE
OR REPLACE STREAM "IN_APP_STREAM_02" (ingest_time TIMESTAMP,
   ticker_symbol VARCHAR(4),
   sector VARCHAR(16),
   price REAL,
```

```
   change REAL);
CREATE
OR REPLACE PUMP "STREAM_PUMP_02" AS
INSERT INTO
   "IN_APP_STREAM_02"
   SELECT
      STREAM ingest_time,
      ticker_symbol,
      sector,
      price,
      change FROM "IN_APP_STREAM_001";
-- Destination in-app stream and third pump
CREATE
OR REPLACE STREAM "DESTINATION_SQL_STREAM" (ingest_time TIMESTAMP,
   ticker_symbol VARCHAR(4),
   sector VARCHAR(16),
   price REAL,
   change REAL);
CREATE
OR REPLACE PUMP "STREAM_PUMP_03" AS
INSERT INTO
   "DESTINATION_SQL_STREAM"
   SELECT
      STREAM ingest_time,
      ticker_symbol,
      sector,
      price,
      change FROM "IN_APP_STREAM_02";
```

Managed Service for Apache Flink Studio

```
Query 1 - % flink.ssql DROP TABLE IF EXISTS SOURCE_SQL_STREAM_001;

CREATE TABLE SOURCE_SQL_STREAM_001 (TICKER_SYMBOL VARCHAR(4),
   SECTOR VARCHAR(16),
   PRICE DOUBLE,
   CHANGE DOUBLE,
   APPROXIMATE_ARRIVAL_TIME TIMESTAMP(3) METADATA

FROM
   'timestamp' VIRTUAL,
   WATERMARK FOR APPROXIMATE_ARRIVAL_TIME AS APPROXIMATE_ARRIVAL_TIME - INTERVAL '1'
 SECOND )
```

```
   PARTITIONED BY (TICKER_SYMBOL) WITH (
      'connector' = 'kinesis',
      'stream' = 'kinesis-analytics-demo-stream',
      'aws.region' = 'us-east-1',
      'scan.stream.initpos' = 'LATEST',
      'format' = 'json',
      'json.timestamp-format.standard' = 'ISO-8601');
DROP TABLE IF EXISTS IN_APP_STREAM_001;

CREATE TABLE IN_APP_STREAM_001 (
   INGEST_TIME TIMESTAMP,
   TICKER_SYMBOL VARCHAR(4),
   SECTOR VARCHAR(16),
   PRICE DOUBLE,
   CHANGE DOUBLE )
PARTITIONED BY (TICKER_SYMBOL) WITH (
      'connector' = 'kinesis',
      'stream' = 'IN_APP_STREAM_001',
      'aws.region' = 'us-east-1',
      'scan.stream.initpos' = 'LATEST',
      'format' = 'json',
      'json.timestamp-format.standard' = 'ISO-8601');

DROP TABLE IF EXISTS IN_APP_STREAM_02;

CREATE TABLE IN_APP_STREAM_02 (
   INGEST_TIME TIMESTAMP,
   TICKER_SYMBOL VARCHAR(4),
   SECTOR VARCHAR(16),
   PRICE DOUBLE,
   CHANGE DOUBLE )
PARTITIONED BY (TICKER_SYMBOL) WITH (
   'connector' = 'kinesis',
   'stream' = 'IN_APP_STREAM_02',
   'aws.region' = 'us-east-1',
   'scan.stream.initpos' = 'LATEST',
   'format' = 'json',
   'json.timestamp-format.standard' = 'ISO-8601');

DROP TABLE IF EXISTS DESTINATION_SQL_STREAM;

CREATE TABLE DESTINATION_SQL_STREAM (
   INGEST_TIME TIMESTAMP, TICKER_SYMBOL VARCHAR(4), SECTOR VARCHAR(16),
   PRICE DOUBLE, CHANGE DOUBLE )
```

```
PARTITIONED BY (TICKER_SYMBOL) WITH (
   'connector' = 'kinesis',
   'stream' = 'DESTINATION_SQL_STREAM',
   'aws.region' = 'us-east-1',
   'scan.stream.initpos' = 'LATEST',
   'format' = 'json',
   'json.timestamp-format.standard' = 'ISO-8601');


Query 2 - % flink.ssql(type =
update
)
   INSERT INTO
      IN_APP_STREAM_001
      SELECT
         APPROXIMATE_ARRIVAL_TIME AS INGEST_TIME,
         TICKER_SYMBOL,
         SECTOR,
         PRICE,
         CHANGE
      FROM
         SOURCE_SQL_STREAM_001;


Query 3 - % flink.ssql(type =
update
)
   INSERT INTO
      IN_APP_STREAM_02
      SELECT
         INGEST_TIME,
         TICKER_SYMBOL,
         SECTOR,
         PRICE,
         CHANGE
      FROM
         IN_APP_STREAM_001;


Query 4 - % flink.ssql(type =
update
)
   INSERT INTO
      DESTINATION_SQL_STREAM
```

```
      SELECT
         INGEST_TIME,
         TICKER_SYMBOL,
         SECTOR,
         PRICE,
         CHANGE
      FROM
         IN_APP_STREAM_02;
```

Transformando los valores DateTime

SQL-based Kinesis Data Analytics application

```
CREATE
OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
   TICKER VARCHAR(4),
   event_time TIMESTAMP,
   five_minutes_before TIMESTAMP,
   event_unix_timestamp BIGINT,
   event_timestamp_as_char VARCHAR(50),
   event_second INTEGER);

CREATE
OR REPLACE PUMP "STREAM_PUMP" AS
INSERT INTO
   "DESTINATION_SQL_STREAM"
   SELECT
      STREAM TICKER,
      EVENT_TIME,
      EVENT_TIME - INTERVAL '5' MINUTE,
      UNIX_TIMESTAMP(EVENT_TIME),
      TIMESTAMP_TO_CHAR('yyyy-MM-dd hh:mm:ss', EVENT_TIME),
      EXTRACT(SECOND
   FROM
      EVENT_TIME)
   FROM
      "SOURCE_SQL_STREAM_001"
```

Managed Service for Apache Flink Studio

```
Query 1 - % flink.ssql(type =
update
```

```
) CREATE TABLE DESTINATION_SQL_STREAM (
   TICKER VARCHAR(4),
   EVENT_TIME TIMESTAMP(3),
   FIVE_MINUTES_BEFORE TIMESTAMP(3),
   EVENT_UNIX_TIMESTAMP INT,
   EVENT_TIMESTAMP_AS_CHAR VARCHAR(50),
   EVENT_SECOND INT)

PARTITIONED BY (TICKER) WITH (
   'connector' = 'kinesis', 'stream' = 'kinesis-analytics-demo-stream',
   'aws.region' = 'us-east-1',
   'scan.stream.initpos' = 'LATEST',
   'format' = 'json',
   'json.timestamp-format.standard' = 'ISO-8601')

Query 2 - % flink.ssql(type =
   update
)
      SELECT
         TICKER,
         EVENT_TIME,
         EVENT_TIME - INTERVAL '5' MINUTE AS FIVE_MINUTES_BEFORE,
         UNIX_TIMESTAMP() AS EVENT_UNIX_TIMESTAMP,
         DATE_FORMAT(EVENT_TIME, 'yyyy-MM-dd hh:mm:ss') AS EVENT_TIMESTAMP_AS_CHAR,
         EXTRACT(SECOND
      FROM
         EVENT_TIME) AS EVENT_SECOND
      FROM
         DESTINATION_SQL_STREAM;
```

Alertas simples

SQL-based Kinesis Data Analytics application

```
CREATE
OR REPLACE STREAM "DESTINATION_SQL_STREAM"(
   ticker_symbol VARCHAR(4),
   sector VARCHAR(12),
   change DOUBLE,
   price DOUBLE);

CREATE
```

```
OR REPLACE PUMP "STREAM_PUMP" AS INSERT INTO "DESTINATION_SQL_STREAM"
SELECT
   STREAM ticker_symbol,
   sector,
   change,
   price
FROM
   "SOURCE_SQL_STREAM_001"
WHERE
   (
      ABS(Change / (Price - Change)) * 100
   )
   > 1
```

Managed Service for Apache Flink Studio

```
Query 1 - % flink.ssql(type =
update
) DROP TABLE IF EXISTS DESTINATION_SQL_STREAM;

CREATE TABLE DESTINATION_SQL_STREAM (
   TICKER_SYMBOL VARCHAR(4),
   SECTOR VARCHAR(4),
   CHANGE DOUBLE,
   PRICE DOUBLE )
PARTITIONED BY (TICKER_SYMBOL) WITH (
   'connector' = 'kinesis',
   'stream' = 'kinesis-analytics-demo-stream',
   'aws.region' = 'us-east-1',
   'scan.stream.initpos' = 'LATEST',
   'format' = 'json',
   'json.timestamp-format.standard' = 'ISO-8601');

Query 2 - % flink.ssql(type =
update
)
   SELECT
      TICKER_SYMBOL,
      SECTOR,
      CHANGE,
      PRICE
   FROM
      DESTINATION_SQL_STREAM
```

```
   WHERE
      (
         ABS(CHANGE / (PRICE - CHANGE)) * 100
      )
      > 1;
```

Alertas limitadas

SQL-based Kinesis Data Analytics application

```
CREATE
OR REPLACE STREAM "CHANGE_STREAM"(
   ticker_symbol VARCHAR(4),
   sector VARCHAR(12),
   change DOUBLE,
   price DOUBLE);

CREATE
OR REPLACE PUMP "change_pump" AS INSERT INTO "CHANGE_STREAM"
SELECT
   STREAM ticker_symbol,
   sector,
   change,
   price
FROM "SOURCE_SQL_STREAM_001"
WHERE
   (
      ABS(Change / (Price - Change)) * 100
   )
   > 1;
-- ** Trigger Count and Limit **
-- Counts "triggers" or those values that evaluated true against the previous where
 clause
-- Then provides its own limit on the number of triggers per hour per ticker symbol
 to what is specified in the WHERE clause

CREATE
OR REPLACE STREAM TRIGGER_COUNT_STREAM (
   ticker_symbol VARCHAR(4),
   change REAL,
   trigger_count INTEGER);
```

```
CREATE
OR REPLACE PUMP trigger_count_pump AS
INSERT INTO
   TRIGGER_COUNT_STREAMSELECT STREAM ticker_symbol,
   change,
   trigger_count
FROM
   (
      SELECT
         STREAM ticker_symbol,
         change,
         COUNT(*) OVER W1 as trigger_countFROM "CHANGE_STREAM" --window to perform
 aggregations over last minute to keep track of triggers
         WINDOW W1 AS
         (
            PARTITION BY ticker_symbol RANGE INTERVAL '1' MINUTE PRECEDING
         )
   )
WHERE
   trigger_count >= 1;
```

Managed Service for Apache Flink Studio

```
Query 1 - % flink.ssql(type =
update
) DROP TABLE IF EXISTS DESTINATION_SQL_STREAM;

CREATE TABLE DESTINATION_SQL_STREAM (
   TICKER_SYMBOL VARCHAR(4),
   SECTOR VARCHAR(4),
   CHANGE DOUBLE, PRICE DOUBLE,
   EVENT_TIME AS PROCTIME())
PARTITIONED BY (TICKER_SYMBOL)
WITH (
   'connector' = 'kinesis',
   'stream' = 'kinesis-analytics-demo-stream',
   'aws.region' = 'us-east-1',
   'scan.stream.initpos' = 'LATEST',
   'format' = 'json',
   'json.timestamp-format.standard' = 'ISO-8601');
DROP TABLE IF EXISTS TRIGGER_COUNT_STREAM;
CREATE TABLE TRIGGER_COUNT_STREAM (
   TICKER_SYMBOL VARCHAR(4),
```

```
   CHANGE DOUBLE,
   TRIGGER_COUNT INT)
PARTITIONED BY (TICKER_SYMBOL);

Query 2 - % flink.ssql(type =
update
)
   SELECT
      TICKER_SYMBOL,
      SECTOR,
      CHANGE,
      PRICE
   FROM
      DESTINATION_SQL_STREAM
   WHERE
      (
         ABS(CHANGE / (PRICE - CHANGE)) * 100
      )
      > 1;

Query 3 - % flink.ssql(type =
update
)
   SELECT *
   FROM(
         SELECT
            TICKER_SYMBOL,
            CHANGE,
            COUNT(*) AS TRIGGER_COUNT
         FROM
            DESTINATION_SQL_STREAM
         GROUP BY
            TUMBLE(EVENT_TIME, INTERVAL '1' MINUTE),
            TICKER_SYMBOL,
            CHANGE
      )
   WHERE
      TRIGGER_COUNT > 1;
```

Agregar resultados parciales de una consulta

SQL-based Kinesis Data Analytics application

```
CREATE
OR REPLACE STREAM "CALC_COUNT_SQL_STREAM"(
   TICKER VARCHAR(4),
   TRADETIME TIMESTAMP,
   TICKERCOUNT DOUBLE);

CREATE
OR REPLACE STREAM "DESTINATION_SQL_STREAM"(
   TICKER VARCHAR(4),
   TRADETIME TIMESTAMP,
   TICKERCOUNT DOUBLE);

CREATE PUMP "CALC_COUNT_SQL_PUMP_001" AS
INSERT INTO
   "CALC_COUNT_SQL_STREAM"(
   "TICKER",
   "TRADETIME",
   "TICKERCOUNT")
   SELECT
      STREAM "TICKER_SYMBOL",
      STEP("SOURCE_SQL_STREAM_001",
      "ROWTIME" BY INTERVAL '1' MINUTE) as "TradeTime",
      COUNT(*) AS "TickerCount "
   FROM
      "SOURCE_SQL_STREAM_001"
   GROUP BY
      STEP("SOURCE_SQL_STREAM_001". ROWTIME BY INTERVAL '1' MINUTE),
      STEP("SOURCE_SQL_STREAM_001"." APPROXIMATE_ARRIVAL_TIME" BY INTERVAL '1'
 MINUTE),
      TICKER_SYMBOL;
CREATE PUMP "AGGREGATED_SQL_PUMP" AS
INSERT INTO
   "DESTINATION_SQL_STREAM" (
   "TICKER",
   "TRADETIME",
   "TICKERCOUNT")
   SELECT
      STREAM "TICKER",
      "TRADETIME",
      SUM("TICKERCOUNT") OVER W1 AS "TICKERCOUNT"
```

```
   FROM
      "CALC_COUNT_SQL_STREAM" WINDOW W1 AS
      (
         PARTITION BY "TRADETIME" RANGE INTERVAL '10' MINUTE PRECEDING
      )
;
```

Managed Service for Apache Flink Studio

```
Query 1 - % flink.ssql(type =
update
) DROP TABLE IF EXISTS SOURCE_SQL_STREAM_001;
CREATE TABLE SOURCE_SQL_STREAM_001 (
   TICKER_SYMBOL VARCHAR(4),
   TRADETIME AS PROCTIME(),
   APPROXIMATE_ARRIVAL_TIME TIMESTAMP(3) METADATA
FROM
   'timestamp' VIRTUAL,
   WATERMARK FOR APPROXIMATE_ARRIVAL_TIME AS APPROXIMATE_ARRIVAL_TIME - INTERVAL '1'
 SECOND)
PARTITIONED BY (TICKER_SYMBOL) WITH (
   'connector' = 'kinesis',
   'stream' = 'kinesis-analytics-demo-stream',
   'aws.region' = 'us-east-1',
   'scan.stream.initpos' = 'LATEST',
   'format' = 'json',
   'json.timestamp-format.standard' = 'ISO-8601');
DROP TABLE IF EXISTS CALC_COUNT_SQL_STREAM;
CREATE TABLE CALC_COUNT_SQL_STREAM (
   TICKER VARCHAR(4),
   TRADETIME TIMESTAMP(3),
   WATERMARK FOR TRADETIME AS TRADETIME - INTERVAL '1' SECOND,
   TICKERCOUNT BIGINT NOT NULL ) PARTITIONED BY (TICKER) WITH (
      'connector' = 'kinesis',
      'stream' = 'CALC_COUNT_SQL_STREAM',
      'aws.region' = 'us-east-1',
      'scan.stream.initpos' = 'LATEST',
      'format' = 'csv');
DROP TABLE IF EXISTS DESTINATION_SQL_STREAM;
CREATE TABLE DESTINATION_SQL_STREAM (
   TICKER VARCHAR(4),
   TRADETIME TIMESTAMP(3),
   WATERMARK FOR TRADETIME AS TRADETIME - INTERVAL '1' SECOND,
```

```
   TICKERCOUNT BIGINT NOT NULL )
   PARTITIONED BY (TICKER) WITH ('connector' = 'kinesis',
      'stream' = 'DESTINATION_SQL_STREAM',
      'aws.region' = 'us-east-1',
      'scan.stream.initpos' = 'LATEST',
      'format' = 'csv');

Query 2 - % flink.ssql(type =
update
)
   INSERT INTO
      CALC_COUNT_SQL_STREAM
      SELECT
         TICKER,
         TO_TIMESTAMP(TRADETIME, 'yyyy-MM-dd HH:mm:ss') AS TRADETIME,
         TICKERCOUNT
      FROM
         (
            SELECT
               TICKER_SYMBOL AS TICKER,
               DATE_FORMAT(TRADETIME, 'yyyy-MM-dd HH:mm:00') AS TRADETIME,
               COUNT(*) AS TICKERCOUNT
            FROM
               SOURCE_SQL_STREAM_001
            GROUP BY
               TUMBLE(TRADETIME, INTERVAL '1' MINUTE),
               DATE_FORMAT(TRADETIME, 'yyyy-MM-dd HH:mm:00'),
               DATE_FORMAT(APPROXIMATE_ARRIVAL_TIME, 'yyyy-MM-dd HH:mm:00'),
               TICKER_SYMBOL
         )
;

Query 3 - % flink.ssql(type =
update
)
   SELECT
      *
   FROM
      CALC_COUNT_SQL_STREAM;

Query 4 - % flink.ssql(type =
update
)
   INSERT INTO
```

```
      DESTINATION_SQL_STREAM
      SELECT
         TICKER,
         TRADETIME,
         SUM(TICKERCOUNT) OVER W1 AS TICKERCOUNT
      FROM
         CALC_COUNT_SQL_STREAM WINDOW W1 AS
         (
            PARTITION BY TICKER
         ORDER BY
            TRADETIME RANGE INTERVAL '10' MINUTE PRECEDING
         )
;

Query 5 - % flink.ssql(type =
update
)
   SELECT
      *
   FROM
      DESTINATION_SQL_STREAM;
```

Transformación de valores de cadena

SQL-based Kinesis Data Analytics application

```
CREATE
OR REPLACE STREAM for cleaned up referrerCREATE
OR REPLACE STREAM "DESTINATION_SQL_STREAM" ( "ingest_time" TIMESTAMP, "referrer"
 VARCHAR(32));
CREATE
OR REPLACE PUMP "myPUMP" AS INSERT INTO "DESTINATION_SQL_STREAM"
SELECT
   STREAM "APPROXIMATE_ARRIVAL_TIME",
   SUBSTRING("referrer", 12,
   (
      POSITION('.com' IN "referrer") - POSITION('www.' IN "referrer") - 4
   )
)
FROM
   "SOURCE_SQL_STREAM_001";
```

Managed Service for Apache Flink Studio

```
Query 1 - % flink.ssql(type =
update
) CREATE TABLE DESTINATION_SQL_STREAM (
   referrer VARCHAR(32),
   ingest_time AS PROCTIME() ) PARTITIONED BY (referrer)
WITH (
   'connector' = 'kinesis',
   'stream' = 'kinesis-analytics-demo-stream',
   'aws.region' = 'us-east-1',
   'scan.stream.initpos' = 'LATEST',
   'format' = 'json',
   'json.timestamp-format.standard' = 'ISO-8601')

Query 2 - % flink.ssql(type =
   update
)
      SELECT
         ingest_time,
         substring(referrer, 12, 6) as referrer
      FROM
         DESTINATION_SQL_STREAM;
```

Sustitución de una subcadena mediante Regex

SQL-based Kinesis Data Analytics application

```
CREATE
OR REPLACE STREAM for cleaned up referrerCREATE
OR REPLACE STREAM "DESTINATION_SQL_STREAM" ( "ingest_time" TIMESTAMP, "referrer"
 VARCHAR(32));
CREATE
OR REPLACE PUMP "myPUMP" AS INSERT INTO "DESTINATION_SQL_STREAM"
SELECT
   STREAM "APPROXIMATE_ARRIVAL_TIME",
   REGEX_REPLACE("REFERRER", 'http://', 'https://', 1, 0)
FROM
   "SOURCE_SQL_STREAM_001";
```

Managed Service for Apache Flink Studio

```
Query 1 - % flink.ssql(type =
update
) CREATE TABLE DESTINATION_SQL_STREAM (
   referrer VARCHAR(32),
   ingest_time AS PROCTIME())
PARTITIONED BY (referrer) WITH (
   'connector' = 'kinesis',
   'stream' = 'kinesis-analytics-demo-stream',
   'aws.region' = 'us-east-1',
   'scan.stream.initpos' = 'LATEST',
   'format' = 'json',
   'json.timestamp-format.standard' = 'ISO-8601')

Query 2 - % flink.ssql(type =
   update
)
      SELECT
         ingest_time,
         REGEXP_REPLACE(referrer, 'http', 'https') as referrer
      FROM
         DESTINATION_SQL_STREAM;
```

Análisis de registros de expresiones regulares

SQL-based Kinesis Data Analytics application

```
CREATE
OR REPLACE STREAM "DESTINATION_SQL_STREAM"(
   sector VARCHAR(24),
   match1 VARCHAR(24),
   match2 VARCHAR(24));
CREATE
OR REPLACE PUMP "STREAM_PUMP" AS
INSERT INTO
   "DESTINATION_SQL_STREAM"
   SELECT
      STREAM T.SECTOR,
      T.REC.COLUMN1,
      T.REC.COLUMN2
   FROM
```

```
      (
         SELECT
            STREAM SECTOR,
            REGEX_LOG_PARSE(SECTOR, '.*([E].).*([R].*)') AS REC
         FROM
            SOURCE_SQL_STREAM_001
      )
      AS T;
```

Managed Service for Apache Flink Studio

```
Query 1 - % flink.ssql(type =
update
) CREATE TABLE DESTINATION_SQL_STREAM (
   CHANGE DOUBLE, PRICE DOUBLE,
   TICKER_SYMBOL VARCHAR(4),
   SECTOR VARCHAR(16))
PARTITIONED BY (SECTOR) WITH (
   'connector' = 'kinesis',
   'stream' = 'kinesis-analytics-demo-stream',
   'aws.region' = 'us-east-1',
   'scan.stream.initpos' = 'LATEST',
   'format' = 'json',
   'json.timestamp-format.standard' = 'ISO-8601')

Query 2 - % flink.ssql(type =
   update
)
SELECT
   *
FROM
   (
      SELECT
         SECTOR,
         REGEXP_EXTRACT(SECTOR, '.([E].).([R].)', 1) AS MATCH1,
         REGEXP_EXTRACT(SECTOR, '.([E].).([R].)', 2) AS MATCH2
      FROM
         DESTINATION_SQL_STREAM
   )
WHERE
   MATCH1 IS NOT NULL
   AND MATCH2 IS NOT NULL;
```

Transformando DateTime valores

SQL-based Kinesis Data Analytics application

```
CREATE
OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
   TICKER VARCHAR(4),
   event_time TIMESTAMP,
   five_minutes_before TIMESTAMP,
   event_unix_timestamp BIGINT,
   event_timestamp_as_char VARCHAR(50),
   event_second INTEGER);
CREATE
OR REPLACE PUMP "STREAM_PUMP" AS
INSERT INTO
   "DESTINATION_SQL_STREAM"
   SELECT
      STREAM TICKER,
      EVENT_TIME,
      EVENT_TIME - INTERVAL '5' MINUTE,
      UNIX_TIMESTAMP(EVENT_TIME),
      TIMESTAMP_TO_CHAR('yyyy-MM-dd hh:mm:ss', EVENT_TIME),
      EXTRACT(SECOND
   FROM
      EVENT_TIME)
   FROM
      "SOURCE_SQL_STREAM_001"
```

Managed Service for Apache Flink Studio

```
Query 1 - % flink.ssql(type =
update
) CREATE TABLE DESTINATION_SQL_STREAM (
   TICKER VARCHAR(4),
   EVENT_TIME TIMESTAMP(3),
   FIVE_MINUTES_BEFORE TIMESTAMP(3),
   EVENT_UNIX_TIMESTAMP INT,
   EVENT_TIMESTAMP_AS_CHAR VARCHAR(50),
   EVENT_SECOND INT) PARTITIONED BY (TICKER)
WITH (
   'connector' = 'kinesis',
   'stream' = 'kinesis-analytics-demo-stream',
   'aws.region' = 'us-east-1',
```

```
   'scan.stream.initpos' = 'LATEST',
   'format' = 'json',
   'json.timestamp-format.standard' = 'ISO-8601')

Query 2 - % flink.ssql(type =
   update
)
      SELECT
         TICKER,
         EVENT_TIME,
         EVENT_TIME - INTERVAL '5' MINUTE AS FIVE_MINUTES_BEFORE,
         UNIX_TIMESTAMP() AS EVENT_UNIX_TIMESTAMP,
         DATE_FORMAT(EVENT_TIME, 'yyyy-MM-dd hh:mm:ss') AS EVENT_TIMESTAMP_AS_CHAR,
         EXTRACT(SECOND
      FROM
         EVENT_TIME) AS EVENT_SECOND
      FROM
         DESTINATION_SQL_STREAM;
```

Ventanas y agregación

SQL-based Kinesis Data Analytics application

```
CREATE
OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
   event_time TIMESTAMP,
   ticker_symbol VARCHAR(4),
   ticker_count INTEGER);
CREATE
OR REPLACE PUMP "STREAM_PUMP" AS
INSERT INTO
   "DESTINATION_SQL_STREAM"
   SELECT
      STREAM EVENT_TIME,
      TICKER,
      COUNT(TICKER) AS ticker_count
   FROM
      "SOURCE_SQL_STREAM_001" WINDOWED BY STAGGER ( PARTITION BY
         TICKER,
         EVENT_TIME RANGE INTERVAL '1' MINUTE);
```

Managed Service for Apache Flink Studio

```
Query 1 - % flink.ssql(type =
update
) CREATE TABLE DESTINATION_SQL_STREAM (
   EVENT_TIME TIMESTAMP(3),
   WATERMARK FOR EVENT_TIME AS EVENT_TIME - INTERVAL '60' SECOND,
   TICKER VARCHAR(4),
   TICKER_COUNT INT) PARTITIONED BY (TICKER)
WITH (
   'connector' = 'kinesis',
   'stream' = 'kinesis-analytics-demo-stream',
   'aws.region' = 'us-east-1',
   'scan.stream.initpos' = 'LATEST',
   'format' = 'json'

Query 2 - % flink.ssql(type =
   update
)
      SELECT
         EVENT_TIME,
         TICKER, COUNT(TICKER) AS ticker_count
      FROM
         DESTINATION_SQL_STREAM
      GROUP BY
         TUMBLE(EVENT_TIME,
         INTERVAL '60' second),
         EVENT_TIME, TICKER;
```

Ventana de saltos con Rowtime

SQL-based Kinesis Data Analytics application

```
CREATE
OR REPLACE STREAM "DESTINATION_SQL_STREAM"(
   TICKER VARCHAR(4),
   MIN_PRICE REAL,
   MAX_PRICE REAL);
CREATE
OR REPLACE PUMP "STREAM_PUMP" AS
INSERT INTO
   "DESTINATION_SQL_STREAM"
```

```
   SELECT
      STREAM TICKER,
      MIN(PRICE),
      MAX(PRICE)
   FROM
      "SOURCE_SQL_STREAM_001"
   GROUP BY
      TICKER,
      STEP("SOURCE_SQL_STREAM_001".
            ROWTIME BY INTERVAL '60' SECOND);
```

Managed Service for Apache Flink Studio

```
Query 1 - % flink.ssql(type =
update
) CREATE TABLE DESTINATION_SQL_STREAM (
   ticker VARCHAR(4),
   price DOUBLE,
   event_time VARCHAR(32),
   processing_time AS PROCTIME())
PARTITIONED BY (ticker) WITH (
   'connector' = 'kinesis',
   'stream' = 'kinesis-analytics-demo-stream',
   'aws.region' = 'us-east-1',
   'scan.stream.initpos' = 'LATEST',
   'format' = 'json',
   'json.timestamp-format.standard' = 'ISO-8601')

Query 2 - % flink.ssql(type =
   update
)
      SELECT
         ticker,
         min(price) AS MIN_PRICE,
         max(price) AS MAX_PRICE
      FROM
         DESTINATION_SQL_STREAM
      GROUP BY
         TUMBLE(processing_time, INTERVAL '60' second),
         ticker;
```

Recuperación de los valores más frecuentes (TOP_K_ITEMS_TUMBLING)

SQL-based Kinesis Data Analytics application

```
CREATE
OR REPLACE STREAM "CALC_COUNT_SQL_STREAM"(TICKER VARCHAR(4),
   TRADETIME TIMESTAMP,
   TICKERCOUNT DOUBLE);
CREATE
OR REPLACE STREAM "DESTINATION_SQL_STREAM"(
   TICKER VARCHAR(4),
   TRADETIME TIMESTAMP,
   TICKERCOUNT DOUBLE);
CREATE PUMP "CALC_COUNT_SQL_PUMP_001" AS INSERT INTO "CALC_COUNT_SQL_STREAM" (
   "TICKER",
   "TRADETIME",
   "TICKERCOUNT")
SELECT
   STREAM"TICKER_SYMBOL",
   STEP("SOURCE_SQL_STREAM_001"."ROWTIME" BY INTERVAL '1' MINUTE) as "TradeTime",
   COUNT(*) AS "TickerCount"
FROM
   "SOURCE_SQL_STREAM_001"
GROUP BY STEP("SOURCE_SQL_STREAM_001".
   ROWTIME BY INTERVAL '1' MINUTE),
   STEP("SOURCE_SQL_STREAM_001".
      "APPROXIMATE_ARRIVAL_TIME" BY INTERVAL '1' MINUTE),
   TICKER_SYMBOL;
CREATE PUMP "AGGREGATED_SQL_PUMP" AS INSERT INTO "DESTINATION_SQL_STREAM" (
   "TICKER",
   "TRADETIME",
   "TICKERCOUNT")
SELECT
   STREAM "TICKER",
   "TRADETIME",
   SUM("TICKERCOUNT") OVER W1 AS "TICKERCOUNT"
FROM
   "CALC_COUNT_SQL_STREAM" WINDOW W1 AS
   (
      PARTITION BY "TRADETIME" RANGE INTERVAL '10' MINUTE PRECEDING
   )
;
```

Managed Service for Apache Flink Studio

```
Query 1 - % flink.ssql(type =
update
) DROP TABLE IF EXISTS DESTINATION_SQL_STREAM;
CREATE TABLE DESTINATION_SQL_STREAM (
   TICKER VARCHAR(4),
   EVENT_TIME TIMESTAMP(3),
   WATERMARK FOR EVENT_TIME AS EVENT_TIME - INTERVAL '1' SECONDS )
PARTITIONED BY (TICKER) WITH (
   'connector' = 'kinesis', 'stream' = 'kinesis-analytics-demo-stream',
   'aws.region' = 'us-east-1',
   'scan.stream.initpos' = 'LATEST',
   'format' = 'json',
   'json.timestamp-format.standard' = 'ISO-8601');

Query 2 - % flink.ssql(type =
update
)
   SELECT
      *
   FROM
      (
         SELECT
            TICKER,
            COUNT(*) as MOST_FREQUENT_VALUES,
            ROW_NUMBER() OVER (PARTITION BY TICKER
         ORDER BY
            TICKER DESC) AS row_num
         FROM
            DESTINATION_SQL_STREAM
         GROUP BY
            TUMBLE(EVENT_TIME, INTERVAL '1' MINUTE),
            TICKER
      )
   WHERE
      row_num <= 5;
```

Elementos aproximados del Top-K

SQL-based Kinesis Data Analytics application

```
CREATE
OR REPLACE STREAM "DESTINATION_SQL_STREAM" (ITEM VARCHAR(1024), ITEM_COUNT DOUBLE);
CREATE
OR REPLACE PUMP "STREAM_PUMP" AS
INSERT INTO
   "DESTINATION_SQL_STREAM"
   SELECT
      STREAM ITEM,
      ITEM_COUNT
   FROM
      TABLE(TOP_K_ITEMS_TUMBLING(CURSOR(
      SELECT
         STREAM *
      FROM
         "SOURCE_SQL_STREAM_001"), 'column1', -- name of column in single quotes10,
 -- number of top items60 -- tumbling window size in seconds));
```

Managed Service for Apache Flink Studio

```
%flinkssql
DROP TABLE IF EXISTS SOURCE_SQL_STREAM_001
CREATE TABLE SOURCE_SQL_STREAM_001 ( TS TIMESTAMP(3), WATERMARK FOR TS as TS -
 INTERVAL '5' SECOND, ITEM VARCHAR(1024),
PRICE DOUBLE)
   WITH ( 'connector' = 'kinesis', 'stream' = 'SOURCE_SQL_STREAM_001',
'aws.region' = 'us-east-1', 'scan.stream.initpos' = 'LATEST', 'format' = 'json',
'json.timestamp-format.standard' = 'ISO-8601');


%flink.ssql(type=update)
SELECT
   *
FROM
   (
      SELECT
         *,
         ROW_NUMBER() OVER (PARTITION BY AGG_WINDOW
      ORDER BY
```

```
         ITEM_COUNT DESC) as rownum
      FROM
         (
            select
               AGG_WINDOW,
               ITEM,
               ITEM_COUNT
            from
               (
                  select
                     TUMBLE_ROWTIME(TS, INTERVAL '60' SECONDS) as AGG_WINDOW,
                     ITEM,
                     count(*) as ITEM_COUNT
                  FROM
                     SOURCE_SQL_STREAM_001
                  GROUP BY
                     TUMBLE(TS, INTERVAL '60' SECONDS),
                     ITEM
               )
         )
   )
where
   rownum <= 3
```

Análisis de registros web (función W3C_LOG_PARSE)

SQL-based Kinesis Data Analytics application

```
CREATE
OR REPLACE STREAM "DESTINATION_SQL_STREAM" ( column1 VARCHAR(16),
   column2 VARCHAR(16),
   column3 VARCHAR(16),
   column4 VARCHAR(16),
   column5 VARCHAR(16),
   column6 VARCHAR(16),
   column7 VARCHAR(16));
CREATE
OR REPLACE PUMP "myPUMP" ASINSERT INTO "DESTINATION_SQL_STREAM"
SELECT
   STREAM l.r.COLUMN1,
   l.r.COLUMN2,
   l.r.COLUMN3,
```

```
   l.r.COLUMN4,
   l.r.COLUMN5,
   l.r.COLUMN6,
   l.r.COLUMN7
FROM
   (
      SELECT
         STREAM W3C_LOG_PARSE("log", 'COMMON')
      FROM
         "SOURCE_SQL_STREAM_001"
   )
   AS l(r);
```

Managed Service for Apache Flink Studio

```
%flink.ssql(type=update)
DROP TABLE IF EXISTS SOURCE_SQL_STREAM_001 CREATE TABLE SOURCE_SQL_STREAM_001 ( log
 VARCHAR(1024))
   WITH ( 'connector' = 'kinesis',
          'stream' = 'SOURCE_SQL_STREAM_001',
          'aws.region' = 'us-east-1',
          'scan.stream.initpos' = 'LATEST',
          'format' = 'json',
          'json.timestamp-format.standard' = 'ISO-8601');

% flink.ssql(type=update)
   select
      SPLIT_INDEX(log, ' ', 0),
      SPLIT_INDEX(log, ' ', 1),
      SPLIT_INDEX(log, ' ', 2),
      SPLIT_INDEX(log, ' ', 3),
      SPLIT_INDEX(log, ' ', 4),
      SPLIT_INDEX(log, ' ', 5),
      SPLIT_INDEX(log, ' ', 6)
   from
      SOURCE_SQL_STREAM_001;
```

División de cadenas en varios campos (función VARIABLE_COLUMN_LOG_PARSE)

SQL-based Kinesis Data Analytics application

```
CREATE
```

```
OR REPLACE STREAM "DESTINATION_SQL_STREAM"( "column_A" VARCHAR(16),
   "column_B" VARCHAR(16),
   "column_C" VARCHAR(16),
   "COL_1" VARCHAR(16),
   "COL_2" VARCHAR(16),
   "COL_3" VARCHAR(16));
CREATE
OR REPLACE PUMP "SECOND_STREAM_PUMP" AS INSERT INTO "DESTINATION_SQL_STREAM"
SELECT
   STREAM t."Col_A",
   t."Col_B",
   t."Col_C",
   t.r."COL_1",
   t.r."COL_2",
   t.r."COL_3"
FROM
   (
      SELECT
         STREAM "Col_A",
         "Col_B",
         "Col_C",
         VARIABLE_COLUMN_LOG_PARSE ("Col_E_Unstructured",
         'COL_1 TYPE VARCHAR(16),
         COL_2 TYPE VARCHAR(16),
         COL_3 TYPE VARCHAR(16)', ',') AS r
      FROM
         "SOURCE_SQL_STREAM_001"
   )
   as t;
```

Managed Service for Apache Flink Studio

```
%flink.ssql(type=update)
DROP TABLE IF EXISTS SOURCE_SQL_STREAM_001 CREATE TABLE SOURCE_SQL_STREAM_001 ( log
 VARCHAR(1024))
   WITH ( 'connector' = 'kinesis',
          'stream' = 'SOURCE_SQL_STREAM_001',
          'aws.region' = 'us-east-1',
          'scan.stream.initpos' = 'LATEST',
          'format' = 'json',
          'json.timestamp-format.standard' = 'ISO-8601');

% flink.ssql(type=update)
```

```
   select
      SPLIT_INDEX(log, ' ', 0),
      SPLIT_INDEX(log, ' ', 1),
      SPLIT_INDEX(log, ' ', 2),
      SPLIT_INDEX(log, ' ', 3),
      SPLIT_INDEX(log, ' ', 4),
      SPLIT_INDEX(log, ' ', 5)
)
from
   SOURCE_SQL_STREAM_001;
```

Uniones

SQL-based Kinesis Data Analytics application

```
CREATE
OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
   ticker_symbol VARCHAR(4),
   "Company" varchar(20),
   sector VARCHAR(12),
   change DOUBLE,
   price DOUBLE);
CREATE
OR REPLACE PUMP "STREAM_PUMP" AS
INSERT INTO
   "DESTINATION_SQL_STREAM"
   SELECT
      STREAM ticker_symbol,
      "c"."Company",
      sector,
      change,
      priceFROM "SOURCE_SQL_STREAM_001"
      LEFT JOIN
         "CompanyName" as "c"
         ON "SOURCE_SQL_STREAM_001".ticker_symbol = "c"."Ticker";
```

Managed Service for Apache Flink Studio

```
Query 1 - % flink.ssql(type =
update
) CREATE TABLE DESTINATION_SQL_STREAM (
   TICKER_SYMBOL VARCHAR(4),
```

```
   SECTOR VARCHAR(12),
   CHANGE INT,
   PRICE DOUBLE )
PARTITIONED BY (TICKER_SYMBOL) WITH (
   'connector' = 'kinesis',
   'stream' = 'kinesis-analytics-demo-stream',
   'aws.region' = 'us-east-1',
   'scan.stream.initpos' = 'LATEST',
   'format' = 'json',
   'json.timestamp-format.standard' = 'ISO-8601');

Query 2 - CREATE TABLE CompanyName (
   Ticker VARCHAR(4),
   Company VARCHAR(4)) WITH (
      'connector' = 'filesystem',
      'path' = 's3://kda-demo-sample/TickerReference.csv',
      'format' = 'csv' );

Query 3 - % flink.ssql(type =
update
)
   SELECT
      TICKER_SYMBOL,
      c.Company,
      SECTOR,
      CHANGE,
      PRICE
   FROM
      DESTINATION_SQL_STREAM
      LEFT JOIN
         CompanyName as c
         ON DESTINATION_SQL_STREAM.TICKER_SYMBOL = c.Ticker;
```

Errores

SQL-based Kinesis Data Analytics application

```
SELECT
   STREAM ticker_symbol,
   sector,
   change,
   (
```

```
      price / 0
   )
   as ProblemColumnFROM "SOURCE_SQL_STREAM_001"
WHERE
   sector SIMILAR TO '%TECH%';
```

Managed Service for Apache Flink Studio

```
Query 1 - % flink.ssql(type =
update
) DROP TABLE IF EXISTS DESTINATION_SQL_STREAM;
CREATE TABLE DESTINATION_SQL_STREAM (
   TICKER_SYMBOL VARCHAR(4),
   SECTOR VARCHAR(16),
   CHANGE DOUBLE,
   PRICE DOUBLE )
PARTITIONED BY (TICKER_SYMBOL) WITH (
   'connector' = 'kinesis',
   'stream' = 'kinesis-analytics-demo-stream',
   'aws.region' = 'us-east-1',
   'scan.stream.initpos' = 'LATEST',
   'format' = 'json',
   'json.timestamp-format.standard' = 'ISO-8601');

Query 2 - % flink.pyflink @udf(input_types = [DataTypes.BIGINT()],
   result_type = DataTypes.BIGINT()) def DivideByZero(price): try: price / 0
except
: return - 1 st_env.register_function("DivideByZero",
   DivideByZero)

   Query 3 - % flink.ssql(type =
update
)
   SELECT
      CURRENT_TIMESTAMP AS ERROR_TIME,
      *
   FROM
      (
         SELECT
            TICKER_SYMBOL,
            SECTOR,
            CHANGE,
            DivideByZero(PRICE) as ErrorColumn
```

```
        FROM
           DESTINATION_SQL_STREAM
        WHERE
           SECTOR SIMILAR TO '%TECH%'
    )
    AS ERROR_STREAM;
```

Migración de cargas de trabajo del bosque de corte aleatorio

Si quiere trasladar cargas de trabajo que utiliza el bosque de corte aleatorio de Kinesis Analytics para SQL a Managed Service para Apache Flink, en esta entrada de blog de AWS se muestra cómo utilizar Managed Service para Apache Flink con el fin de ejecutar un algoritmo RCF en línea en la detección de anomalías.

Sustitución de Kinesis Data Firehose como origen por Kinesis Data Streams

Consulte Converting-KDASQL-KDAStudio/para ver un tutorial completo.

En el siguiente ejercicio, cambiará su flujo de datos para usar Amazon Managed Service para Apache Flink Studio. Esto también implicará cambiar de Amazon Kinesis Data Firehose a Amazon Kinesis Data Streams.

En primer lugar, compartimos una arquitectura típica de KDA-SQL, antes de mostrar cómo puede sustituirla mediante Amazon Managed Service para Apache Flink Studio y Amazon Kinesis Data Streams. Como alternativa, puede lanzar la plantilla aquí: CloudFormation

Amazon Kinesis Data Analytics-SQL y Amazon Kinesis Data Firehose

Este es el flujo de arquitectura SQL de Amazon Kinesis Data Analytics:

En primer lugar, examinamos la configuración de Amazon Kinesis Data Analytics-SQL y Amazon Kinesis Data Firehose anteriores. El caso de uso es un mercado bursátil en el que los datos de

negociación, incluidos el precio y el precio de las acciones, se transmiten desde fuentes externas a los sistemas Amazon Kinesis. Amazon Kinesis Data Analytics para SQL utiliza el flujo de entrada para ejecutar consultas en ventana, como Tumbling Window, a fin de determinar el volumen de operaciones y el precio de negociación `min`, `max` y `average` durante un período de un minuto para cada cotización bursátil.

Amazon Kinesis Data Analytics-SQL está configurado para ingerir datos de la API Amazon Kinesis Data Firehose. Tras el procesamiento, Amazon Kinesis Data Analytics-SQL envía los datos procesados a otra Amazon Kinesis Data Firehose, que luego guarda la salida en un bucket de Amazon S3.

En este caso, utiliza Amazon Kinesis Data Generator. Amazon Kinesis Data Generator le permite enviar datos de prueba a sus flujos de entrega de Amazon Kinesis Data Streams o Amazon Kinesis Data Firehose. Para empezar, siga las instrucciones que aparecen aquí. Utilice la CloudFormation plantilla aquí en lugar de la que se proporciona en las instrucciones:.

Una vez que ejecute la CloudFormation plantilla, la sección de resultados proporcionará la URL del generador de datos de Amazon Kinesis. Inicie sesión en el portal con el ID y la contraseña de Cognito que configuró aquí. Seleccione la región y el nombre del flujo de destino. Para ver el estado actual, elija los flujos de Amazon Kinesis Data Firehose Delivery. Para ver el estado nuevo, elija los flujos de Amazon Kinesis Data Firehose. Puede crear varias plantillas, en función de sus requisitos, y probarlas con el botón Probar plantilla antes de enviarlas al flujo de destino.

A continuación, se presenta un ejemplo de carga útil con Amazon Kinesis Data Generator. El generador de datos se dirige a la entrada de Amazon Kinesis Firehose Streams para transmitir los datos de forma continua. El cliente del SDK de Amazon Kinesis también puede enviar datos de otros productores.

```
2023-02-17 09:28:07.763,"AAPL",5032023-02-17 09:28:07.763,
"AMZN",3352023-02-17 09:28:07.763,
"GOOGL",1852023-02-17 09:28:07.763,
"AAPL",11162023-02-17 09:28:07.763,
"GOOGL",1582
```

El siguiente JSON se utiliza para generar una serie aleatoria de fecha y hora de negociación, cotización bursátil y precio bursátil:

```
date.now(YYYY-MM-DD HH:mm:ss.SSS),
"random.arrayElement(["AAPL","AMZN","MSFT","META","GOOGL"])",
```

```
random.number(2000)
```

Una vez que seleccione Enviar datos, el generador empezará a enviar datos simulados.

Los sistemas externos transmiten los datos a Amazon Kinesis Data Firehose. Con aplicaciones de Amazon Kinesis Data Analytics para SQL, puede analizar datos de flujo utilizando SQL estándar. El servicio le permite crear y ejecutar código SQL en orígenes de streaming para realizar análisis de series temporales, alimentar paneles en tiempo real y crear métricas en tiempo real. Las aplicaciones de Amazon Kinesis Data Analytics para SQL podrían crear un flujo de destino a partir de consultas SQL en el flujo de entrada y enviar el flujo de destino a otra Amazon Kinesis Data Firehose. El Amazon Kinesis Data Firehose de destino podría enviar los datos analíticos a Amazon S3 como estado final.

El código heredado de Amazon Kinesis Data Analytics-SQL se basa en una extensión de SQL Standard.

Se utiliza la siguiente consulta en Amazon Kinesis Data Analytics-SQL. Primero debe crear un flujo de destino para el resultado de la consulta. A continuación, usaría `PUMP`, que es un objeto de repositorio de Amazon Kinesis Data Analytics (una extensión del estándar de SQL) que ofrece una funcionalidad de consulta `INSERT INTO stream SELECT ... FROM` en constante ejecución, que permite ingresar los resultados de una consulta de manera constante en una secuencia determinada.

```
CREATE
OR REPLACE STREAM "DESTINATION_SQL_STREAM" (EVENT_TIME TIMESTAMP,
INGEST_TIME TIMESTAMP,
TICKER VARCHAR(16),
VOLUME BIGINT,
AVG_PRICE DOUBLE,
MIN_PRICE DOUBLE,
MAX_PRICE DOUBLE);

CREATE
OR REPLACE PUMP "STREAM_PUMP" AS
INSERT INTO
   "DESTINATION_SQL_STREAM"
   SELECT
      STREAM STEP("SOURCE_SQL_STREAM_001"."tradeTimestamp" BY INTERVAL '60' SECOND) AS
 EVENT_TIME,
      STEP("SOURCE_SQL_STREAM_001".ROWTIME BY INTERVAL '60' SECOND) AS
 "STREAM_INGEST_TIME",
```

```
    "ticker",
     COUNT(*) AS VOLUME,
    AVG("tradePrice") AS AVG_PRICE,
    MIN("tradePrice") AS MIN_PRICE,
    MAX("tradePrice") AS MAX_PRICEFROM "SOURCE_SQL_STREAM_001"
GROUP BY
    "ticker",
    STEP("SOURCE_SQL_STREAM_001".ROWTIME BY INTERVAL '60' SECOND),
    STEP("SOURCE_SQL_STREAM_001"."tradeTimestamp" BY INTERVAL '60' SECOND);
```

El SQL anterior usa dos ventanas de tiempo, `tradeTimestamp` que proviene de la carga útil del flujo entrante y `ROWTIME.tradeTimestamp` también denominado `Event Time` o `client-side time`. Suele ser conveniente utilizar estos momentos en análisis, ya que es el momento en el que se produjo un evento. No obstante, muchas fuentes de eventos como, por ejemplo, clientes de teléfonos móviles y web, no tienen relojes de confianza, lo que puede provocar tiempos inexactos. Además, los problemas de conectividad pueden hacer que los registros aparezcan en la secuencia y no lo en el mismo orden los eventos.

Las secuencias en la aplicación incluyen una columna especial llamada `ROWTIME`. Almacena una marca temporal cuando Amazon Kinesis Data Analytics inserta una fila en la primera secuencia en la aplicación. `ROWTIME` refleja la marca temporal en la que Amazon Kinesis Data Analytics insertó un registro en la primera secuencia en la aplicación después de leer desde el origen de streaming. Este valor `ROWTIME` se mantiene en toda su aplicación.

SQL determina el número de ticker como `volume`, `min`, `max` y `average` lo valora en un intervalo de 60 segundos.

Utilizar cada uno de estos tiempos en las consultas en ventana basadas en el tiempo tiene ventajas y desventajas. Le recomendamos que elija uno o varios de estos tiempos, y una estrategia para abordar las posibles desventajas en función de su caso de uso.

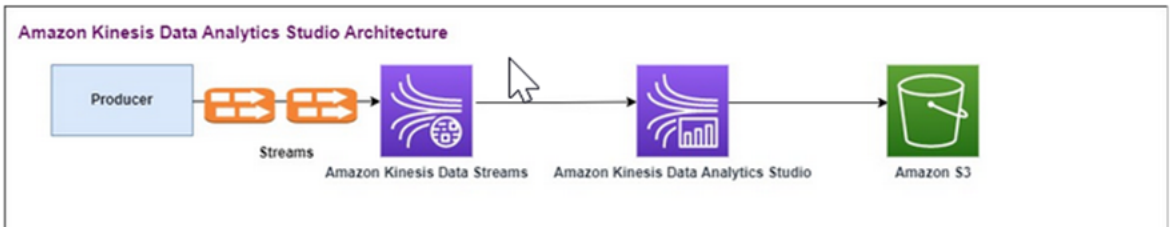
Recomendamos una estrategia de dos ventanas que utilice dos ventanas basadas en el tiempo: una `ROWTIME` y una para los otros tiempos, como el tiempo de evento.

- Utilice `ROWTIME` como la primera ventana, que controla la frecuencia con la que la consulta emite los resultados, tal y como se muestra en el siguiente ejemplo. No se utiliza como tiempo lógico.
- Utilice uno de los otros tiempos que es el tiempo lógico que desea asociar a su análisis. Este tiempo representa cuándo se produjo el evento. En el siguiente ejemplo, el objetivo de análisis es agrupar los registros y devolver un recuento por cada símbolo.

Amazon Managed Service para Apache Flink Studio

En la arquitectura actualizada, se sustituye Amazon Kinesis Data Firehose por Amazon Kinesis Data Streams. Las aplicaciones de Amazon Kinesis Data Analytics para SQL se sustituyen por Amazon Managed Service para Apache Flink Studio. El código de Apache Flink se ejecuta de forma interactiva en un cuaderno Apache Zeppelin. Amazon Managed Service para Apache Flink Studio envía los datos de comercio agregado a un bucket de Amazon S3 para su almacenamiento. Los pasos se muestran a continuación:

Este es el flujo de arquitectura de Amazon Managed Service para Apache Flink:



Cree de un flujo de datos de Kinesis

Para crear un flujo de datos con la consola

1. Inicie sesión en la consola de Kinesis Consola de administración de AWS y ábrala en https://console.aws.amazon.com /kinesis.
2. En la barra de navegación, expanda el selector de regiones y seleccione una región.
3. Elija Create data stream (Crear flujo de datos).
4. En la página Crear flujo de Kinesis, escriba un nombre para su flujo de datos y, a continuación, elija el modo de capacidad Bajo demanda predeterminado.

   Con el modo Bajo demanda, puede seleccionar Crear flujo de Kinesis para crear su flujo de datos.

   En la página Flujos de Kinesis, el valor Estado del flujo es Creándose mientras se crea. Cuando el flujo está listo para usarse, el valor Estado cambia a Activo.
5. Elija el nombre del flujo. La página Detalles del flujo muestra un resumen de la configuración del flujo, junto con información de monitoreo.
6. En el generador de datos de Amazon Kinesis, cambie la Stream/delivery transmisión por la nueva Amazon Kinesis Data Streams: TRADE_SOURCE_STREAM.

El JSON y la carga útil serán los mismos que los que utilizó para Amazon Kinesis Data Analytics-SQL. Utilice el generador de datos de Amazon Kinesis para generar algunos ejemplos de datos de carga útil de negociación y diríjase al flujo de datos TRADE_SOURCE_STREAM para este ejercicio:

```
{{date.now(YYYY-MM-DD HH:mm:ss.SSS)}},
"{{random.arrayElement(["AAPL","AMZN","MSFT","META","GOOGL"])}}",
{{random.number(2000)}}
```

7. Consola de administración de AWS Vaya a Managed Service for Apache Flink y, a continuación, seleccione Crear aplicación.
8. En el panel de navegación izquierdo, elija Bloc de notas de Studio y, a continuación, seleccione Crear bloc de notas de Studio.
9. Escriba el nombre del bloc de notas de Studio.
10. En AWS Glue database, proporcione una base de datos AWS Glue existente que defina los metadatos de sus fuentes y destinos. Si no tiene una AWS Glue base de datos, elija Crear y haga lo siguiente:

    a. En la consola AWS Glue, selecciona Bases de datos en Catálogo de datos en el menú de la izquierda.

    b. Elija Crear base de datos.

    c. En la página Crear base de datos, ingrese el nombre de la base de datos. En la sección Ubicación - opcional, elija Examinar Amazon S3 y seleccione el bucket de Amazon S3. Si aún no tiene configurado un bucket de Amazon S3, puede omitir este paso y volver a él más tarde.

    d. (Opcional). Ingrese la descripción de la base de datos.

    e. Elija Creación de base de datos.
11. Elija Crear bloc de notas.
12. Una vez creado el bloc de notas, seleccione Ejecutar.
13. Una vez que el cuaderno se haya iniciado correctamente, abra un cuaderno Zeppelin seleccionando Abrir en Apache Zeppelin.
14. En la página del bloc de notas de Zeppelin, selecciona Crear nueva nota y asígnale un nombre. MarketDataFeed

El código SQL de Flink se explica a continuación, pero primero así es como se ve la pantalla de un bloc de notas Zeppelin. Cada ventana del bloc de notas es un bloque de códigos independiente y se pueden ejecutar de una en una.

Código de Amazon Managed Service para Apache Flink Studio

Amazon Managed Service para Apache Flink utiliza Zeppelin Notebooks para ejecutar el código. En este ejemplo, la asignación se realiza a código ssql basado en Apache Flink 1.13. El código del cuaderno Zeppelin se muestra debajo de un bloque a la vez.

Antes de ejecutar cualquier código en su bloc de notas Zeppelin, debe ejecutar los comandos de configuración de Flink. Si necesita cambiar algún ajuste de configuración después de ejecutar el código (ssql, Python o Scala), tendrá que detener y reiniciar el cuaderno. En este ejemplo, tendrá que establecer puntos de control. Se requieren puntos de control para poder transmitir datos a un archivo en Amazon S3. Esto permite que los datos que se transmiten a Amazon S3 se vacíen en un archivo. La siguiente afirmación establece el intervalo en 5000 milisegundos.

```
%flink.conf
execution.checkpointing.interval 5000
```

`%flink.conf` indica que este bloque son declaraciones de configuración. Para obtener más información sobre la configuración de Flink, incluidos los puntos de control, consulte Puntos de control de Apache Flink.

La tabla de entrada para la fuente Amazon Kinesis Data Streams se crea con el código ssql de Flink que aparece a continuación. Tenga en cuenta que el `TRADE_TIME` campo almacena lo date/time creado por el generador de datos.

```
%flink.ssql

DROP TABLE IF EXISTS TRADE_SOURCE_STREAM;
CREATE TABLE TRADE_SOURCE_STREAM (--`arrival_time` TIMESTAMP(3) METADATA FROM
 'timestamp' VIRTUAL,
TRADE_TIME TIMESTAMP(3),
WATERMARK FOR TRADE_TIME as TRADE_TIME - INTERVAL '5' SECOND,TICKER STRING,PRICE
 DOUBLE,
STATUS STRING)WITH ('connector' = 'kinesis','stream' = 'TRADE_SOURCE_STREAM',
'aws.region' = 'us-east-1','scan.stream.initpos' = 'LATEST','format' = 'csv');
```

Puede ver el flujo de entrada con esta declaración:

```
%flink.ssql(type=update)-- testing the source stream

select * from TRADE_SOURCE_STREAM;
```

Antes de enviar los datos agregados a Amazon S3, puede verlos directamente en Amazon Managed Service para Apache Flink Studio con una consulta de selección en una ventana desplegable. Esto agrega los datos de negociación en intervalos de tiempo de un minuto. Tenga en cuenta que la sentencia %flink.ssql debe tener una designación (type=update):

```
%flink.ssql(type=update)

select TUMBLE_ROWTIME(TRADE_TIME,
INTERVAL '1' MINUTE) as TRADE_WINDOW,
TICKER, COUNT(*) as VOLUME,
AVG(PRICE) as AVG_PRICE,
MIN(PRICE) as MIN_PRICE,
MAX(PRICE) as MAX_PRICE FROM TRADE_SOURCE_STREAMGROUP BY TUMBLE(TRADE_TIME, INTERVAL
 '1' MINUTE), TICKER;
```

A continuación, podrá crear una tabla para el destino en Amazon S3. Tiene que utilizar una marca de agua. Una marca de agua es una métrica de progreso que indica un momento en el que está seguro de que no se producirán más eventos retrasados. El motivo de la marca de agua es tener en cuenta las llegadas tardías. El intervalo `‘5’ Second` permite que las operaciones entren en Amazon Kinesis Data Streams con 5 segundos de retraso y que se sigan incluyendo si tienen una marca de tiempo dentro de la ventana. Para obtener más información, consulte Generating Watermarks.

```
%flink.ssql(type=update)

DROP TABLE IF EXISTS TRADE_DESTINATION_S3;
CREATE TABLE TRADE_DESTINATION_S3 (
TRADE_WINDOW_START TIMESTAMP(3),
WATERMARK FOR TRADE_WINDOW_START as TRADE_WINDOW_START - INTERVAL '5' SECOND,
TICKER STRING,
VOLUME BIGINT,
AVG_PRICE DOUBLE,
MIN_PRICE DOUBLE,
MAX_PRICE DOUBLE)
WITH ('connector' = 'filesystem','path' = 's3://trade-destination/','format' = 'csv');
```

Esta declaración inserta los datos en `TRADE_DESTINATION_S3`. `TUMPLE_ROWTIME` es la marca de tiempo del límite superior inclusivo de la ventana de saltos.

```
%flink.ssql(type=update)

insert into TRADE_DESTINATION_S3
select TUMBLE_ROWTIME(TRADE_TIME,
INTERVAL '1' MINUTE),
TICKER, COUNT(*) as VOLUME,
AVG(PRICE) as AVG_PRICE,
MIN(PRICE) as MIN_PRICE,
MAX(PRICE) as MAX_PRICE FROM TRADE_SOURCE_STREAM
GROUP BY TUMBLE(TRADE_TIME, INTERVAL '1' MINUTE), TICKER;
```

Deje que su estado de cuenta se ejecute durante 10 a 20 minutos para acumular algunos datos en Amazon S3. A continuación, aborte su instrucción.

Esto cierra el archivo en Amazon S3 para que se pueda ver.

Este es el aspecto del contenido:

Puede usar la plantilla de CloudFormation para crear la infraestructura.

CloudFormation creará los siguientes recursos en su AWS cuenta:

- Amazon Kinesis Data Streams
- Amazon Managed Service para Apache Flink Studio
- AWS Glue base de datos
- Bucket de Amazon S3
- Roles y políticas de IAM para que Amazon Managed Service para Apache Flink Studio acceda a los recursos adecuados

Importe el bloc de notas y cambie el nombre del bucket de Amazon S3 por el nuevo bucket de Amazon S3 creado por CloudFormation.

```
%flink.ssql(type=update)

DROP TABLE IF EXISTS  TRADE_DESTINATION_S3;
CREATE TABLE TRADE_DESTINATION_S3 (
TRADE_WINDOW_START TIMESTAMP(3),
WATERMARK FOR TRADE_WINDOW_START as TRADE_WINDOW_START - INTERVAL '5' SECOND,
TICKER STRING,
VOLUME BIGINT,
AVG_PRICE DOUBLE,
MIN_PRICE DOUBLE,
MAX_PRICE DOUBLE)
WITH ('connector' = 'filesystem','path' = 's3://kda-studio-test-stack-markettradinganalyticsc[redacted]/','format' = 'csv');
```

Ver más

Estos son algunos recursos adicionales que puede utilizar para obtener más información sobre el uso de Managed Service para Apache Flink Studio:

- Guía para desarrolladores de Managed Service para Apache Flink Studio Notebooks
- Documentación de Apache Flink 1.13
- Taller de Managed Service for Apache Flink Studio
- Creación de ventanas de Apache Flink
- Guía para desarrolladores de Amazon Kinesis Data Analytics: escritura desde un flujo de Kinesis Data Analytics a un bucket de S3

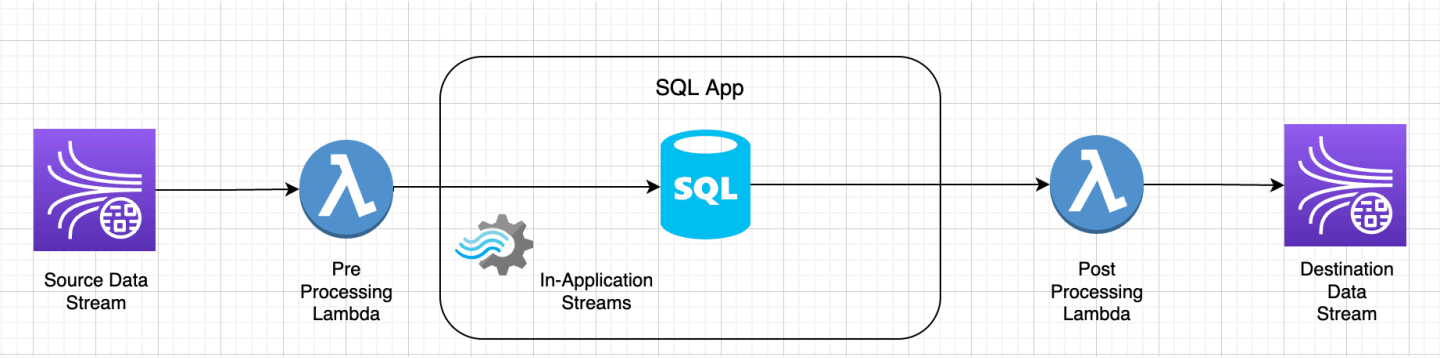
Aprovechar las funciones definidas por el usuario () UDFs

El propósito del patrón es demostrar cómo aprovechar las libretas Zeppelin de Kinesis Data Analytics-Studio para procesar datos UDFs en la transmisión de Kinesis. Managed Service para Apache Flink Studio utiliza Apache Flink para proporcionar capacidades analíticas avanzadas, que incluyen semántica de procesamiento de una sola vez, ventanas temporales de eventos,

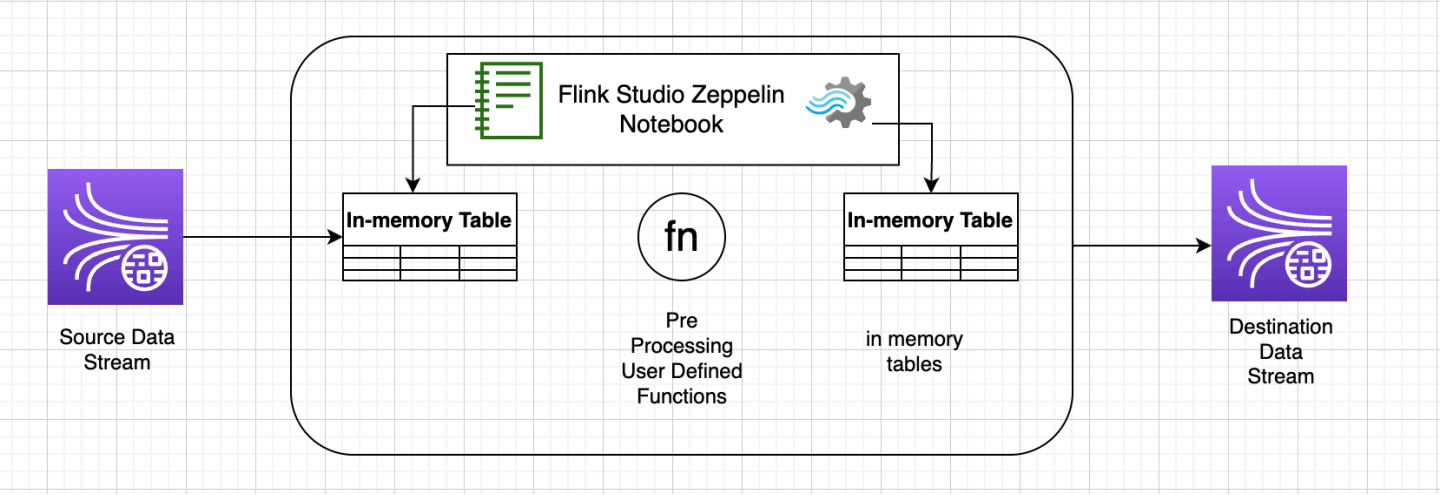
extensibilidad mediante funciones definidas por el usuario e integraciones de clientes, compatibilidad con lenguajes imperativos, estado de aplicación duradero, escalado horizontal, soporte para múltiples orígenes de datos, integraciones extensibles y más. Son fundamentales para garantizar la precisión, la integridad, la coherencia y la fiabilidad del procesamiento de los flujos de datos y no están disponibles con Amazon Kinesis Data Analytics para SQL.

En este ejemplo de aplicación, demostraremos cómo aprovechar UDFs el cuaderno Zeppelin de KDA-Studio para procesar datos en la transmisión de Kinesis. Los blocs de notas de Studio para Kinesis Data Analytics le permiten consultar flujos de datos de forma interactiva en tiempo real y crear y ejecutar fácilmente aplicaciones de procesamiento de flujos mediante SQL, Python y Scala estándares. Con unos pocos clics Consola de administración de AWS, puede abrir un bloc de notas sin servidor para consultar flujos de datos y obtener resultados en cuestión de segundos. Para obtener más información, consulte Uso de un bloc de notas de Studio con Kinesis Data Analytics para Apache Flink.

Funciones Lambda utilizadas para el pre/post procesamiento de datos en aplicaciones KDA-SQL:



Funciones definidas por el usuario para el pre/post procesamiento de datos con los cuadernos Zeppelin de KDA-Studio



Funciones UDFs definidas por el usuario ()

Para reutilizar la lógica empresarial habitual en un operador, puede resultar útil hacer referencia a una función definida por el usuario para transformar el flujo de datos. Esto se puede hacer desde el bloc de notas Managed Service para Apache Flink Studio o como un archivo jar de aplicación con referencia externa. El uso de funciones definidas por el usuario puede simplificar las transformaciones o los enriquecimientos de datos que se podrían realizar a través del flujo de datos.

En su bloc de notas, hará referencia a un sencillo contenedor de aplicaciones Java que tiene la funcionalidad de anonimizar números de teléfono personales. También puedes escribir Python o Scala UDFs para usarlos en el cuaderno. Elegimos una aplicación Java jar para resaltar la funcionalidad de importar una aplicación jar a un bloc de notas de Pyflink.

Configuración del entorno

Para seguir esta guía e interactuar con sus datos de flujo, utilizará un script AWS CloudFormation para lanzar los siguientes recursos:

- Flujo de datos de origen y destino de Kinesis
- Base de datos Glue
- rol de IAM
- Aplicación Managed Service para Apache Flink Studio
- Función de Lambda para iniciar la aplicación Managed Service para Apache Flink Studio
- Rol de Lambda para ejecutar la anterior función de Lambda
- Recurso personalizado para invocar la función de Lambda

Descarga la CloudFormation plantilla aquí.

Crea la CloudFormation pila

1. Ve a Consola de administración de AWS y elige CloudFormationen la lista de servicios.
2. En la CloudFormationpágina, selecciona Pilas y, a continuación, selecciona Crear pila con nuevos recursos (estándar).
3. En la página Crear pila, elija Cargar un archivo de plantilla y, a continuación, elija el `kda-flink-udf.yml` que haya descargado anteriormente. Elija el archivo y después elija Siguiente.

4. Asigne un nombre a la plantilla, como `kinesis-UDF` de modo que sea fácil de recordar, y actualice los parámetros de entrada, como flujo de entrada, si desea un nombre diferente. Elija Siguiente.
5. En la página Configurar opciones de pila, añada Etiquetas si lo desea y, a continuación, seleccione Siguiente.
6. En la página de revisión, marque las casillas que permiten la creación de recursos de IAM y, a continuación, seleccione Enviar.

El lanzamiento de la CloudFormation pila puede tardar entre 10 y 15 minutos, en función de la región en la que lo hagas. Cuando vea el estado `CREATE_COMPLETE` de toda la pila, estará listo para continuar.

Uso del bloc de notas de Managed Service para Apache Flink Studio

Los blocs de notas de Studio para Kinesis Data Analytics le permiten consultar flujos de datos de forma interactiva en tiempo real y crear y ejecutar fácilmente aplicaciones de procesamiento de flujos mediante SQL, Python y Scala estándar. Con unos pocos clics Consola de administración de AWS, puedes abrir una libreta sin servidor para consultar flujos de datos y obtener resultados en cuestión de segundos.

Un bloc de notas es un entorno de desarrollo basado en la web. Con los blocs de notas, obtiene una experiencia de desarrollo interactiva sencilla combinada con las capacidades avanzadas de procesamiento de flujos de datos que proporciona Apache Flink. Los cuadernos de Studio utilizan la tecnología Apache Zeppelin y utilizan Apache Flink como motor de procesamiento de flujos. Los blocs de notas de Studio combinan estas tecnologías a la perfección para que los desarrolladores con todas las habilidades puedan acceder a los análisis avanzados de los flujos de datos.

Apache Zeppelin proporciona a sus blocs de notas de Studio un conjunto completo de herramientas de análisis, entre las que se incluyen las siguientes:

- Visualización de datos
- Exportación de datos a un archivo CSV
- Control del formato de salida para facilitar el análisis

Uso del bloc de notas

1. Vaya a Amazon Kinesis Consola de administración de AWS y elija Amazon Kinesis en la lista de servicios.
2. En la página de navegación de la izquierda, elija Aplicaciones de análisis y, a continuación, elija blocs de notas de Studio.
3. Compruebe que el KinesisDataAnalyticsStudioportátil esté funcionando.
4. Elija el bloc de notas y, a continuación, elija Abrir en Apache Zeppelin.
5. Descargue el archivo de Data Producer Zeppelin Notebook que utilizará para leer y cargar datos en Kinesis Stream.
6. Importe el bloc de notas Zeppelin `Data Producer`. Asegúrese de modificar la entrada `STREAM_NAME` y `REGION` en el código del bloc de notas. El nombre del flujo de entrada se encuentra en la salida de la pila CloudFormation.
7. Ejecute el bloc de notas Data Producer pulsando el botón Ejecutar este párrafo para insertar datos de muestra en la entrada de Kinesis Data Stream.
8. Mientras se cargan los datos de muestra, descargue MaskPhoneNumber-Interactive Notebook, que leerá los datos de entrada, anonimizará los números de teléfono del flujo de entrada y almacenará los datos anónimos en el flujo de salida.
9. Importe el bloc de notas Zeppelin `MaskPhoneNumber-interactive`.
10. Ejecute cada párrafo del bloc de notas.

    a. En el párrafo 1, se importa una función definida por el usuario para anonimizar los números de teléfono.

    ```
    %flink(parallelism=1)
    import com.mycompany.app.MaskPhoneNumber
    stenv.registerFunction("MaskPhoneNumber", new MaskPhoneNumber())
    ```

    b. En el siguiente párrafo, creará una tabla en memoria para leer los datos del flujo de entrada. Asegúrese de que el nombre de la transmisión y la región sean correctos. AWS

    ```
    %flink.ssql(type=update)

    DROP TABLE IF EXISTS customer_reviews;

    CREATE TABLE customer_reviews (
    customer_id VARCHAR,
    ```

```
product VARCHAR,
review VARCHAR,
phone VARCHAR
)
WITH (
'connector' = 'kinesis',
'stream' = 'KinesisUDFSampleInputStream',
'aws.region' = 'us-east-1',
'scan.stream.initpos' = 'LATEST',
'format' = 'json');
```

c. Compruebe si los datos están cargados en la tabla en memoria.

```
%flink.ssql(type=update)
select * from customer_reviews
```

d. Invoque la función definida por el usuario para anonimizar el número de teléfono.

```
%flink.ssql(type=update)
select customer_id, product, review, MaskPhoneNumber('mask_phone', phone) as
 phoneNumber from customer_reviews
```

e. Ahora que los números de teléfono están enmascarados, cree una vista con un número enmascarado.

```
%flink.ssql(type=update)

DROP VIEW IF EXISTS sentiments_view;

CREATE VIEW
    sentiments_view
AS
  select customer_id, product, review, MaskPhoneNumber('mask_phone', phone) as
 phoneNumber from customer_reviews
```

f. Compruebe los datos.

```
%flink.ssql(type=update)
select * from sentiments_view
```

g. Cree una tabla en memoria para la salida de Kinesis Stream. Asegúrese de que el nombre de la transmisión y AWS la región sean correctos.

```
%flink.ssql(type=update)

DROP TABLE IF EXISTS customer_reviews_stream_table;

CREATE TABLE customer_reviews_stream_table (
customer_id VARCHAR,
product VARCHAR,
review VARCHAR,
phoneNumber varchar
)
WITH (
'connector' = 'kinesis',
'stream' = 'KinesisUDFSampleOutputStream',
'aws.region' = 'us-east-1',
'scan.stream.initpos' = 'TRIM_HORIZON',
'format' = 'json');
```

h. Inserte registros actualizados en el flujo de Kinesis de destino.

```
%flink.ssql(type=update)
INSERT INTO customer_reviews_stream_table
SELECT customer_id, product, review, phoneNumber
FROM sentiments_view
```

i. Vea y verifique los datos del flujo de Kinesis de destino.

```
%flink.ssql(type=update)
select * from customer_reviews_stream_table
```

Promoción de un bloc de notas como aplicación

Ahora que ha probado el código de su bloc de notas de forma interactiva, implementará el código como una aplicación de flujo con un estado duradero. Primero tendrá que modificar la configuración de la aplicación para especificar una ubicación para su código en Amazon S3.

1. En Consola de administración de AWS, elija su bloc de notas y, en Implementar como configuración de aplicación (opcional), elija Editar.
2. En Destino del código en Amazon S3, elija el bucket de Amazon S3 que crearon los CloudFormation scripts. El proceso puede demorar unos minutos.

3. No podrá promocionar la nota tal como está. Si lo intenta, se producirá un error ya que no se admiten las instrucciones `Select`. Para evitar este problema, descargue el cuaderno MaskPhoneNumber-Streaming Zeppelin.
4. Importe el bloc de notas Zeppelin `MaskPhoneNumber-streaming`.
5. Abre la nota y selecciona Acciones para. KinesisDataAnalyticsStudio
6. Elija Build MaskPhoneNumber -Streaming y exporte a S3. Asegúrese de cambiar el nombre de la aplicación y de no incluir caracteres especiales.
7. Seleccione Crear y exportar. La configuración de la aplicación de flujo tardará unos minutos.
8. Una vez que se complete la compilación, elija Implementar mediante la consola de AWS .
9. En la página siguiente, revise la configuración y asegúrese de elegir el rol de IAM correcto. A continuación, seleccione Crear aplicación de streaming.
10. Después de unos minutos, verá un mensaje que indica que la aplicación de flujo se creó correctamente.

Para obtener más información sobre la implementación de aplicaciones con un estado y límites duraderos, consulte Implementación como una aplicación con un estado duradero.

Limpieza

Si lo desea, ahora puede desinstalar la pila CloudFormation. Esto eliminará todos los servicios que configuró anteriormente.

¿Qué son las aplicaciones de Amazon Kinesis Data Analytics para SQL?

Con las aplicaciones de Amazon Kinesis Data Analytics para SQL, puede utilizar SQL estándar para procesar y analizar datos de streaming. El servicio le permite crear rápidamente y ejecutar un potente código SQL en orígenes de streaming para realizar análisis de serie temporal, alimentar paneles en tiempo real y crear métricas en tiempo real.

Para comenzar a utilizar Kinesis Data Analytics, debe crear una aplicación de Kinesis Data Analytics que lea y procese constantemente datos de streaming. El servicio admite la ingestión de datos de orígenes de flujos de Amazon Kinesis Data Streams y Amazon Data Firehose. A continuación, cree código SQL con el editor interactivo y pruébelo con datos de streaming en vivo. También puede configurar los destinos a los que desea que Kinesis Data Analytics envíe los resultados.

Kinesis Data Analytics admite Amazon Data Firehose (Amazon S3, Amazon Redshift, Amazon Service y Splunk) y OpenSearch Amazon AWS Lambda Kinesis Data Streams como destinos.

¿Cuándo debo usar Amazon Kinesis Data Analytics?

Amazon Kinesis Data Analytics permite crear rápidamente un código SQL que lee, procesa y almacena datos de manera continua en casi tiempo real. Puede utilizar consultas SQL estándar con los datos de streaming para crear aplicaciones que transformen y proporcionen información valiosa a partir de los datos. A continuación se indican algunos escenarios de ejemplo para utilizar Kinesis Data Analytics::

- Generar análisis de series de tiempo: puede calcular métricas en ventanas de tiempo y, a continuación, transmitir los valores a Amazon S3 o Amazon Redshift a través de una secuencia de entrega de datos de Kinesis.
- Alimentar paneles en tiempo real: puede enviar los resultados de datos de streaming agregados y procesados descendentes para alimentar paneles en tiempo real.
- Crear métricas en tiempo real: puede crear métricas personalizadas y disparadores para usarlos en la monitorización, las notificaciones y las alarmas en tiempo real.

Para obtener información sobre los elementos del lenguaje SQL compatibles con Kinesis Data Analytics, consulte Referencia de SQL de Amazon Kinesis Data Analytics.

¿Es la primera vez que utiliza Amazon Kinesis Data Analytics?

Si es la primera vez que utiliza Amazon Kinesis Data Analytics, le recomendamos que lea las siguientes secciones en orden:

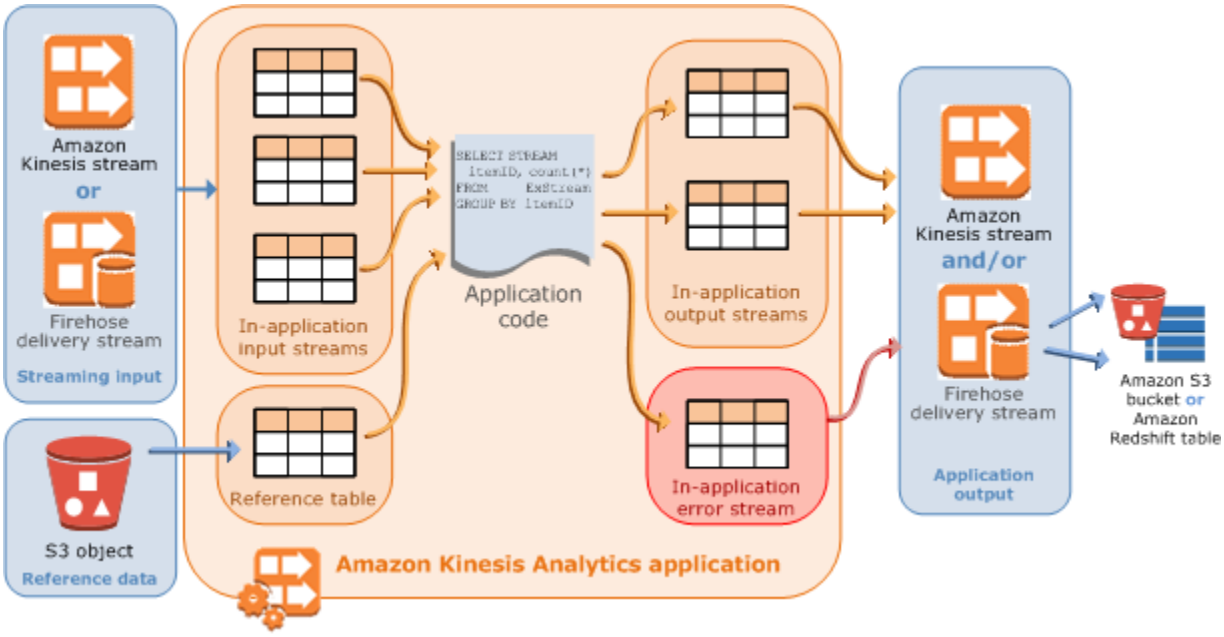
1. Lea la sección “Cómo funciona” de esta guía. En esta sección, se presentan varios componentes de Kinesis Data Analytics con los que puede trabajar para crear end-to-end una experiencia. Para obtener más información, consulte Aplicaciones de Amazon Kinesis Data Analytics para SQL: cómo funciona.
2. Haga los ejercicios de la introducción. Para obtener más información, consulte Introducción a aplicaciones de Amazon Kinesis Data Analytics para SQL.
3. Explore los conceptos de SQL de streaming. Para obtener más información, consulte Conceptos de SQL de Streaming.
4. Consulte los ejemplos adicionales. Para obtener más información, consulte Ejemplos de Kinesis Data Analytics para SQL.

Aplicaciones de Amazon Kinesis Data Analytics para SQL: cómo funciona

> **Note**
>
> Después del 12 de septiembre de 2023, no podrá crear nuevas aplicaciones con Kinesis Data Firehose como origen si aún no utiliza Kinesis Data Analytics para SQL. Para obtener más información, consulte Límites.

Una aplicación es el recurso principal en Amazon Kinesis Data Analytics que puede crear en su cuenta. Puede crear y administrar aplicaciones mediante la API de Kinesis Data Analytics Consola de administración de AWS o la API. Kinesis Data Analytics proporciona operaciones de la API para la administración de aplicaciones. Para ver la lista de operaciones de la API, consulte Acciones.

Las aplicaciones de análisis de datos de Kinesis Data Analytics leen y procesan datos de streaming en tiempo real de forma constante. El código de la aplicación se escribe mediante SQL para procesar los datos de streaming entrantes y generar los resultados. A continuación, Kinesis Data Analytics escribe la salida en el destino que se haya configurado. El siguiente diagrama ilustra la arquitectura de una aplicación típica.



Cada aplicación tiene un nombre, descripción, ID de versión y estado. Amazon Kinesis Data Analytics asigna un ID de versión la primera vez que crea una aplicación. Se actualiza este ID

de versión cuando actualiza la configuración de cualquier aplicación. Por ejemplo, si añade una configuración de entrada, añade o elimina un origen de datos de referencia, añade o elimina la configuración de salida o actualiza el código de la aplicación, Kinesis Data Analytics actualiza el ID de versión de la aplicación actual. Kinesis Data Analytics también mantiene marcas temporales de cuando se creó una aplicación y de su última actualización.

Además de estas propiedades básicas, cada aplicación se compone de lo siguiente:

- Entrada: es el origen de streaming de la aplicación. Puede seleccionar un flujo de datos de Kinesis o un flujo de entrega de datos de Firehose como origen de flujo. En la configuración de entrada, debe asignar el origen de streaming a una secuencia de entrada en la aplicación. La secuencia en la aplicación se asemeja a una tabla que se actualiza constantemente, en la que puede realizar operaciones `SELECT` e `INSERT SQL`. En su código de aplicación, puede crear flujos en la aplicación adicionales para almacenar los resultados intermedios de la consulta.

  Si lo desea, puede dividir un único origen de streaming en varias secuencias de entrada en la aplicación para mejorar el rendimiento. Para obtener más información, consulte Límites y Configuración de entrada de la aplicación.

  Amazon Kinesis Data Analytics proporciona una columna de marca temporal denominada Marcas temporales y la comuna ROWTIME para cada secuencia en la aplicación. Puede utilizar esta columna en las consultas en ventana basadas en el tiempo. Para obtener más información, consulte Consultas en ventana.

  Si lo desea, puede configurar un origen de datos de referencia para enriquecer el flujo de datos de entrada en la aplicación. Se produce una tabla de referencia en la aplicación. Debe almacenar los datos de referencia como un objeto en el bucket S3. Cuando se inicia la aplicación, Amazon Kinesis Data Analytics lee el objeto de Amazon S3 y crea una tabla en la aplicación. Para obtener más información, consulte Configuración de entrada de la aplicación.

- Código de la aplicación: una serie de instrucciones SQL que procesan entradas y producen salidas. Puede escribir instrucciones de SQL en a secuencias en la aplicación y con tablas de referencia. También puede escribir consultas JOIN para combinar datos de ambos orígenes.

  Para obtener información sobre los elementos del lenguaje SQL compatibles con Kinesis Data Analytics, consulte Referencia de SQL de Amazon Kinesis Data Analytics.

  En su forma más sencilla, el código de la aplicación puede ser una única instrucción SQL que selecciona de una entrada de streaming e introduce los resultados en una salida de streaming. También puede ser una serie de instrucciones SQL en las que la salida de una alimenta la salida de la siguiente instrucción SQL. Además, puede escribir el código de aplicación para dividir una secuencia de entrada en varias secuencias. A continuación, puede aplicar consultas adicionales para procesar estas secuencias. Para obtener más información, consulte Código de la aplicación.

- Salida: en el código de la aplicación, los resultados de las consultas van a secuencias en la aplicación. En el código de la aplicación, puede crear una o más secuencias en la aplicación adicionales para conservar resultados intermedios. Además, como opción, puede configurar la salida de la aplicación para que se envíen datos de secuencias en la aplicación, que mantengan la salida de la aplicación (se conocen también como reproducciones de salida en la aplicación), para destinos externos. Los destinos externos pueden ser un flujo de entrega de Firehose o un flujo de datos de Kinesis. Tenga en cuenta lo siguiente en relación con estos destinos:

  - Puede configurar una transmisión de entrega de Firehose para escribir los resultados en Amazon S3, Amazon Redshift o OpenSearch Amazon Service (ServiceOpenSearch).

  - También puede escribir la salida de la aplicación en un destino personalizado, en lugar de Amazon S3 o Amazon Redshift. Para ello, debe especificar un flujo de datos de Kinesis como destino en la configuración de salida. A continuación, se configura AWS Lambda para sondear la transmisión e invocar la función Lambda. El código de la función de Lambda recibe datos de secuencia como entrada. En el código de la función de Lambda puede escribir los datos entrantes en su destino personalizado. Para obtener más información, consulte Uso AWS Lambda con Amazon Kinesis Data Analytics.

Para obtener más información, consulte Configuración de salida de la aplicación.

Además, tenga en cuenta lo siguiente:

- Amazon Kinesis Data Analytics necesita permisos para leer los registros de un origen de streaming y escribir la salida de la aplicación en los destinos externos. Use los roles de IAM para otorgar estos permisos.

- Kinesis Data Analytics proporciona automáticamente una secuencia de errores en la aplicación para cada aplicación. Si la aplicación tiene problemas al procesar determinados registros (por ejemplo, por un retraso o porque no coincide el tipo), ese registro se escribe en la secuencia de errores. Puede configurar la salida de la aplicación para indicar a Kinesis Data Analytics que conserve los datos de la secuencia de errores en un destino externo para su posterior evaluación. Para obtener más información, consulte Gestión de errores.

- Amazon Kinesis Data Analytics garantiza que se escriben los registros de salida de la aplicación en el destino configurado. Utiliza "al menos un" modelo de entrega y procesamiento, incluso si se produce una interrupción de la aplicación. Para obtener más información, consulte Modelo de entrega para conservar la salida de las aplicaciones en destinos externos.

Temas

Configuración de entrada de la aplicación

La aplicación de Amazon Kinesis Data Analytics puede recibir entradas desde un único origen de streaming y, opcionalmente, puede utilizar un origen de datos de referencia. Para obtener más

información, consulte Aplicaciones de Amazon Kinesis Data Analytics para SQL: cómo funciona. Las secciones en este tema describen las fuentes de entrada de la aplicación.

Temas

Configuración de origen de streaming

En el momento de crear una aplicación, se especifica una fuente de streaming. También puede modificar una entrada después de crear la aplicación. Amazon Kinesis Data Analytics admite los siguientes orígenes de streaming para la aplicación:

- Un flujo de datos de Kinesis
- Un flujo de entrega de Firehose

> **Note**
>
> Después del 12 de septiembre de 2023, no podrá crear nuevas aplicaciones con Kinesis Data Firehose como origen si aún no utiliza Kinesis Data Analytics para SQL. Los clientes actuales que utilizan aplicaciones de Kinesis Data Analytics para SQL con `KinesisFirehoseInput` pueden seguir añadiendo aplicaciones con `KinesisFirehoseInput` dentro de una cuenta existente mediante Kinesis Data Analytics. Si ya es cliente y desea crear una nueva cuenta con aplicaciones de Kinesis Data Analytics para SQL con `KinesisFirehoseInput`, puede crear un caso mediante el formulario de aumento del límite de servicio. Para obtener más información, consulte el Centro de AWS Support. Recomendamos probar siempre las aplicaciones nuevas antes de pasarlas a producción.

> **Note**
>
> Si el flujo de datos de Kinesis está cifrado, Kinesis Data Analytics obtiene acceso a los datos de la secuencia cifrada sin problemas y sin necesidad de configuración adicional. Kinesis Data Analytics no almacena datos no cifrados leídos de Kinesis Data Streams. Para obtener más información, consulte ¿Qué es el cifrado en el servidor para las secuencias de Kinesis?

Kinesis Data Analytics explora constantemente el origen de streaming para detectar datos nuevos y los ingiere en secuencias en la aplicación según la configuración de entrada.

> **Note**
>
> Puede añadir una secuencia de Kinesis Stream siempre que la entrada de la aplicación no afecte a los datos de la secuencia. Si otro recurso, como un flujo de entrega de Firehose, también accedió al mismo flujo de Kinesis, tanto el flujo de entrega de Firehose como la aplicación de Kinesis Data Analytics recibirán los mismos datos. Sin embargo, el rendimiento y la limitación pueden verse afectados.

El código de la aplicación puede consultar la secuencia en la aplicación. Como parte de la configuración de entrada debe proporcionar lo siguiente:

- Origen de streaming: debe proporcionar el nombre de recurso de Amazon (ARN) de la secuencia y un rol de IAM que Kinesis Data Analytics pueda asumir para obtener acceso a la secuencia en su nombre.
- Prefijo del nombre de secuencia en la aplicación: cuando se inicia la aplicación, Kinesis Data Analytics crea la secuencia en la aplicación especificada. En el código de la aplicación, puede obtener acceso a la secuencia en la aplicación usando este nombre.

  Si lo desea, puede asignar un origen de streaming a varias secuencias en la aplicación. Para obtener más información, consulte Límites. En este caso, Amazon Kinesis Data Analytics crea el número especificado de transmisiones en la aplicación con los siguientes nombres *prefix*_001: *prefix*_002, y. *prefix* _003 De forma predeterminada, Kinesis Data Analytics asigna la fuente de transmisión a una transmisión de la aplicación denominada. *prefix* _001

  Existe un límite en la velocidad que puede insertar filas en una secuencia en la aplicación. Por lo tanto, Kinesis Data Analytics admite varias secuencias en la aplicación para que se puedan

procesar registros en la aplicación a un ritmo más rápido. Si la aplicación no mantiene el ritmo de los datos del origen de streaming, puede añadir unidades de paralelismo para mejorar el desempeño.

- Esquema de mapeo: se debe describir el formato de los registros (JSON, CSV) del origen de streaming. También se debe describir cómo se mapea cada registro de la secuencia a las columnas de la secuencia en la aplicación que se crea. Aquí es donde proporciona los nombres de columnas y tipos de datos.

> **Note**
>
> Kinesis Data Analytics añade comillas alrededor de los identificadores (nombre de la secuencia y de las columnas) al crear la secuencia de entrada en la aplicación. Al consultar esta secuencia y las columnas, debe especificarlas entre comillas utilizando exactamente las mismas letras mayúsculas y minúsculas. Para obtener más información sobre los identificadores, consulte Identificadores en la Referencia de SQL de Amazon Managed Service para Apache Flink.

Puede crear una aplicación y configurar las entradas en la consola de Amazon Kinesis Data Analytics. La consola a continuación realiza las llamadas a la API necesarias. Puede configurar la entrada de la aplicación al crear una nueva aplicación de API o al añadirla configuración de entrada en una aplicación existente. Para obtener más información, consulte CreateApplication y AddApplicationInput. A continuación se presenta la parte de la configuración de entrada `Createapplication` del cuerpo de solicitud de API:

```
"Inputs": [
       {
           "InputSchema": {
               "RecordColumns": [
                   {
                       "Mapping": "string",
                       "Name": "string",
                       "SqlType": "string"
                   }
               ],
               "RecordEncoding": "string",
               "RecordFormat": {
                   "MappingParameters": {
```

```
                        "CSVMappingParameters": {
                            "RecordColumnDelimiter": "string",
                            "RecordRowDelimiter": "string"
                        },
                        "JSONMappingParameters": {
                            "RecordRowPath": "string"
                        }
                    },
                    "RecordFormatType": "string"
                }
            },
            "KinesisFirehoseInput": {
                "ResourceARN": "string",
                "RoleARN": "string"
            },
            "KinesisStreamsInput": {
                "ResourceARN": "string",
                "RoleARN": "string"
            },
            "Name": "string"
        }
    ]
```

Configuración de origen de referencia

Si lo desea, también puede añadir una fuente de datos de referencia a una aplicación existente para enriquecer los datos procedentes de las fuentes de streaming. Los datos de referencia se deben almacenar como objeto en el bucket de Amazon S3. Cuando se inicia la aplicación, Amazon Kinesis Data Analytics lee el objeto de Amazon S3 y crea una tabla de referencia en la aplicación. El código de la aplicación puede después unirla con una secuencia en la aplicación.

Los datos de referencia se almacenan en el objeto de Amazon S3 utilizando formatos compatibles (CSV, JSON). Por ejemplo, suponga que su aplicación realiza análisis de órdenes bursátiles. Suponga los siguientes formatos de registro en el origen de streaming:

```
Ticker, SalePrice, OrderId

AMZN     $700        1003
XYZ      $250        1004
...
```

En este caso, podría considerar mantener un origen de datos de referencia para proporcionar detalles para cada símbolo de cotización, como, por ejemplo, el nombre de la empresa.

```
Ticker, Company
AMZN, Amazon
XYZ, SomeCompany
...
```

Puede añadir un origen de datos de referencia de aplicación con la API o con la consola. Amazon Kinesis Data Analytics ofrece las siguientes acciones de API para administrar los orígenes de datos de referencia:

- AddApplicationReferenceDataSource
- UpdateApplication

Para obtener información sobre cómo añadir datos de referencia mediante la consola, consulte Ejemplo: Agregar datos de referencia a una aplicación de Kinesis Data Analytics.

Tenga en cuenta lo siguiente:

- Si la aplicación está en ejecución, Kinesis Data Analytics crea una tabla de referencia en la aplicación y carga los datos de referencia inmediatamente.
- Si la aplicación no se está ejecutando (por ejemplo, está en estado listo), Kinesis Data Analytics solo guarda la configuración de entrada actualizada. Cuando la aplicación comienza a ejecutarse, Kinesis Data Analytics carga los datos de referencia como una tabla en la aplicación.

Supongamos que desea actualizar los datos después de que Kinesis Data Analytics cree la tabla de referencia en la aplicación. Puede que haya actualizado el objeto de Amazon S3 o que desee utilizar otro objeto de Amazon S3. En este caso, puede llamar explícitamente a UpdateApplication o elegir Acciones, Sincronizar tabla de datos de referencia en la consola. Kinesis Data Analytics no actualiza la tabla de referencia en la aplicación automáticamente.

Existe un límite en el tamaño del objeto de Amazon S3 que se puede crear como origen de datos de referencia. Para obtener más información, consulte Límites. Si el tamaño del objeto supera el límite, Kinesis Data Analytics no puede cargar los datos. El estado de la aplicación aparece como en ejecución, pero no se leen los datos.

Cuando añada un origen de datos de referencia, proporcione la siguiente información:

- Nombre del bucket de S3 y de la clave de objeto: además del nombre del bucket y de la clave de objeto, también debe proporcionar un rol de IAM que Kinesis Data Analytics pueda asumir para leer el objeto en su nombre.
- Nombre de tabla de referencia en la aplicación: crea esta tabla en la aplicación y la rellena al leer el objeto de Amazon S3. El nombre de esta tabla se especifica en el código de la aplicación.
- Esquema de mapeo: describe el formato de registro (JSON, CSV), codificación de los datos almacenados en el objeto de Amazon S3. También debe describir cómo cada elemento de datos se correlaciona con las columnas de la tabla de referencia en la aplicación.

A continuación se muestra el cuerpo de la solicitud en la solicitud de API `AddApplicationReferenceDataSource`.

```
{
    "applicationName": "string",
    "CurrentapplicationVersionId": number,
    "ReferenceDataSource": {
        "ReferenceSchema": {
            "RecordColumns": [
                {
                    "IsDropped": boolean,
                    "Mapping": "string",
                    "Name": "string",
                    "SqlType": "string"
                }
            ],
            "RecordEncoding": "string",
            "RecordFormat": {
                "MappingParameters": {
                    "CSVMappingParameters": {
                        "RecordColumnDelimiter": "string",
                        "RecordRowDelimiter": "string"
                    },
                    "JSONMappingParameters": {
                        "RecordRowPath": "string"
                    }
                },
                "RecordFormatType": "string"
            }
        },
        "S3ReferenceDataSource": {
            "BucketARN": "string",
```

```
            "FileKey": "string",
            "ReferenceRoleARN": "string"
        },
        "TableName": "string"
    }
}
```

Trabajando con JSONPath

> **Note**
>
> Después del 12 de septiembre de 2023, no podrá crear nuevas aplicaciones con Kinesis Data Firehose como origen si aún no utiliza Kinesis Data Analytics para SQL. Para obtener más información, consulte Límites.

JSONPath es una forma estandarizada de consultar los elementos de un objeto JSON. JSONPath utiliza expresiones de ruta para navegar por los elementos, los elementos anidados y las matrices de un documento JSON. Para obtener más información acerca de JSON, consulte Introducing JSON.

Amazon Kinesis Data Analytics JSONPath utiliza expresiones en el esquema de origen de la aplicación para identificar los elementos de datos de una fuente de streaming que contiene datos en formato JSON.

Para obtener más información sobre cómo mapear los datos de streaming al flujo de entrada de la aplicación, consulte the section called “Mapeo de elementos de origen de streaming a columnas de entrada de SQL”.

Acceder a JSON Elements con JSONPath

A continuación, encontrará información sobre cómo usar JSONPath expresiones para acceder a varios elementos de datos con formato JSON. Para los ejemplos de esta sección, suponga que la secuencia de origen contiene el siguiente registro de JSON:

```
{
  "customerName":"John Doe",
  "address":
  {
    "streetAddress":
```

```
    [
      "number":"123",
      "street":"AnyStreet"
    ],
    "city":"Anytown"
  }
  "orders":
  [
    { "orderId":"23284", "itemName":"Widget", "itemPrice":"33.99" },
    { "orderId":"63122", "itemName":"Gadget", "itemPrice":"22.50" },
    { "orderId":"77284", "itemName":"Sprocket", "itemPrice":"12.00" }
  ]
}
```

Acceso a los elementos de JSON

Para consultar un elemento de los datos de JSON JSONPath, utilice la siguiente sintaxis. A continuación, $ representa la raíz de la jerarquía de datos y `elementName` es el nombre del nodo del elemento de consulta.

```
$.elementName
```

La siguiente expresión consulta el `customerName` elemento en el ejemplo de JSON anterior.

```
$.customerName
```

La expresión anterior devuelve lo siguiente del registro de JSON anterior.

```
John Doe
```

> **Note**
>
> Las expresiones de ruta distinguen entre mayúsculas y minúsculas. La expresión `$.customername` vuelve `null` desde el anterior ejemplo de JSON.

> **Note**
>
> Si no aparece el elemento en la ubicación donde se especifica la expresión de la ruta, la expresión devuelve `null`. La siguiente expresión vuelve `null` desde el ejemplo anterior de JSON, ya que no existe ningún elemento coincidente.
>
> ```
> $.customerId
> ```

Acceso a los elementos de JSON anidados

Para consultar un elemento de JSON anidado, utilice la siguiente sintaxis.

```
$.parentElement.element
```

La siguiente expresión consulta el `city` elemento en el ejemplo de JSON anterior.

```
$.address.city
```

La expresión anterior devuelve lo siguiente del registro de JSON anterior.

```
Anytown
```

Puede consultar más niveles de subelementos utilizando la siguiente sintaxis.

```
$.parentElement.element.subElement
```

La siguiente expresión consulta el `street` elemento en el ejemplo de JSON anterior.

```
$.address.streetAddress.street
```

La expresión anterior devuelve lo siguiente del registro de JSON anterior.

```
AnyStreet
```

Acceso a matrices

Puede obtener acceso a los datos de una matriz JSON de las siguientes maneras:

- Recuperar todos los elementos de la matriz en una sola fila.
- Recuperar cada elemento de la matriz como una fila independiente.

Recuperar todos los elementos de una matriz en una sola fila

Para consultar todo el contenido de una matriz como una sola fila, utilice la siguiente sintaxis.

```
$.arrayObject[0:]
```

La siguiente expresión consulta todos los contenidos del `orders` elemento en el ejemplo de JSON anterior que se usa en esta sección. Devuelve los contenidos del arreglo en una única columna en una sola fila.

```
$.orders[0:]
```

La expresión anterior devuelve lo siguiente del registro de JSON de ejemplo que se usa en esta sección.

```
[{"orderId":"23284","itemName":"Widget","itemPrice":"33.99"},
{"orderId":"61322","itemName":"Gadget","itemPrice":"22.50"},
{"orderId":"77284","itemName":"Sprocket","itemPrice":"12.00"}]
```

Recuperar todos los elementos de una matriz en filas independientes

Para consultar los elementos individuales en una matriz como filas distintas, utilice la siguiente sintaxis.

```
$.arrayObject[0:].element
```

La siguiente expresión consulta los `orderId` elementos en la expresión JSON anterior y devuelve cada elemento de matriz como una fila distinta.

```
$.orders[0:].orderId
```

La expresión anterior devuelve lo siguiente del registro de JSON anterior, con cada elemento de datos devuelto como una fila distinta.

23284
63122
77284

Note

Si expresiones que consulta elementos no de matriz se incluyen en un esquema que consulta los elementos de matriz individuales, los elementos no de matriz se repiten para cada elemento de la matriz. Por ejemplo, suponga que un esquema para el ejemplo anterior JSON incluye las siguientes expresiones:

- $.customerName
- $.orders[0:].orderId

En este caso, las filas de datos devueltas del elemento de la secuencia de entrada de muestra se parecen a lo siguiente, con el `name` elemento repetido para cada `orderId` elemento.

John Doe	23284
John Doe	63122
John Doe	77284

Note

Las siguientes limitaciones se aplican a las expresiones de matriz en Amazon Kinesis Data Analytics:

- Solo un nivel de cancelación de referencias es compatible con una matriz de expresión. No se admite el siguiente formato de expresión.

```
$.arrayObject[0:].element[0:].subElement
```

- Solo una matriz puede aplanarse en un esquema. Se puede hacer referencia a varias matrices: se devuelven como una fila que contiene todos los elementos de la matriz. Sin embargo, solo una matriz puede tener cada uno de sus elementos devueltos como filas individuales.

  Un esquema que contienen elementos en el siguiente formato es válido. Este formato devuelve los contenidos de la segunda matriz como una única columna, repetida para cada elemento en la primera matriz.

```
$.arrayObjectOne[0:].element
$.arrayObjectTwo[0:]
```

  Un esquema que contiene elementos en el siguiente formato no es válido.

```
$.arrayObjectOne[0:].element
$.arrayObjectTwo[0:].element
```

Otras consideraciones

Las consideraciones adicionales para trabajar con JSONPath él son las siguientes:

- Si un elemento individual de las JSONPath expresiones del esquema de la aplicación no accede a ninguna matriz, se crea una sola fila en el flujo de entrada de la aplicación para cada registro JSON procesado.
- Cuando se aplana una matriz (es decir, sus elementos se devuelven como filas individuales), cualquier elemento que fata genera la creación de un valor nulo en el flujo de la aplicación.
- Una matriz siempre se aplana a al menos una fila. Si no se devuelven valores (es decir, la matriz está vacía o ninguno de sus elementos se ha consultado), se devuelve una sola fila con todos los valores nulos.

  En la siguiente expresión devuelve los registros con valores nulos del ejemplo de JSON anterior, ya que no existe ningún elemento coincidentes en la ruta especificada.

```
$.orders[0:].itemId
```

  La expresión anterior devuelve lo siguiente del registro de ejemplo de JSON anterior.

null
nulo
null

Temas relacionados

- Presentación de JSON

Mapeo de elementos de origen de streaming a columnas de entrada de SQL

> **Note**
>
> Después del 12 de septiembre de 2023, no podrá crear nuevas aplicaciones con Kinesis Data Firehose como origen si aún no utiliza Kinesis Data Analytics para SQL. Para obtener más información, consulte Límites.

Con Amazon Kinesis Data Analytics, puede procesar y analizar datos de streaming en formatos JSON o CSV mediante SQL estándar.

- Para procesar y analizar los datos CSV de streaming, debe asignar los nombres de columna y los tipos de datos para las columnas de la secuencia de entrada. Su aplicación importa una columna de la secuencia de entrada por definición de columna, en orden.

  No tiene que incluir todas las columnas en la secuencia de entrada de la aplicación, pero no puede omitir columnas de la secuencia de origen. Por ejemplo, puede importar las primeras tres columnas de una secuencia de entrada que contiene cinco elementos, pero no puede importar solo las columnas 1, 2 y 4.
- Para procesar y analizar los datos JSON de transmisión, usa JSONPath expresiones para mapear elementos JSON de una fuente de transmisión a columnas SQL en una secuencia de entrada. Para obtener más información sobre el uso JSONPath con Amazon Kinesis Data Analytics, Trabajando con JSONPath consulte. Las columnas en la tabla SQL poseen los tipos de datos

que se asignan a partir de tipos de JSON. Para conocer los tipos de datos compatibles, consulte Tipos de datos. Para obtener más información sobre la conversión de datos JSON en datos SQL, consulte Mapeo de tipos de datos JSON para los tipos de datos SQL.

Para obtener más información sobre cómo configurar las secuencias de entrada, consulte Configuración de entrada de la aplicación.

Mapeo de datos JSON a las columnas SQL

Puede asignar elementos JSON a columnas de entrada mediante la API de Kinesis Data Analytics Consola de administración de AWS o la API.

- Para asignar elementos a columnas con la consola, consulte Uso del editor de esquemas.
- Para asignar elementos a columnas mediante API de Kinesis Data Analytics, consulte la siguiente sección.

Para asignar elementos JSON a columnas en la secuencia de entrada en la aplicación, necesita un esquema con la siguiente información para cada columna:

- Expresión de origen: la JSONPath expresión que identifica la ubicación de los datos de la columna.
- Nombre de la columna: el nombre que las consultas SQL utilizan para hacer referencia a los datos.
- Tipo de datos: el tipo de datos SQL para la columna.

Uso de la API

Para asignar elementos de un origen de streaming a columnas de entrada, puede utilizar la acción CreateApplication de API de Kinesis Data Analytics. Para crear la secuencia en la aplicación, especifique un esquema para transformar los datos en una versión esquematizada utilizada en SQL. La CreateApplication acción configura su aplicación para que reciba entradas de un solo origen de streaming. Para asignar elementos JSON o columnas CSV a las columnas SQL, debe crear un RecordColumn objeto en la SourceSchema `RecordColumns` matriz. El RecordColumn objeto tiene el siguiente esquema:

```
{
    "Mapping": "String",
    "Name": "String",
```

```
    "SqlType": "String"
}
```

Los campos en el RecordColumn objeto tienen los siguientes valores:

- `Mapping`: la JSONPath expresión que identifica la ubicación de los datos en el registro del flujo de entrada. Este valor no está presente para un esquema de entrada para una secuencia de origen en formato CSV.
- `Name`: el nombre de columna en el flujo de datos SQL en la aplicación.
- `SqlType`: el tipo de datos de los datos en el flujo de datos SQL en la aplicación.

Ejemplo del esquema de entrada JSON

El siguiente ejemplo ilustra el formato del `InputSchema` valor para un esquema JSON.

```
"InputSchema": {
    "RecordColumns": [
        {
            "SqlType": "VARCHAR(4)",
            "Name": "TICKER_SYMBOL",
            "Mapping": "$.TICKER_SYMBOL"
        },
        {
            "SqlType": "VARCHAR(16)",
            "Name": "SECTOR",
            "Mapping": "$.SECTOR"
        },
        {
            "SqlType": "TINYINT",
            "Name": "CHANGE",
            "Mapping": "$.CHANGE"
        },
        {
            "SqlType": "DECIMAL(5,2)",
            "Name": "PRICE",
            "Mapping": "$.PRICE"
        }
    ],
    "RecordFormat": {
        "MappingParameters": {
```

```
            "JSONMappingParameters": {
                "RecordRowPath": "$"
            }
        },
        "RecordFormatType": "JSON"
    },
    "RecordEncoding": "UTF-8"
}
```

Ejemplo de esquema de entrada de CSV

El siguiente ejemplo ilustra el formato del `InputSchema` valor de un esquema en formato de valor separado por coma (CSV).

```
"InputSchema": {
    "RecordColumns": [
        {
            "SqlType": "VARCHAR(16)",
            "Name": "LastName"
        },
        {
            "SqlType": "VARCHAR(16)",
            "Name": "FirstName"
        },
        {
            "SqlType": "INTEGER",
             "Name": "CustomerId"
        }
    ],
    "RecordFormat": {
        "MappingParameters": {
            "CSVMappingParameters": {
                "RecordColumnDelimiter": ",",
                "RecordRowDelimiter": "\n"
            }
        },
        "RecordFormatType": "CSV"
    },
    "RecordEncoding": "UTF-8"
}
```

Mapeo de tipos de datos JSON para los tipos de datos SQL

Los tipos de datos JSON se convierten en tipos de datos SQL correspondientes en función del esquema de entrada de la aplicación. Para obtener información acerca de los tipos de datos de SQL compatibles, consulte Tipos de datos. Amazon Kinesis Data Analytics convierte los tipos de datos de JSON en tipos de datos SQL conforme a las reglas siguientes.

Literal nulo

Un literal nulo en la secuencia de entrada de JSON (es decir, `"City":null`) se convierte en un SQL nulo independientemente del tipo de datos de destino.

Literal booleano

Un literal booleano en la secuencia de entrada de JSON (es decir, `"Contacted":true`) se convierte en datos SQL de la siguiente manera:

- Numérico, (DECIMAL, INT, etc.): `true` se convierte en 1; `false` se convierte en 0.
- Binario (BINARY o VARBINARY):
  - `true`: El resultado tiene un conjunto de bits más bajo y ha borrado bits restantes.
  - `false`: El resultado ha borrado todos los bits.

  La conversión a VARBINARY genera un valor 1 bit de largo.
- BOOLEAN: se convierte al correspondiente valor BOOLEANO SQL correspondiente.
- Carácter (CHAR o VARCHAR): se convierte al valor de cadena correspondiente (`true` o `false`). El valor está truncado para adaptarse a la longitud del campo.
- Datetime (DATE, TIME o TIMESTAMP): falla la conversión y un error de coerción se escribe en la secuencia de errores.

Número

Un literal de número en la secuencia de entrada de JSON (es decir, `"CustomerId":67321`) se convierte en datos SQL de la siguiente manera:

- Numérico (DECIMAL, INT, etc.): se convierte directamente. Si el valor convertido supera el tamaño o precisión del tipo de datos de destino (es decir, convertir `123.4` a INT), falla la conversión y se genera un error de coerción en la secuencia de errores.

- Binario (BINARY o VARBINARY): falla la conversión y se genera un error de coerción en la secuencia de errores.
- BOOLEAN:
  - `0`: se convierte en `false`.
  - Todos otros números: se convierten en `true`.
- Carácter (CHAR o VARCHAR): se convierte en una representación de cadena del número.
- Datetime (DATE, TIME o TIMESTAMP): falla la conversión y un error de coerción se escribe en la secuencia de errores.

Cadena

Un valor de cadena en la secuencia de entrada de JSON (es decir, `"CustomerName":"John Doe"`) se convierte en datos SQL de la siguiente manera:

- Numérico (DECIMAL, INT, etc.): Amazon Kinesis Data Analytics intenta convertir el valor en el tipo de datos de destino. Si el valor no se puede convertir, falla la conversión y se genera un error de coerción en la secuencia de errores.
- Binario (BINARY o VARBINARY): Si la cadena de origen es literal binaria válida (es decir, `X'3F67A23A'` con un número par de f), el valor se convierte en el tipo de datos de destino. De lo contrario, falla la conversión y se genera un error de coerción en la secuencia de errores.
- BOOLEAN: Si la cadena de origen es `"true"`, se convierte en `true`. Esta comparación distingue entre mayúsculas y minúsculas. De lo contrario, se convierte en `false`.
- Carácter (CHAR o VARCHAR): se convierte en el valor de cadena en la entrada. Si el valor es mayor que el tipo de datos de destino, es truncado y no se escribe ningún error en la secuencia de errores.
- Datetime (DATE, TIME o TIMESTAMP): si la cadena de origen está en un formato que puede convertirse en el valor de destino, el valor se convierte. De lo contrario, falla la conversión y se genera un error de coerción en la secuencia de errores.

  Los formatos de marca temporal válidos son:

  - "1992-02-14"
  - "1992-02-14 18:35:44.0"

Matriz u objeto

Una matriz u objeto en la secuencia de entrada de JSON se convierte en datos SQL de la siguiente manera:

- Carácter (CHAR o VARCHAR): se convierte en el texto de origen de la matriz u objeto. Consulte Acceso a matrices.
- Todos los otros tipos de datos: falla la conversión y se genera un error de coerción en la secuencia de errores.

Para ver un ejemplo de la matriz de JSON, consulte Trabajando con JSONPath.

Temas relacionados

- Configuración de entrada de la aplicación
- Tipos de datos
- Uso del editor de esquemas
- CreateApplication
- RecordColumn
- SourceSchema

Uso de la función de detección de esquema en datos de streaming

> **Note**
>
> Después del 12 de septiembre de 2023, no podrá crear nuevas aplicaciones con Kinesis Data Firehose como origen si aún no utiliza Kinesis Data Analytics para SQL. Para obtener más información, consulte Límites.

Proporcionar un esquema de entrada que describa cómo los registros de la transmisión se asignan a una transmisión integrada en la aplicación puede resultar engorroso y propenso a errores. Puede utilizar la API DiscoverInputSchema (denominada API de detección) para inferir un esquema. Al utilizar muestras aleatorias de registros de la fuente de streaming, la API puede deducir un esquema (es decir, los nombres de las columnas, los tipos de datos y la posición del elemento de datos en los datos entrantes).

> **Note**
>
> Para utilizar la API de detección para generar un esquema a partir de un archivo almacenado en Amazon S3, consulte Uso de la función de detección de esquema en datos estáticos.

La consola usa la API Discovery para generar un esquema para una fuente de transmisión específica. Con la consola, también puede actualizar el esquema, lo que incluye agregar o eliminar columnas, cambiar los nombres de las columnas o los tipos de datos, etc. Sin embargo, realice los cambios con cuidado para asegurarse de que no crea un esquema no válido.

Una vez que finalice un esquema para su secuencia en la aplicación, existen las funciones que se pueden utilizar para manipular cadena y valores de datetime. Puede utilizar estas funciones en el código de la aplicación a la hora de trabajar con filas en la secuencia en la aplicación resultante. Para obtener más información, consulte Ejemplo: transformación DateTime de valores.

Nombres de columnas durante la detección de esquemas

Durante la detección de esquemas, Amazon Kinesis Data Analytics intenta conservar cuanto puede del nombre original de la columna del origen de entrada de streaming, excepto en los casos siguientes:

- El nombre de la columna de secuencia de origen es una palabra clave SQL reservada como `TIMESTAMP`, `USER`, `VALUES` o `YEAR`.
- El nombre de la columna de secuencia de origen contiene caracteres no compatibles. Solo letras, números y guiones bajos (_) son compatibles.
- El nombre de la columna de secuencia de origen comienza con un número.
- El nombre de la columna de secuencia de origen tiene más de 100 caracteres de largo.

Si se ha cambiado el nombre de la columna, la columna de esquema con nombre cambiado comienza con `COL_`. En algunos casos, ninguno de los nombres de columna original se pueden conservar, por ejemplo, si todo el nombre tiene caracteres no compatibles. En este caso, la columna se denomina `COL_#`, con # que indica el lugar de la columna en el orden de la columna.

Una vez completada la detección, puede actualizar el esquema utilizando la consola para añadir o eliminar columnas, o cambiar los nombres de columna, los tipos de datos o el tamaño de los datos.

Ejemplos de nombres de columna sugeridos por la detección

Nombre de columna de secuencia de origen	Nombre de columna sugerido por la detección
USER	COL_USER
USER@DOMAIN	COL_USERDOMAIN
@@	COL_0

Problemas de la detección de esquemas

¿Qué sucede si Kinesis Data Analytics no infiere un esquema para un determinado origen de streaming?

Kinesis Data Analytics infiere el esquema para formatos comunes, como CSV y JSON, que están codificados en UTF-8. Kinesis Data Analytics admite cualquier registro codificado en UTF-8, incluido texto sin procesar, como los registros y logs de las aplicaciones, con una columna personalizada y un delimitador de filas. Si Kinesis Data Analytics no infiere un esquema, es posible definir un esquema manualmente con el editor de esquemas de la consola (o utilizando la API).

Si los datos no siguen un patrón (que puede especificar con el editor de esquemas), puede definir un esquema como un único tipo de columnas VARCHAR (N), donde N es el mayor número de caracteres que espera que incluya un registro. A partir de ahí, puede utilizar la hora y la fecha de manipulación de cadenas para estructurar los datos una vez que estén en una secuencia en la aplicación. Para ver ejemplos, consulte Ejemplo: transformación DateTime de valores.

Uso de la función de detección de esquema en datos estáticos

> **Note**
>
> Después del 12 de septiembre de 2023, no podrá crear nuevas aplicaciones con Kinesis Data Firehose como origen si aún no utiliza Kinesis Data Analytics para SQL. Para obtener más información, consulte Límites.

La característica de detección de esquema puede generar un esquema a partir de los datos de una secuencia o de los datos de un archivo estático almacenado en un bucket de Amazon S3.

Supongamos que desea generar un esquema para una aplicación de Kinesis Data Analytics con fines de demostración o si no se dispone de datos de streaming activos. Puede utilizar la característica de detección de esquemas en un archivo estático que contiene una muestra de datos con el formato esperado para los datos de referencia o de streaming. Kinesis Data Analytics puede ejecutar la detección de esquemas en datos de muestra a partir de un archivo JSON o CSV que está almacenado en un bucket de Amazon S3. El uso de la detección de esquemas en un archivo de datos utiliza la consola o la API DiscoverInputSchema con el parámetro `S3Configuration` especificado.

Ejecución de la detección de esquemas mediante la consola

Para ejecutar la detección en un archivo estático mediante la consola, haga lo siguiente:

1. Añada un objeto de datos de referencia a un bucket de S3.
2. Elija Conectar datos de referencia en la página principal de la aplicación en la consola de Kinesis Data Analytics.
3. Proporcione los datos de bucket, ruta y rol de IAM para obtener acceso al objeto de Amazon S3 que contiene los datos de referencia.
4. Elija Detectar esquema.

Para obtener más información sobre cómo añadir datos de referencia y descubrir el esquema en la consola, consulte Ejemplo: Agregar datos de referencia a una aplicación de Kinesis Data Analytics.

Ejecución de la detección de esquemas mediante la API

Para ejecutar la detección en un archivo estático mediante la API, se debe proporcionar la API con una estructura `S3Configuration` con la siguiente información:

- `BucketARN`: el nombre de recurso de Amazon (ARN) del bucket de Amazon S3 que contiene el archivo. Para ver el formato del ARN de un bucket de Amazon S3, consulte Amazon Resource Names (ARNs) y Amazon Service Namespaces: Amazon Simple Storage Service (Amazon S3).
- `RoleARN`: el ARN de un rol de IAM con la política `AmazonS3ReadOnlyAccess`. Para obtener más información sobre cómo aplicar una política a una función, consulte Modificación de un rol.
- `FileKey`: El nombre de archivo del objeto.

Para generar un esquema a partir de un objeto de Amazon S3 utilizando el API **`DiscoverInputSchema`**

1. Asegúrese de tener la configuración adecuada. AWS CLI Para obtener más información, consulte Paso 2: Configura el AWS Command Line Interface ()AWS CLI en la sección Introducción.
2. Cree un archivo denominado `data.csv` con el siguiente contenido:

```
year,month,state,producer_type,energy_source,units,consumption
2001,1,AK,TotalElectricPowerIndustry,Coal,ShortTons,47615
2001,1,AK,ElectricGeneratorsElectricUtilities,Coal,ShortTons,16535
2001,1,AK,CombinedHeatandPowerElectricPower,Coal,ShortTons,22890
2001,1,AL,TotalElectricPowerIndustry,Coal,ShortTons,3020601
2001,1,AL,ElectricGeneratorsElectricUtilities,Coal,ShortTons,2987681
```

3. Inicie sesión en la consola de Amazon S3 en https://console.aws.amazon.com/s3/.
4. Cree un bucket de Amazon S3 y cargue el archivo `data.csv` que ha creado. Anote el ARN del bucket creado. Para obtener información sobre la creación de un bucket de Amazon S3 y cargar un archivo, consulte Introducción a Amazon Simple Storage Service.
5. Abra la consola de IAM en https://console.aws.amazon.com/iam/. Cree una función con la política `AmazonS3ReadOnlyAccess`. Anote el ARN de la nueva función. Para obtener más información acerca de cómo crear una función, consulte Crear un rol para delegar permisos a un servicio de Amazon. Para obtener más información sobre cómo aplicar una política a una función, consulte Modificación de un rol.
6. Ejecute el siguiente `DiscoverInputSchema` comando en el AWS CLI, sustituyéndolo ARNs por su bucket de Amazon S3 y su función de IAM:

```
$aws kinesisanalytics discover-input-schema --s3-configuration '{ "RoleARN":
 "arn:aws:iam::123456789012:role/service-role/your-IAM-role", "BucketARN":
 "arn:aws:s3:::your-bucket-name", "FileKey": "data.csv" }'
```

7. La respuesta tiene un aspecto similar a la siguiente:

```
{
    "InputSchema": {
        "RecordEncoding": "UTF-8",
        "RecordColumns": [
            {
                "SqlType": "INTEGER",
```

```
                "Name": "COL_year"
            },
            {
                "SqlType": "INTEGER",
                "Name": "COL_month"
            },
            {
                "SqlType": "VARCHAR(4)",
                "Name": "state"
            },
            {
                "SqlType": "VARCHAR(64)",
                "Name": "producer_type"
            },
            {
                "SqlType": "VARCHAR(4)",
                "Name": "energy_source"
            },
            {
                "SqlType": "VARCHAR(16)",
                "Name": "units"
            },
            {
                "SqlType": "INTEGER",
                "Name": "consumption"
            }
        ],
        "RecordFormat": {
            "RecordFormatType": "CSV",
            "MappingParameters": {
                "CSVMappingParameters": {
                    "RecordRowDelimiter": "\r\n",
                    "RecordColumnDelimiter": ","
                }
            }
        }
    },
    "RawInputRecords": [
        "year,month,state,producer_type,energy_source,units,consumption
\r\n2001,1,AK,TotalElectricPowerIndustry,Coal,ShortTons,47615\r
\n2001,1,AK,ElectricGeneratorsElectricUtilities,Coal,ShortTons,16535\r
\n2001,1,AK,CombinedHeatandPowerElectricPower,Coal,ShortTons,22890\r
\n2001,1,AL,TotalElectricPowerIndustry,Coal,ShortTons,3020601\r
\n2001,1,AL,ElectricGeneratorsElectricUtilities,Coal,ShortTons,2987681"
```

```
    ],
    "ParsedInputRecords": [
        [
            null,
            null,
            "state",
            "producer_type",
            "energy_source",
            "units",
            null
        ],
        [
            "2001",
            "1",
            "AK",
            "TotalElectricPowerIndustry",
            "Coal",
            "ShortTons",
            "47615"
        ],
        [
            "2001",
            "1",
            "AK",
            "ElectricGeneratorsElectricUtilities",
            "Coal",
            "ShortTons",
            "16535"
        ],
        [
            "2001",
            "1",
            "AK",
            "CombinedHeatandPowerElectricPower",
            "Coal",
            "ShortTons",
            "22890"
        ],
        [
            "2001",
            "1",
            "AL",
            "TotalElectricPowerIndustry",
            "Coal",
```

```
            "ShortTons",
            "3020601"
        ],
        [
            "2001",
            "1",
            "AL",
            "ElectricGeneratorsElectricUtilities",
            "Coal",
            "ShortTons",
            "2987681"
        ]
    ]
}
```

Procesamiento previo de registros con una función de Lambda

> **Note**
>
> Después del 12 de septiembre de 2023, no podrá crear nuevas aplicaciones con Kinesis Data Firehose como origen si aún no utiliza Kinesis Data Analytics para SQL. Para obtener más información, consulte Límites.

Si los datos de su transmisión necesitan conversión de formato, transformación, enriquecimiento o filtrado, puede preprocesar los datos mediante una función. AWS Lambda Puede hacerlo antes de que se ejecute el código SQL de la aplicación o antes de que la aplicación cree un esquema a partir deel flujo de datos.

El uso de una función de Lambda para el procesamiento previo de registros es útil en las siguientes situaciones:

- Transformar registros a partir de otros formatos (como KPL o GZIP) en formatos que Kinesis Data Analytics puede analizar. Kinesis Data Analytics actualmente admite formatos de datos JSON o CSV.
- Expandir datos en un formato que sea más asequible para operaciones como, por ejemplo, agregación o detección de anomalías. Por ejemplo, si se almacenan juntos en una cadena varios valores de datos, puede expandir los datos en columnas independientes.

- El enriquecimiento de datos con otros servicios de Amazon, como la extrapolación o la corrección de errores.
- Aplicación de transformación de cadenas complejas en campos de registros.
- Filtrado de datos para limpieza de datos.

Uso de una función de Lambda para procesamiento previo de registros

Al crear su aplicación de Kinesis Data Analytics, habilita el procesamiento previo de en la página Conectar con una fuente.

Para utilizar una función de Lambda para procesamiento previo de registros en una aplicación de Kinesis Data Analytics

1. Inicie sesión en la consola Managed Service for Apache Flink Consola de administración de AWS y ábrala en /kinesisanalytics. https://console.aws.amazon.com
2. En la página Conectar a un origen de la aplicación, elija Habilitado en la sección Procesamiento previo de registros con AWS Lambda.
3. Para utilizar una función de Lambda que haya creado, elija la función en la lista desplegable Función de Lambda .
4. Para crear una función de Lambda nueva a partir de una de las plantillas de procesamiento previo de Lambda, elija la plantilla en la lista desplegable. A continuación, elija View <template name> in Lambda (Ver <nombre de la plantilla> en Lambda) para editar la función.
5. Para crear una nueva función de Lambda, elija Crear nueva. Para obtener información sobre la creación de una función Lambda, consulte Creación de una función HelloWorld Lambda y Explore la consola en la Guía para desarrolladores.AWS Lambda
6. Elija la versión de la función de Lambda que desea utilizar. Para utilizar la versión más reciente, elija $LATEST.

Cuando se elige o se crea una función de Lambda para el procesamiento previo de los registros, estos se procesan antes de que se ejecute el código SQL de la aplicación o de que la aplicación genere un esquema a partir de ellos.

Permisos de procesamiento previo de Lambda

Para utilizar el procesamiento previo de Lambda, el rol de IAM de la aplicación requiere la política de permisos siguiente:

```
    {
      "Sid": "UseLambdaFunction",
      "Effect": "Allow",
      "Action": [
          "lambda:InvokeFunction",
          "lambda:GetFunctionConfiguration"
      ],
      "Resource": "<FunctionARN>"
  }
```

Métricas de procesamiento previo de Lambda

Puede utilizar Amazon CloudWatch para supervisar el número de invocaciones a Lambda, los bytes procesados, los éxitos y los errores, etc. Para obtener información sobre CloudWatch las métricas emitidas por el preprocesamiento de Kinesis Data Analytics Lambda, consulte Amazon Kinesis Analytics Metrics.

Uso AWS Lambda con la biblioteca de Kinesis Producer

Kinesis Producer Library (KPL) agrega pequeños registros formateados por el usuario en registros de mayor tamaño, hasta 1 MB, para utilizar mejor el rendimiento de Amazon Kinesis Data Streams. La Kinesis Client Library (KCL) para Java es compatible con la desagrupación de estos registros. Sin embargo, debe utilizar un módulo especial para desagregar los registros cuando los utilice AWS Lambda como consumidor de sus transmisiones.

Para obtener el código y las instrucciones del proyecto necesarios, consulte los módulos de desagregación de la biblioteca de productores de Kinesis para obtener más información. AWS Lambda GitHub Puede usar los componentes de este proyecto para procesar datos serializados de KPL AWS Lambda en Java, Node.js y Python. Estos componentes también se pueden utilizar como parte de una aplicación KCL multilenguaje.

Modelo de respuesta de datos de entrada de eventos de preprocesamiento de datos Model/Record

Para realizar el procesamiento previo de los registros, la función de Lambda debe respetar los modelos necesarios de datos de entrada de eventos y de respuesta de registros.

Modelo de datos de entrada de eventos

Kinesis Data Analytics lee continuamente los datos de su flujo de datos de Kinesis o de su flujo de entrega de Firehose. Para cada lote de registros que recupera, el servicio administra cómo se

transfiere cada lote a su función de Lambda. Su función recibe una lista de registros como entrada. Dentro de su función, itera a través de la lista y aplica su lógica de negocio para llevar a cabo sus requisitos de procesamiento previo (tales como el enriquecimiento o la conversión de formato de datos).

El modelo de entrada a su función de preprocesamiento varía ligeramente, en función de si los datos se han recibido desde un flujo de datos de Kinesis o un flujo de entrega de Firehose.

Si el origen es un flujo de entrega de Firehose, el modelo de datos de entrada de eventos es el siguiente:

Modelo de datos de solicitud de Kinesis Data Firehose

Campo	Description (Descripción)
`invocationId`	El ID de invocación de Lambda (GUID aleatorio).
`applicationArn`	El nombre de recurso de Amazon (ARN) de la aplicación de Kinesis Data Analytics.
`streamArn`	ARN de secuencia de entrega
registros	

Campo	Description (Descripción)	
`recordId`	ID de registro (GUID aleatorio)	
`kinesisFirehoseRecordMetadata`		

Campo	Description (Descripción)
`approximateArrivalTimestamp`	Hora de llegada aproximada del registro de secuencia de entrega

Campo	Description (Descripción)

Campo	Description (Descripción)	
`data`	Carga útil de registro de origen codificada en Base64	

En el siguiente ejemplo, se muestra la entrada de una secuencia de entrega de Firehose:

```
{
   "invocationId":"00540a87-5050-496a-84e4-e7d92bbaf5e2",
   "applicationArn":"arn:aws:kinesisanalytics:us-east-1:12345678911:application/lambda-
test",
   "streamArn":"arn:aws:firehose:us-east-1:AAAAAAAAAAAA:deliverystream/lambda-test",
   "records":[
      {
         "recordId":"49572672223665514422805246926656954630972486059535892482",
         "data":"aGVsbG8gd29ybGQ=",
         "kinesisFirehoseRecordMetadata":{
            "approximateArrivalTimestamp":1520280173
         }
      }
   ]
}
```

Si el origen es un flujo de datos de Kinesis, el modelo de datos de entrada de eventos es el siguiente:

Modelo de datos de solicitud de secuencias de Kinesis.

Campo	Description (Descripción)
`invocationId`	El ID de invocación de Lambda (GUID aleatorio).
`applicationArn`	ARN de la aplicación de Kinesis Data Analytics
`streamArn`	ARN de secuencia de entrega
registros	

Campo	Description (Descripción)

Campo	Description (Descripción)	
`recordId`	ID de registro en función del número de secuencia de registro de Kinesis	
`kinesisStreamRecordMetadata`		

Campo	Description (Descripción)
`sequenceNumber`	Número de secuencia del registro de secuencia de Kinesis
`partitionKey`	Clave de partición del registro de secuencia de Kinesis
`shardId`	`ShardId` del registro de secuencia de Kinesis
`approximateArrivalTimestamp`	Hora de llegada aproximada del registro de secuencia de entrega

data	Carga útil de registro de origen codificada en Base64	

En el siguiente ejemplo, se muestra la entrada de un flujo de datos de Kinesis:

```
{
  "invocationId": "00540a87-5050-496a-84e4-e7d92bbaf5e2",
  "applicationArn": "arn:aws:kinesisanalytics:us-east-1:12345678911:application/lambda-
test",
  "streamArn": "arn:aws:kinesis:us-east-1:AAAAAAAAAAAA:stream/lambda-test",
  "records": [
    {
      "recordId": "49572672223665514422805246926656954630972486059535892482",
      "data": "aGVsbG8gd29ybGQ=",
      "kinesisStreamRecordMetadata":{
            "shardId" :"shardId-000000000003",
            "partitionKey":"7400791606",

 "sequenceNumber":"49572672223665514422805246926656954630972486059535892482",
            "approximateArrivalTimestamp":1520280173
         }
    }
  ]
}
```

Modelo de respuesta de registros

Deben devolverse todos los registros devueltos por la función de preprocesamiento de Lambda (con registro IDs) que se envíen a la función de Lambda. Deben contener los siguientes parámetros, de lo contrario, Kinesis Data Analytics los rechaza y los trata como error de procesamiento previo de datos. La parte de carga de datos del registro se puede transformar para cumplir los requisitos de procesamiento previo.

Modelo de datos de respuesta

registros

Campo	Description (Descripción)
`recordId`	El ID de registro se transfiere desde Kinesis Data Analytics hacia Lambda durante la invocación. El registro transformado debe contener el mismo ID de registro. Cualquier discrepancia entre el ID del registro original y el ID del registro transformado se trata como un error de procesamiento previo de datos.

Campo	Description (Descripción)
`result`	Es el estado de la transformación de los datos del registro. Los valores posibles son: • `Ok`: el registro se ha transformado correctamente. Kinesis Data Analytics ingiere el registro para el procesamiento de SQL. • `Dropped`: el registro lo ha descartado de forma intencion ada la lógica de procesamiento. Kinesis Data Analytics recibe el registro para el procesamiento de SQL. El campo de carga de datos es opcional para un registro `Dropped`. • `ProcessingFailed` : el registro no se ha podido transformar. Kinesis Data Analytics considera que lo ha procesado sin éxito la función de Lambda y escribe un error en la secuencia de errores. Para obtener más informaci ón acerca de la secuencia de errores, consulte Gestión de errores. El campo de carga de datos es opcional para un registro `ProcessingFailed` .
`data`	Es la carga útil de datos transformados después codificar los en base64. Cada carga de datos puede contener varios documentos JSON si el formato de datos de adquisición de la aplicación es JSON. O bien, cada una puede contener varias filas CSV (con un delimitador de filas especificado en cada fila) si el formato de datos de adquisición de la aplicación es CSV. El servicio de Kinesis Data Analytics analiza y procesa correctamente los datos con varios documentos JSON o filas CSV dentro de la misma carga de datos.

En el siguiente ejemplo, se muestra el resultado de una función Lambda:

```
{
  "records": [
    {
      "recordId": "49572672223665514422805246926656954630972486059535892482",
      "result": "Ok",
```

```
      "data": "SEVMTE8gV09STEQ="
    }
  ]
}
```

Errores comunes de procesamiento previo de datos

A continuación se indican los motivos habituales por los que un procesamiento previo puede generar un error.

- No todos los registros (con registro IDs) de un lote que se envían a la función Lambda se devuelven al servicio Kinesis Data Analytics.
- En la respuesta falta el campo de ID de registro, estado o carga de datos. El campo de carga de datos es opcional para un registro `Dropped` o `ProcessingFailed`.
- Los tiempos de espera de la función de Lambda no son suficientes para el procesamiento previo de los datos.
- La respuesta de la función de Lambda supera los límites de respuesta impuestos por el servicio de AWS Lambda .

Para errores de procesamiento previo de datos, Kinesis Data Analytics sigue reintentando las invocaciones de Lambda en el mismo conjunto de registros hasta que tiene éxito. Puede supervisar las siguientes CloudWatch métricas para obtener información sobre los errores.

- `MillisBehindLatest` de aplicación de Kinesis Data Analytics: indica el retraso que lleva la aplicación al leer desde el origen de streaming.
- Métricas de la `InputPreprocessing` CloudWatch aplicación Kinesis Data Analytics: indican el número de éxitos y fracasos, entre otras estadísticas. Para obtener más información, consulte Amazon Kinesis Analytics Metrics.
- AWS Lambda CloudWatch métricas y registros de funciones.

Creación de funciones de Lambda para el procesamiento previo

Una aplicación de Amazon Kinesis Data Analytics puede utilizar funciones de Lambda para realizar el procesamiento previo de registros a medida que estos se ingieren en la aplicación. Kinesis Data Analytics proporciona las plantillas siguientes en la consola para usarlas como punto de partida para el procesamiento previo de los datos.

Temas

Creación de una función de Lambda de procesamiento previo en Node.js

Las siguientes plantillas están disponibles en la consola de Kinesis Data Analytics para crear funciones de Lambda de procesamiento previo en Node.js:

Lambda Blueprint	Lenguaje y versión	Description (Descripción)
Procesamiento de entrada general de Kinesis Data Analytics	Node.js 6.10	Un procesador previo de registros de Kinesis Data Analytics que recibe registros JSON o CSV como entrada y los devuelve con un estado de procesamiento. Utilice este procesador como punto de partida para la lógica de transformación personalizada.
Procesamiento de entrada comprimida	Node.js 6.10	Un procesador de registros de Kinesis Data Analytics que recibe registros JSON o CSV comprimidos (GZIP o compresión Deflate) como entrada y devuelve los registros sin comprimir con un estado de procesamiento.

Creación de una función de Lambda de procesamiento previo en Python

Las siguientes plantillas están disponibles en la consola para crear funciones de Lambda de procesamiento previo en Python:

Lambda Blueprint	Lenguaje y versión	Description (Descripción)
Procesamiento de entrada general de Kinesis Analytics	Python 2.7	Un procesador previo de registros de Kinesis Data Analytics que recibe registros JSON o CSV como entrada y los devuelve con

Lambda Blueprint	Lenguaje y versión	Description (Descripción)
		un estado de procesamiento. Utilice este procesador como punto de partida para la lógica de transformación personalizada.
Procesamiento de entrada KPL	Python 2.7	Un procesador de registros de Kinesis Data Analytics que recibe agregados de Kinesis Producer Library (KPL) de registros JSON o CSV como entrada y devuelve registros sin agregar con un estado de procesamiento.

Creación de una función de Lambda de procesamiento previo en Java

Para crear una función de Lambda para el procesamiento previo de registros en Java, utilice las clases de eventos de Java.

El código siguiente ilustra un ejemplo de función de Lambda para el procesamiento previo de registros creada en Java:

```
public class LambdaFunctionHandler implements
        RequestHandler<KinesisAnalyticsStreamsInputPreprocessingEvent,
 KinesisAnalyticsInputPreprocessingResponse> {

    @Override
    public KinesisAnalyticsInputPreprocessingResponse handleRequest(
            KinesisAnalyticsStreamsInputPreprocessingEvent event, Context context) {
        context.getLogger().log("InvocatonId is : " + event.invocationId);
        context.getLogger().log("StreamArn is : " + event.streamArn);
        context.getLogger().log("ApplicationArn is : " + event.applicationArn);

        List<KinesisAnalyticsInputPreprocessingResponse.Record> records = new
 ArrayList<KinesisAnalyticsInputPreprocessingResponse.Record>();
        KinesisAnalyticsInputPreprocessingResponse response = new
 KinesisAnalyticsInputPreprocessingResponse(records);

        event.records.stream().forEach(record -> {
            context.getLogger().log("recordId is : " + record.recordId);
            context.getLogger().log("record aat is :" +
 record.kinesisStreamRecordMetadata.approximateArrivalTimestamp);
```

```
            // Add your record.data pre-processing logic here.

            // response.records.add(new Record(record.recordId,
 KinesisAnalyticsInputPreprocessingResult.Ok, <preprocessedrecordData>));
        });
        return response;
    }

}
```

Creación de una función de Lambda de procesamiento previo en .NET

Para crear una función de Lambda para el procesamiento previo de registros en .NET, utilice las clases de eventos de .NET.

El código siguiente ilustra un ejemplo de función de Lambda para el procesamiento previo de registros creada en C#:

```
public class Function
    {
        public KinesisAnalyticsInputPreprocessingResponse
 FunctionHandler(KinesisAnalyticsStreamsInputPreprocessingEvent evnt, ILambdaContext
 context)
        {
            context.Logger.LogLine($"InvocationId: {evnt.InvocationId}");
            context.Logger.LogLine($"StreamArn: {evnt.StreamArn}");
            context.Logger.LogLine($"ApplicationArn: {evnt.ApplicationArn}");

            var response = new KinesisAnalyticsInputPreprocessingResponse
            {
                Records = new List<KinesisAnalyticsInputPreprocessingResponse.Record>()
            };

            foreach (var record in evnt.Records)
            {
                context.Logger.LogLine($"\tRecordId: {record.RecordId}");
                context.Logger.LogLine($"\tShardId: {record.RecordMetadata.ShardId}");
                context.Logger.LogLine($"\tPartitionKey:
 {record.RecordMetadata.PartitionKey}");
                context.Logger.LogLine($"\tRecord ApproximateArrivalTime:
 {record.RecordMetadata.ApproximateArrivalTimestamp}");
                context.Logger.LogLine($"\tData: {record.DecodeData()}");
```

```
                // Add your record preprocessig logic here.

                var preprocessedRecord = new
KinesisAnalyticsInputPreprocessingResponse.Record
                {
                    RecordId = record.RecordId,
                    Result = KinesisAnalyticsInputPreprocessingResponse.OK
                };
                preprocessedRecord.EncodeData(record.DecodeData().ToUpperInvariant());
                response.Records.Add(preprocessedRecord);
            }
            return response;
        }
    }
```

Para obtener más información sobre cómo crear funciones de Lambda para el procesamiento previo y como destino en .NET, consulte `Amazon.Lambda.KinesisAnalyticsEvents`.

Paralelizar secuencias de entrada para mejorar el desempeño

> **Note**
>
> Después del 12 de septiembre de 2023, no podrá crear nuevas aplicaciones con Kinesis Data Firehose como origen si aún no utiliza Kinesis Data Analytics para SQL. Para obtener más información, consulte Límites.

Las aplicaciones de Amazon Kinesis Data Analytics admiten múltiples secuencias de entrada en la aplicación, para escalar una aplicación más allá del rendimiento de una única secuencia de entrada en la aplicación. Para obtener más información acerca de las secuencias de entrada en la aplicación, consulte Aplicaciones de Amazon Kinesis Data Analytics para SQL: cómo funciona.

En casi todos los casos, Amazon Kinesis Data Analytics escala la aplicación para gestionar la capacidad de los flujos de Kinesis o los flujos de origen de Firehose que se envían a la aplicación. Sin embargo, si el rendimiento de su secuencia de origen supera el rendimiento de una única secuencia de entrada en la aplicación, usted puede de forma explícita, aumentar el número de secuencias de entrada en la aplicación que utilice la aplicación. Puede hacerlo con el parámetro `InputParallelism`.

Cuando el parámetro `InputParallelism` es mayor que uno, Amazon Kinesis Data Analytics divide de manera uniforme las particiones de su secuencia de origen entre las secuencias en la aplicación. Por ejemplo, si su secuencia de origen tiene 50 fragmentos, y establece `InputParallelism` en 2, cada secuencia de entrada en la aplicación recibe la entrada de 25 fragmentos de secuencia de origen.

Al aumentar el número de secuencias en la aplicación, su aplicación deberá tener acceso a los datos de cada secuencia de forma explícita. Para obtener más información sobre cómo tener acceso a varias secuencias en la aplicación en el código, consulte Acceso a secuencias en la aplicación separadas en la aplicación de Amazon Kinesis Data Analytics.

Aunque las particiones de los flujos de datos y los flujos de Kinesis están divididas entre los flujos en la aplicación de la misma forma, se diferencian en cómo aparecen en la aplicación:

- Los registros de un flujo de datos de Kinesis incluyen un campo `shard_id` que puede utilizar para identificar la partición de origen del registro.
- Los registros de un flujo de entrega de Firehose no incluyen un campo que identifica el fragmento o la partición de origen del registro. Esto se debe a que Firehose abstrae esta información de la aplicación.

Evaluar si debe aumentar el número de secuencias de entrada en la aplicación

En la mayoría de los casos, una única secuencia de entrada en la aplicación puede gestionar el rendimiento de una única secuencia de origen, en función de la complejidad y del tamaño de los datos en las secuencias de entrada. Para determinar si necesitas aumentar el número de flujos de entrada en la aplicación, puedes monitorear `MillisBehindLatest` las métricas `InputBytes` y las métricas en Amazon CloudWatch.

Si la `InputBytes` métrica es superior a 100 MB/sec (o prevé que superará esta tasa), esto puede provocar un aumento `MillisBehindLatest` y un aumento del impacto de los problemas de aplicación. Para solucionar este problema, le recomendamos que configure las siguientes opciones del lenguaje para su aplicación:

- Utilice varias secuencias e instancias de aplicaciones de Kinesis Data Analytics para SQL si su aplicación tiene necesidades de escalado superiores a 100 MB/segundo.
- Utilice Kinesis Data Analytics for Java Applications si desea usar una sola secuencia y aplicación.

Si la métrica `MillisBehindLatest` tiene una de las siguientes características, debería aumentar el valor de `InputParallelism` de su aplicación:

- La métrica `MillisBehindLatest` se aumenta gradualmente, lo que indica que la aplicación queda rezagada con respecto a los datos más recientes de la secuencia.
- La métrica `MillisBehindLatest` está constantemente por encima de 1 000 (un segundo).

No es necesario para aumentar la configuración `InputParallelism` de su aplicación si lo siguiente es verdadero:

- La métrica `MillisBehindLatest` se reduce gradualmente, lo que indica que la aplicación está alcanzando a los datos más recientes de la secuencia.
- La métrica `MillisBehindLatest` está por debajo de 1 000 (un segundo).

Para obtener más información sobre su uso CloudWatch, consulte la Guía CloudWatch del usuario.

Implementación de múltiples secuencias de entrada en la aplicación

Puede definir el número de secuencias de entrada en la aplicación cuando se crea una aplicación utilizando CreateApplication. Puede definir este número una vez creada la aplicación utilizando UpdateApplication.

> **Note**
>
> Solo puede establecer la configuración `InputParallelism` mediante la API de Amazon Kinesis Data Analytics o la AWS CLI. No puede establecer esta configuración mediante Consola de administración de AWS. Para obtener información sobre cómo configurar el AWS CLI, consultePaso 2: Configura el AWS Command Line Interface ()AWS CLI.

Establecer un nuevo recuento de secuencia de entrada en la aplicación

El siguiente ejemplo ilustra cómo utilizar la acción de la API `CreateApplication` para establecer un nuevo recuento de secuencia de entrada de aplicación en 2.

Para obtener más información acerca de `CreateApplication`, consulte CreateApplication.

```
{
```

```
    "ApplicationCode": "<The SQL code the new application will run on the input
  stream>",
    "ApplicationDescription": "<A friendly description for the new application>",
    "ApplicationName": "<The name for the new application>",
    "Inputs": [
     {
       "InputId": "ID for the new input stream",
       "InputParallelism": {
         "Count": 2
     }],
    "Outputs": [ ... ],
  }]
}
```

Establecer una recuento de la secuencia de entrada de aplicación existente

El siguiente ejemplo ilustra cómo utilizar la acción de la API `UpdateApplication` para establecer un recuento existente de secuencia de entrada de aplicación en 2.

Para obtener más información acerca de `Update_Application`, consulte UpdateApplication.

```
{
   "InputUpdates": [
      {
         "InputId": "yourInputId",
         "InputParallelismUpdate": {
            "CountUpdate": 2
         }
      }
   ],
}
```

Acceso a secuencias en la aplicación separadas en la aplicación de Amazon Kinesis Data Analytics

Para utilizar varias secuencias de entrada en la aplicación en su aplicación, debe seleccionar de forma explícita entre las distintas secuencias. Los siguientes códigos de ejemplo demuestran cómo consultar múltiples secuencias de entrada en la aplicación creada durante el tutorial de introducción.

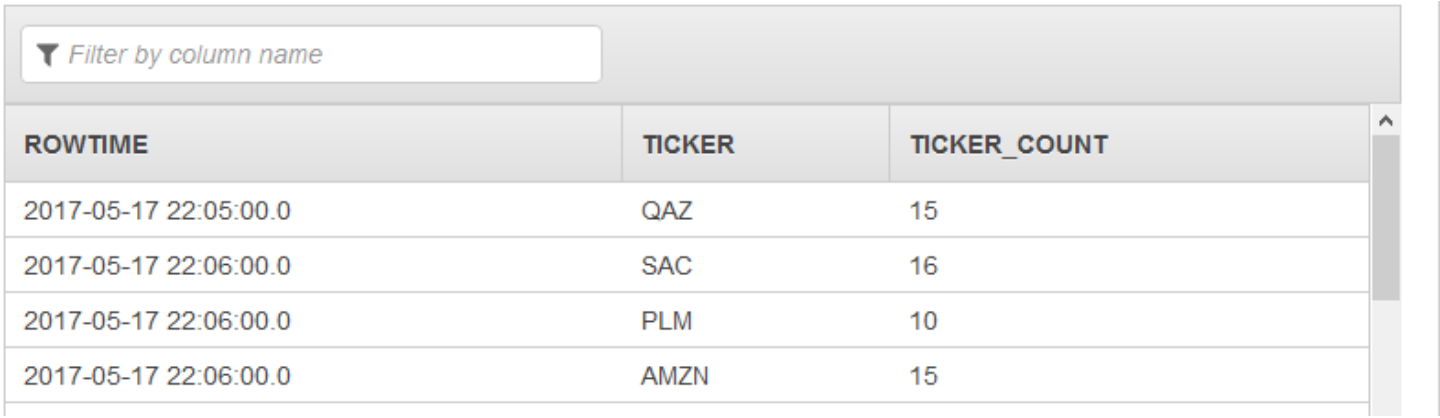
En el siguiente ejemplo, cada secuencia de origen es agregada primera utilizando COUNT antes de que se combinen en una única secuencia en la aplicación llamada `in_application_stream001`.

La suma de las secuencias de origen de antemano ayuda a garantizar que la secuencia combinada en la aplicación pueda gestionar el tráfico de varias secuencias sin llegar a estar sobrecargada.

> **Note**
>
> Para ejecutar este ejemplo y obtener resultados de ambas secuencias de entrada en la aplicación, actualice el número de fragmentos de su secuencia de origen y el parámetro `InputParallelism` de la aplicación.

```
CREATE OR REPLACE STREAM in_application_stream_001 (
    ticker VARCHAR(64),
    ticker_count INTEGER
);

CREATE OR REPLACE PUMP pump001 AS
INSERT INTO in_application_stream_001
SELECT STREAM ticker_symbol, COUNT(ticker_symbol)
FROM source_sql_stream_001
GROUP BY STEP(source_sql_stream_001.rowtime BY INTERVAL '60' SECOND),
    ticker_symbol;

CREATE OR REPLACE PUMP pump002 AS
INSERT INTO in_application_stream_001
SELECT STREAM ticker_symbol, COUNT(ticker_symbol)
FROM source_sql_stream_002
GROUP BY STEP(source_sql_stream_002.rowtime BY INTERVAL '60' SECOND),
    ticker_symbol;
```

El ejemplo de código anterior obtiene un resultado similar en `in_application_stream001` al del siguiente:



Consideraciones adicionales

Al utilizar múltiples secuencias de entrada, debe tener cuidado con lo siguiente:

- El número máximo de secuencias de entrada en la aplicación es 64.
- Las secuencias de entrada en la aplicación se distribuyen de manera uniforme entre los fragmentos de la secuencia de entrada de la aplicación.
- Las ganancias de desempeño al añadir secuencias en la aplicación no se escalan linealmente. Es decir, duplicar la cantidad de secuencias en la aplicación no duplica el rendimiento. Con un tamaño de filas típico, cada secuencia en la aplicación puede conseguir un rendimiento de aproximadamente 5 000 a 15 000 filas por segundo. Al aumentar el recuento de secuencia en la aplicación a 10, puede conseguir un rendimiento de 20 000 a 30 000 filas por segundo. La velocidad del rendimiento depende del recuento, de los tipos de datos y del tamaño de los datos de los campos en la secuencia de entrada.
- Algunas funciones agregadas (por ejemplo, AVG) puede producir resultados inesperados a la hora de aplicarlas a las secuencias de entrada particionadas en diferentes fragmentos. Como necesita ejecutar la operación agregada en fragmentos individuales antes de combinarlos en una secuencia agregada, los resultados se ponderarán hacia la secuencia que contenga más registros.
- Si su aplicación sigue teniendo un rendimiento deficiente (lo que se refleja en una `MillisBehindLatest` métrica alta) después de aumentar el número de transmisiones de entrada, es posible que haya alcanzado el límite de unidades de procesamiento de Kinesis ()KPUs. Para obtener más información, consulte Escalado automático de aplicaciones para incrementar el desempeño.

Código de la aplicación

El código de la aplicación es una serie de instrucciones SQL que procesan entradas y generan salidas. Estas instrucciones SQL operan con secuencias en la aplicación y con tablas de referencia. Para obtener más información, consulte Aplicaciones de Amazon Kinesis Data Analytics para SQL: cómo funciona.

Para obtener información sobre los elementos del lenguaje SQL compatibles con Kinesis Data Analytics, consulte Referencia de SQL de Amazon Kinesis Data Analytics.

En bases de datos relacionales, se trabaja con tablas, utilizando instrucciones INSERT para añadir registros y la instrucción SELECT para consultar los datos. En Amazon Kinesis Data Analytics, trabaja con flujos. Puede escribir una instrucción SQL para consultar estas secuencias. Los resultados de la consulta de una secuencia en la aplicación siempre se envían a otra secuencia en la aplicación. Cuando realice análisis complejos, puede crear varias secuencias en la aplicación para guardar los resultados de los análisis intermedios. Por último, puede configurar la salida de la aplicación para que conserve los resultados del análisis final (de una o más secuencias en la aplicación) en destinos externos. A continuación se muestra un patrón típico para escribir código de la aplicación:

- La instrucción SELECT siempre se utiliza en el contexto de una instrucción INSERT. Es decir, cuando selecciona filas, introduce resultados en otra secuencia en la aplicación.
- La instrucción INSERT siempre se utiliza en el contexto de una bomba. Es decir, que utiliza bombas para escribir a una secuencia en la aplicación.

El siguiente ejemplo de aplicación el código lee los registros de una secuencia en la aplicación (SOURCE_SQL_STREAM_001) y la escribe en otra secuencia en la aplicación (DESTINATION_SQL_STREAM). Puede insertar registros en secuencias en la aplicación utilizando bombas, tal y como se muestra a continuación:

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (ticker_symbol VARCHAR(4),
                                                   change DOUBLE,
                                                   price DOUBLE);
-- Create a pump and insert into output stream.
CREATE OR REPLACE PUMP "STREAM_PUMP" AS

  INSERT INTO "DESTINATION_SQL_STREAM"
    SELECT STREAM ticker_symbol, change,price
```

```
    FROM   "SOURCE_SQL_STREAM_001";
```

> **Note**
>
> Los identificadores que especifique para los nombres de secuencia y de columna siguen las convenciones estándar de SQL. Por ejemplo, si pone comillas en torno a un identificador, el identificador distingue entre mayúsculas y minúsculas. Si no las pone, el identificador adoptará las mayúsculas de forma predeterminada. Para obtener más información sobre los identificadores, consulte Identificadores en la Referencia de SQL de Amazon Managed Service para Apache Flink.

El código de la aplicación puede contener muchas instrucciones SQL. Por ejemplo:

- Puede escribir consultas SQL de manera secuencial donde el resultado de una instrucción SQL alimenta la siguiente instrucción SQL.
- También puede escribir consultas SQL que se ejecutan independientes entre sí. Por ejemplo, puede escribir dos instrucciones SQL que consulten la misma secuencia en la aplicación, pero enviar resultados en diferentes secuencias en la aplicación. A continuación, puede consultar las nuevas secuencias en la aplicación de manera independiente.

También puede crear secuencias en la aplicación para almacenar resultados de consultas intermedias. Puede insertar los datos en secuencias en la aplicación utilizando bombas. Para obtener más información, consulte Secuencias y bombeos en la aplicación.

Si añade una tabla de referencia en la aplicación, puede escribir SQL para unir los datos de las secuencias en la aplicación y las tablas de referencia. Para obtener más información, consulte Ejemplo: Agregar datos de referencia a una aplicación de Kinesis Data Analytics.

De acuerdo con la aplicación de la configuración de salida, Amazon Kinesis Data Analytics escribe datos de secuencias en la aplicación específicas a los destinos externos en función de la configuración de salida de la aplicación. Asegúrese de que el código de la aplicación escribe a las secuencias en la aplicación especificadas en la configuración de salida.

Para obtener más información, consulte los temas siguientes:

- Conceptos de SQL de Streaming
- Referencia de SQL de Amazon Kinesis Data Analytics

Configuración de salida de la aplicación

En el código de la aplicación, se escribe el resultado de las sentencias SQL en una o más secuencias de la aplicación. Si lo desea, puede añadir una configuración de salida a su aplicación para conservar todo lo escrito en una transmisión de la aplicación a un destino externo, como una transmisión de datos de Amazon Kinesis, una transmisión de entrega de Firehose o una función. AWS Lambda

Existe un límite en cuanto al número de destinos externos que puede utilizar para conservar datos en la salida de una aplicación. Para obtener más información, consulte Límites.

> **Note**
>
> Le recomendamos que utilice un destino externo para almacenar los datos de secuencia de errores en la aplicación para que pueda investigar los errores.

En cada una de estas configuraciones de salida, debe proporcionar lo siguiente:

- Nombre de la secuencia en la aplicación: la secuencia que desea conservar en un destino externo.

  Kinesis Data Analytics busca la secuencia en la aplicación que ha especificado en la configuración de salida. (El nombre de la transmisión distingue entre mayúsculas y minúsculas y debe coincidir exactamente). Asegúrese de que el código de su aplicación cree esta transmisión dentro de la aplicación.
- Destino externo: puede conservar los datos en un flujo de datos de Kinesis, un flujo de entrega de Firehose o una función de Lambda. Debe proporcionar el nombre de recurso de Amazon (ARN) de la transmisión o función. Además proporciona un rol de IAM que Kinesis Data Analytics puede asumir para escribir la secuencia o función en su nombre. Debe describir el formato de registro (JSON o CSV) que Kinesis Data Analytics debe utilizar a la hora de escribir en el destino externo.

Si Kinesis Data Analytics no puede escribir en el destino de streaming o de Lambda, el servicio sigue intentándolo de forma indefinida. Con ello se crea resistencia y la aplicación se queda retrasada. Si el problema no se resuelve, la aplicación finalmente detiene el procesamiento de datos nuevos. Puede monitorizar las métricas de Kinesis Data Analytics y establecer alarmas para los errores. Para obtener más información sobre las métricas y las alarmas, consulte Uso de Amazon CloudWatch Metrics y Creación de CloudWatch alarmas de Amazon.

Puede configurar la salida de la aplicación usando la Consola de administración de AWS. La consola realiza la llamada a la API para guardar la configuración.

Creación de una salida mediante el AWS CLI

En esta sección, se describe cómo crear la sección `Outputs` del cuerpo de la solicitud para una operación `CreateApplication` o `AddApplicationOutput`.

Creación de una salida de secuencias de Kinesis

El siguiente fragmento de JSON muestra la sección `Outputs` del cuerpo de la solicitud `CreateApplication` que se utiliza para crear un destino hacia un flujo de datos de Amazon Kinesis.

```
"Outputs": [
   {
       "DestinationSchema": {
           "RecordFormatType": "string"
       },
       "KinesisStreamsOutput": {
           "ResourceARN": "string",
           "RoleARN": "string"
       },
       "Name": "string"
   }

]
```

Creación de una salida de flujo de entrega de Firehose

El siguiente fragmento de JSON muestra la sección `Outputs` del cuerpo de la solicitud `CreateApplication` que se utiliza para crear un destino hacia un flujo de entrega de Amazon Data Firehose.

```
"Outputs": [
   {
       "DestinationSchema": {
           "RecordFormatType": "string"
       },
       "KinesisFirehoseOutput": {
           "ResourceARN": "string",
```

```
            "RoleARN": "string"
        },
        "Name": "string"
    }
]
```

Creación de una salida de la función de Lambda

El siguiente fragmento de JSON muestra la `Outputs` sección del cuerpo de la `CreateApplication` solicitud para crear un destino de AWS Lambda función.

```
"Outputs": [
    {
        "DestinationSchema": {
            "RecordFormatType": "string"
        },
        "LambdaOutput": {
            "ResourceARN": "string",
            "RoleARN": "string"
        },
        "Name": "string"
    }
]
```

Uso de una función de Lambda como salida

Si AWS Lambda se utiliza como destino, puede realizar más fácilmente el posprocesamiento de los resultados de SQL antes de enviarlos a un destino final. Entre las tareas de procesamiento posterior más frecuentes se incluyen las siguientes:

- Acumulación de varias filas en un solo registro
- Combinación de resultados actuales con resultados anteriores para procesar los datos que llegan tarde
- Efectuar la entrega en diferentes destinos en función del tipo de información
- Traducción del formato de los registros (por ejemplo, traducirlos a Protobuf)
- Manipulación o transformación de cadenas
- Enriquecimiento de datos después del procesamiento analítico
- Procesamiento personalizado para casos de uso geoespaciales

- Cifrado de datos

Las funciones Lambda pueden entregar información analítica a una variedad de AWS servicios y otros destinos, incluidos los siguientes:

- Amazon Simple Storage Service (Amazon S3)
- Personalizadas APIs
- Amazon DynamoDB
- Amazon Aurora
- Amazon Redshift
- Amazon Simple Notification Service (Amazon SNS)
- Amazon Simple Queue Service (Amazon SQS)
- Amazon CloudWatch

Para obtener más información acerca de cómo crear aplicaciones de Lambda, consulte Introducción a AWS Lambda.

Temas

- Permisos para utilizar Lambda como salida
- Métricas para utilizar Lambda como salida
- Modelo de datos de entrada de eventos y modelo de respuesta de registros para utilizar Lambda como salida
- Frecuencia de invocación de Lambda como salida
- Adición de una función de Lambda para su uso como salida
- Errores comunes de Lambda como salida
- Creación de funciones de Lambda como destinos de aplicaciones

Permisos para utilizar Lambda como salida

Para utilizar Lambda como salida, el rol de IAM de salida de Lambda de la aplicación requiere la política de permisos siguiente:

```
{
   "Sid": "UseLambdaFunction",
```

```
    "Effect": "Allow",
    "Action": [
        "lambda:InvokeFunction",
        "lambda:GetFunctionConfiguration"
    ],
    "Resource": "FunctionARN"
}
```

Métricas para utilizar Lambda como salida

Utiliza Amazon CloudWatch para monitorizar el número de bytes enviados, los éxitos y los errores, etc. Para obtener información sobre CloudWatch las métricas emitidas por Kinesis Data Analytics con Lambda como salida, consulte Amazon Kinesis Analytics Metrics.

Modelo de datos de entrada de eventos y modelo de respuesta de registros para utilizar Lambda como salida

Para enviar los registros de salida de Kinesis Data Analytics, la función de Lambda debe respetar los modelos de datos de entrada de eventos y de respuesta de registros.

Modelo de datos de entrada de eventos

Kinesis Data Analytics envía continuamente los registros de salida de la aplicación a la Lambda como salida con el siguiente modelo de solicitud. Dentro de la función, debe iterar por la lista y aplicar la lógica de negocio para cumplir los requisitos de salida (tales como la transformación de datos antes de enviarlos a su destino final).

Campo	Description (Descripción)
`invocationId`	El ID de invocación de Lambda (GUID aleatorio).
`applicationArn`	El nombre de recurso de Amazon (ARN) de la aplicación de análisis de datos de Kinesis Data Analytics.
registros	

Campo	Description (Descripción)
`recordId`	ID de registro (GUID aleatorio)

Campo	Description (Descripción)

Campo	Description (Descripción)
`lambdaDeliveryRecordMetadata`	

Campo	Description (Descripción)
`retryHint`	Número de reintentos de entrega

Campo	Description (Descripción)
data	Carga del registro de salida codificada en Base64

> **Note**
>
> `retryHint` es un valor que aumenta para cada error de entrega. Este valor no se conserva de forma duradera y se restablece si la aplicación se interrumpe.

Modelo de respuesta de registros

Cada registro enviado a su Lambda como función de salida (con registro IDs) debe confirmarse con una `Ok` o `DeliveryFailed` y debe contener los siguientes parámetros. De lo contrario, Kinesis Data Analytics lo considera un error de entrega.

registros

Campo	Description (Descripción)
`recordId`	El ID de registro se transfiere desde Kinesis Data Analytics hacia Lambda durante la invocación. Cualquier discrepancia entre el ID del registro original y el ID del registro reconocido se considera un error de entrega.

Campo	Description (Descripción)
`result`	El estado de entrega del registro. Los valores posibles son los siguientes: • `Ok`: el registro se ha transformado correctamente y se ha enviado a su destino final. Kinesis Data Analytics ingiere el registro para el procesamiento de SQL. • `DeliveryFailed` : el registro no se ha entregado correctamente a su destino final por la función de Lambda como salida. Kinesis Data Analytics vuelve a intentar continuamente el envío de los registros cuya entrega no se ha realizado a la función de Lambda como salida.

Frecuencia de invocación de Lambda como salida

Una aplicación de análisis de datos de Kinesis Data Analytics almacena en búfer los registros de salida e invoca con frecuencia la función de destino de AWS Lambda .

- Si los registros se emiten a la transmisión de destino dentro de la aplicación de análisis de datos como una ventana giratoria, la función de AWS Lambda destino se invoca por cada activador de la ventana oscilante. Por ejemplo, si se utiliza una ventana de saltos de 60 segundos para emitir los registros a la secuencia en la aplicación de destino, la función de Lambda se invoca una vez cada 60 segundos.
- Si los registros se emiten a la secuencia en la aplicación de destino dentro de la aplicación como ventana deslizante o de consulta continua, la función de destino de Lambda se invoca aproximadamente una vez por segundo.

> **Note**
>
> Se aplican los límites de tamaño de carga de solicitud de invocación para cada función de Lambda. Si se superan esos límites, los registros salientes se dividen y se envían a través de varias llamadas de funciones de Lambda.

Adición de una función de Lambda para su uso como salida

El siguiente procedimiento muestra cómo añadir una función de Lambda como salida para una aplicación de Kinesis Data Analytics.

1. Inicie sesión en la consola Managed Service for Apache Flink Consola de administración de AWS y ábrala en /kinesisanalytics. https://console.aws.amazon.com
2. Elija la aplicación en la lista y, a continuación, elija Application details.
3. En la sección Destination, elija Connect new destination.
4. En el elemento Destination, elija la función AWS Lambda .
5. En la sección Entregar registros a AWS Lambda, elija una función de Lambda existente o elija Crear nueva.
6. Si va a crear una función de Lambda nueva, haga lo siguiente:
   a. Elija una de las plantillas que se proporcionan. Para obtener más información, Creación de funciones de Lambda como destinos de aplicaciones.
   b. Se abre la página Create Function (Crear función) en una nueva pestaña del navegador web. En el cuadro Name (Nombre), indique un nombre significativo para la función (por ejemplo, **`myLambdaFunction`**).
   c. Actualice la plantilla con la funcionalidad de procesamiento posterior que necesite su aplicación. Para obtener más información acerca de la creación de una función de Lambda, consulte Introducción en la Guía para desarrolladores de AWS Lambda .
   d. En la consola de Kinesis Data Analytics, en la lista función de Lambda, elija la función de Lambda que acaba de crear. Elija $LATEST para la versión de la función de Lambda.
7. En la sección In-application stream, elija Choose an existing in-application stream. En In-application stream name, elija la secuencia de salida de la aplicación. Los resultados de la secuencia de salida seleccionada se envían a la función de salida de Lambda.
8. Deje el resto del formulario con los valores predeterminados, y elija Save and continue.

Ahora, la aplicación envía los registros de la secuencia en la aplicación a la función de Lambda. Puedes ver los resultados de la plantilla predeterminada en la CloudWatch consola de Amazon. Monitorice la métrica `AWS/KinesisAnalytics/LambdaDelivery.OkRecords` para ver el número de registros entregados a la función de Lambda.

Errores comunes de Lambda como salida

A continuación se indican los motivos comunes por los que puede no efectuarse la entrega a una función de Lambda.

- No todos los registros (con registro IDs) de un lote que se envían a la función Lambda se devuelven al servicio Kinesis Data Analytics.
- En la respuesta falta el campo de estado o el ID de registro.
- Los tiempos de espera de la función de Lambda no son suficientes para ejecutar la lógica de negocio de la función de Lambda.
- La lógica de negocio de la función de Lambda no captura todos los errores, lo que provoca tiempos de espera y resistencias que se deben a las excepciones no controladas. En ocasiones se denominan mensajes de “píldora venenosa”.

Para errores de entrega de datos, Kinesis Data Analytics sigue reintentando las invocaciones de Lambda en el mismo conjunto de registros hasta que tiene éxito. Para obtener información sobre los errores, puede supervisar las siguientes métricas: CloudWatch

- Lambda como métrica de CloudWatch salida de la aplicación Kinesis Data Analytics: indica el número de éxitos y fracasos, entre otras estadísticas. Para obtener más información, consulte Amazon Kinesis Analytics Metrics.
- AWS Lambda CloudWatch métricas y registros de funciones.

Creación de funciones de Lambda como destinos de aplicaciones

La aplicación Kinesis Data Analytics puede AWS Lambda utilizar funciones como salida. Kinesis Data Analytics proporciona plantillas para crear funciones de Lambda que se utilizan como destino de las aplicaciones. Utilice estas plantillas como punto de partida para el procesamiento posterior de la salida de las aplicaciones.

Temas

- Creación de una función de Lambda como destino en Node.js
- Creación de una función de Lambda como destino en Python
- Creación de una función de Lambda como destino en Java
- Creación de una función de Lambda como destino en .NET

Creación de una función de Lambda como destino en Node.js

La siguiente plantilla para crear una función de Lambda de destino en Node.js está disponible en la consola:

Esquema de Lambda como salida	Lenguaje y versión	Description (Descripción)
`kinesis-analytics-output`	Node.js 12.x	Envía los registros de salida de una aplicación de Kinesis Data Analytics a un destino personalizado.

Creación de una función de Lambda como destino en Python

Las siguientes plantillas para crear una función de Lambda de destino en Python están disponibles en la consola:

Esquema de Lambda como salida	Lenguaje y versión	Description (Descripción)
`kinesis-analytics-output-sns`	Python 2.7	Entregue registros de salida desde una aplicación de Kinesis Data Analytics a Amazon SNS.
`kinesis-analytics-output-ddb`	Python 2.7	Entregue registros de salida desde una aplicación de Kinesis Data Analytics a Amazon DynamoDB.

Creación de una función de Lambda como destino en Java

Para crear una función de Lambda como destino en Java, utilice las clases de eventos de Java.

El código siguiente ilustra un ejemplo de función de Lambda como destino creada en Java:

```
public class LambdaFunctionHandler
```

```
        implements RequestHandler<KinesisAnalyticsOutputDeliveryEvent,
 KinesisAnalyticsOutputDeliveryResponse> {

    @Override
    public KinesisAnalyticsOutputDeliveryResponse
 handleRequest(KinesisAnalyticsOutputDeliveryEvent event,
            Context context) {
        context.getLogger().log("InvocatonId is : " + event.invocationId);
        context.getLogger().log("ApplicationArn is : " + event.applicationArn);

        List<KinesisAnalyticsOutputDeliveryResponse.Record> records = new
 ArrayList<KinesisAnalyticsOutputDeliveryResponse.Record>();
        KinesisAnalyticsOutputDeliveryResponse response = new
 KinesisAnalyticsOutputDeliveryResponse(records);

        event.records.stream().forEach(record -> {
            context.getLogger().log("recordId is : " + record.recordId);
            context.getLogger().log("record retryHint is :" +
 record.lambdaDeliveryRecordMetadata.retryHint);
            // Add logic here to transform and send the record to final destination of
 your choice.
            response.records.add(new Record(record.recordId,
 KinesisAnalyticsOutputDeliveryResponse.Result.Ok));
        });
        return response;
    }

}
```

Creación de una función de Lambda como destino en .NET

Para crear una función de Lambda como destino en .NET, utilice las clases de eventos de .NET.

El código siguiente ilustra un ejemplo de función de Lambda como destino creada en C#:

```
public class Function
    {
        public KinesisAnalyticsOutputDeliveryResponse
 FunctionHandler(KinesisAnalyticsOutputDeliveryEvent evnt, ILambdaContext context)
        {
            context.Logger.LogLine($"InvocationId: {evnt.InvocationId}");
            context.Logger.LogLine($"ApplicationArn: {evnt.ApplicationArn}");
```

```
            var response = new KinesisAnalyticsOutputDeliveryResponse
            {
                Records = new List<KinesisAnalyticsOutputDeliveryResponse.Record>()
            };

            foreach (var record in evnt.Records)
            {
                context.Logger.LogLine($"\tRecordId: {record.RecordId}");
                context.Logger.LogLine($"\tRetryHint:
{record.RecordMetadata.RetryHint}");
                context.Logger.LogLine($"\tData: {record.DecodeData()}");

                // Add logic here to send to the record to final destination of your
choice.

                var deliveredRecord = new KinesisAnalyticsOutputDeliveryResponse.Record
                {
                    RecordId = record.RecordId,
                    Result = KinesisAnalyticsOutputDeliveryResponse.OK
                };
                response.Records.Add(deliveredRecord);
            }
            return response;
        }
    }
```

Para obtener más información sobre cómo crear funciones de Lambda para el procesamiento previo y como destino en .NET, consulte [Amazon.Lambda.KinesisAnalyticsEvents].

Modelo de entrega para conservar la salida de las aplicaciones en destinos externos

Amazon Kinesis Data Analytics utiliza un modelo de entrega de tipo "al menos una vez" para enviar la salida de las aplicaciones a los destinos configurados. Cuando una aplicación se está ejecutando, Kinesis Data Analytics tiene puntos de comprobación internos. Estos puntos de comprobación son momentos determinados en los que se han entregado registros de salida en los destinos sin pérdida de datos. El servicio utiliza los puntos de control necesarios para garantizar que la salida de la aplicación se entregue al menos una vez a los destinos configurados.

En una situación normal, la aplicación procesa los datos entrantes de forma continua. Kinesis Data Analytics graba el resultado en los destinos configurados, como un flujo de datos de Kinesis

o un flujo de entrega de Firehose. Sin embargo, es posible que la aplicación se interrumpa ocasionalmente; por ejemplo:

- Puede optar por detener la aplicación y reiniciarla más adelante.
- Puede eliminar el rol de IAM que Kinesis Data Analytics necesita para escribir la salida de la aplicación en el destino configurado. Sin el rol de IAM, Kinesis Data Analytics no tiene permisos para escribir en un destino externo.
- Una interrupción de la red o errores en otros servicios internos provocan que se detenga la ejecución de la aplicación momentáneamente.

Cuando la aplicación se reinicia, Kinesis Data Analytics se asegura de que continúa el procesamiento y escribe la salida desde un punto anterior o igual al momento en que se produjo el error. Esto ayuda a asegurarse de que no se pierde la entrega de ninguna salida de la aplicación a los destinos configurados.

Supongamos que ha configurado varios destinos desde la misma secuencia en la aplicación. Después de que la aplicación se recupere de un error, Kinesis Data Analytics reanuda la conservación de la salida en los destinos configurados desde el último registro que se entregó en el destino más lento. Esto podría resultar en el mismo registro de salida enviado más de una vez a otros destinos. En este caso, es obligatorio gestionar las posibles duplicidades en el destino de forma externa.

Gestión de errores

Amazon Kinesis Data Analytics le devuelve los errores de API o SQL directamente. Para obtener más información sobre las operaciones de API, consulte Acciones. Para obtener más información sobre el control de errores de SQL, consulte Referencia de SQL de Amazon Kinesis Data Analytics.

Amazon Kinesis Data Analytics informa de errores de tiempo de ejecución mediante una secuencia de errores en la aplicación denominada `error_stream`.

Informar de los errores mediante una secuencia de errores en la aplicación

Amazon Kinesis Data Analytics informa de errores de tiempo de ejecución mediante una secuencia de errores en la aplicación denominada `error_stream`. Los siguientes ejemplos muestran algunos errores que pueden ocurrir:

- Un registro leído desde el origen de streaming no cumple el esquema de entrada.
- El código de la aplicación especifica división por cero.
- Las filas están desordenadas (por ejemplo, un registro aparece en la secuencia con un valor `ROWTIME` que un usuario modificó y provoca hace que el registro esté desordenado).
- Los datos de la secuencia de origen no se pueden convertir al tipo de datos especificado en el esquema (Error de coerción). Para obtener información sobre qué tipos de datos se pueden convertir, consulte Mapeo de tipos de datos JSON para los tipos de datos SQL

Recomendamos que gestione estos errores mediante programación en el código SQL o que conserve los datos en la secuencia de errores en un destino externo. Esto requiere que añada una configuración de salida (consulte Configuración de salida de la aplicación) a la aplicación. Para ver un ejemplo sobre cómo funciona la secuencia de errores en la aplicación, consulte Ejemplo: Exploración de la secuencia de errores en la aplicación.

> **Note**
>
> La aplicación de Kinesis Data Analytics no puede obtener acceso mediante programación a la secuencia de errores, ni tampoco a modificarla, debido a que dicha secuencia se crea con la cuenta del sistema. Debe utilizar la salida de error para determinar los errores que pueda encontrar la aplicación. A continuación, escriba el código SQL de la aplicación para gestionar las condiciones de error previstas.

Esquema de la secuencia de errores

La secuencia de errores tiene el siguiente esquema:

Campo	Tipo de datos	Notas
ERROR_TIME	TIMESTAMP	El momento en el que se produjo el error
ERROR_LEVEL	VARCHAR (10)	
ERROR_NAME	VARCHAR (32)	
MESSAGE	VARCHAR (4096)	

DATA_ROWTIME	TIMESTAMP	El tiempo de la fila del registro entrante
DATA_ROW	VARCHAR (49152)	Los datos codificados en hexadecimal en la fila original. Puede utilizar bibliotecas estándar para decodificar de forma hexadecimal este valor, o utilizar recursos web como este convertidor de hexadecimal a cadena.
PUMP_NAME	VARCHAR (128)	La bomba de origen, tal y como se define con `CREATE PUMP`

Escalado automático de aplicaciones para incrementar el desempeño

Amazon Kinesis Data Analytics escala de manera elástica su aplicación para atender el rendimiento de datos de su flujo de origen y la complejidad de sus consultas en la mayoría de situaciones. Kinesis Data Analytics aprovisiona capacidad mediante unidades de procesamiento de Kinesis (KPU). Una única KPU le proporciona la memoria (4 GB), así como la informática y redes correspondientes.

El límite predeterminado para su aplicación es KPUs de 64. Para obtener instrucciones sobre cómo solicitar un aumento de este límite, consulte Solicitar un aumento de límite en Amazon Service Limits.

Uso del etiquetado

En esta sección, se describe cómo añadir etiquetas de metadatos de clave-valor a las aplicaciones de Kinesis Data Analytics. Estas etiquetas se pueden utilizar para lo siguiente:

- Determinar la facturación de las aplicaciones individuales de Kinesis Data Analytics. Para obtener más información, consulte Uso de etiquetas de asignación de costos en la Guía de administración de costos y facturación de AWS .

- Controlar el acceso a los recursos de la aplicación en función de las etiquetas. Para obtener más información, consulte Controlling Access Using Tags en la Guía del usuario de .
- Fines definidos por el usuario. Puede definir la funcionalidad de la aplicación en función de la presencia de etiquetas del usuario.

Tenga en cuenta la siguiente información sobre el etiquetado:

- El número máximo de etiquetas de la aplicación incluye las etiquetas del sistema. El número máximo de etiquetas de la aplicación definidas por el usuario es 50.
- Si una acción incluye una lista de etiquetas que tiene valores `Key` duplicados, el servicio genera una `InvalidArgumentException`.

Este tema contiene las siguientes secciones:

Adición de etiquetas al crear una aplicación

Las etiquetas se añaden al crear una aplicación mediante el `tags` parámetro de la CreateApplicationacción.

En la siguiente solicitud de ejemplo, se muestra el nodo `Tags` de una solicitud `CreateApplication`:

```
"Tags": [
    {
        "Key": "Key1",
        "Value": "Value1"
    },
    {
        "Key": "Key2",
        "Value": "Value2"
    }
]
```

Incorporación o actualización de etiquetas para una aplicación existente

Las etiquetas se añaden a una aplicación mediante la TagResourceacción. No puede añadir etiquetas a una aplicación mediante la UpdateApplicationacción.

Para actualizar una etiqueta existente, añada otra etiqueta con la misma clave.

En la siguiente solicitud de ejemplo de la acción `TagResource`, se añaden nuevas etiquetas o se actualizan las etiquetas existentes:

```
{
   "ResourceARN": "string",
   "Tags": [
      {
         "Key": "NewTagKey",
         "Value": "NewTagValue"
      },
      {
         "Key": "ExistingKeyOfTagToUpdate",
         "Value": "NewValueForExistingTag"
      }
   ]
}
```

Mostrar las etiquetas de una aplicación

Para enumerar las etiquetas existentes, utilice la ListTagsForResourceacción.

En la siguiente solicitud de ejemplo de la acción `ListTagsForResource`, se muestran las etiquetas de una aplicación:

```
{
   "ResourceARN": "arn:aws:kinesisanalytics:us-west-2:012345678901:application/
MyApplication"
}
```

Eliminación de etiquetas de una aplicación

Para eliminar etiquetas de una aplicación, utilice la UntagResourceacción.

En la siguiente solicitud de ejemplo de la acción `UntagResource`, se eliminan etiquetas de una aplicación:

```
{
   "ResourceARN": "arn:aws:kinesisanalytics:us-west-2:012345678901:application/
MyApplication",
   "TagKeys": [ "KeyOfFirstTagToRemove", "KeyOfSecondTagToRemove" ]
}
```

Introducción a aplicaciones de Amazon Kinesis Data Analytics para SQL

A continuación, encontrará algunos temas que le ayudarán a empezar a usar aplicaciones de Amazon Kinesis Data Analytics para SQL. Si es la primera vez que utiliza aplicaciones de Kinesis Data Analytics para SQL, le recomendamos que revise los conceptos y la terminología de Aplicaciones de Amazon Kinesis Data Analytics para SQL: cómo funciona antes de realizar los pasos de la sección Introducción.

Temas

Inscríbase en un Cuenta de AWS

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

Para suscribirte 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 o mensaje de texto 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, puede ver la actividad de su 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 Consola de administración de AWScomo 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. Active la autenticación multifactor (MFA) para el usuario raíz.

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

Creación de un usuario con acceso administrativo

1. Activar 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 su uso 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.

Inicio de sesión como usuario con acceso de administrador

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

  Para obtener ayuda para iniciar sesión con un usuario del Centro de identidades de IAM, 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 de 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 .

Paso 1: Configurar una cuenta de y crear un usuario administrador

Antes de usar Amazon Kinesis Data Analytics por primera vez, complete las siguientes tareas:

1. Inscríbase en AWS
2. Creación de un usuario de IAM

Inscríbase en AWS

Al suscribirse a Amazon Web Services, Cuenta de AWS se suscribe automáticamente a todos los servicios de Amazon Kinesis Data Analytics AWS, incluido Amazon Kinesis Data Analytics. Solo se le cobrará por los servicios que utilice.

Con Kinesis Data Analytics, paga solo por los recursos que usa. Si es un AWS cliente nuevo, puede empezar a utilizar Kinesis Data Analytics de forma gratuita. Para obtener más información, consulte Nivel de uso gratuito de AWS.

Si ya tiene una Cuenta de AWS, pase a la siguiente tarea. Si no tiene una Cuenta de AWS, lleve a cabo los pasos del siguiente procedimiento para crearla.

Para crear un Cuenta de AWS

1. Abre el 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 o mensaje de texto 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.

Anota tu Cuenta de AWS ID porque lo necesitarás para la siguiente tarea.

Creación de un usuario de IAM

Los servicios de AWS, como Amazon Kinesis Data Analytics, requieren que proporcione credenciales al acceder a ellos para que el servicio pueda determinar si tiene permisos para acceder a los recursos que son propiedad de ese servicio. La consola requiere que especifique la contraseña. Puede crear claves de acceso para acceder Cuenta de AWS a la API AWS CLI o a la API. Sin embargo, no te recomendamos que accedas AWS con las credenciales de tu Cuenta de AWS. En su lugar, le recomendamos que utilice AWS Identity and Access Management (IAM). Cree un usuario de IAM, añada el usuario a un grupo de IAM con permisos administrativos y, a continuación, conceda permisos administrativos al usuario de IAM que ha creado. A continuación, podrá acceder AWS mediante una URL especial y las credenciales de ese usuario de IAM.

Si te has registrado AWS, pero no has creado un usuario de IAM para ti, puedes crear uno mediante la consola de IAM.

En los ejercicios de introducción de esta guía se presupone que tiene un usuario (`adminuser`) con privilegios de administrador. Siga el procedimiento para crear `adminuser` en su cuenta.

Para crear un usuario administrador e iniciar sesión en la consola

1. Cree un usuario administrador llamado `adminuser` en su Cuenta de AWS. Para obtener instrucciones, consulte Creación del primer grupo de administradores y usuarios de IAM en la Guía del usuario de IAM.
2. Un usuario puede iniciar sesión en ella Consola de administración de AWS mediante una URL especial. Para obtener más información, consulte Cómo inician sesión los usuarios en su cuenta en la Guía del usuario de IAM.

Para obtener más información sobre IAM, consulte lo siguiente:

- AWS Identity and Access Management (IAM)
- Cómo empezar con IAM
- Guía del usuario de IAM

Paso siguiente

Paso 2: Configura el AWS Command Line Interface ()AWS CLI

Inscríbase en una Cuenta de AWS

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

Para suscribirte 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 o mensaje de texto 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, puede ver la actividad de su 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 Consola de administración de AWScomo 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. Active la autenticación multifactor (MFA) para el usuario raíz.

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

Creación de un usuario con acceso administrativo

1. Activar 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 su uso 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.

Inicio de sesión como usuario con acceso de administrador

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

  Para obtener ayuda para iniciar sesión con un usuario del Centro de identidades de IAM, 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 de 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 .

Paso 2: Configura el AWS Command Line Interface ()AWS CLI

Siga los pasos para descargar y configurar el AWS Command Line Interface (AWS CLI).

> **Important**
>
> No los necesita AWS CLI para realizar los pasos del ejercicio de introducción. Sin embargo, algunos de los ejercicios de esta guía utilizan la AWS CLI. Puede omitir este paso e ir aPaso 3: crear la aplicación inicial de Amazon Kinesis Data Analytics, y luego configurarlo AWS CLI más adelante cuando lo necesite.

Para configurar el AWS CLI

1. Descargue y configure la AWS CLI. Para obtener instrucciones, consulte los siguientes temas en la Guía del usuario de AWS Command Line Interface :

   - Cómo configurarse con el AWS Command Line Interface

- Configurando el AWS Command Line Interface

2. Agregue un perfil con nombre para el usuario administrador en el archivo de AWS CLI configuración. Este perfil se utiliza al ejecutar los AWS CLI comandos. Para obtener más información sobre los perfiles con nombre, consulte Perfiles con nombre en la Guía del usuario de la AWS Command Line Interface .

```
[profile adminuser]
aws_access_key_id = adminuser access key ID
aws_secret_access_key = adminuser secret access key
region = aws-region
```

   Para obtener una lista de las disponibles Regiones de AWS, consulte Regiones y puntos finales en la Referencia general de Amazon Web Services.

3. Verifique la configuración introduciendo el siguiente comando de ayuda en el símbolo del sistema:

```
aws help
```

Paso siguiente

Paso 3: crear la aplicación inicial de Amazon Kinesis Data Analytics

Paso 3: crear la aplicación inicial de Amazon Kinesis Data Analytics

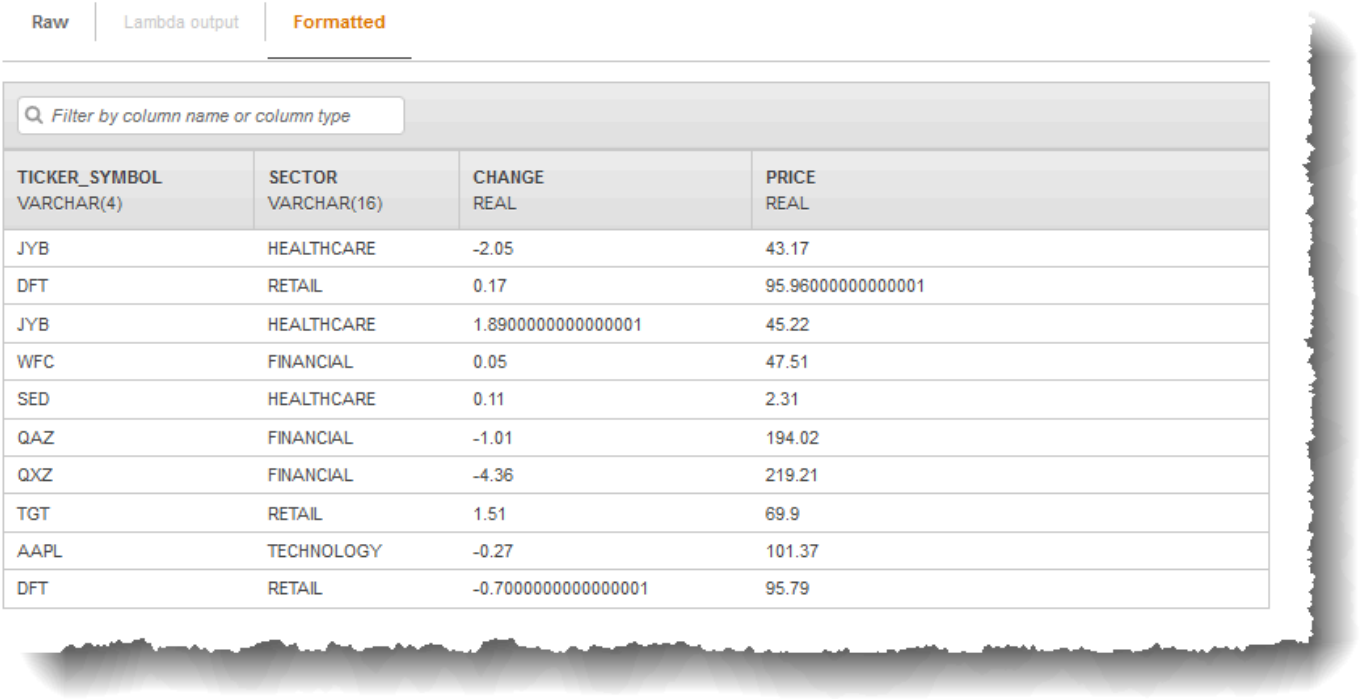
Mediante los pasos de esta sección, puede crear la primera aplicación de Kinesis Data Analytics utilizando la consola.

> **Note**
>
> Le recomendamos que consulte Aplicaciones de Amazon Kinesis Data Analytics para SQL: cómo funciona antes de probar el ejercicio de introducción.

En este ejercicio de introducción, puede utilizar la consola para trabajar con la secuencia de demostración o las plantillas con el código de la aplicación.

- Si decide utilizar la secuencia de demostración, la consola crea un flujo de datos de Kinesis en su cuenta denominada `kinesis-analytics-demo-stream`.

  Una aplicación de Kinesis Data Analytics requiere un origen de streaming. Para esta origen, varios ejemplos de SQL de esta guía utilizar la secuencia de demostración `kinesis-analytics-demo-stream`. La consola también ejecuta un script que añade continuamente datos de muestra (registros de operaciones bursátiles simuladas) a esta secuencia, tal y como se muestra a continuación.



TICKER_SYMBOL VARCHAR(4)	SECTOR VARCHAR(16)	CHANGE REAL	PRICE REAL
JYB	HEALTHCARE	-2.05	43.17
DFT	RETAIL	0.17	95.96000000000001
JYB	HEALTHCARE	1.8900000000000001	45.22
WFC	FINANCIAL	0.05	47.51
SED	HEALTHCARE	0.11	2.31
QAZ	FINANCIAL	-1.01	194.02
QXZ	FINANCIAL	-4.36	219.21
TGT	RETAIL	1.51	69.9
AAPL	TECHNOLOGY	-0.27	101.37
DFT	RETAIL	-0.7000000000000001	95.79

Puede utilizar `kinesis-analytics-demo-stream` como el origen de streaming para su aplicación en este ejercicio.

> **Note**
>
> La secuencia de demostración permanece en su cuenta. Puede utilizarla para probar otros ejemplos que aparecen en esta guía. Sin embargo, cuando salga de la consola, el script que utiliza la consola dejará de rellenar los datos. Cuando sea necesario, la consola proporcionará la opción de comenzar a rellenar la secuencia nuevamente.

- Si decide usar las plantillas con ejemplo de código de la aplicación, puede utilizar el código de plantilla que proporciona la consola para realizar análisis sencillos en la secuencia de demostración.

Puede utilizar estas características para configurar de manera rápida su primera aplicación de la siguiente manera:

1. Crear una aplicación: solo tiene que proporcionar un nombre. La consola crea la aplicación y el servicio establece el estado de la aplicación en `READY`.

2. Configurar la entrada: en primer lugar, debe añadir un origen de streaming, la secuencia de demostración. Debe crear una secuencia de demostración en la consola antes de poder utilizarla. A continuación, la consola toma una muestra al azar de los registros de la secuencia de demostración e infiere un esquema para la secuencia de entrada en la aplicación que se crea. Los consola asigna el nombre `SOURCE_SQL_STREAM_001` a la secuencia en la aplicación.

   La consola utiliza la API de detección para inferir el esquema. Si es necesario, puede editar el esquema inferido. Para obtener más información, consulte DiscoverInputSchema. Kinesis Data Analytics utiliza este esquema para crear una secuencia en la aplicación.

   Cuando se inicia la aplicación, Kinesis Data Analytics lee la secuencia de demostración de forma continua e introduce filas en la secuencia de entrada en la aplicación `SOURCE_SQL_STREAM_001`.

3. Especificar el código de la aplicación: puede utilizar una plantilla (llamada Continuous filter) que proporciona el siguiente código:

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM"
  (symbol VARCHAR(4), sector VARCHAR(12), CHANGE DOUBLE, price DOUBLE);

-- Create pump to insert into output.
CREATE OR REPLACE PUMP "STREAM_PUMP" AS
   INSERT INTO "DESTINATION_SQL_STREAM"
      SELECT STREAM ticker_symbol, sector, CHANGE, price
      FROM "SOURCE_SQL_STREAM_001"
      WHERE sector SIMILAR TO '%TECH%';
```

El código de la aplicación consulta la secuencia en la aplicación `SOURCE_SQL_STREAM_001`. El código inserta las filas resultantes en otra secuencia en la aplicación (`DESTINATION_SQL_STREAM`), utilizando bombas. Para obtener más información sobre este patrón de codificación, consulte Código de la aplicación.

Para obtener información sobre los elementos del lenguaje SQL compatibles con Kinesis Data Analytics, consulte Referencia de SQL de Amazon Kinesis Data Analytics.

4. Configurar la salida: en este ejercicio, no va a configurar salidas. Es decir, no va a preservar los datos en la secuencia en la aplicación que ha creado su aplicación en un destino externo. Lo que va a hacer es verificar los resultados de consulta en la consola. Existen más ejemplos en esta guía que ilustran de qué manera configurar la salida. Para ver un ejemplo, consulte Ejemplo: Creación de alertas simples.

> **Important**
>
> El ejercicio utiliza la Región Este de EE. UU. (Norte de Virginia) para configurar la aplicación. Puede utilizar cualquiera de los compatibles Regiones de AWS.

Paso siguiente

Paso 3.1: Cree una aplicación

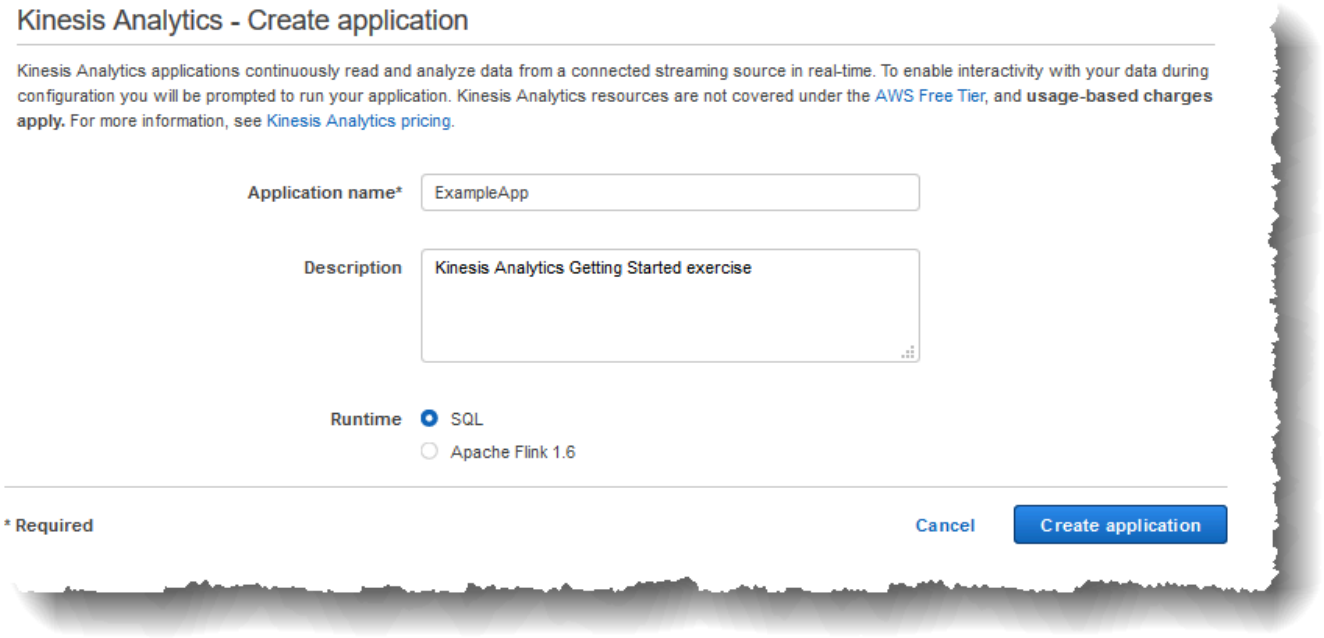
Paso 3.1: Cree una aplicación

En esta sección, va a crear una aplicación de análisis de datos de Amazon Kinesis Data Analytics. Puede configurar la entrada de la aplicación en el próximo paso.

Para crear una aplicación de análisis de datos

1. Inicie sesión en la consola Managed Service for Apache Flink Consola de administración de AWS y ábrala en https://console.aws.amazon.com/kinesisanalytics.
2. Elija Creación de aplicación.

3. En la página Create application (Crear aplicación), escriba un nombre de aplicación, escriba una descripción y elija SQL para la opción Runtime de la aplicación. A continuación, elija Create application (Crear aplicación).



Esto crea una aplicación de Kinesis Data Analytics con el estado READY. La consola muestra el centro de la aplicación donde puede configurar la entrada y la salida.

> **Note**
>
> Para crear una aplicación, la operación CreateApplication requiere únicamente el nombre de aplicación. Puede añadirla configuración de entrada y salida después de crear una aplicación en la consola.

En el paso siguiente, puede configurar la entrada para la aplicación. En la configuración de entrada, añade un origen de datos de streaming a la aplicación y descubre un esquema para una secuencia de entrada en la aplicación de los datos de muestra en el origen de streaming.

Paso siguiente

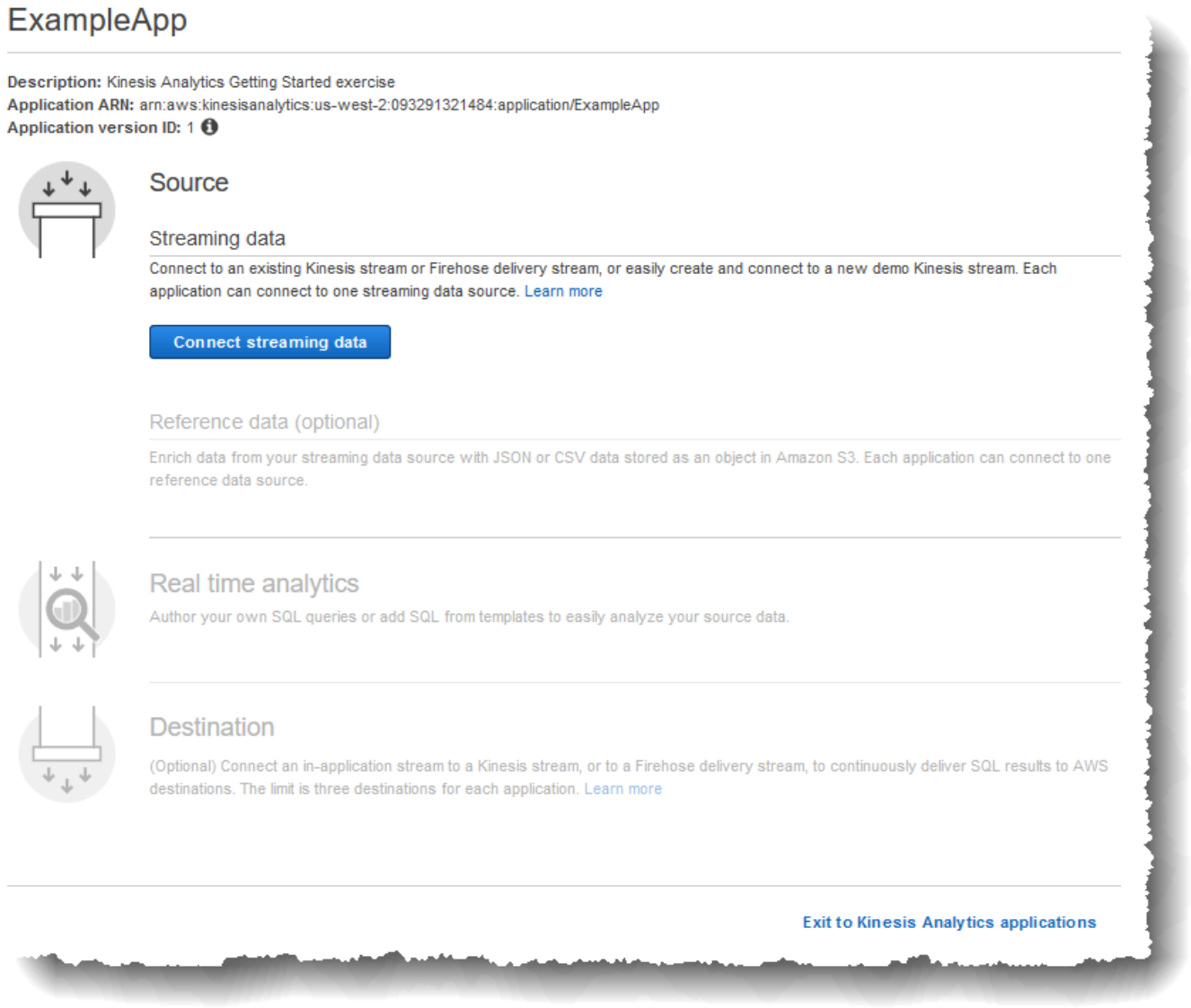
Paso 3.2: Configurar la entrada

Paso 3.2: Configurar la entrada

La aplicación necesita un origen de streaming. Para ayudarle a empezar, la consola puede crear una secuencia de demostración (denominada `kinesis-analytics-demo-stream`). La consola también ejecuta un script que rellena los registros en la secuencia.

Para añadir un origen de streaming a su aplicación

1. En el página del centro de aplicaciones en la consola, elija Connect streaming data (Conectar datos de streaming).



2. En la página que aparece, revise lo siguiente:

- Source en la que especifica un origen de streaming para la aplicación. Puede seleccionar una fuente de secuencia o crear una. En este ejercicio, deberá crear una nueva secuencia: la secuencia de demostración.

  De forma predeterminada, la consola nombra la secuencia de entrada en la aplicación que se crea como `INPUT_SQL_STREAM_001`. Para este ejercicio, mantenga este nombre tal como aparece.

- Nombre de referencia de secuencia: esta opción muestra el nombre de la secuencia de entrada en la aplicación, `SOURCE_SQL_STREAM_001`, que se crea. Puede cambiar el nombre, pero para este ejercicio mantenga este nombre.

  En la configuración de entrada, asigna la secuencia de demostración a una secuencia de entrada en la aplicación que se crea. Cuando inicia la aplicación, Amazon Kinesis Data Analytics lee la secuencia de demostración de forma continua e introduce filas en la secuencia de entrada en la aplicación. Usted consulta esta secuencia de entrada en la aplicación en el código de la aplicación.

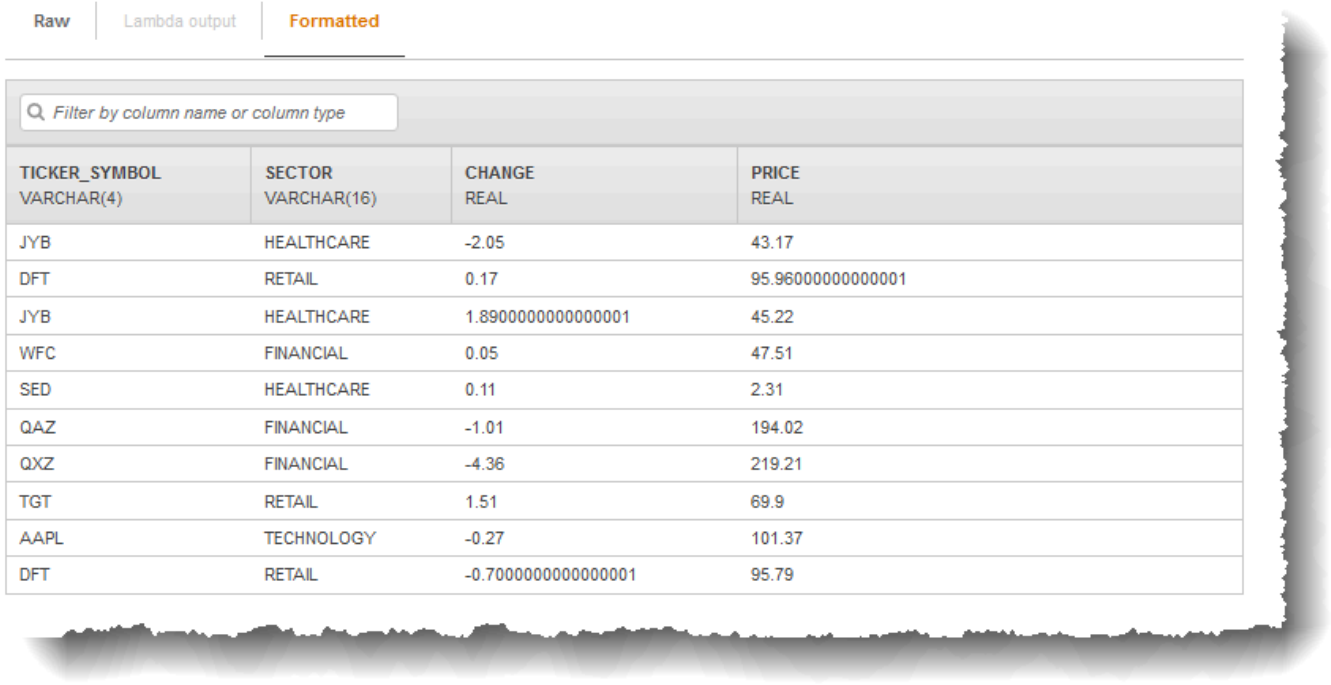
- Preprocesamiento de registros con AWS Lambda: esta opción permite especificar una AWS Lambda expresión que modifica los registros del flujo de entrada antes de que se ejecute el código de la aplicación. En este ejercicio, deje la opción Disabled seleccionada. Para obtener más información acerca del procesamiento previo de Lambda, consulte Procesamiento previo de registros con una función de Lambda.

Después de proporcionar toda la información de esta página, la consola envía una solicitud de actualización (consulte UpdateApplication) para añadirla configuración de entrada de la aplicación.

3. En la página Source, elija Configure a new stream.
4. Elija Create demo stream. La consola configura la entrada de la aplicación realizando las siguientes operaciones:

- La consola crea un flujo de datos de Kinesis denominada `kinesis-analytics-demo-stream`.
- La consola rellena la secuencia con datos de muestra de cotizaciones bursátiles.
- Mediante la acción de entrada DiscoverInputSchema, la consola infiere un esquema leyendo los registros de ejemplo de la secuencia. El esquema que se infiere es el esquema de la secuencia de entrada en la aplicación que se crea. Para obtener más información, consulte Configuración de entrada de la aplicación.
- La consola muestra el esquema inferido y los datos de muestra que lee desde el origen de streaming a fin de inferir el esquema.

La consola exhibe los registros de muestra en el origen de streaming.



TICKER_SYMBOL VARCHAR(4)	SECTOR VARCHAR(16)	CHANGE REAL	PRICE REAL
JYB	HEALTHCARE	-2.05	43.17
DFT	RETAIL	0.17	95.96000000000001
JYB	HEALTHCARE	1.8900000000000001	45.22
WFC	FINANCIAL	0.05	47.51
SED	HEALTHCARE	0.11	2.31
QAZ	FINANCIAL	-1.01	194.02
QXZ	FINANCIAL	-4.36	219.21
TGT	RETAIL	1.51	69.9
AAPL	TECHNOLOGY	-0.27	101.37
DFT	RETAIL	-0.7000000000000001	95.79

Esto es lo que se muestra en la página Stream sample de la consola:

- La pestaña Raw stream sample muestra los registros de secuencias sin procesar muestreados mediante la acción de la API DiscoverInputSchema para inferir el esquema.
- La pestaña Formatted stream sample exhibe la versión tabular de los datos en la pestaña Raw stream sample.

- Si elige Edit schema, puede editar el esquema inferido. Para este ejercicio, no cambie el esquema inferido. Para obtener más información sobre la edición de un esquema, consulte Uso del editor de esquemas.

  Si elige Rediscover schema, puede solicitar a la consola que ejecute de nuevo DiscoverInputSchema e infiera el esquema.

5. Elija Guardar y continuar.

   Ahora tiene una aplicación con la configuración de entrada incorporada. En el siguiente paso, añada el código SQL para realizar un análisis de los datos en la secuencia de entrada en la aplicación.

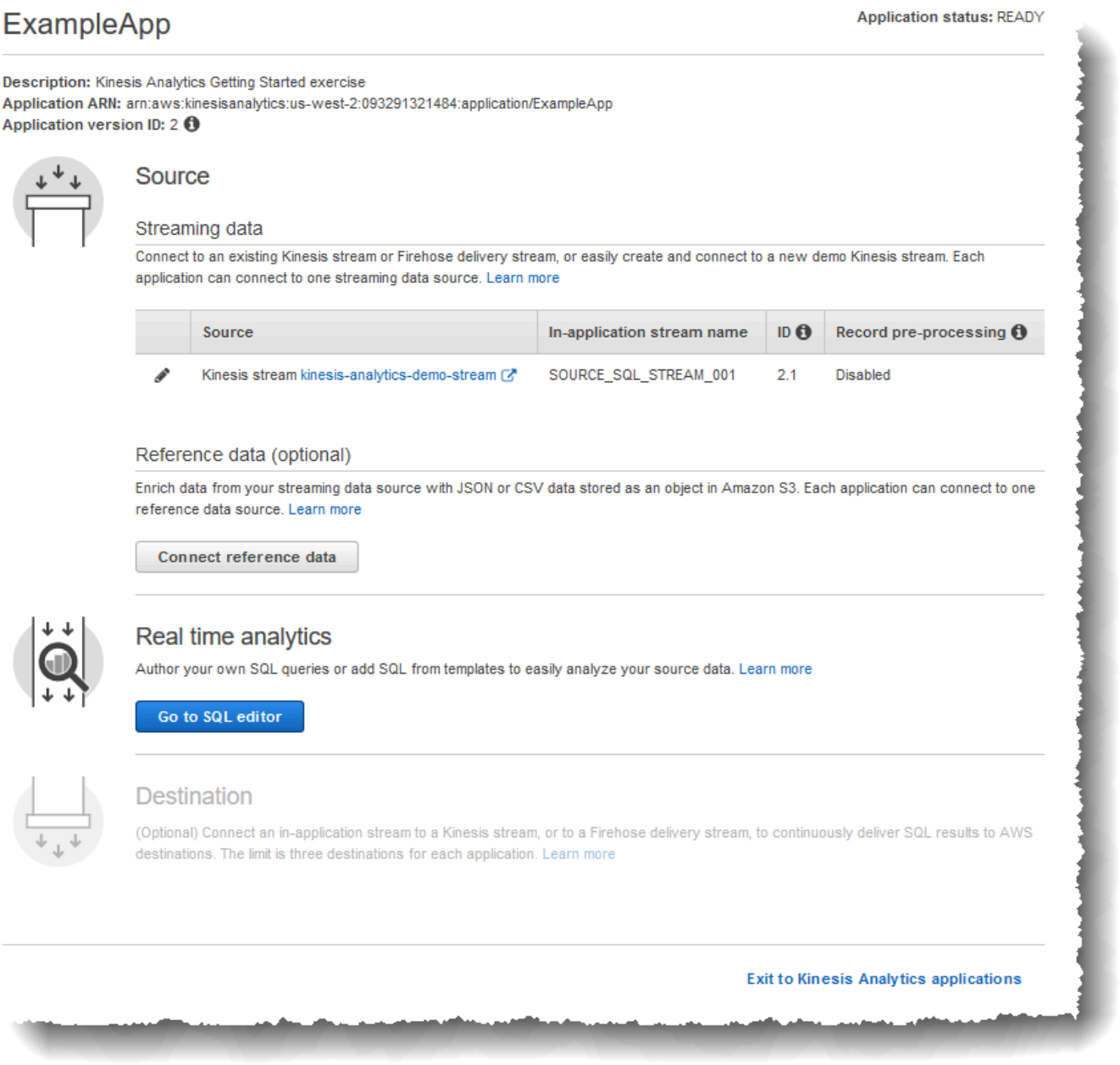
Paso siguiente

Paso 3.3: Añadir análisis en tiempo real (añadir el código de la aplicación)

Paso 3.3: Añadir análisis en tiempo real (añadir el código de la aplicación)

Puede escribir sus propias consultas SQL para la secuencia en la aplicación, pero para el paso siguiente utilice una de las plantillas que proporciona el código de muestra.

1. En la página del centro de la aplicación, elija Go to SQL editor.



2. En el cuadro ¿Desea empezar a ejecutar "«? ExampleApp cuadro de diálogo, seleccione Sí, iniciar la aplicación.

   La consola envía una solicitud para comenzar la aplicación (consulte StartApplication) y luego aparece la página de editor de SQL.

3. La consola abre la página de editor de SQL. Revise la página, incluso los botones (Add SQL from templates, Save and run SQL) y las distintas pestañas.
4. En el editor de SQL, elija Add SQL from templates.

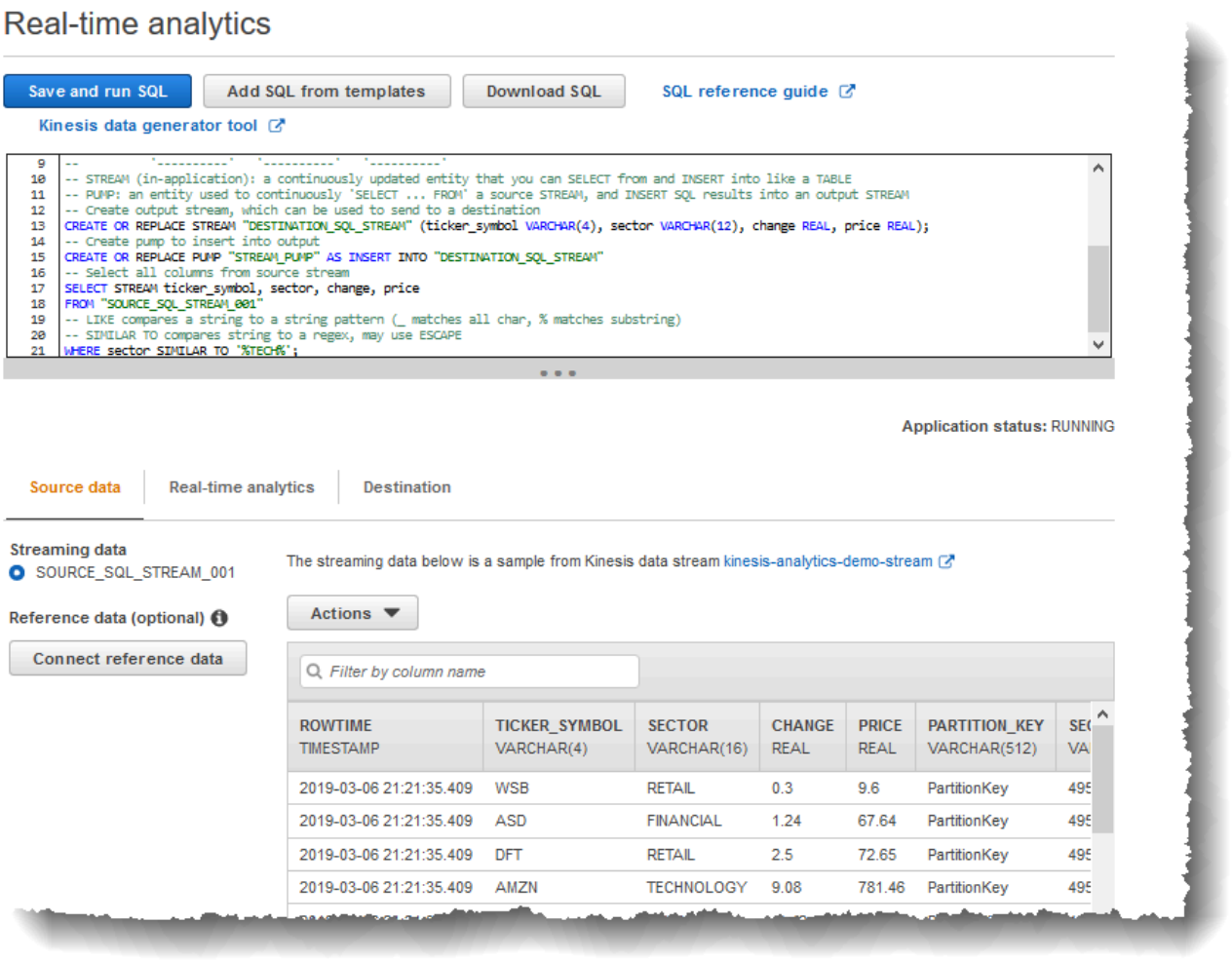
5. En la lista de plantillas disponibles, elija Continuous filter. El código de muestra lee datos de una secuencia en la aplicación (la cláusula `WHERE` filtra las filas) y los introduce en otra secuencia en la aplicación de la siguiente manera:

   - Crea la secuencia en la aplicación `DESTINATION_SQL_STREAM`.
   - Crea una bomba `STREAM_PUMP` y la utiliza para seleccionar filas de `SOURCE_SQL_STREAM_001` e insertarlas en el `DESTINATION_SQL_STREAM`.

6. Elija Add this SQL to editor.
7. Pruebe el código de la aplicación de la siguiente manera:

   Recuerde que ya ha comenzado la aplicación (el estado es RUNNING). Por lo tanto, Amazon Kinesis Data Analytics ya está leyendo de manera continua desde el origen de streaming y añadiendo filas a la secuencia en la aplicación `SOURCE_SQL_STREAM_001`.

   a. En el Editor de SQL, elija Save and run SQL. La consola primero envía la solicitud de actualización para guardar el código de la aplicación. Luego el código se ejecuta de forma continua.
   b. Puede ver los resultados en la pestaña Real-time analytics.



El editor de SQL tiene las siguientes pestañas:

- La pestaña Source data muestra una secuencia de entrada en la aplicación que se asigna al origen de streaming. Elija la secuencia en la aplicación y podrá ver los datos entrantes. Observe las columnas adicionales de la secuencia de entrada en la aplicación que no se especificaron en la configuración de entrada y que Incluyen las siguientes columnas de marca de tiempo:

  - ROWTIME: cada fila en una secuencia en la aplicación tiene una columna especial denominada `ROWTIME`. Esta columna es la marca de tiempo cuando Amazon Kinesis Data Analytics inserta la fila en la primera secuencia en la aplicación (la secuencia de entrada en la aplicación que se asigna al origen de streaming).

- Approximate_Arrival_Time: cada registro de Kinesis Data Analytics incluye un valor denominado `Approximate_Arrival_Time`. Este valor es la marca de tiempo aproximado de llegada que se establece cuando el origen de streaming recibe y almacena correctamente el registro. Cuando Kinesis Data Analytics lee registros de un origen de streaming, incluye esta columna en la secuencia de entrada en la aplicación.

Estos valores de marca de tiempo resultan útiles en consultas de ventanas basadas en el tiempo. Para obtener más información, consulte Consultas en ventana.

- La pestaña Real-time analytics muestra todas las demás secuencias en la aplicación que crea el código de la aplicación. También incluye la secuencia de errores. Kinesis Data Analytics envía las filas que no puede procesar a la secuencia de errores. Para obtener más información, consulte Gestión de errores.

  Elija `DESTINATION_SQL_STREAM` para ver las filas que el código de su aplicación ha insertado. Observe las columnas adicionales que el código de la aplicación no ha creado. Estas columnas incluyen las columnas de marca de tiempo `ROWTIME`. Kinesis Data Analytics simplemente copia estos valores de la fuente (`SOURCE_SQL_STREAM_001`).

- La pestaña Destino muestra el destino externo en el que Kinesis Data Analytics escribe los resultados de la consulta. Aún no ha configurado ningún destino externo para la salida de la aplicación.

Paso siguiente

Paso 3.4 (opcional): actualizar el código de la aplicación

Paso 3.4 (opcional): actualizar el código de la aplicación

En este paso, explore cómo actualizar el código de la aplicación.

Para actualizar el código de la aplicación

1. Cree otra secuencia en la aplicación del siguiente modo:

- Cree otra secuencia en la aplicación llamada DESTINATION_SQL_STREAM_2.
- Cree una bomba y luego utilícela para insertar filas en la secuencia recién creada seleccionando filas de DESTINATION_SQL_STREAM.

En el editor de SQL, añada este código al código de la aplicación:

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM_2"
           (ticker_symbol VARCHAR(4),
            change        DOUBLE,
            price         DOUBLE);

CREATE OR REPLACE PUMP "STREAM_PUMP_2" AS
   INSERT INTO "DESTINATION_SQL_STREAM_2"
      SELECT STREAM ticker_symbol, change, price
      FROM   "DESTINATION_SQL_STREAM";
```

Guarde y ejecute el código. Otros flujos en la aplicación aparecerán en la pestaña Real-time analytics.

2. Cree dos secuencias en la aplicación. Filtre las filas de SOURCE_SQL_STREAM_001 basándose en el símbolo de la cotización y luego insértelas en estas secuencias independientes.

Añada las siguientes instrucciones SQL al código de la aplicación:

```
CREATE OR REPLACE STREAM "AMZN_STREAM"
           (ticker_symbol VARCHAR(4),
            change        DOUBLE,
            price         DOUBLE);

CREATE OR REPLACE PUMP "AMZN_PUMP" AS
   INSERT INTO "AMZN_STREAM"
      SELECT STREAM ticker_symbol, change, price
      FROM   "SOURCE_SQL_STREAM_001"
      WHERE  ticker_symbol SIMILAR TO '%AMZN%';

CREATE OR REPLACE STREAM "TGT_STREAM"
           (ticker_symbol VARCHAR(4),
            change        DOUBLE,
            price         DOUBLE);

CREATE OR REPLACE PUMP "TGT_PUMP" AS
```

```
INSERT INTO "TGT_STREAM"
   SELECT STREAM ticker_symbol, change, price
   FROM   "SOURCE_SQL_STREAM_001"
   WHERE  ticker_symbol SIMILAR TO '%TGT%';
```

Guarde y ejecute el código. Tenga en cuenta las secuencias en la aplicación adicionales en la pestaña Real-time analytics.

Ya tiene la primera aplicación de análisis de datos de Amazon Kinesis Data Analytics en funcionamiento. En este ejercicio ha hecho lo siguiente:

- Ha creado su primera aplicación de análisis de datos de Kinesis Data Analytics.

- Ha configurado la entrada de la aplicación que identificaba la secuencia de demostración como el origen de streaming y se la ha asignado a una secuencia en la aplicación (`SOURCE_SQL_STREAM_001`) que se crea. Kinesis Data Analytics lee de manera continua la secuencia de demostración e introduce los registros en la secuencia en la aplicación.

- El código de la aplicación solicitó el `SOURCE_SQL_STREAM_001` y escribió la salida en otra secuencia en la aplicación denominada `DESTINATION_SQL_STREAM`.

Ahora tiene la opción de configurar la salida de la aplicación a fin de escribir la salida de la aplicación en un destino externo. Es decir, puede configurar la salida de la aplicación para escribir registros de `DESTINATION_SQL_STREAM` en un destino externo. Para este ejercicio, este paso es opcional. Para obtener información sobre cómo configurar el destino, vaya al siguiente paso.

Paso siguiente

Paso 4 (opcional): Editar el esquema y el código SQL utilizando la consola.

Paso 4 (opcional): Editar el esquema y el código SQL utilizando la consola

A continuación, puede encontrar información acerca de cómo editar un esquema inferido y cómo editar código SQL para Amazon Kinesis Data Analytics . Para ello, utilice el editor de esquemas y el editor de SQL incluidos en la consola de Kinesis Data Analytics.

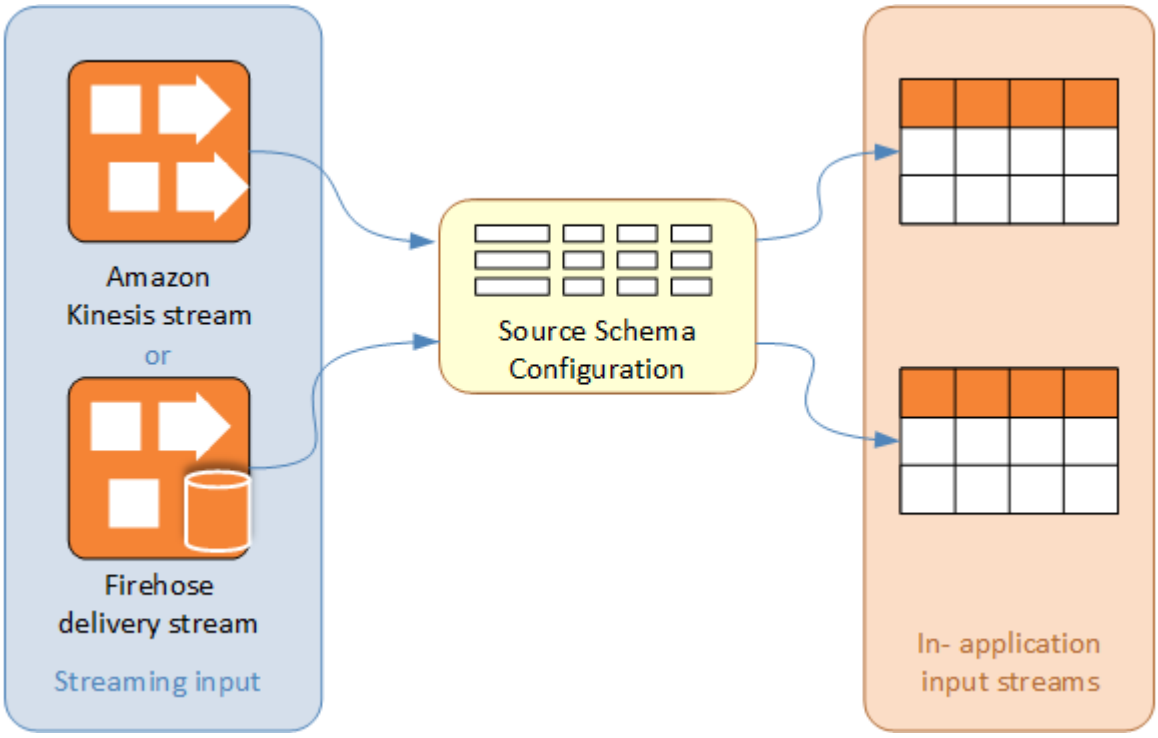
> **Note**
>
> Para acceder a los datos de la consola o probarlos, el rol del usuario de inicio de sesión debe tener el permiso `kinesisanalytics:GetApplicationState`. Para obtener más información sobre los permisos de aplicación de Kinesis Data Analytics, consulte Información general sobre la administración del acceso.

Temas

- Uso del editor de esquemas
- Trabajar con el editor de SQL

Uso del editor de esquemas

El esquema de la secuencia de entrada de una aplicación de Amazon Kinesis Data Analytics define la forma en que los datos de la secuencia están disponibles para las consultas de SQL de la aplicación.



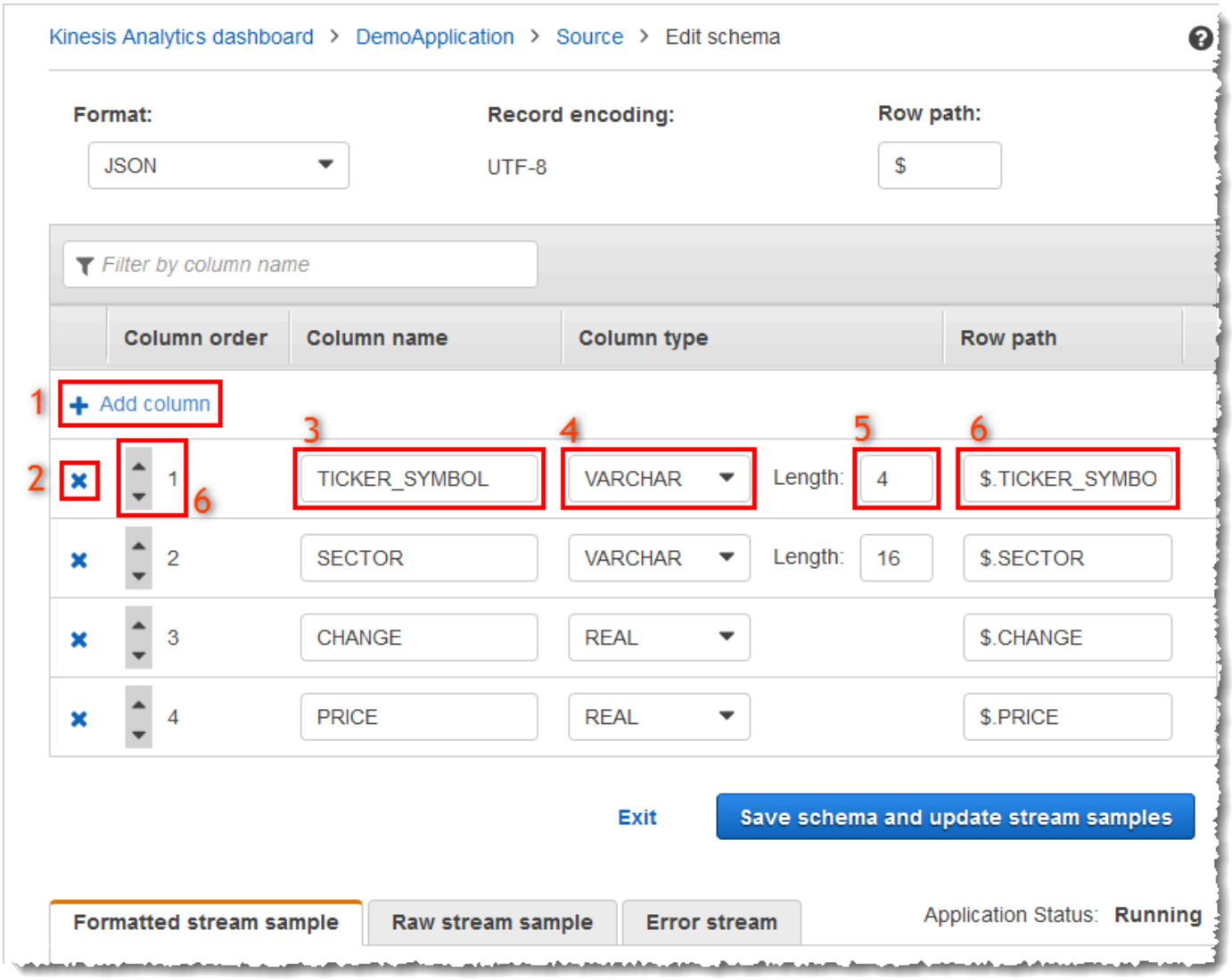
El esquema contiene criterios de selección para determinar qué parte de la entrada de streaming se transforma en una columna de datos en la secuencia de entrada en la aplicación. Esta entrada puede ser una de las siguientes:

- Una JSONPath expresión para los flujos de entrada de JSON. JSONPath es una herramienta para consultar datos de JSON.
- Un número de columna para secuencias de entrada con formato de valores separados por comas (CSV).
- Un nombre de columna y un tipo de datos SQL para presentar los datos en el flujo de datos en la aplicación. El tipo de datos también contiene un máximo de caracteres o datos binarios.

La consola intenta generar el esquema utilizando DiscoverInputSchema. Si la detección de esquema produce un error o devuelve un esquema incorrecto o incompleto, debe editar el esquema manualmente mediante el editor de esquemas.

Pantalla principal del editor de esquemas

En la siguiente captura de pantalla se muestra la pantalla principal del editor de esquemas.



Puede aplicar las siguientes modificaciones al esquema:

- Añadir una columna (1): es posible que necesite añadir una columna de datos si no se detecta un elemento de datos automáticamente.
- Eliminar una columna (2): puede excluir los datos de la secuencia de origen si la aplicación no los requiere. Esta exclusión no afecta los datos de la secuencia de origen. Si se excluyen los datos, esos datos simplemente no se ponen a disposición de la aplicación.
- Cambiar el nombre a una columna (3). El nombre de una columna no puede quedar vacío, debe tener más de un carácter y no debe contener palabras claves reservadas de SQL. El nombre también debe cumplir los criterios de nomenclatura de los identificadores normales de SQL: el nombre debe comenzar con una letra y solo puede contener letras, números y guiones bajos.

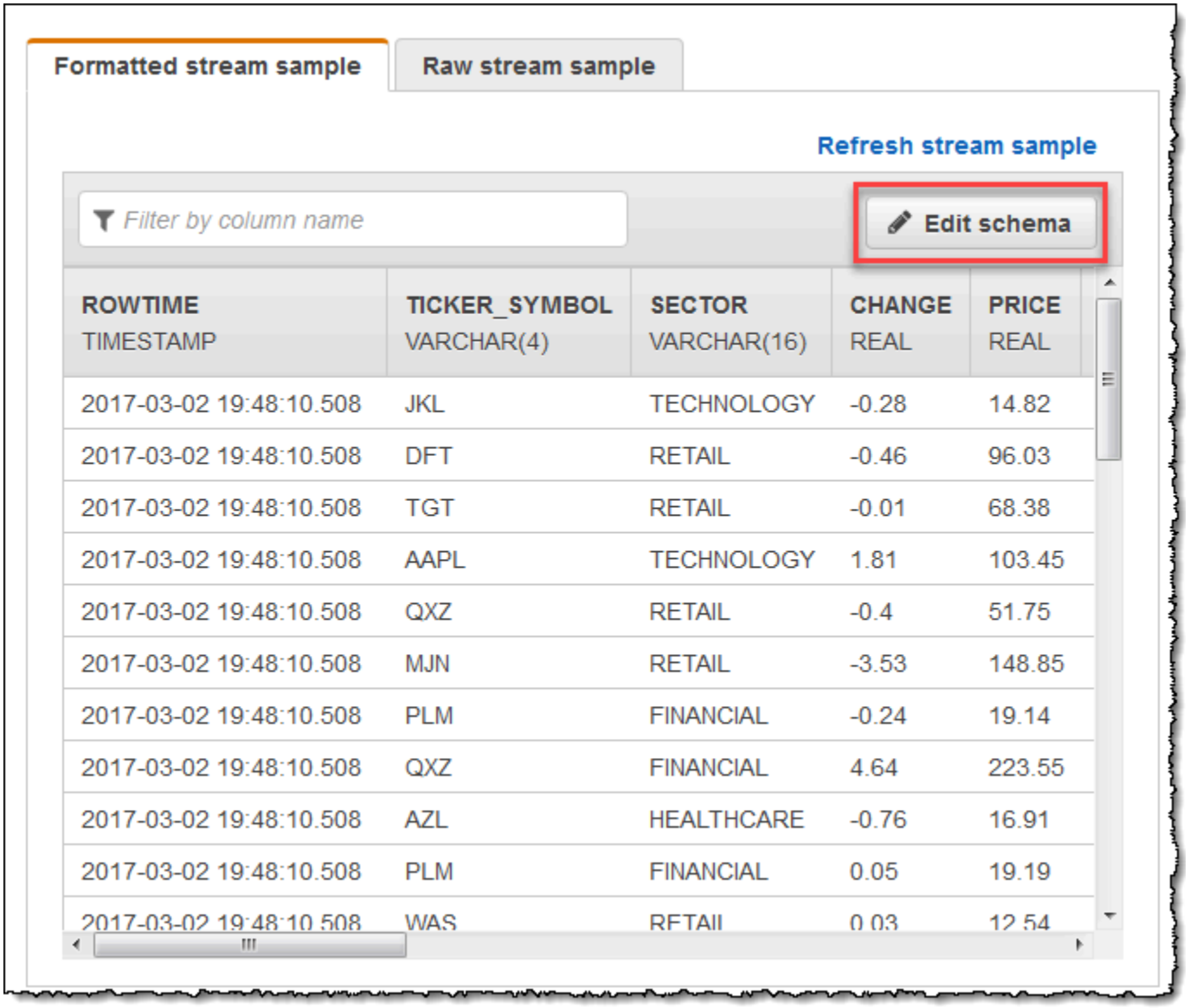
- Cambiar el tipo de datos (4) o la longitud (5) de una columna: puede especificar un tipo de datos compatibles para una columna. Si especifica un tipo de datos incompatible, la columna se rellena con valores NULL o no se rellena de ningún modo la secuencia en la aplicación. En este último caso, los errores están escritos en la secuencia de errores. Si especifica una longitud para una columna que es demasiado pequeña, los datos entrantes están truncados.
- Cambiar los criterios de selección de una columna (6): puede editar la JSONPath expresión o el orden de las columnas del CSV utilizado para determinar el origen de los datos de una columna. Para cambiar los criterios de selección para un esquema JSON, introduzca un nuevo valor para la expresión de la ruta de la fila. Un esquema CSV utiliza el orden de la columna como criterios de selección. Para cambiar los criterios de selección para un esquema CSV, cambie el orden de las columnas.

Edición del esquema para un origen de streaming

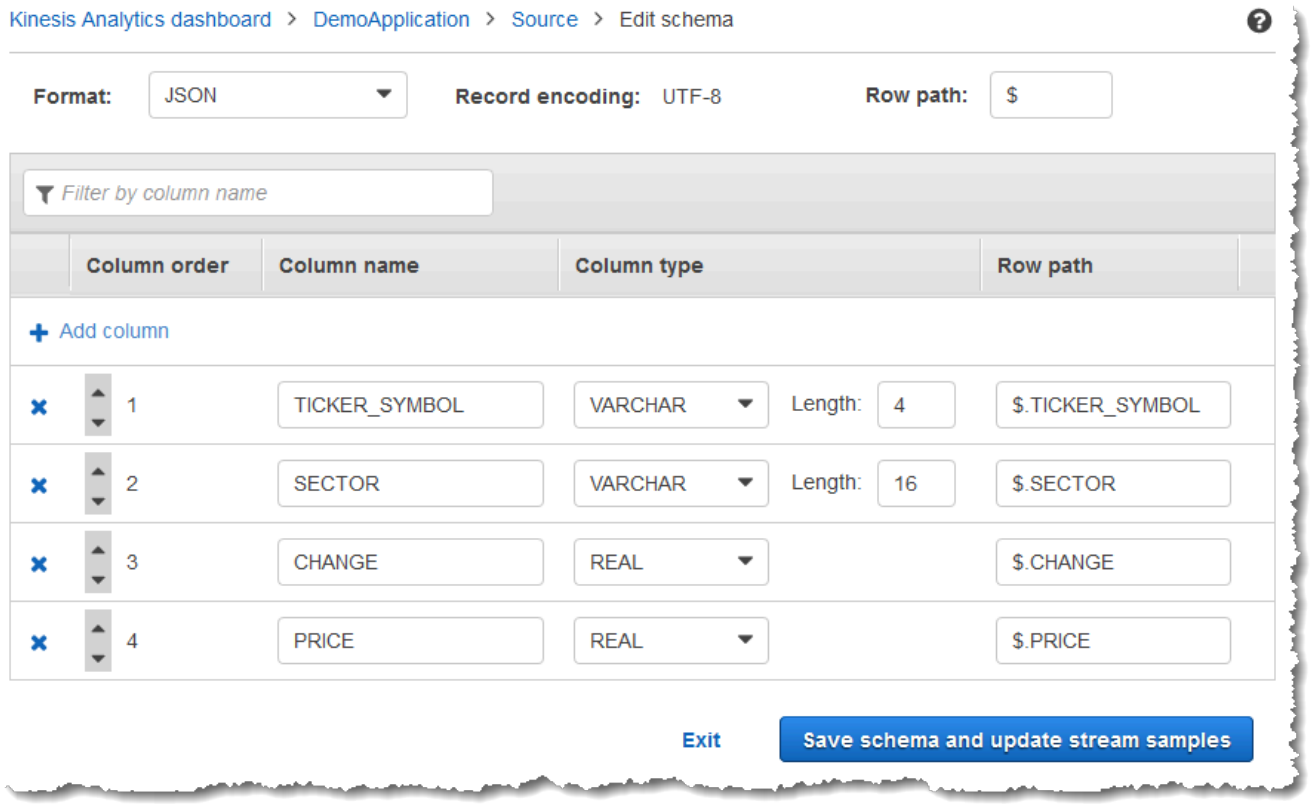
Si necesita editar un esquema para un origen de streaming, siga estos pasos.

Editar el esquema para un origen de streaming

1. En la página Source, elija Edit schema.



2. En la página Edit schema, edite el esquema de la fuente.



3. En Format (Formato), elija JSON o CSV. Para el formato JSON o CSV, se admite la codificación ISO 8859-1.

Para obtener más información sobre la edición del esquema para el formato JSON o CSV, consulte los procedimientos en las próximas secciones.

Edición de esquemas JSON

Puede editar un esquema JSON siguiendo los siguientes pasos.

Para editar un esquema JSON

1. En el editor de esquemas, elija Add column para añadir una columna.

   Una nueva columna aparece en la posición de la primera columna. Para cambiar el orden de una columna, elija las flechas hacia arriba y abajo que se encuentran junto al nombre de la columna.

   Para obtener una nueva columna, proporcione la siguiente información:

- En Column name, escriba un nombre.

  El nombre de una columna no puede quedar vacío, debe tener más de un carácter y no debe contener palabras claves reservadas de SQL. También debe cumplir con los criterios de identificadores normales de SQL: debe comenzar con una letra y solo puede contener letras, números y guiones bajos.

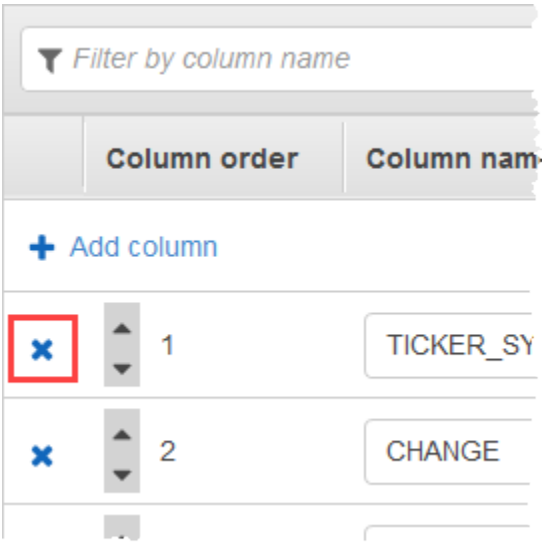
- En Column type, escriba un tipo de datos SQL.

  Un tipo de columna puede ser cualquier tipo de datos SQL compatible. Si el nuevo tipo de datos es CHAR, VARBINARY o VARCHAR, especifique una longitud en Length. Para obtener más información, consulte Data Types.

- En Row path, proporcione una ruta de la fila. Una ruta de fila es una JSONPath expresión válida que se asigna a un elemento JSON.

  > **Note**
  >
  > El valor base de Row path es la ruta del origen de nivel superior que contiene los datos que se deben importar. Este valor es $ de forma predeterminada. Para obtener más información, consulta `RecordRowPath` en `JSONMappingParameters`.

2. Para eliminar una columna, elija el icono x situado junto al número de la columna.

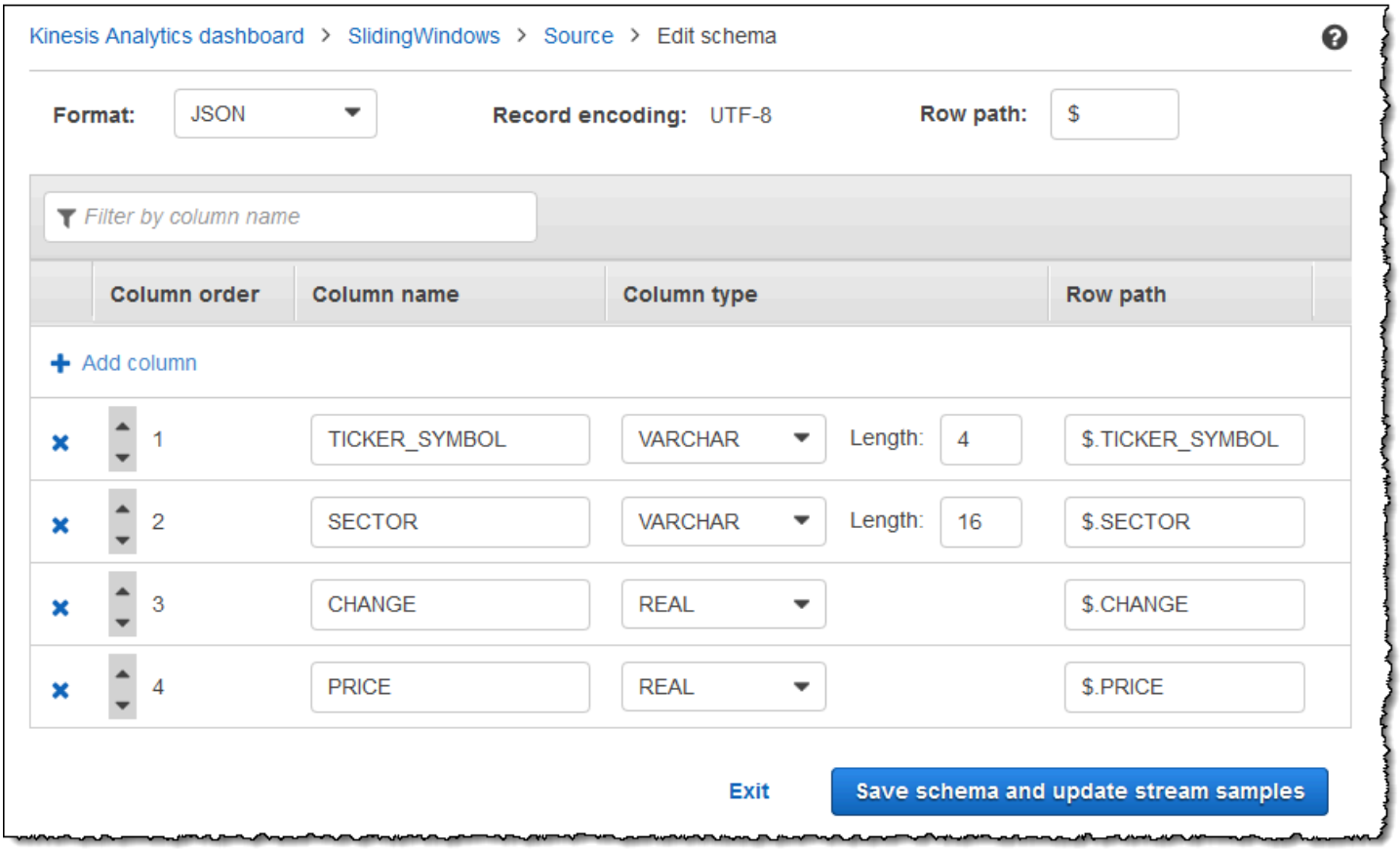


3. Para cambiar el nombre de una columna, introduzca un nuevo nombre en Column name (Nombre de columna). El nuevo nombre de una columna no puede quedar vacío, debe tener más de un carácter y no debe contener palabras claves reservadas de SQL. También debe

cumplir con los criterios de identificadores normales de SQL: debe comenzar con una letra y solo puede contener letras, números y guiones bajos.

4. Para cambiar el tipo de datos de una columna, elija un tipo de datos nuevo en Column type. Si el nuevo tipo de datos es `CHAR`, `VARBINARY` o `VARCHAR`, especifique una longitud en Length (Longitud). Para obtener más información, consulte Data Types.
5. Elija Save schema and update stream para guardar los cambios.

El esquema modificado aparece en el editor y será similar al siguiente.



Si su esquema tiene varias filas, puede filtrarlas usando Filter by column name. Por ejemplo, para editar los nombres de una columna que empieza con `P`, como la columna `Price`, introduzca `P` en la casilla Filtrar por nombre de columna.

Edición de esquemas CSV

Para editar un esquema CSV siga los pasos que se indican a continuación.

Para editar un esquema CSV

1. En el editor de esquemas, en Row delimiter, elija el delimitador utilizado por el flujo de datos entrantes. Este es el delimitador entre los registros de datos en su secuencia, como, por ejemplo, un carácter de nueva línea.
2. En Column delimiter, elija el delimitador utilizado por el flujo de datos entrantes. Este es el delimitador entre los campos de datos en su secuencia, como, por ejemplo, una coma.
3. Para añadir una columna, elija Add column.

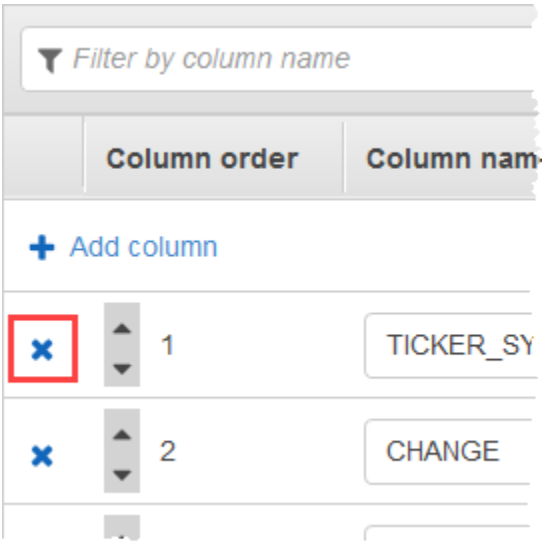
   Una nueva columna aparece en la posición de la primera columna. Para cambiar el orden de una columna, elija las flechas hacia arriba y abajo que se encuentran junto al nombre de la columna.

   Para obtener una nueva columna, proporcione la siguiente información:

   - En Column name (Nombre de columna), introduzca un nombre.

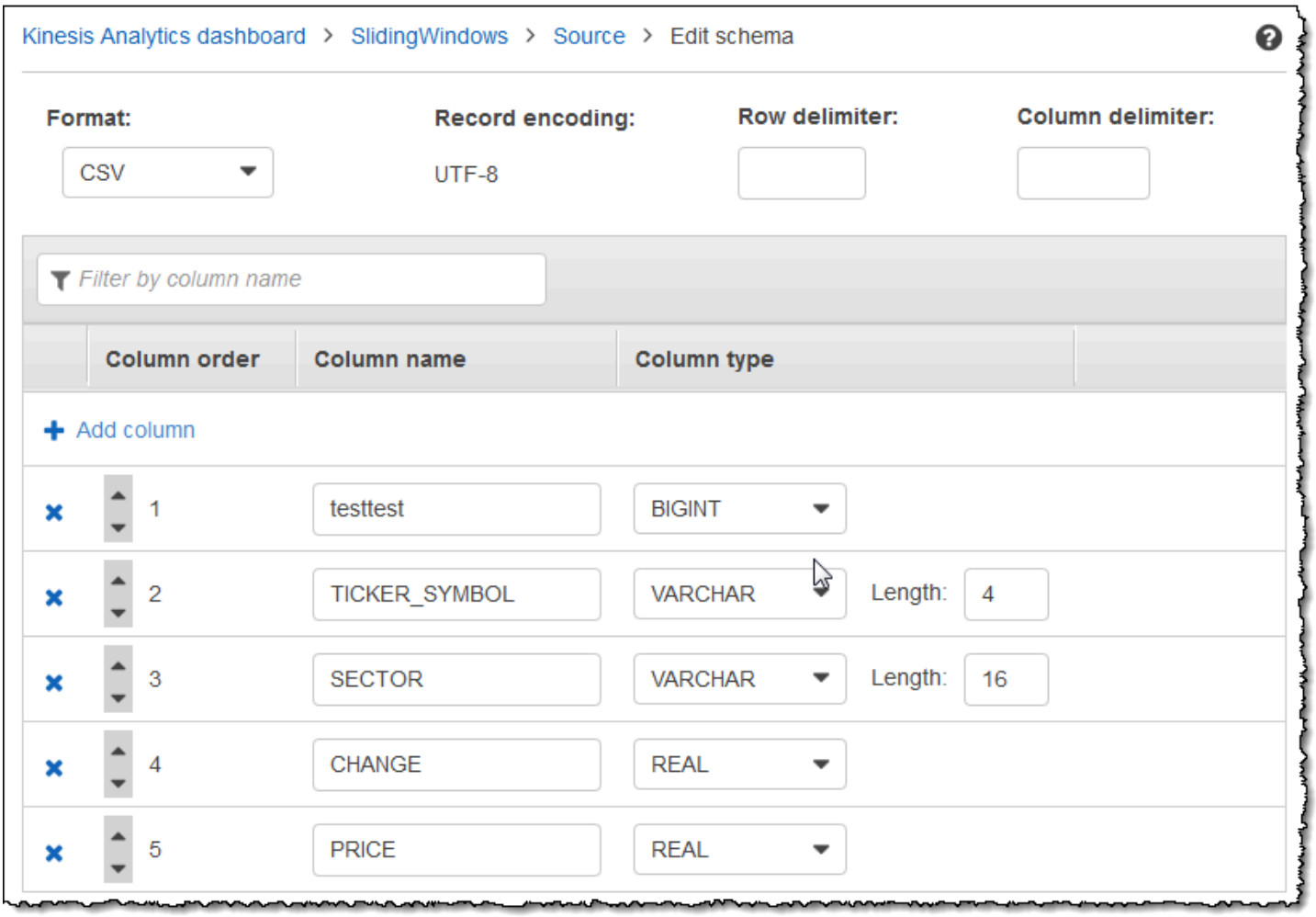
     El nombre de una columna no puede quedar vacío, debe tener más de un carácter y no debe contener palabras claves reservadas de SQL. También debe cumplir con los criterios de identificadores normales de SQL: debe comenzar con una letra y solo puede contener letras, números y guiones bajos.
   - En Column type (Tipo de columna), introduzca un tipo de datos SQL.

     Un tipo de columna puede ser cualquier tipo de datos SQL compatible. Si el nuevo tipo de datos es CHAR, VARBINARY o VARCHAR, especifique una longitud en Length. Para obtener más información, consulte Data Types.
4. Para eliminar una columna, elija el icono x situado junto al número de la columna.



5. Para cambiar el nombre de una columna, introduzca un nuevo nombre en Column name (Nombre de columna). El nuevo nombre de una columna no puede quedar vacío, debe tener más de un carácter y no debe contener palabras claves reservadas de SQL. También debe cumplir con los criterios de identificadores normales de SQL: debe comenzar con una letra y solo puede contener letras, números y guiones bajos.
6. Para cambiar el tipo de datos de una columna, elija un tipo de datos nuevo en Column type. Si el nuevo tipo de datos es CHAR, VARBINARY o VARCHAR, especifique una longitud en Length. Para obtener más información, consulte Data Types.
7. Elija Save schema and update stream para guardar los cambios.

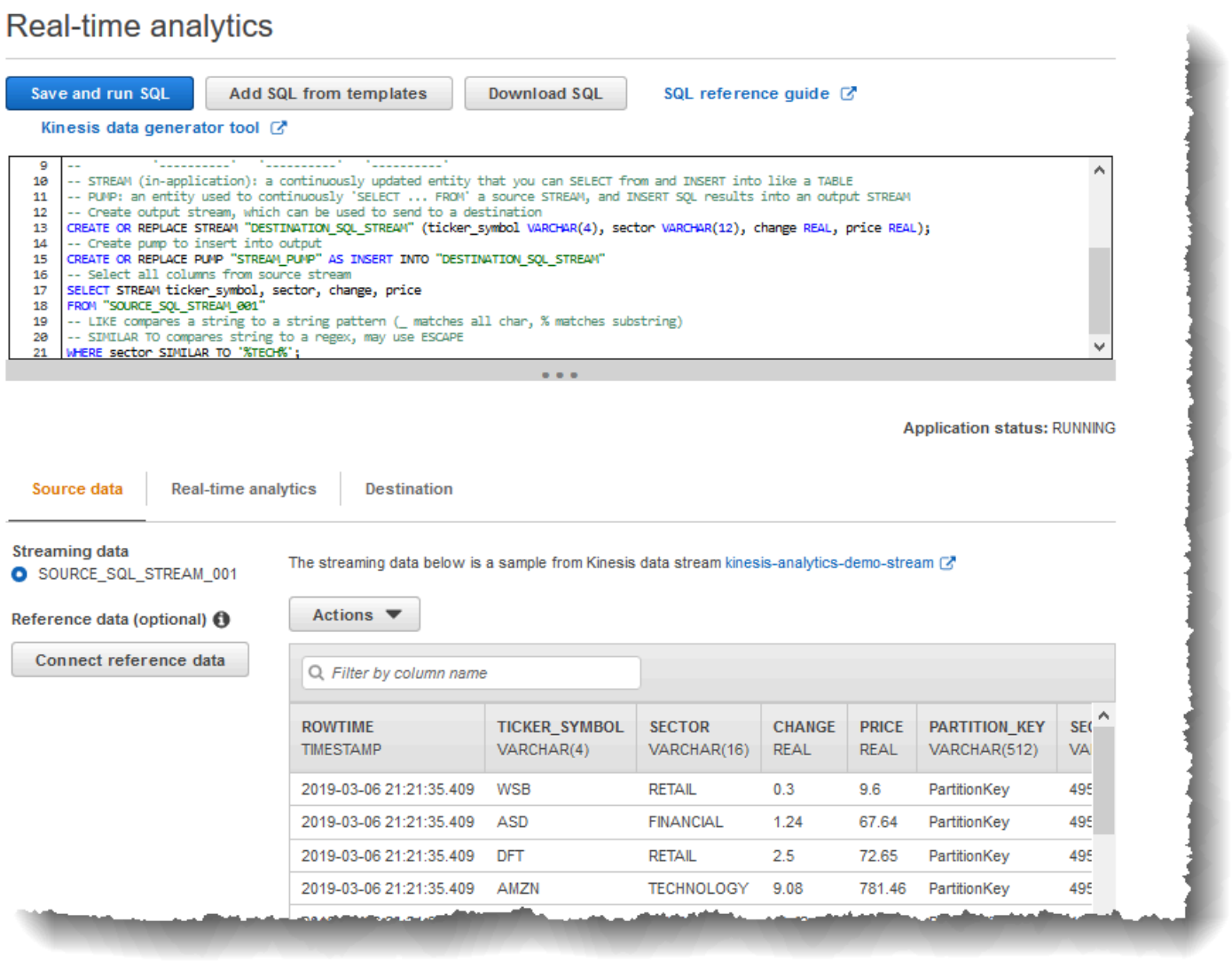
El esquema modificado aparece en el editor y será similar al siguiente.



Si su esquema tiene varias filas, puede filtrarlas usando Filter by column name. Por ejemplo, para editar los nombres de una columna que empieza con `P`, como la columna `Price`, introduzca `P` en la casilla Filtrar por nombre de columna.

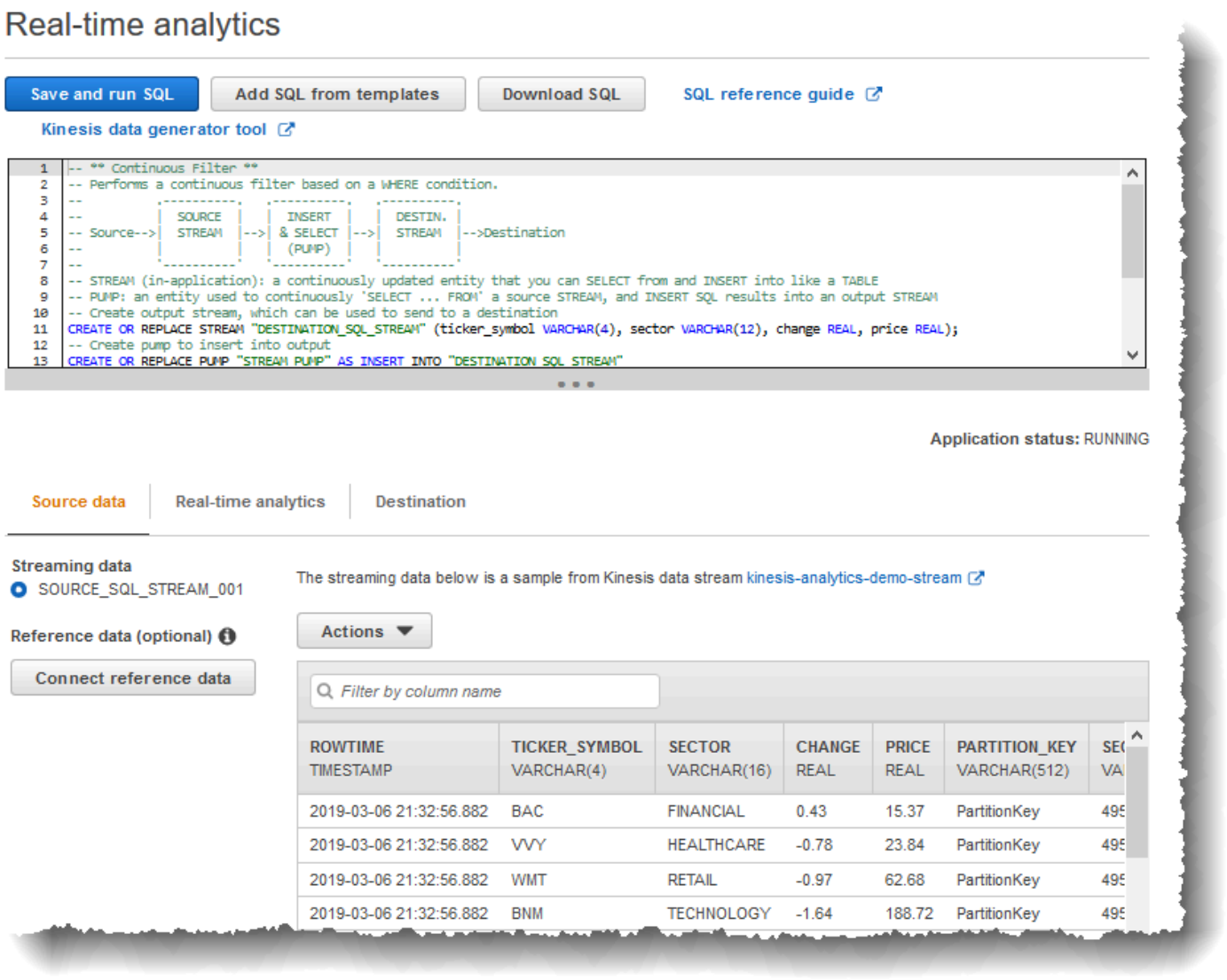
Trabajar con el editor de SQL

A continuación, puede encontrar información sobre secciones del editor de SQL y cómo funciona cada una. En el editor de SQL, puede crear su propio código o elegir Add SQL from templates. Una plantilla de SQL le ofrece código SQL de ejemplo que le puede ayudar a escribir aplicaciones de análisis de datos comunes de Amazon Kinesis Data Analytics. Las aplicaciones de ejemplo en esta guía utilizan algunas de estas plantillas. Para obtener más información, consulte Ejemplos de Kinesis Data Analytics para SQL.



Tipos de datos de fuente

La pestaña Source data identifica un origen de streaming. Identifica también la secuencia de entrada en la aplicación que se asocia a esta fuente y que proporciona la configuración de entrada de la aplicación.



Amazon Kinesis Data Analytics ofrece las siguientes columnas de marca de tiempo, por lo que no es necesario que proporcione el mapeo explícito en su configuración de entrada:

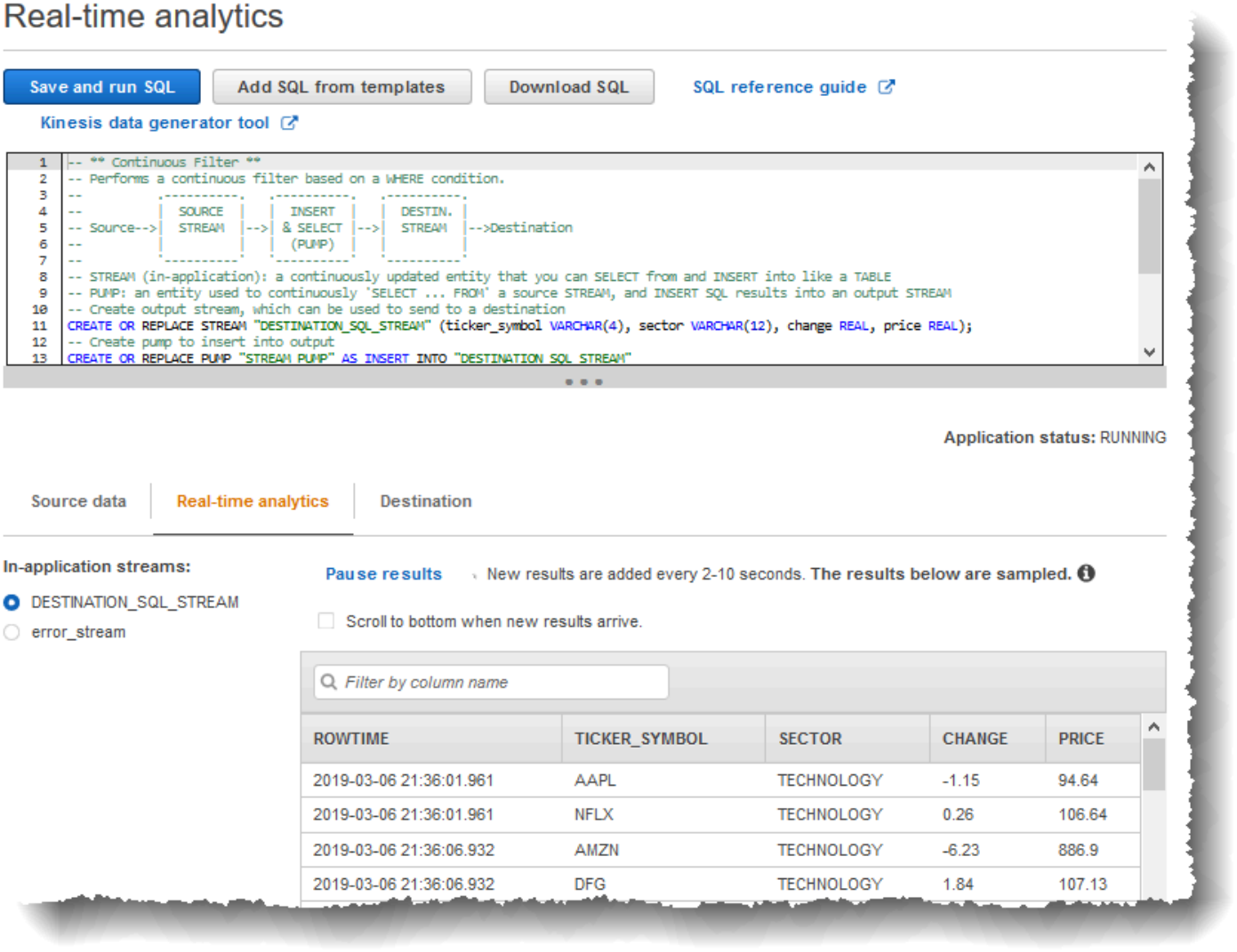
- ROWTIME: cada fila en una secuencia en la aplicación tiene una columna especial denominada `ROWTIME`. Esta columna es la marca temporal correspondiente al momento en el que Kinesis Data Analytics insertó la fila en la primera secuencia en la aplicación.
- Approximate_Arrival_Time: los registros en su origen de streaming incluyen la columna `Approximate_Arrival_Timestamp`. Se trata de la marca de tiempo aproximado de llegada que se establece cuando el origen de streaming recibe y almacena correctamente el registro. Kinesis Data Analytics proporciona esta columna en la secuencia de entrada en la aplicación como `Approximate_Arrival_Time`. Amazon Kinesis Data Analytics proporciona esta columna

exclusivamente en la secuencia de entrada en la aplicación en la que se asigna al origen de streaming.

Estos valores de marca de tiempo resultan útiles en consultas de ventanas basadas en el tiempo. Para obtener más información, consulte Consultas en ventana.

Pestaña Real-Time Analytics

La pestaña Real-time analytics muestra todas las secuencias en la aplicación que su código de aplicación crea. Este grupo de secuencias incluye la secuencia de errores (`error_stream`) que Amazon Kinesis Data Analytics proporciona para todas las aplicaciones.



Pestaña Destination

La pestaña Destination (Destino) le permite configurar la salida de la aplicación, para conservar secuencias en la aplicación en destinos externos. Puede configurar la salida para continuar con los datos en cualquiera de las secuencias en la aplicación en un destino externo. Para obtener más información, consulte Configuración de salida de la aplicación.

Conceptos de SQL de Streaming

Amazon Kinesis Data Analytics implementa el estándar ANSI 2008 SQL con extensiones. Estas extensiones le permiten procesar los datos de streaming. Los siguientes temas cubren de los conceptos clave de SQL de streaming.

Temas

Secuencias y bombeos en la aplicación

Al configurar la entrada de aplicación, asigna el origen de streaming a una secuencia en la aplicación que se crea. La información fluye desde el origen de streaming en la secuencia en la aplicación. Una secuencia en la aplicación funciona como una tabla que puede consultar a través de instrucciones SQL, pero se denomina una secuencia ya que representa un flujo de datos continuo.

> **Note**
>
> No confunda los flujos en la aplicación con los flujos de datos de Amazon Kinesis Data Streams y los flujos de entrega de Firehose. Las secuencias en la aplicación existen solo en el contexto de una aplicación de Amazon Kinesis Data Analytics. Los flujos de datos de Kinesis y los flujos de entrega de Firehose existen independientemente de su aplicación. Puede configurarlas como un origen de streaming en la configuración de entrada de su aplicación o como destino en la configuración de salida.

También puede crear más secuencias en la aplicación como se necesite, para almacenar resultados intermedios de consultas. La creación de una secuencia en la aplicación es un proceso de dos pasos. En primer lugar, debe crear una secuencia en la aplicación y, a continuación, se envían datos a ella. Por ejemplo, suponga que la configuración de entrada de la aplicación crea una secuencia

en la aplicación llamada INPUTSTREAM. En el siguiente ejemplo, debe crear otra secuencia (TEMPSTREAM) y, a continuación, envíe datos desde INPUTSTREAM a la misma.

1. Crear una secuencia en la aplicación (TEMPSTREAM) con tres columnas, tal y como se muestra a continuación:

   ```
   CREATE OR REPLACE STREAM "TEMPSTREAM" (
      "column1" BIGINT NOT NULL,
      "column2" INTEGER,
      "column3" VARCHAR(64));
   ```

   Los nombres de columna se especifican entre comillas, de manera que distingue entre mayúsculas y minúsculas. Para obtener más información, consulte Identificadores en la Referencia de SQL de Amazon Kinesis Data Analytics.

2. Insertar datos a la secuencia mediante una bomba. Una bomba es una consulta continua, que inserta información de una secuencia en la aplicación a otra secuencia en la aplicación. La siguiente instrucción crea una bomba (SAMPLEPUMP) e introduce los datos en TEMPSTREAM seleccionando registros de otra secuencia (INPUTSTREAM).

   ```
   CREATE OR REPLACE PUMP "SAMPLEPUMP" AS
   INSERT INTO "TEMPSTREAM" ("column1",
                             "column2",
                             "column3")
   SELECT STREAM inputcolumn1,
                 inputcolumn2,
                 inputcolumn3
   FROM "INPUTSTREAM";
   ```

Puede tener varios escritores insertados en una secuencia en la aplicación, y puede haber varios lectores seleccionados de la secuencia. Piense en una transmisión dentro de la aplicación como la implementación de un paradigma publish/subscribe de mensajería. En este paradigma, la fila de datos, incluida la hora de creación y la hora de la recepción, puede ser procesada, interpretada y reenviada por una cascada de instrucciones SQL de streaming, sin tener que estar almacenada en un RDBMS tradicional.

Una vez creada una secuencia en la aplicación, puede realizar consultas SQL normales.

> **Note**
>
> Cuando se consultan secuencias, la mayoría de las instrucciones SQL se unen usando una ventana basada en filas o en el tiempo. Para obtener más información, consulte Consultas en ventana.

También puede unir secuencias. Para ver ejemplos de cómo unir secuencias, consulte Operaciones de streaming de datos: uniones de secuencias.

Marcas temporales y la comuna ROWTIME

Las secuencias en la aplicación incluyen una columna especial llamada `ROWTIME`. Almacena una marca temporal cuando Amazon Kinesis Data Analytics inserta una fila en la primera secuencia en la aplicación. `ROWTIME` refleja la marca temporal en la que Amazon Kinesis Data Analytics insertó un registro en la primera secuencia en la aplicación después de leer desde el origen de streaming. Este valor `ROWTIME` se mantiene en toda su aplicación.

> **Note**
>
> Cuando se bombean registros de una secuencia en la aplicación a otra, no es necesario copiar la columna `ROWTIME` porque ya lo hace Amazon Kinesis Data Analytics.

Amazon Kinesis Data Analytics garantiza que los valores de `ROWTIME` aumentan de forma monotómica. Puede utilizar esta marca temporal en las consultas en ventana basadas en el tiempo. Para obtener más información, consulte Consultas en ventana.

Puede tener acceso a la columna ROWTIME en la instrucción `SELECT` al igual que con cualquier otra columna de la secuencia en la aplicación. Por ejemplo:

```
SELECT STREAM ROWTIME,
              some_col_1,
              some_col_2
FROM  SOURCE_SQL_STREAM_001
```

Descripción de los distintos tiempos en análisis de streaming

Además de `ROWTIME`, existen otros tipos de tiempo en aplicaciones de streaming en tiempo real. Estos son:

- Tiempo de eventos: la marca temporal de cuando se produjo el evento. A esto también se le llama el lado de tiempo del cliente. Suele ser conveniente utilizar estos momentos en análisis, ya que es el momento en el que se produjo un evento. No obstante, muchas fuentes de eventos como, por ejemplo, clientes de teléfonos móviles y web, no tienen relojes de confianza, lo que puede provocar tiempos inexactos. Además, los problemas de conectividad pueden hacer que los registros aparezcan en la secuencia y no lo en el mismo orden los eventos.

- Tiempo de ingestión: la marca temporal de cuándo se añadió un registro a un origen de streaming. Amazon Kinesis Data Streams incluye un campo llamado `APPROXIMATE_ARRIVAL_TIME` en todos los registros que proporciona esta marca temporal. Esto también se denomina a veces tiempo del servidor. Este tiempo de ingestión suele ser una aproximación cercana al tiempo de evento. Si existe algún tipo de retraso en la adquisición de registros en la secuencia, se pueden producir inexactitudes, que suelen ser raras. Además, el tiempo de ingestión no suele estar fuera de lugar, si bien eso puede ocurrir debido a la naturaleza distribuida del streaming de datos. Por lo tanto, el tiempo de ingestión es un reflejo bastante preciso y en orden del tiempo de evento.

- Tiempo de procesamiento: la marca temporal de cuando Amazon Kinesis Data Analytics inserta un fila en la primera secuencia en la aplicación. Amazon Kinesis Data Analytics proporciona esta marca temporal en la columna `ROWTIME` de cada secuencia en la aplicación. El tiempo de procesamiento aumenta siempre de forma monótona. Sin embargo, no será preciso si la aplicación se rezaga. (Si una aplicación se rezaga, el tiempo de procesamiento no refleja con precisión la hora del evento). Este `ROWTIME` es preciso en relación con el reloj, pero podría no ser el momento en el que el evento en que el evento realmente ocurrió.

Utilizar cada uno de estos tiempos en las consultas en ventana basadas en el tiempo tiene ventajas y desventajas. Le recomendamos que elija uno o varios de estos tiempos, y una estrategia para abordar las posibles desventajas en función de su caso de uso.

> **Note**
>
> Si utiliza ventanas basadas en filas, el tiempo no será un problema y puede ignorar esta sección.

Recomendamos una estrategia de dos ventanas que utilice dos ventanas basadas en el tiempo: una `ROWTIME` y una para los otros tiempos (tiempo de ingestión o de evento).

- Utilice `ROWTIME` como la primera ventana, que controla la frecuencia con la que la consulta emite los resultados, tal y como se muestra en el siguiente ejemplo. No se utiliza como tiempo lógico.
- Utilice uno de los otros tiempos que es el tiempo lógico que desea asociar a su análisis. Este tiempo representa cuándo se produjo el evento. En el siguiente ejemplo, el objetivo de análisis es agrupar los registros y devolver un recuento por cada símbolo.

La ventaja de esta estrategia es que puede utilizar una hora que represente cuándo se produjo el evento. Puede gestionar adecuadamente situaciones en las que la aplicación se queda rezagada o cuando los eventos llegan desordenados. Si la aplicación se rezaga al traer registros en la secuencia en la aplicación, estos se siguen agrupando por el momento lógico en la segunda ventana. La consulta utiliza `ROWTIME` para garantizar el orden de procesamiento. Los registros que están retrasados (la marca temporal de ingestión muestra un valor anteriores comparación con el `ROWTIME` valor) también se procesan correctamente.

Considere realizar la siguiente consulta con a la secuencia de demostración utilizada en el ejercicio de introducción. La consulta utiliza la cláusula `GROUP BY` y emite el recuento de cada símbolo en una ventana de saltos de un minuto.

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM"
    ("ingest_time"    timestamp,
    "APPROXIMATE_ARRIVAL_TIME" timestamp,
    "ticker_symbol"  VARCHAR(12),
    "symbol_count"        integer);


CREATE OR REPLACE PUMP "STREAM_PUMP" AS
    INSERT INTO "DESTINATION_SQL_STREAM"
    SELECT STREAM STEP("SOURCE_SQL_STREAM_001".ROWTIME BY INTERVAL '60' SECOND) AS
 "ingest_time",
```

```
        STEP("SOURCE_SQL_STREAM_001".APPROXIMATE_ARRIVAL_TIME BY INTERVAL '60' SECOND)
 AS "APPROXIMATE_ARRIVAL_TIME",
        "TICKER_SYMBOL",
        COUNT(*) AS "symbol_count"
    FROM "SOURCE_SQL_STREAM_001"
    GROUP BY "TICKER_SYMBOL",
        STEP("SOURCE_SQL_STREAM_001".ROWTIME BY INTERVAL '60' SECOND),
        STEP("SOURCE_SQL_STREAM_001".APPROXIMATE_ARRIVAL_TIME BY INTERVAL '60' SECOND);
```

En `GROUP BY`, primero agrupe los registros según `ROWTIME` en una ventana de un minuto y, a continuación, según `APPROXIMATE_ARRIVAL_TIME`.

Los valores de marca temporal del resultado se redondean hacia abajo al intervalo de 60 segundos más próximo. El primer grupo de resultados emitido por la consulta muestra registros del primer minuto. El segundo grupo de resultados emitido muestra registros de los minutos siguientes según `ROWTIME`. El último registro indica que la aplicación se retrasó al traer el registro en la secuencia en la aplicación (muestra un valor `ROWTIME` con retraso en comparación la marca temporal de la ingestión).

```
ROWTIME                  INGEST_TIME     TICKER_SYMBOL  SYMBOL_COUNT

--First one minute window.
2016-07-19 17:05:00.0    2016-07-19 17:05:00.0    ABC      10
2016-07-19 17:05:00.0    2016-07-19 17:05:00.0    DEF      15
2016-07-19 17:05:00.0    2016-07-19 17:05:00.0    XYZ      6
--Second one minute window.
2016-07-19 17:06:00.0    2016-07-19 17:06:00.0    ABC      11
2016-07-19 17:06:00.0    2016-07-19 17:06:00.0    DEF      11
2016-07-19 17:06:00.0    2016-07-19 17:05:00.0    XYZ      1  ***

***late-arriving record, instead of appearing in the result of the
first 1-minute windows (based on ingest_time, it is in the result
of the second 1-minute window.
```

Puede combinar los resultados para obtener un recuento preciso por minuto insertando los resultados en una base de datos posterior. Por ejemplo, puede configurar la salida de la aplicación para que conserve los resultados en un flujo de entrega de Firehose que pueda escribir en una tabla de Amazon Redshift. Una vez que los resultados estén en la tabla de Amazon Redshift puede consultar la tabla para calcular el grupo de recuento total por `Ticker_Symbol`. En el caso de `XYZ`, el total es correcto (6+1), aunque un registro haya llegado tarde.

Consultas continuas

Una consulta a través de una secuencia se ejecuta de forma continua en los datos de streaming. Esta ejecución continua hace posibles situaciones como la capacidad de las aplicaciones para consultar continuamente una secuencia y generar alertas.

En el ejercicio de introducción, dispone de una secuencia en la aplicación denominada `SOURCE_SQL_STREAM_001`. Recibe constantemente los precios de valores a partir de una secuencia de demostración (un flujo de datos de Kinesis). El esquema es el siguiente:

```
(TICKER_SYMBOL VARCHAR(4),
 SECTOR varchar(16),
 CHANGE REAL,
 PRICE REAL)
```

Supongamos que está interesado en los cambios de cotizaciones superiores al 15 por ciento. Puede utilizar la siguiente consulta en el código de la aplicación. Esta consulta se ejecuta sin interrupción y emite registros cuando se detecta un cambio en las cotizaciones superior al 15 por ciento.

```
SELECT STREAM TICKER_SYMBOL, PRICE
      FROM   "SOURCE_SQL_STREAM_001"
      WHERE  (ABS((CHANGE / (PRICE-CHANGE)) * 100)) > 15
```

Utilice el siguiente procedimiento para configurar una aplicación de Amazon Kinesis Data Analytics y probar esta consulta.

Para probar la consulta

1. Cree una aplicación siguiendo el ejercicio de introducción.
2. Sustituya la instrucción `SELECT` en el código de la aplicación por la consulta `SELECT` anterior. El código de la aplicación resultante se muestra a continuación:

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (ticker_symbol VARCHAR(4),
                                                   price DOUBLE);
-- CREATE OR REPLACE PUMP to insert into output
CREATE OR REPLACE PUMP "STREAM_PUMP" AS
  INSERT INTO "DESTINATION_SQL_STREAM"
      SELECT STREAM TICKER_SYMBOL,
                    PRICE
```

```
FROM   "SOURCE_SQL_STREAM_001"
WHERE  (ABS((CHANGE / (PRICE-CHANGE)) * 100)) > 15;
```

Consultas en ventana

Las consultas SQL en el código de la aplicación pueden ejecutarse de forma continua a través de secuencias en la aplicación. En una secuencia en la aplicación representa un datos ilimitados que fluyen de forma continua a través de su aplicación. Por lo tanto, para obtener resultados de esta salida que se actualiza de forma constante, a menudo se unen consultas utilizando una ventana definida en términos de tiempo o de filas. Estas también se denominan SQL en ventana.

Para una consulta en ventana basada en el tiempo, debe especificar el tamaño de la ventana expresado en unidades de tiempo (por ejemplo, una ventana de un minuto). Esto requiere una columna de marca temporal en la secuencia en la aplicación que aumente de forma monótona. (La marca de tiempo de una fila nueva es superior o igual a la fila anterior.) Amazon Kinesis Data Analytics proporciona una columna de marca temporal llamada `ROWTIME` para cada secuencia en la aplicación. Puede utilizar esta columna al especificar las consultas basadas en el tiempo. Para su aplicación, puede elegir otra opción de marca temporal. Para obtener más información, consulte Marcas temporales y la comuna ROWTIME.

Para una consulta en ventana basada en filas, debe especificar el tamaño de la ventana expresado como un número de filas.

Puede especificar una consulta para procesar registros en una ventana de saltos, una ventana deslizante una ventana escalonada, según las necesidades de la aplicación. Kinesis Data Analytics admite los siguientes tipos de ventanas:

- Ventanas escalonadas: una consulta que agrupa los datos usando ventanas basadas en tiempo con clave que se abren cuando llegan los datos. Las claves permiten que se superpongan varias ventanas. Esta es la forma recomendada de agregar datos mediante ventanas basadas en el tiempo, ya que Stagger Windows reduce el retraso o out-of-order los datos en comparación con las ventanas giratorias.
- Ventanas de saltos de tamaño constante: una consulta que agrupa los datos usando ventanas de tiempo que se abren y cierran a intervalos regulares.
- Ventanas deslizantes: una consulta que agrega datos continuamente, utilizando un intervalo de tiempo fijo o de número de filas.

Ventanas escalonadas

El uso de las ventanas escalonadas es un método de definición de ventanas adecuado para analizar grupos de datos que llegan erráticamente. Es ideal para cualquier caso de uso de análisis de series temporales, como un conjunto de ventas relacionadas o registros de un archivo de registros.

Por ejemplo, Registros de flujo de VPC tiene una ventana de captura de 10 minutos aproximadamente. Sin embargo, pueden tener una ventana de captura de hasta 15 minutos si acumula datos en el cliente. Las ventanas escalonadas son ideales para agrupar estos registros para su análisis.

Las ventanas escalonadas solucionan el problema de los registros relacionados no incluidos en la misma ventana de tiempo restringido, por ejemplo, si se han utilizado ventanas de saltos.

Resultados parciales con ventanas de saltos

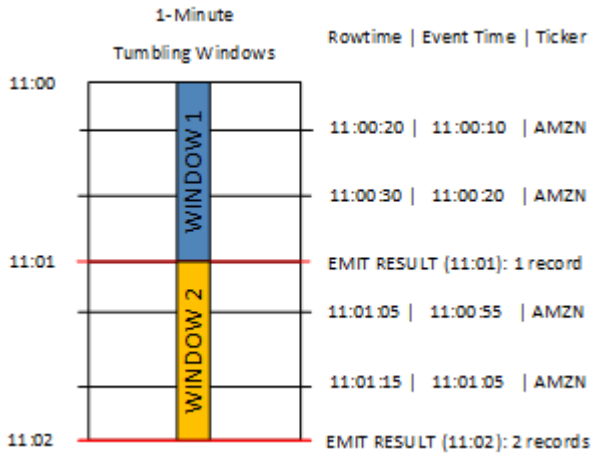
El uso Ventanas de saltos de tamaño constante para agregar out-of-order datos o datos tiene ciertas limitaciones.

Si las ventanas de saltos se utilizan para analizar grupos de datos relacionados con el tiempo, los registros individuales podrían caer en ventanas separadas. Por lo tanto, los resultados parciales de cada ventana se deben combinar más tarde para ofrecer resultados completos para cada grupo de registros.

En la siguiente consulta de ventana de saltos, los registros se agrupan en ventanas por hora de fila, hora del evento y clave de cotización:

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
    TICKER_SYMBOL VARCHAR(4),
    EVENT_TIME timestamp,
    TICKER_COUNT     DOUBLE);

CREATE OR REPLACE PUMP "STREAM_PUMP" AS
  INSERT INTO "DESTINATION_SQL_STREAM"
    SELECT STREAM
        TICKER_SYMBOL,
        FLOOR(EVENT_TIME TO MINUTE),
        COUNT(TICKER_SYMBOL) AS TICKER_COUNT
    FROM "SOURCE_SQL_STREAM_001"
    GROUP BY ticker_symbol, FLOOR(EVENT_TIME TO MINUTE),
 STEP("SOURCE_SQL_STREAM_001".ROWTIME BY INTERVAL '1' MINUTE);
```

En el siguiente diagrama, una aplicación cuenta el número de operaciones que recibe, en función del momento en que se han producido las operaciones (hora del evento) con un grado de detalle de un minuto. La aplicación puede utilizar una ventana de saltos para agrupar los datos en función de la hora de la fila y la hora del evento. La aplicación recibe cuatro registros que llegan minuto a minuto. Agrupa los registros por hora de la fila, hora del evento y clave de cotización. Como algunos de los registros llegan después de que termine la primera ventana de saltos, los registros no entrarán todos en la misma ventana de saltos de un minuto.



El diagrama anterior tiene los siguientes eventos.

ROWTIME	EVENT_TIME	TICKER_SYMBOL
11:00:20	11:00:10	AMZN
11:00:30	11:00:20	AMZN
11:01:05	11:00:55	AMZN
11:01:15	11:01:05	AMZN

El conjunto de resultados de la aplicación de la ventana de saltos tiene un aspecto similar al siguiente.

ROWTIME	EVENT_TIME	TICKER_SYMBOL	COUNT
11:01:00	11:00:00	AMZN	2

ROWTIME	EVENT_TIME	TICKER_SYMBOL	COUNT
11:02:00	11:00:00	AMZN	1
11:02:00	11:01:00	AMZN	1

En el conjunto de resultados anterior, se devuelven tres resultados:

- Un registro con un `ROWTIME` de 11:01:00 que agrupa los dos primeros registros.
- Un registro a las 11:02:00 que agrupa solo el tercer registro. Este registro tiene un `ROWTIME` dentro de la segunda ventana, pero un `EVENT_TIME` dentro de la primera ventana.
- Un registro a las 11:02:00 que agrupa solo el cuarto registro.

Para analizar todo el conjunto de resultados, los registros deben agruparse en el almacén de persistencia. Esto añade complejidad y requisitos de procesamiento a la aplicación.

Resultados completos con ventanas escalonadas

Para mejorar la precisión de los registros de datos de tiempo, Kinesis Data Analytics ofrece un nuevo tipo de ventana llamada ventanas escalonadas. En este tipo de ventana, la ventana se abre cuando llega el primer evento que coincide con la clave de partición y no a un intervalo de tiempo fijo. La ventana se cierra en función de la antigüedad especificada, que se mide desde el momento en que se abrió la ventana.

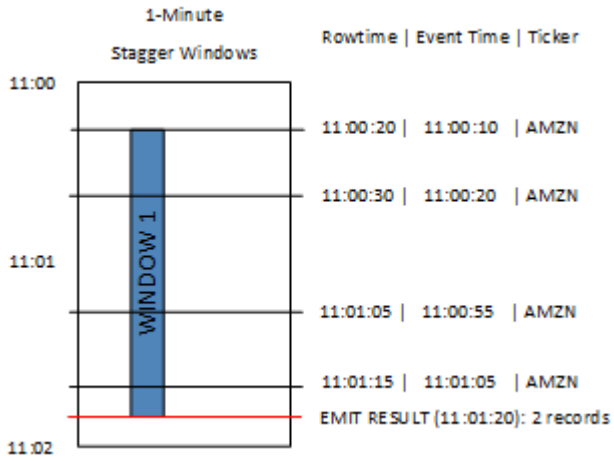
Una ventana escalonada es una ventana con restricción de tiempo distinta para cada grupo de claves de una cláusula de ventana. La aplicación agrupa cada resultado de la cláusula de ventana en su propia ventana de tiempo, en lugar de usar una sola ventana para todos los resultados.

En la siguiente consulta de ventana escalonada, los registros se agrupan en ventanas por hora del evento y clave de cotización:

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
    ticker_symbol    VARCHAR(4),
    event_time       TIMESTAMP,
    ticker_count     DOUBLE);

CREATE OR REPLACE PUMP "STREAM_PUMP" AS
  INSERT INTO "DESTINATION_SQL_STREAM"
```

```
    SELECT STREAM
        TICKER_SYMBOL,
        FLOOR(EVENT_TIME TO MINUTE),
        COUNT(TICKER_SYMBOL) AS ticker_count
    FROM "SOURCE_SQL_STREAM_001"
    WINDOWED BY STAGGER (
            PARTITION BY FLOOR(EVENT_TIME TO MINUTE), TICKER_SYMBOL RANGE INTERVAL '1'
 MINUTE);
```

En el siguiente diagrama, los registros se agrupan en ventanas escalonadas por hora del evento y clave de cotización:



El diagrama anterior tiene los siguientes eventos, que son los mismos que los que analizó la aplicación de ventana de saltos:

ROWTIME	EVENT_TIME	TICKER_SYMBOL
11:00:20	11:00:10	AMZN
11:00:30	11:00:20	AMZN
11:01:05	11:00:55	AMZN
11:01:15	11:01:05	AMZN

El conjunto de resultados de la aplicación de ventana escalonada tiene un aspecto similar al siguiente.

ROWTIME	EVENT_TIME	TICKER_SYMBOL	Recuento
11:01:20	11:00:00	AMZN	3
11:02:15	11:01:00	AMZN	1

El registro devuelto agrupa los tres primeros registros de entrada. Los registros se agrupan en ventanas escalonadas de un minuto. La ventana escalonada comienza cuando la aplicación recibe el primer registro AMZN (con un `ROWTIME` de 11:00:20). Cuando finaliza la ventana escalonada de un minuto (a las 11:01:20), se escribe un registro con los resultados incluidos dentro de la ventana escalonada (en función de `ROWTIME` y `EVENT_TIME`) en la secuencia de salida. Gracias al uso de una ventana escalonada, todos los registros en los que el valor de `ROWTIME` y `EVENT_TIME` esté comprendido en una ventana de un minuto se emitirán como un único resultado.

El último registro (con un `EVENT_TIME` fuera de la agregación de un minuto) se agrega por separado. Esto se debe a que `EVENT_TIME` es una de las claves de partición que se utiliza para separar los registros en conjuntos de resultados y la clave de partición `EVENT_TIME` para la primera ventana es `11:00`.

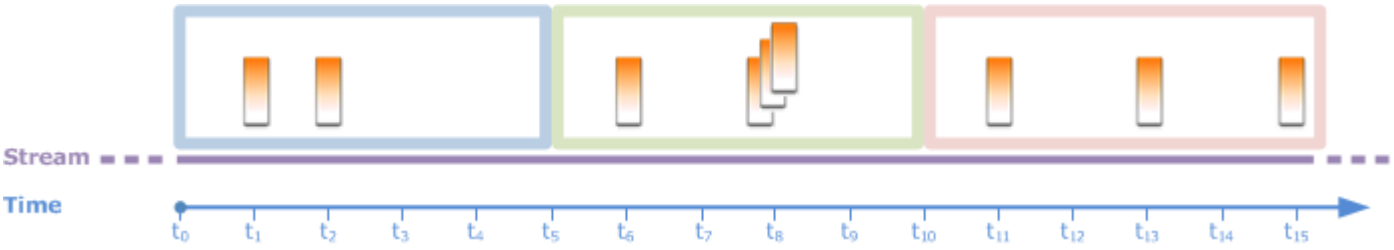
La sintaxis de la ventana escalonada se define en una cláusula especial, `WINDOWED BY`. Esta cláusula se utiliza en lugar de la cláusula `GROUP BY` para streaming de agregaciones. La cláusula aparece inmediatamente detrás de la cláusula `WHERE` opcional y antes de la cláusula `HAVING`.

La ventana escalonada se define en la cláusula `WINDOWED BY` y toma dos parámetros: claves de partición y duración de la ventana. Las claves de partición dividen el flujo del flujo de datos entrante y definen cuándo se abre la ventana. Una ventana escalonada se abre cuando el primer evento con una clave de partición única aparece en la secuencia. La ventana escalonada se cierra después de un periodo de tiempo fijo definido por la duración de la ventana. La sintaxis se muestra en el siguiente ejemplo de código:

```
...
FROM <stream-name>
WHERE <... optional statements...>
WINDOWED BY STAGGER(
 PARTITION BY <partition key(s)>
 RANGE INTERVAL <window length, interval>
);
```

Ventanas de saltos de tamaño constante (Agregados utilizando GROUP BY)

Cuando una consulta en ventana procesa cada ventana de forma que no se superpongan, la ventana se denomina ventana de saltos. En este caso, cada registro de una secuencia en la aplicación pertenece a una ventana específica. Se procesa solo una vez (cuando la consulta procesa la ventana a la que pertenece el registro).



Por ejemplo, una consulta mediante una cláusula `GROUP BY` procesa las filas en una ventana de saltos. La secuencia de demostración en el ejercicio de introducción recibe datos de cotizaciones de valores que se asignan a la secuencia en la aplicación `SOURCE_SQL_STREAM_001` en su aplicación. Esta secuencia tiene el siguiente esquema.

```
(TICKER_SYMBOL VARCHAR(4),
 SECTOR varchar(16),
 CHANGE REAL,
 PRICE REAL)
```

En el código de la aplicación, suponga que desea encontrar los precios (min, max) agregados de cada símbolo a lo largo de un periodo de un minuto. Puede utilizar la siguiente consulta.

```
SELECT STREAM ROWTIME,
              Ticker_Symbol,
              MIN(Price) AS Price,
              MAX(Price) AS Price
FROM     "SOURCE_SQL_STREAM_001"
GROUP BY Ticker_Symbol,
         STEP("SOURCE_SQL_STREAM_001".ROWTIME BY INTERVAL '60' SECOND);
```

El ejemplo anterior es una consulta en ventana que está basada en el tiempo. La consulta agrupa los registros por valores `ROWTIME`. Para notificar cada minuto, la función `STEP` redondea hacia abajo los valores `ROWTIME` al minuto más próximo.

> **Note**
>
> También puede utilizar la función `FLOOR` para agrupar registros en ventanas. Sin embargo, `FLOOR` solo puede redondear los valores temporales hacia abajo a una unidad de tiempo completa (hora, minuto, segundo, etc.). Se recomienda `STEP` para agrupar los registros en ventanas de saltos, ya que puede redondear los valores hacia abajo a un intervalo arbitrario, por ejemplo, 30 segundos.

Este es un ejemplo de una ventana que no se superpone (de saltos). La cláusula `GROUP BY` agrupa los registros en una ventana de un minuto y cada registro pertenece a una ventana específica (sin solapamiento). La consulta emite un registro de salida por minuto, que proporciona el precio de cotización registrado min/max en ese minuto específico. Este tipo de consulta es útil para generar informes periódicos a partir de el flujo de datos de entrada. En este ejemplo, los informes se generan cada minuto.

Para probar la consulta

1. Configure una aplicación siguiendo el ejercicio de introducción.
2. Sustituya la instrucción `SELECT` en el código de la aplicación por la consulta `SELECT` anterior. El código de la aplicación resultante se muestra a continuación:

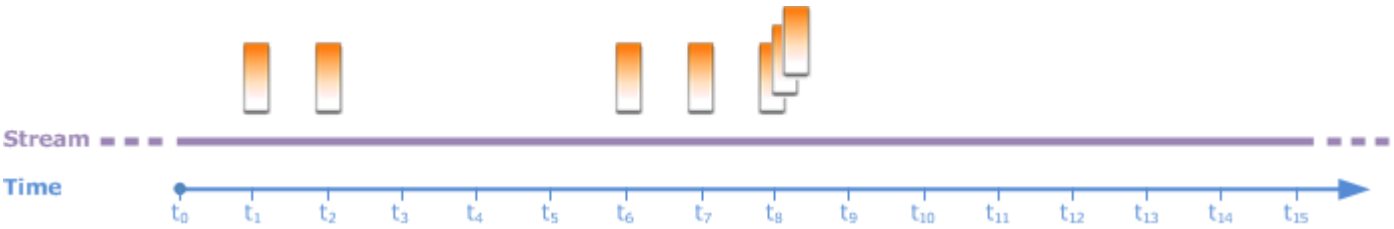
```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
                                    ticker_symbol VARCHAR(4),
                                    Min_Price     DOUBLE,
                                    Max_Price     DOUBLE);
-- CREATE OR REPLACE PUMP to insert into output
CREATE OR REPLACE PUMP "STREAM_PUMP" AS
  INSERT INTO "DESTINATION_SQL_STREAM"
    SELECT STREAM Ticker_Symbol,
                  MIN(Price) AS Min_Price,
                  MAX(Price) AS Max_Price
    FROM    "SOURCE_SQL_STREAM_001"
    GROUP BY Ticker_Symbol,
             STEP("SOURCE_SQL_STREAM_001".ROWTIME BY INTERVAL '60' SECOND);
```

Ventanas deslizantes

En lugar de agrupar los registros utilizando `GROUP BY`, puede definir ventana basada en tiempo o filas. Para ello, añada una cláusula `WINDOW` explícita.

En este caso, como la ventana se mueve con el tiempo, Amazon Kinesis Data Analytics emite una salida cuando aparecen nuevos registros en la secuencia. Kinesis Data Analytics emite esta salida al procesar las filas en la ventana. Las ventanas pueden superponerse en este tipo de procesamiento, y un registro pueden formar parte de varias ventanas y ser procesado con cada ventana. El siguiente ejemplo ilustra una ventana deslizante.

Considere una simple consulta que calcule registros en la secuencia. En este ejemplo se supone una ventana de 5 segundos. En el siguiente ejemplo de secuencia, llegan nuevos registros en el tiempo t_1, t_2, t_6 y t_7 y tres registros llegan en el tiempo t_8 segundos.



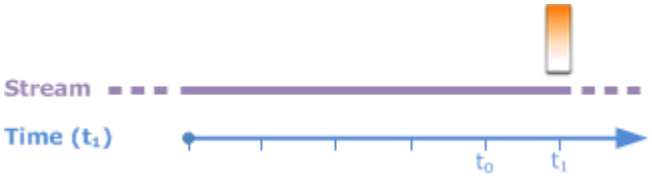
Tenga en cuenta lo siguiente:

- En el ejemplo se supone una ventana de 5 segundos. La ventana de 5 segundos varía de forma continua con el tiempo.
- Por cada fila que introduce en una ventana, la ventana deslizante emite una fila de salida. Poco después que se inicie la aplicación, verá que la consulta emite una salida para cada nuevo registro que aparece en la secuencia, a pesar de no haya pasado todavía la ventana de 5 segundos. Por ejemplo, la consulta emite una salida cuando un registro aparece en el primer segundo y segundo segundo. Después, la consulta procesa los registros en la ventana de 5 segundos.
- Las ventanas varían con el tiempo. Si se un registro anterior en la secuencia cae fuera de la ventana, la consulta no emite una salida a menos que también exista un nuevo registro en la secuencia que entre en esa ventana de 5 segundos.

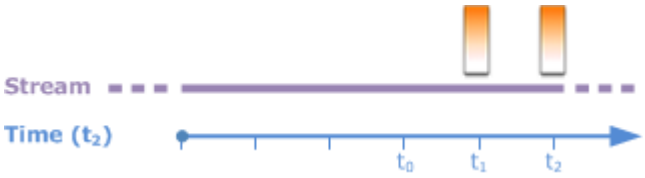
Imagine que la consulta empieza a ejecutarse en t_0. Entonces, ocurriría lo siguiente:

1. La consulta empieza en el tiempo t_0. La consulta no emite una salida (valor de recuento), porque no existen registros en ese momento.

2. En el tiempo t_1, un nuevo registro aparece en la secuencia y la consulta emite el valor de recuento 1.



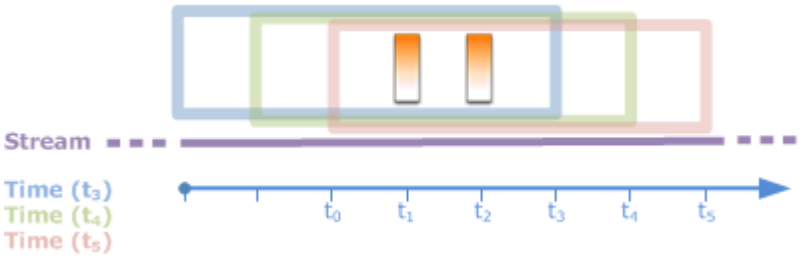
3. En el tiempo t_2, otro registro aparece y la consulta emite recuento 2.



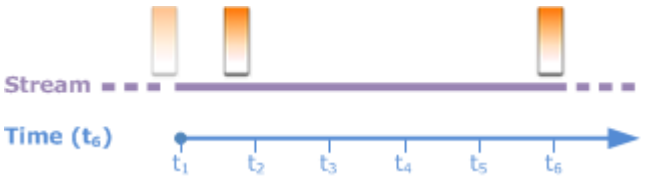
4. La ventana de 5 segundos se desliza con el tiempo:

   - En t_3, la ventana deslizante abarca de t_3 a t_0
   - En t4 (ventana deslizante de t_4 a t_0)
   - En t_5, la ventana deslizante abarca de t_5 a t_0

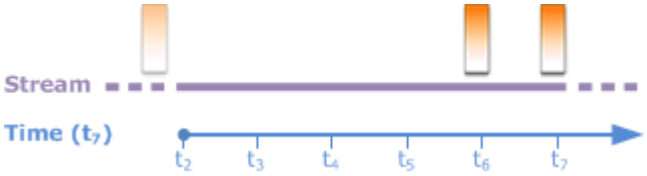
   Todas estas veces, la ventana de 5 segundos tiene los mismos registros, no hay nuevos registros. Por lo tanto, la consulta no emite ninguna salida.



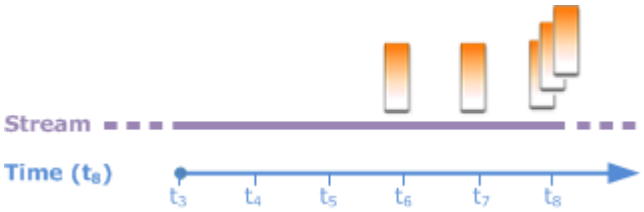
5. En el tiempo t_6, la ventana de 5 segundos es (t_6 a t_1). La consulta detecta un nuevo registro en t_6, por lo que emite la salida 2. El registro en t_1 ya no está en la ventana y no se tiene en cuenta.



6. En el tiempo t_7, la ventana de 5 segundos es t_7 a t_2. La consulta detecta un nuevo registro en t_7, por lo que emite la salida 2. El registro en t_2 ya no está en la ventana de 5 segundos y, por lo tanto, no se tiene en cuenta.



7. En el tiempo t_8, la ventana de 5 segundos es t_8 a t_3. La consulta detecta tres nuevos registros, y, por lo tanto, emite el recuento de registro 5.



Resumiendo, la ventana es de tamaño fijo y varía con el tiempo. La consulta emite salidas cuando aparecen nuevos registros.

> **Note**
>
> Le recomendamos que utilice una ventana deslizante de no más de una hora. Si utiliza una ventana más larga, la aplicación tardará mas en reiniciarse después del mantenimiento normal del sistema. Esto se debe a que los datos de origen deben leerse de la secuencia de nuevo.

A continuación, se describe un ejemplo de consultas que utiliza la cláusula `WINDOW` para definir y realizar agregados de ventanas. Como las consultas no especifican `GROUP BY`, la consulta utiliza la ventana deslizante para procesar registros en la secuencia.

Ejemplo 1: Procesar una secuencia mediante una ventana deslizante de 1 minuto

Considere la secuencia de demostración en el ejercicio de introducción que rellena la secuencia en la aplicación `SOURCE_SQL_STREAM_001`. A continuación se muestra el esquema.

```
(TICKER_SYMBOL VARCHAR(4),
 SECTOR varchar(16),
```

```
  CHANGE REAL,
  PRICE REAL)
```

Supongamos que desea que su aplicación calcule agregados mediante una ventana deslizante de 1 minuto. Es decir, para cada nuevo registro que aparezca en la secuencia, desea que la aplicación emita una salida aplicando sumas en los registros de la anterior ventana de 1 minuto.

Puede utilizar la siguiente consulta en ventana basada en el tiempo. La consulta utiliza la cláusula `WINDOW` para definir el intervalo de 1 minuto. `PARTITION BY` en la cláusula `WINDOW` agrupa los registros por valor de símbolo dentro de la ventana deslizante.

```
SELECT STREAM ticker_symbol,
              MIN(Price) OVER W1 AS Min_Price,
              MAX(Price) OVER W1 AS Max_Price,
              AVG(Price) OVER W1 AS Avg_Price
FROM   "SOURCE_SQL_STREAM_001"
WINDOW W1 AS (
   PARTITION BY ticker_symbol
   RANGE INTERVAL '1' MINUTE PRECEDING);
```

Para probar la consulta

1. Configure una aplicación siguiendo el Ejercicio de introducción.
2. Sustituya la instrucción `SELECT` en el código de la aplicación por la consulta `SELECT` anterior. A continuación se muestra el código de la aplicación resultante.

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
                         ticker_symbol VARCHAR(10),
                         Min_Price     double,
                         Max_Price     double,
                         Avg_Price     double);
CREATE OR REPLACE PUMP "STREAM_PUMP" AS
   INSERT INTO "DESTINATION_SQL_STREAM"
     SELECT STREAM ticker_symbol,
                   MIN(Price) OVER W1 AS Min_Price,
                   MAX(Price) OVER W1 AS Max_Price,
                   AVG(Price) OVER W1 AS Avg_Price
     FROM   "SOURCE_SQL_STREAM_001"
     WINDOW W1 AS (
        PARTITION BY ticker_symbol
        RANGE INTERVAL '1' MINUTE PRECEDING);
```

Ejemplo 2: Consulta que aplica agregados en una ventana deslizante

La siguiente consulta sobre la secuencia de demostración, devuelve el promedio del porcentaje de cambio en el precio de cada símbolo, en una ventana de 10 segundos.

```
SELECT STREAM Ticker_Symbol,
              AVG(Change / (Price - Change)) over W1 as Avg_Percent_Change
FROM "SOURCE_SQL_STREAM_001"
WINDOW W1 AS (
        PARTITION BY ticker_symbol
        RANGE INTERVAL '10' SECOND PRECEDING);
```

Para probar la consulta

1. Configure una aplicación siguiendo el Ejercicio de introducción.
2. Sustituya la instrucción SELECT en el código de la aplicación por la consulta SELECT anterior. A continuación se muestra el código de la aplicación resultante.

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
                            ticker_symbol VARCHAR(10),
                            Avg_Percent_Change double);
CREATE OR REPLACE PUMP "STREAM_PUMP" AS
   INSERT INTO "DESTINATION_SQL_STREAM"
      SELECT STREAM Ticker_Symbol,
                    AVG(Change / (Price - Change)) over W1 as Avg_Percent_Change
      FROM "SOURCE_SQL_STREAM_001"
      WINDOW W1 AS (
              PARTITION BY ticker_symbol
              RANGE INTERVAL '10' SECOND PRECEDING);
```

Ejemplo 3: Consultar datos de varias ventanas deslizantes en la misma secuencia

Puede escribir consultas de modo que emitan una salida en la cual cada valor de columna se calcule mediante diferentes ventanas deslizantes definidas para la misma secuencia.

En el siguiente ejemplo, la consulta emite el símbolo de salida, el precio, a2 y a10. Emite una salida para los símbolos de cotización cuya media móvil de dos filas cruza la media móvil de diez filas. Los valores de columna a2 y a10 se derivan de ventanas deslizantes de dos y diez filas.

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
                           ticker_symbol      VARCHAR(12),
                           price              double,
                           average_last2rows  double,
                           average_last10rows double);

CREATE OR REPLACE PUMP "myPump" AS INSERT INTO "DESTINATION_SQL_STREAM"
SELECT STREAM ticker_symbol,
              price,
              avg(price) over last2rows,
              avg(price) over last10rows
FROM SOURCE_SQL_STREAM_001
WINDOW
    last2rows AS (PARTITION BY ticker_symbol ROWS 2 PRECEDING),
    last10rows AS (PARTITION BY ticker_symbol ROWS 10 PRECEDING);
```

Para probar esta consulta con la secuencia de demostración, siga el procedimiento descrito en Ejemplo 1.

Operaciones de streaming de datos: uniones de secuencias

Puede tener varias secuencias en la aplicación en su aplicación. Puede escribir `JOIN` consultas para evaluar los datos de estas secuencias. Suponga, por ejemplo, que tiene la siguiente secuencia en la aplicación:

- OrderStream— Recibe los pedidos de existencias que se están realizando.

```
(orderId SqlType, ticker SqlType, amount SqlType, ROWTIME TimeStamp)
```

- TradeStream— Recibe las operaciones bursátiles resultantes de esos pedidos.

```
(tradeId SqlType, orderId SqlType, ticker SqlType, amount SqlType, ticker SqlType,
 amount SqlType, ROWTIME TimeStamp)
```

Los siguientes son ejemplos de consultas `JOIN` que correlacionan datos en estas secuencias.

Ejemplo 1: Informar sobre las órdenes que se negocian en menos de un minuto

En este ejemplo, su consulta une `OrderStream` y `TradeStream`. No obstante, como solo queremos operaciones que tienen lugar en el minuto siguiente a las órdenes, la consulta define la ventana de 1 minuto para `TradeStream`. Para obtener información sobre consultas en ventana, consulte Ventanas deslizantes.

```
SELECT STREAM
     ROWTIME,
     o.orderId, o.ticker, o.amount AS orderAmount,
     t.amount AS tradeAmount
FROM OrderStream AS o
JOIN TradeStream OVER (RANGE INTERVAL '1' MINUTE PRECEDING) AS t
ON   o.orderId = t.orderId;
```

Puede definir las ventanas de forma explícita utilizando la cláusula `WINDOW` y escribiendo las consultas anteriores de la siguiente manera:

```
SELECT STREAM
    ROWTIME,
    o.orderId, o.ticker, o.amount AS orderAmount,
    t.amount AS tradeAmount
FROM OrderStream AS o
JOIN TradeStream OVER t
ON o.orderId = t.orderId
WINDOW t AS
    (RANGE INTERVAL '1' MINUTE PRECEDING)
```

Al incluir esta consulta en el código de la aplicación, este se ejecuta de forma continua. Para cada uno de los registros que llegan en `OrderStream`, la aplicación emite una salida si hay operaciones dentro de la ventana de 1 minuto a partir del momento en que se cursó la orden.

La unión en las consultas anteriores es una unión interna donde la consulta emite los registros en `OrderStream` para los que existe un registro en `TradeStream` (y viceversa). Con una unión exterior puede crear otro escenario interesante. Supongamos que desea órdenes bursátiles para las que no se han producido operaciones al cabo de un minuto de emitir la orden y también operaciones comunicadas dentro de la misma ventana, pero para otras órdenes. Este es un ejemplo de unión exterior.

```
SELECT STREAM
    ROWTIME,
    o.orderId, o.ticker, o.amount AS orderAmount,
    t.ticker, t.tradeId, t.amount AS tradeAmount,
FROM OrderStream AS o
LEFT OUTER JOIN TradeStream OVER (RANGE INTERVAL '1' MINUTE PRECEDING) AS t
ON    o.orderId = t.orderId;
```

Ejemplos de Kinesis Data Analytics para SQL

En esta sección se proporcionan ejemplos de cómo crear y utilizar aplicaciones en Amazon Kinesis Data Analytics. Incluyen ejemplos de código e step-by-step instrucciones que le ayudarán a crear aplicaciones de Kinesis Data Analytics y a probar sus resultados.

Antes de explorar estos ejemplos, le recomendamos que en primer lugar examine Aplicaciones de Amazon Kinesis Data Analytics para SQL: cómo funciona y Introducción a aplicaciones de Amazon Kinesis Data Analytics para SQL.

Temas

Ejemplos: Transformación de datos

Hay ocasiones en que el código de la aplicación debe realizar un procesamiento previo de los registros de entrada antes de que se realice ningún análisis en Amazon Kinesis Data Analytics. Esto puede ocurrir por varias razones, como, por ejemplo, que los registros no cumplan con los formatos de registro admitidos, lo que puede derivar en columnas sin normalizar en las secuencias de entrada en la aplicación.

Esta sección proporciona ejemplos de cómo utilizar las funciones de cadena disponibles para normalizar los datos, cómo extraer información necesaria a partir de columnas de cadena, y así sucesivamente. La sección también menciona funciones de fecha y hora que podrían resultarle de utilidad.

Procesamiento de secuencias con Lambda

Para obtener información sobre el preprocesamiento de transmisiones con AWS Lambda, consulteProcesamiento previo de registros con una función de Lambda.

Temas

Ejemplos: Transformación de valores de cadena

Amazon Kinesis Data Analytics admite formatos como JSON y CSV en los registros de un origen de streaming. Para obtener más información, consulte RecordFormat. Estos registros se asignan a filas en una secuencia en la aplicación según la configuración de entrada. Para obtener más información, consulte Configuración de entrada de la aplicación. La configuración de entrada especifica cómo se mapean los campos de los registros del origen de streaming a las columnas en una secuencia en la aplicación.

Este mapeo funciona cuando los registros del origen de streaming tienen formatos compatibles, lo que da como resultado una secuencia en la aplicación con datos normalizados. Pero, ¿qué sucede si los datos en el origen de streaming no se ajustan a los estándares admitidos? Por ejemplo, ¿qué sucede si el origen de streaming contiene datos de secuencias de clics, sensores de IoT y registros de aplicaciones?

Estudie estos ejemplos:

- El origen de streaming contiene registros de aplicaciones: los registros de aplicaciones tienen el formato de registro estándar de Apache y se escriben en la secuencia utilizando el formato JSON.

  ```
  {
     "Log":"192.168.254.30 - John [24/May/2004:22:01:02 -0700] "GET /icons/
  apache_pb.gif HTTP/1.1" 304 0"
  }
  ```

  Para obtener más información sobre el formato de registro estándar de Apache, consulte la sección de archivos de registro en el sitio web de Apache.

- El origen de streaming contiene datos semiestructurados: en el siguiente ejemplo se muestran dos registros. El valor del campo `Col_E_Unstructured` es una serie de valores separados por comas. Existen cinco columnas: las cuatro primeras tienen valores de tipo cadena y la última columna contiene valores separados por comas.

```
{ "Col_A" : "string",
  "Col_B" : "string",
  "Col_C" : "string",
  "Col_D" : "string",
  "Col_E_Unstructured" : "value,value,value,value"}

{ "Col_A" : "string",
  "Col_B" : "string",
  "Col_C" : "string",
  "Col_D" : "string",
  "Col_E_Unstructured" : "value,value,value,value"}
```

- Los registros de tu fuente de streaming contienen URLs, y necesitas, una parte del nombre de dominio URL para el análisis.

```
{ "referrer" : "http://www.amazon.com"}
{ "referrer" : "http://www.stackoverflow.com" }
```

En estos casos, el siguiente proceso de dos pasos por lo general funciona para crear secuencias en la aplicación que contienen datos normalizados:

1. Configure la entrada de la aplicación para asignar el campo sin estructurar a una columna del tipo de `VARCHAR(N)` en la secuencia de entrada en la aplicación que se crea.
2. En el código de la aplicación, utilice funciones de cadena para dividir esta columna en varias columnas y luego guarde las filas en otra secuencia en la aplicación. La secuencia en la aplicación que ha creado el código de su aplicación tendrá datos normalizados. Por lo tanto, podrá realizar análisis con esta secuencia en la aplicación.

Amazon Kinesis Data Analytics proporciona las siguientes operaciones de cadena, funciones SQL estándar y extensiones de SQL estándar para trabajar con columnas de cadena:

- Operadores de cadenas: operadores como, por ejemplo, `LIKE` y `SIMILAR` son útiles para comparar cadenas. Para obtener más información, consulte Operadores de cadena en la Referencia de SQL de Amazon Managed Service para Apache Flink.
- Funciones SQL: las siguientes funciones resultan útiles para manipular las cadenas individuales. Para obtener más información, consulte Funciones de cadena y búsqueda en la Referencia de SQL de Amazon Managed Service para Apache Flink.

- `CHAR_LENGTH`: devuelve la longitud de una cadena.
- `INITCAP`: devuelve una versión convertida de la cadena de entrada para que el primer carácter de cada palabra delimitada por espacios utilice mayúsculas y todos los demás caracteres utilicen minúsculas.
- `LOWER/UPPER`: convierte una cadena en minúsculas o mayúsculas.
- `OVERLAY`: reemplaza una parte del argumento de la primera cadena (la cadena original) por el argumento de la segunda cadena (la cadena de sustitución).
- `POSITION`: busca una cadena dentro de otra.
- `REGEX_REPLACE`: reemplaza una subcadena por otra.
- `SUBSTRING` (SUBCADENA): extrae una parte de una cadena de origen a partir de una posición específica.
- `TRIM`: elimina instancias del carácter especificado del principio o del final de la cadena de origen.

- Extensiones SQL: son útiles para trabajar con cadenas no estructuradas, como registros y URIs. Para obtener más información, consulte Funciones de análisis de registros en la Referencia de SQL de Amazon Managed Service para Apache Flink.
  - `FAST_REGEX_LOG_PARSER`: funciona de manera similar al analizador regex, pero se necesitan varios "atajos" para garantizar resultados más rápidos. Por ejemplo, el analizador Fast Regex se detiene en la primera coincidencia que encuentra (conocido como semántica perezosa).
  - `FIXED_COLUMN_LOG_PARSE` – Analiza los campos de ancho fijo y los convierte automáticamente en los tipos de SQL dados.
  - `REGEX_LOG_PARSE`: analiza una cadena basándose en los patrones de expresiones regulares de Java predeterminados.
  - `SYS_LOG_PARSE`— Analiza las entradas que se encuentran habitualmente en los registros UNIX/Linux del sistema.
  - `VARIABLE_COLUMN_LOG_PARSE`: divide una cadena de entrada en campos separados por un carácter delimitador o una cadena delimitadora.
  - `W3C_LOG_PARSE`: se puede utilizar para formatear rápidamente los logs de Apache.

Para ver ejemplos de uso de estas funciones, consulte los temas siguientes:

Temas

- Ejemplo: Extracción de una parte de una cadena (función SUBSTRING)

- Ejemplo: Sustitución de una subcadena mediante Regex (función REGEX_REPLACE)
- Ejemplo: Análisis de cadenas de log basado en expresiones regulares (función REGEX_LOG_PARSE)
- Ejemplo: Análisis de logs web (función W3C_LOG_PARSE)
- Ejemplo: División de cadenas en varios campos (función VARIABLE_COLUMN_LOG_PARSE)

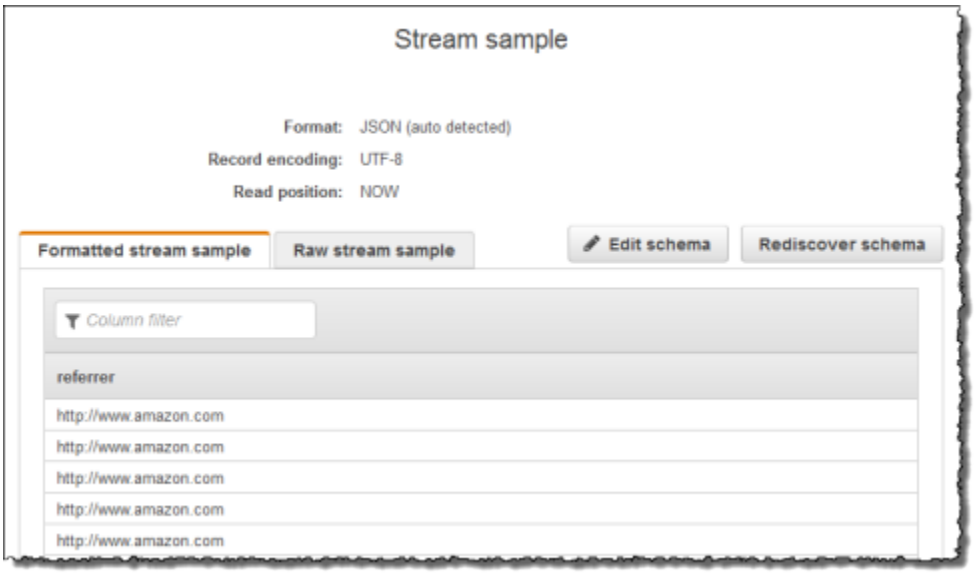
Ejemplo: Extracción de una parte de una cadena (función SUBSTRING)

Este ejemplo utiliza la función `SUBSTRING` para transformar una cadena en Amazon Kinesis Data Analytics. La función `SUBSTRING` extrae una parte de una cadena de origen a partir de una posición específica. Para obtener más información, consulte SUBSTRING en la Referencia de SQL de Amazon Managed Service para Apache Flink.

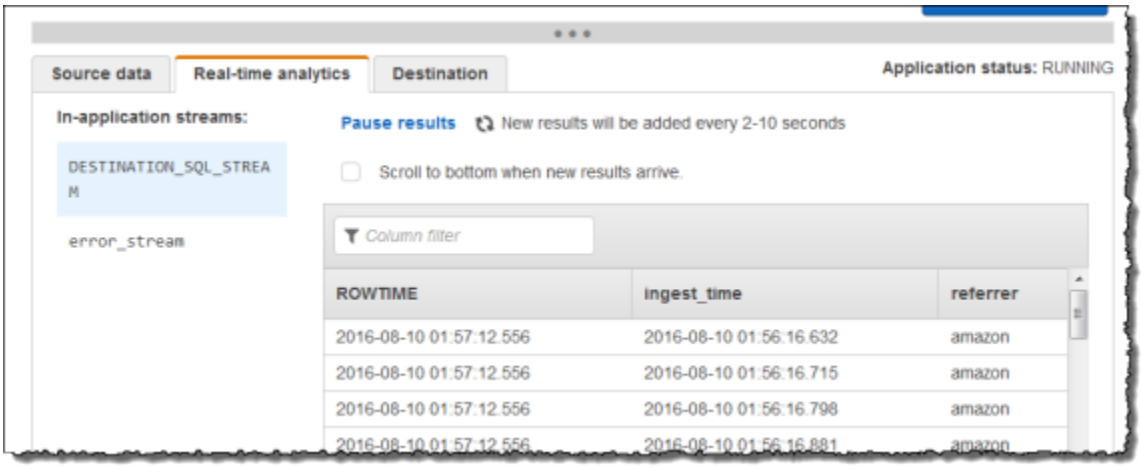
En este ejemplo, escribirá los siguientes registros en un flujo de datos de Amazon Kinesis.

```
{ "REFERRER" : "http://www.amazon.com" }
{ "REFERRER" : "http://www.amazon.com"}
{ "REFERRER" : "http://www.amazon.com"}
...
```

A continuación, creará una aplicación de análisis de datos de Kinesis Data Analytics en la consola, con el flujo de datos de Kinesis como origen de streaming. El proceso de detección lee los registros de muestra en el origen de streaming e infiere un esquema en la aplicación con una columna (`REFERRER`), tal como se muestra a continuación.



A continuación, utilizará el código de la aplicación con la función SUBSTRING para analizar la cadena de URL para recuperar el nombre de la empresa. Por último, insertará los datos resultantes en otra secuencia en la aplicación, tal y como se muestra a continuación:



Temas

- Paso 1: crear un flujo de datos de Kinesis
- Paso 2: creación de una aplicación de Kinesis Data Analytics

Paso 1: crear un flujo de datos de Kinesis

Cree un flujo de datos de Amazon Kinesis y rellene los registros del log como se indica a continuación:

1. Inicie sesión en la consola de Kinesis Consola de administración de AWS y ábrala en https://console.aws.amazon.com /kinesis.
2. Elija Flujos de datos en el panel de navegación.
3. Elija Crear secuencia de Kinesis y a continuación cree una secuencia con una partición. Para obtener más información, consulte Crear secuencia en la Guía para desarrolladores de Amazon Kinesis Data Streams.
4. Ejecute el siguiente código de Python para rellenar los registros de logs de muestra. Este código de ejemplo escribe continuamente el mismo registro de log en la secuencia.

```
import json
import boto3
```

```
STREAM_NAME = "ExampleInputStream"


def get_data():
    return {"REFERRER": "http://www.amazon.com"}


def generate(stream_name, kinesis_client):
    while True:
        data = get_data()
        print(data)
        kinesis_client.put_record(
            StreamName=stream_name, Data=json.dumps(data),
 PartitionKey="partitionkey"
        )


if __name__ == "__main__":
    generate(STREAM_NAME, boto3.client("kinesis"))
```

Paso 2: creación de una aplicación de Kinesis Data Analytics

A continuación, cree una aplicación de análisis de datos de Kinesis Data Analytics de la siguiente manera:

1. Abra la consola de Managed Service for Apache Flink en /kinesisanalytics. https://console.aws.amazon.com
2. Elija Create application (Crear aplicación), escriba el nombre de la aplicación y elija Create application (Crear aplicación).
3. En la página de detalles de la aplicación, elija Conectar datos de streaming.
4. En la página Connect to source (Conectarse al origen), haga lo siguiente:
   a. Elija la secuencia que ha creado en la sección anterior.
   b. Elija la opción para crear un rol de IAM.
   c. Seleccione Detectar esquema. Espere a que la consola muestre el esquema inferido y los registros de muestra utilizados para inferir en el esquema de la secuencia en la aplicación que ha creado. El esquema inferido solo tiene una columna.
   d. Elija Guardar y continuar.

5. En la página de detalles de la aplicación, elija Go to SQL editor (Ir al editor de SQL). Para iniciar la aplicación, elija Yes, start application (Sí, iniciar la aplicación) en el cuadro de diálogo que aparece.
6. En el editor de SQL, escriba el código de la aplicación y verifique los resultados como se indica a continuación:

   a. Copie el siguiente código de la aplicación y péguelo en el editor.

```
-- CREATE OR REPLACE STREAM for cleaned up referrer
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
    "ingest_time" TIMESTAMP,
    "referrer" VARCHAR(32));

CREATE OR REPLACE PUMP "myPUMP" AS
   INSERT INTO "DESTINATION_SQL_STREAM"
      SELECT STREAM
         "APPROXIMATE_ARRIVAL_TIME",
         SUBSTRING("referrer", 12, (POSITION('.com' IN "referrer") -
 POSITION('www.' IN "referrer") - 4))
      FROM "SOURCE_SQL_STREAM_001";
```

   b. Elija Save and run SQL. En la pestaña Real-time analytics (Análisis en tiempo real), puede ver todas las secuencias en la aplicación creadas por esta y comprobar los datos.

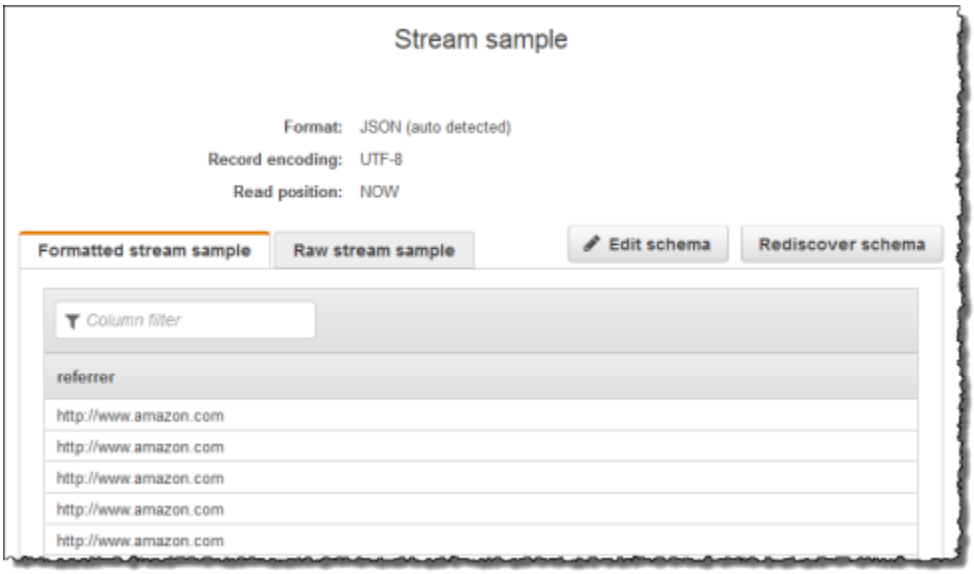
Ejemplo: Sustitución de una subcadena mediante Regex (función REGEX_REPLACE)

En este ejemplo se utiliza la función `REGEX_REPLACE` para transformar una cadena en Amazon Kinesis Data Analytics. `REGEX_REPLACE` reemplaza una subcadena por una subcadena alternativa. Para obtener más información, consulte REGEX_REPLACE en la Referencia de SQL de Amazon Managed Service para Apache Flink.

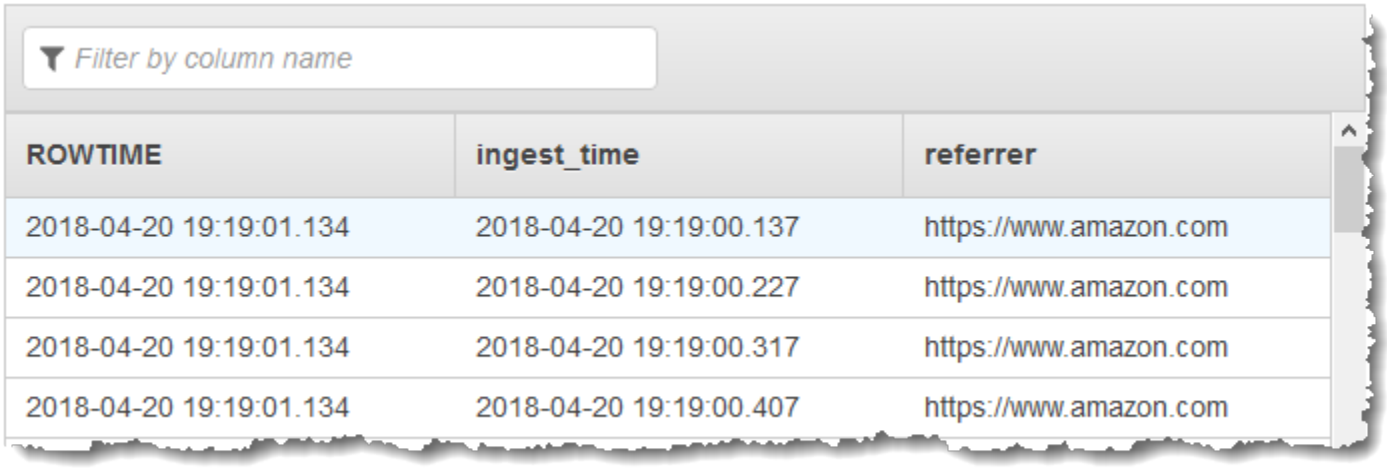
En este ejemplo, escribirá los siguientes registros en un flujo de datos de Amazon Kinesis.

```
{ "REFERRER" : "http://www.amazon.com" }
{ "REFERRER" : "http://www.amazon.com"}
{ "REFERRER" : "http://www.amazon.com"}
...
```

A continuación, creará una aplicación de análisis de datos de Kinesis Data Analytics en la consola, con el flujo de datos de Kinesis como origen de streaming. El proceso de detección lee los registros de muestra en el origen de streaming e infiere un esquema en la aplicación con una columna (REFERRER), tal como se muestra a continuación.



A continuación, utilizará el código de la aplicación con la función `REGEX_REPLACE` para convertir la URL de forma que use `https://` en lugar de `http://`. Por último, insertará los datos resultantes en otra secuencia en la aplicación, tal y como se muestra a continuación:



Temas

- Paso 1: crear un flujo de datos de Kinesis
- Paso 2: creación de una aplicación de Kinesis Data Analytics

Paso 1: crear un flujo de datos de Kinesis

Cree un flujo de datos de Amazon Kinesis y rellene los registros del log como se indica a continuación:

1. Inicie sesión en la consola de Kinesis Consola de administración de AWS y ábrala en https://console.aws.amazon.com /kinesis.
2. Elija Flujos de datos en el panel de navegación.
3. Elija Crear secuencia de Kinesis y a continuación cree una secuencia con una partición. Para obtener más información, consulte Crear secuencia en la Guía para desarrolladores de Amazon Kinesis Data Streams.
4. Ejecute el siguiente código de Python para rellenar los registros de logs de muestra. Este código de ejemplo escribe continuamente el mismo registro de log en la secuencia.

```
import json
import boto3

STREAM_NAME = "ExampleInputStream"


def get_data():
    return {"REFERRER": "http://www.amazon.com"}


def generate(stream_name, kinesis_client):
    while True:
        data = get_data()
        print(data)
        kinesis_client.put_record(
            StreamName=stream_name, Data=json.dumps(data),
 PartitionKey="partitionkey"
        )


if __name__ == "__main__":
    generate(STREAM_NAME, boto3.client("kinesis"))
```

Paso 2: creación de una aplicación de Kinesis Data Analytics

A continuación, cree una aplicación de análisis de datos de Kinesis Data Analytics de la siguiente manera:

1. Abra la consola de Managed Service for Apache Flink en /kinesisanalytics. https://console.aws.amazon.com
2. Elija Create application (Crear aplicación), escriba el nombre de la aplicación y elija Create application (Crear aplicación).
3. En la página de detalles de la aplicación, elija Conectar datos de streaming.
4. En la página Connect to source (Conectarse al origen), haga lo siguiente:
   a. Elija la secuencia que ha creado en la sección anterior.
   b. Elija la opción para crear un rol de IAM.
   c. Seleccione Detectar esquema. Espere a que la consola muestre el esquema inferido y los registros de muestra utilizados para inferir en el esquema de la secuencia en la aplicación que ha creado. El esquema inferido solo tiene una columna.
   d. Elija Guardar y continuar.
5. En la página de detalles de la aplicación, elija Go to SQL editor (Ir al editor de SQL). Para iniciar la aplicación, elija Yes, start application (Sí, iniciar la aplicación) en el cuadro de diálogo que aparece.
6. En el editor de SQL, escriba el código de la aplicación y verifique los resultados como se indica a continuación:
   a. Copie el siguiente código de la aplicación y péguelo en el editor:

```
-- CREATE OR REPLACE STREAM for cleaned up referrer
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
    "ingest_time" TIMESTAMP,
    "referrer" VARCHAR(32));

CREATE OR REPLACE PUMP "myPUMP" AS
   INSERT INTO "DESTINATION_SQL_STREAM"
      SELECT STREAM
         "APPROXIMATE_ARRIVAL_TIME",
         REGEX_REPLACE("REFERRER", 'http://', 'https://', 1, 0)
```

```
        FROM "SOURCE_SQL_STREAM_001";
```

b. Elija Save and run SQL. En la pestaña Real-time analytics (Análisis en tiempo real), puede ver todas las secuencias en la aplicación creadas por esta y comprobar los datos.

Ejemplo: Análisis de cadenas de log basado en expresiones regulares (función REGEX_LOG_PARSE)

En este ejemplo se utiliza la función `REGEX_LOG_PARSE` para transformar una cadena en Amazon Kinesis Data Analytics. `REGEX_LOG_PARSE` analiza una cadena en función de los patrones predeterminados de expresiones regulares de Java. Para obtener más información, consulte [REGEX_LOG_PARSE](#) en la Referencia de SQL de Amazon Managed Service para Apache Flink.

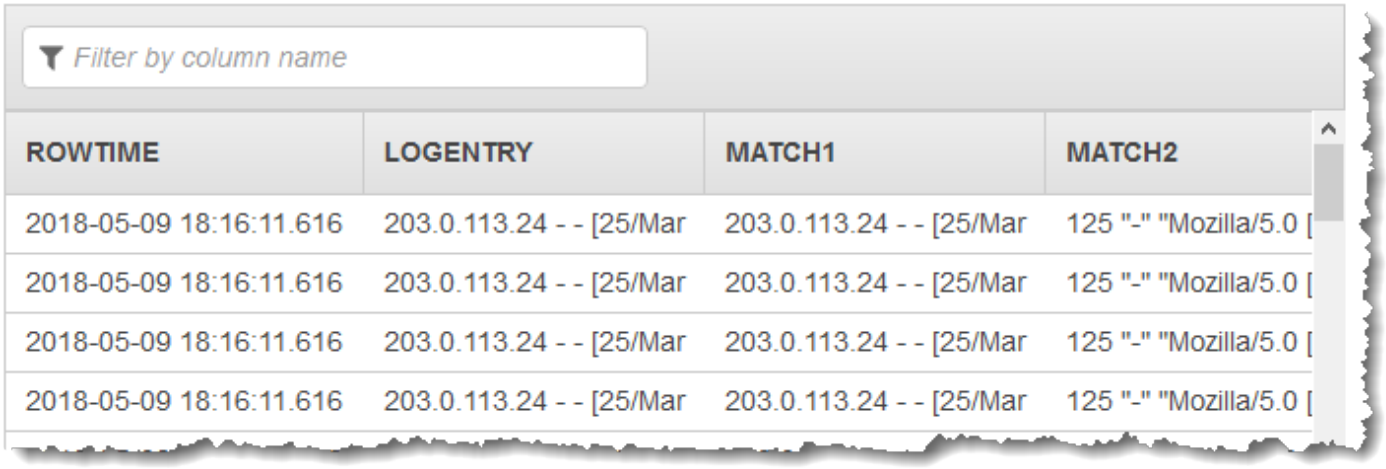
En este ejemplo, escribirá los siguientes registros en un flujo de datos de Amazon Kinesis.

```
{"LOGENTRY": "203.0.113.24 - - [25/Mar/2018:15:25:37 -0700] \"GET /index.php HTTP/1.1\"
 200 125 \"-\" \"Mozilla/5.0 [en] Gecko/20100101 Firefox/52.0\""}
{"LOGENTRY": "203.0.113.24 - - [25/Mar/2018:15:25:37 -0700] \"GET /index.php HTTP/1.1\"
 200 125 \"-\" \"Mozilla/5.0 [en] Gecko/20100101 Firefox/52.0\""}
{"LOGENTRY": "203.0.113.24 - - [25/Mar/2018:15:25:37 -0700] \"GET /index.php HTTP/1.1\"
 200 125 \"-\" \"Mozilla/5.0 [en] Gecko/20100101 Firefox/52.0\""}
...
```

A continuación, creará una aplicación de análisis de datos de Kinesis Data Analytics en la consola, con el flujo de datos de Kinesis como origen de streaming. El proceso de detección lee los registros de muestra en el origen de streaming e infiere un esquema en la aplicación con una columna (LOGENTRY), tal como se muestra a continuación.

ROWTIME TIMESTAMP	LOGENTRY VARCHAR(256)
2018-05-09 18:12:18.552	203.0.113.24 - - [25/Mar/2018:15:25:37 -0700] "GET /index.php HTTP/1.1"
2018-05-09 18:12:18.552	203.0.113.24 - - [25/Mar/2018:15:25:37 -0700] "GET /index.php HTTP/1.1"
2018-05-09 18:12:18.552	203.0.113.24 - - [25/Mar/2018:15:25:37 -0700] "GET /index.php HTTP/1.1"
2018-05-09 18:12:18.552	203.0.113.24 - - [25/Mar/2018:15:25:37 -0700] "GET /index.php HTTP/1.1"

A continuación, utilizará el código de la aplicación con la función REGEX_LOG_PARSE para analizar la cadena de log con el fin de recuperar los elementos de datos. Por último, insertará los datos resultantes en otra secuencia en la aplicación, tal y como se muestra en la siguiente captura de pantalla:



Temas

- Paso 1: crear un flujo de datos de Kinesis
- Paso 2: creación de una aplicación de Kinesis Data Analytics

Paso 1: crear un flujo de datos de Kinesis

Cree un flujo de datos de Amazon Kinesis y rellene los registros del log como se indica a continuación:

1. Inicie sesión en la consola de Kinesis Consola de administración de AWS y ábrala en https://console.aws.amazon.com /kinesis.
2. Elija Flujos de datos en el panel de navegación.
3. Elija Crear secuencia de Kinesis y a continuación cree una secuencia con una partición. Para obtener más información, consulte Crear secuencia en la Guía para desarrolladores de Amazon Kinesis Data Streams.
4. Ejecute el siguiente código de Python para rellenar los registros de logs de muestra. Este código de ejemplo escribe continuamente el mismo registro de log en la secuencia.

```
import json
```

```
import boto3

STREAM_NAME = "ExampleInputStream"


def get_data():
    return {
        "LOGENTRY": "203.0.113.24 - - [25/Mar/2018:15:25:37 -0700] "
        '"GET /index.php HTTP/1.1" 200 125 "-" '
        '"Mozilla/5.0 [en] Gecko/20100101 Firefox/52.0"'
    }


def generate(stream_name, kinesis_client):
    while True:
        data = get_data()
        print(data)
        kinesis_client.put_record(
            StreamName=stream_name, Data=json.dumps(data),
 PartitionKey="partitionkey"
        )


if __name__ == "__main__":
    generate(STREAM_NAME, boto3.client("kinesis"))
```

Paso 2: creación de una aplicación de Kinesis Data Analytics

A continuación, cree una aplicación de análisis de datos de Kinesis Data Analytics de la siguiente manera:

1. Abra la consola de Managed Service for Apache Flink en /kinesisanalytics. https://console.aws.amazon.com
2. Elija Create application y especifique un nombre de aplicación.
3. En la página de detalles de la aplicación, elija Conectar datos de streaming.
4. En la página Connect to source (Conectarse al origen), haga lo siguiente:
   a. Elija la secuencia que ha creado en la sección anterior.
   b. Elija la opción para crear un rol de IAM.

c. Seleccione Detectar esquema. Espere a que la consola muestre el esquema inferido y los registros de muestra utilizados para inferir en el esquema de la secuencia en la aplicación que ha creado. El esquema inferido solo tiene una columna.

d. Elija Guardar y continuar.

5. En la página de detalles de la aplicación, elija Go to SQL editor (Ir al editor de SQL). Para iniciar la aplicación, elija Yes, start application (Sí, iniciar la aplicación) en el cuadro de diálogo que aparece.

6. En el editor de SQL, escriba el código de la aplicación y verifique los resultados como se indica a continuación:

a. Copie el siguiente código de la aplicación y péguelo en el editor.

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (logentry VARCHAR(24), match1
 VARCHAR(24), match2 VARCHAR(24));

CREATE OR REPLACE PUMP "STREAM_PUMP" AS INSERT INTO "DESTINATION_SQL_STREAM"
    SELECT STREAM T.LOGENTRY, T.REC.COLUMN1, T.REC.COLUMN2
    FROM
         (SELECT STREAM LOGENTRY,
             REGEX_LOG_PARSE(LOGENTRY, '(\w.+) (\d.+) (\w.+) (\w.+)') AS REC
             FROM SOURCE_SQL_STREAM_001) AS T;
```

b. Elija Save and run SQL. En la pestaña Real-time analytics (Análisis en tiempo real), puede ver todas las secuencias en la aplicación creadas por esta y comprobar los datos.

Ejemplo: Análisis de logs web (función W3C_LOG_PARSE)

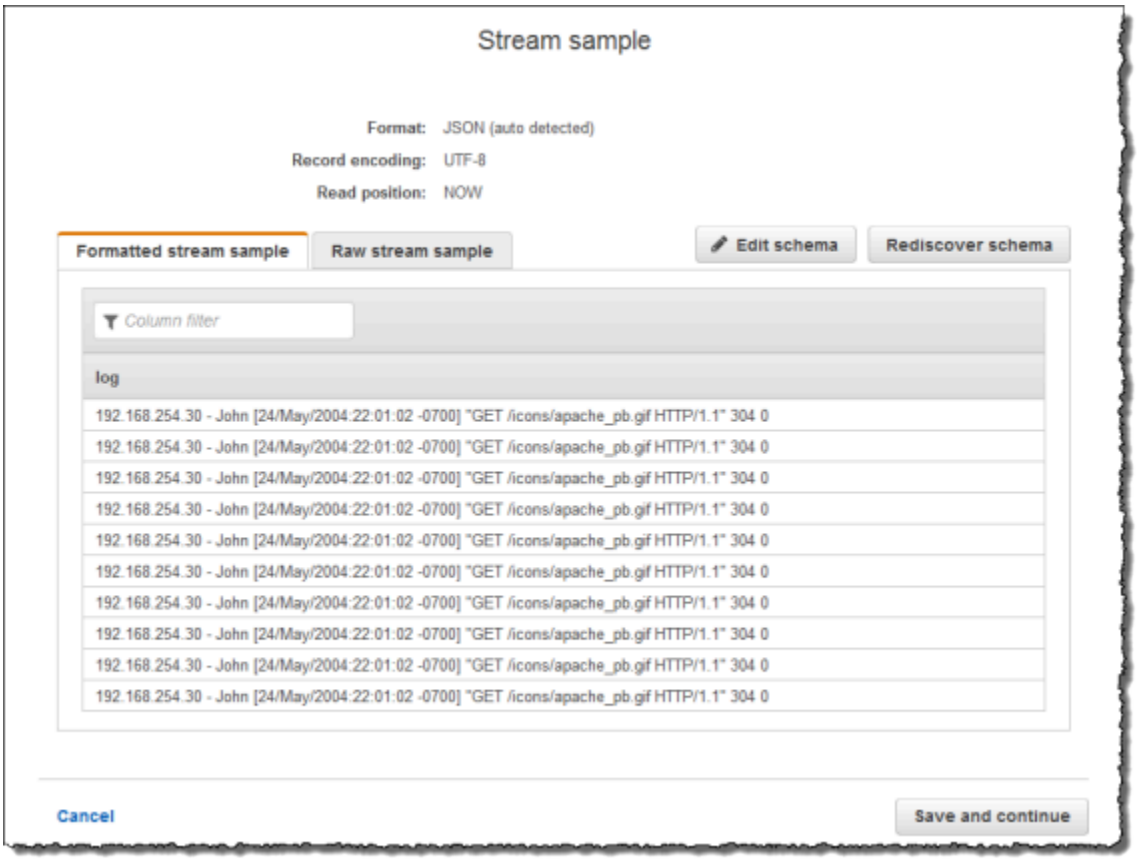
Este ejemplo utiliza la función `W3C_LOG_PARSE` para transformar una cadena en Amazon Kinesis Data Analytics. Puede utilizar `W3C_LOG_PARSE` para formatear rápidamente los logs de Apache. Para obtener más información, consulte W3C_LOG_PARSE en la Referencia de SQL de Amazon Managed Service para Apache Flink.

En este ejemplo escribirá los registros de logs en un flujo de datos de Amazon Kinesis. Estos son los ejemplos de registros:

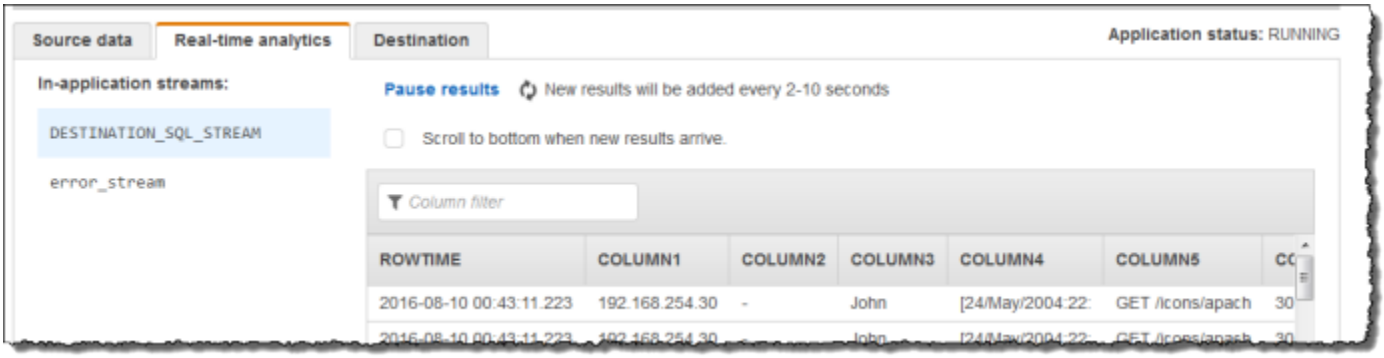
```
{"Log":"192.168.254.30 - John [24/May/2004:22:01:02 -0700] "GET /icons/apache_pba.gif
 HTTP/1.1" 304 0"}
```

```
{"Log":"192.168.254.30 - John [24/May/2004:22:01:03 -0700] "GET /icons/apache_pbb.gif
 HTTP/1.1" 304 0"}
{"Log":"192.168.254.30 - John [24/May/2004:22:01:04 -0700] "GET /icons/apache_pbc.gif
 HTTP/1.1" 304 0"}
...
```

A continuación, creará una aplicación de análisis de datos de Kinesis Data Analytics en la consola, con el flujo de datos de Kinesis como origen de streaming. El proceso de detección lee los registros de muestra en el origen de streaming e infiere un esquema en la aplicación con una columna (registro), tal como se muestra a continuación:



Luego, utiliza el código de la aplicación con la función `W3C_LOG_PARSE` para analizar el registro y crear otra secuencia en la aplicación con diferentes campos de registro en columnas independientes, como se muestra a continuación:



Temas

- [Paso 1: crear un flujo de datos de Kinesis]
- [Paso 2: creación de una aplicación de Kinesis Data Analytics]

Paso 1: crear un flujo de datos de Kinesis

Cree un flujo de datos de Amazon Kinesis y rellene los registros del log como se indica a continuación:

1. [Inicie sesión en la consola de Kinesis Consola de administración de AWS y ábrala en https://console.aws.amazon.com /kinesis.]
2. Elija Flujos de datos en el panel de navegación.
3. Elija Crear secuencia de Kinesis y a continuación cree una secuencia con una partición. Para obtener más información, consulte [Crear secuencia] en la Guía para desarrolladores de Amazon Kinesis Data Streams.
4. Ejecute el siguiente código de Python para rellenar los registros de logs de muestra. Este código de ejemplo escribe continuamente el mismo registro de log en la secuencia.

```
import json
import boto3

STREAM_NAME = "ExampleInputStream"


def get_data():
    return {
        "log": "192.168.254.30 - John [24/May/2004:22:01:02 -0700] "
        '"GET /icons/apache_pb.gif HTTP/1.1" 304 0'
```

```
    }


def generate(stream_name, kinesis_client):
    while True:
        data = get_data()
        print(data)
        kinesis_client.put_record(
            StreamName=stream_name, Data=json.dumps(data),
 PartitionKey="partitionkey"
        )


if __name__ == "__main__":
    generate(STREAM_NAME, boto3.client("kinesis"))
```

Paso 2: creación de una aplicación de Kinesis Data Analytics

Cree una aplicación de análisis de datos de Kinesis Data Analytics de la siguiente manera:

1. Abra la consola de Managed Service for Apache Flink en /kinesisanalytics. https://console.aws.amazon.com
2. Elija Create application (Crear aplicación), escriba el nombre de la aplicación y elija Create application (Crear aplicación).
3. En la página de detalles de la aplicación, elija Conectar datos de streaming.
4. En la página Connect to source (Conectarse al origen), haga lo siguiente:
   a. Elija la secuencia que ha creado en la sección anterior.
   b. Elija la opción para crear un rol de IAM.
   c. Seleccione Detectar esquema. Espere a que la consola muestre el esquema inferido y los registros de muestra utilizados para inferir en el esquema de la secuencia en la aplicación que ha creado. El esquema inferido solo tiene una columna.
   d. Elija Guardar y continuar.
5. En la página de detalles de la aplicación, elija Go to SQL editor (Ir al editor de SQL). Para iniciar la aplicación, elija Yes, start application (Sí, iniciar la aplicación) en el cuadro de diálogo que aparece.

6. En el editor de SQL, escriba el código de la aplicación y verifique los resultados como se indica a continuación:

   a. Copie el siguiente código de la aplicación y péguelo en el editor.

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
column1 VARCHAR(16),
column2 VARCHAR(16),
column3 VARCHAR(16),
column4 VARCHAR(16),
column5 VARCHAR(16),
column6 VARCHAR(16),
column7 VARCHAR(16));

CREATE OR REPLACE PUMP "myPUMP" AS
INSERT INTO "DESTINATION_SQL_STREAM"
        SELECT STREAM
            l.r.COLUMN1,
            l.r.COLUMN2,
            l.r.COLUMN3,
            l.r.COLUMN4,
            l.r.COLUMN5,
            l.r.COLUMN6,
            l.r.COLUMN7
        FROM (SELECT STREAM W3C_LOG_PARSE("log", 'COMMON')
              FROM "SOURCE_SQL_STREAM_001") AS l(r);
```

   b. Elija Save and run SQL. En la pestaña Real-time analytics (Análisis en tiempo real), puede ver todas las secuencias en la aplicación creadas por esta y comprobar los datos.

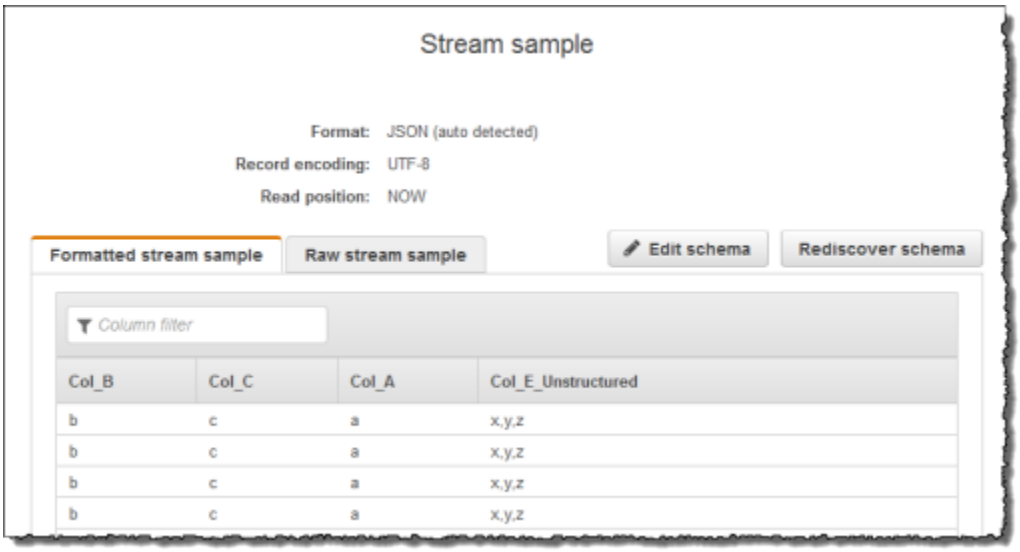
Ejemplo: División de cadenas en varios campos (función VARIABLE_COLUMN_LOG_PARSE)

En este ejemplo se utiliza la función `VARIABLE_COLUMN_LOG_PARSE` para manipular cadenas en Kinesis Data Analytics. `VARIABLE_COLUMN_LOG_PARSE` divide una cadena de entrada en campos separados por un carácter delimitador o una cadena delimitadora. Para obtener más información, consulte VARIABLE_COLUMN_LOG_PARSE en la Referencia de SQL de Amazon Managed Service para Apache Flink.

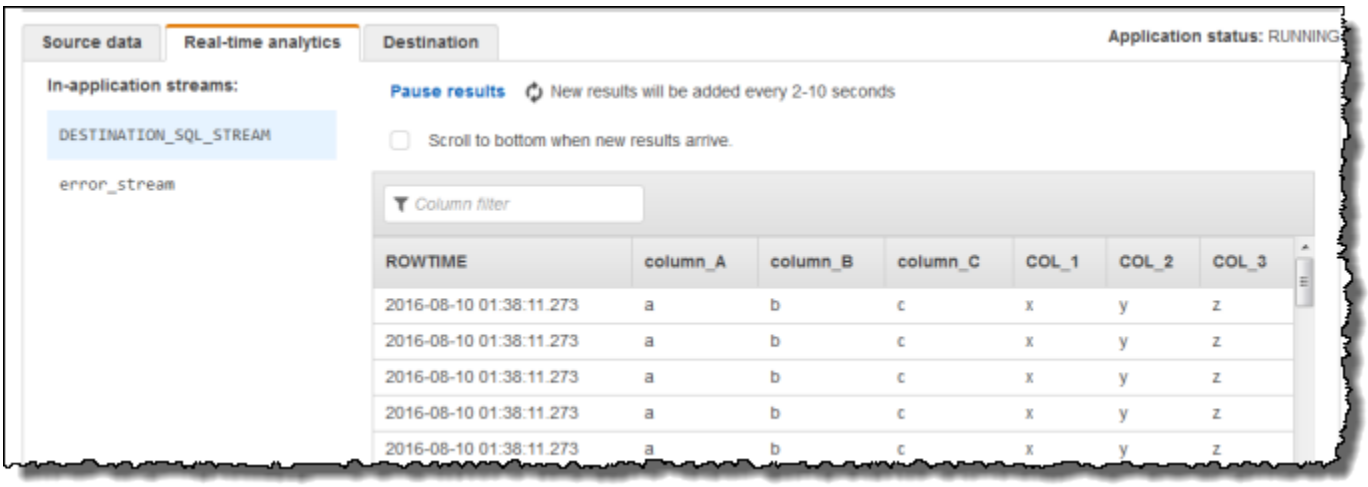
En este ejemplo, escriba los registros semiestructurados en un flujo de datos de Amazon Kinesis. Los registros de ejemplo son los siguientes:

```
{ "Col_A" : "string",
  "Col_B" : "string",
  "Col_C" : "string",
  "Col_D_Unstructured" : "value,value,value,value"}
{ "Col_A" : "string",
  "Col_B" : "string",
  "Col_C" : "string",
  "Col_D_Unstructured" : "value,value,value,value"}
```

A continuación, creará una aplicación de análisis de datos de Kinesis Data Analytics en la consola, con el flujo de datos de Kinesis como origen de streaming. El proceso de detección lee los registros de muestra en el origen de streaming e infiere un esquema en la aplicación con cuatro columnas, tal como se muestra a continuación:



A continuación, utiliza el código de la aplicación con la función `VARIABLE_COLUMN_LOG_PARSE` para analizar los valores separados por comas e insertar filas normalizadas en otra secuencia en la aplicación, como se muestra a continuación:



Temas

- Paso 1: crear un flujo de datos de Kinesis
- Paso 2: creación de una aplicación de Kinesis Data Analytics

Paso 1: crear un flujo de datos de Kinesis

Cree un flujo de datos de Amazon Kinesis y rellene los registros del log como se indica a continuación:

1. Inicie sesión en la consola de Kinesis Consola de administración de AWS y ábrala en https://console.aws.amazon.com /kinesis.
2. Elija Flujos de datos en el panel de navegación.
3. Elija Crear secuencia de Kinesis y a continuación cree una secuencia con una partición. Para obtener más información, consulte Crear secuencia en la Guía para desarrolladores de Amazon Kinesis Data Streams.
4. Ejecute el siguiente código de Python para rellenar los registros de logs de muestra. Este código de ejemplo escribe continuamente el mismo registro de log en la secuencia.

```
import json
import boto3

STREAM_NAME = "ExampleInputStream"


def get_data():
```

```
    return {"Col_A": "a", "Col_B": "b", "Col_C": "c", "Col_E_Unstructured":
 "x,y,z"}


def generate(stream_name, kinesis_client):
    while True:
        data = get_data()
        print(data)
        kinesis_client.put_record(
            StreamName=stream_name, Data=json.dumps(data),
 PartitionKey="partitionkey"
        )


if __name__ == "__main__":
    generate(STREAM_NAME, boto3.client("kinesis"))
```

Paso 2: creación de una aplicación de Kinesis Data Analytics

Cree una aplicación de análisis de datos de Kinesis Data Analytics de la siguiente manera:

1. Abra la consola de Managed Service for Apache Flink en /kinesisanalytics. https://console.aws.amazon.com
2. Elija Create application (Crear aplicación), escriba el nombre de la aplicación y elija Create application (Crear aplicación).
3. En la página de detalles de la aplicación, elija Conectar datos de streaming.
4. En la página Connect to source (Conectarse al origen), haga lo siguiente:
   a. Elija la secuencia que ha creado en la sección anterior.
   b. Elija la opción para crear un rol de IAM.
   c. Seleccione Detectar esquema. Espere a que la consola muestre el esquema inferido y los registros de muestra utilizados para inferir en el esquema de la secuencia en la aplicación que ha creado. Observe que el esquema inferido solo tiene una columna.
   d. Elija Guardar y continuar.

5. En la página de detalles de la aplicación, elija Go to SQL editor (Ir al editor de SQL). Para iniciar la aplicación, elija Yes, start application (Sí, iniciar la aplicación) en el cuadro de diálogo que aparece.
6. En el editor de SQL, escriba el código de la aplicación y verifique los resultados:
   a. Copie el siguiente código de la aplicación y péguelo en el editor:

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM"(
            "column_A" VARCHAR(16),
            "column_B" VARCHAR(16),
            "column_C" VARCHAR(16),
            "COL_1" VARCHAR(16),
            "COL_2" VARCHAR(16),
            "COL_3" VARCHAR(16));

CREATE OR REPLACE PUMP "SECOND_STREAM_PUMP" AS
INSERT INTO "DESTINATION_SQL_STREAM"
   SELECT STREAM  t."Col_A", t."Col_B", t."Col_C",
                  t.r."COL_1", t.r."COL_2", t.r."COL_3"
   FROM (SELECT STREAM
           "Col_A", "Col_B", "Col_C",
           VARIABLE_COLUMN_LOG_PARSE ("Col_E_Unstructured",
                                      'COL_1 TYPE VARCHAR(16), COL_2 TYPE
 VARCHAR(16), COL_3 TYPE VARCHAR(16)',
                                      ',') AS r
         FROM "SOURCE_SQL_STREAM_001") as t;
```

   b. Elija Save and run SQL. En la pestaña Real-time analytics (Análisis en tiempo real), puede ver todas las secuencias en la aplicación creadas por esta y comprobar los datos.

Ejemplo: transformación DateTime de valores

Amazon Kinesis Data Analytics admite la conversión de columnas en marcas temporales. Por ejemplo, es posible que desee utilizar su propia marca temporal como parte de una cláusula `GROUP BY` y utilizarla como otra ventana basada en el tiempo, además de la columna `ROWTIME`. Kinesis Data Analytics proporciona operaciones y funciones de SQL para trabajar con campos de fecha y hora.

- Operadores de fecha y hora: puede realizar operaciones aritméticas en tipos de datos de intervalos, fechas y tiempos. Para obtener más información, consulte Operadores de fecha, tiempos e intervalos en la Referencia de SQL de Amazon Managed Service para Apache Flink.

- Funciones de SQL: incluyen las siguientes. Para obtener más información, consulte Funciones de hora y tiempo en la Referencia de SQL de Amazon Managed Service para Apache Flink.
  - `EXTRACT()`: extrae un campo a partir de una expresión de intervalo, fecha, hora o marca temporal.
  - `CURRENT_TIME`: devuelve la hora a la que se ejecuta la consulta (UTC).
  - `CURRENT_DATE`: devuelve la fecha en la que se ejecuta la consulta (UTC).
  - `CURRENT_TIMESTAMP`: devuelve la marca temporal del momento en que se ejecuta la consulta (UTC).
  - `LOCALTIME`: devuelve la hora actual del momento en el que se ejecuta la consulta según lo establecido en el entorno en el que se está ejecutando Kinesis Data Analytics (UTC).
  - `LOCALTIMESTAMP`: devuelve la marca temporal actual según lo establecido en el entorno en el que se está ejecutando Kinesis Data Analytics (UTC).

- Extensiones de SQL: incluyen las siguientes. Para obtener más información, consulte Funciones de fecha y hora y Funciones de conversión de fecha y hora en la Referencia de SQL de Amazon Managed Service para Apache Flink.
  - `CURRENT_ROW_TIMESTAMP`: devuelve una nueva marca temporal para cada fila de la secuencia.
  - `TSDIFF`: devuelve la diferencia entre dos marcas temporales en milisegundos.
  - `CHAR_TO_DATE`: convierte una cadena en una fecha.
  - `CHAR_TO_TIME`: convierte una cadena en una hora.
  - `CHAR_TO_TIMESTAMP`: convierte una cadena en una marca temporal.
  - `DATE_TO_CHAR`: convierte una fecha en una cadena.
  - `TIME_TO_CHAR`: convierte una hora en una cadena.
  - `TIMESTAMP_TO_CHAR`: convierte una marca temporal en una cadena.

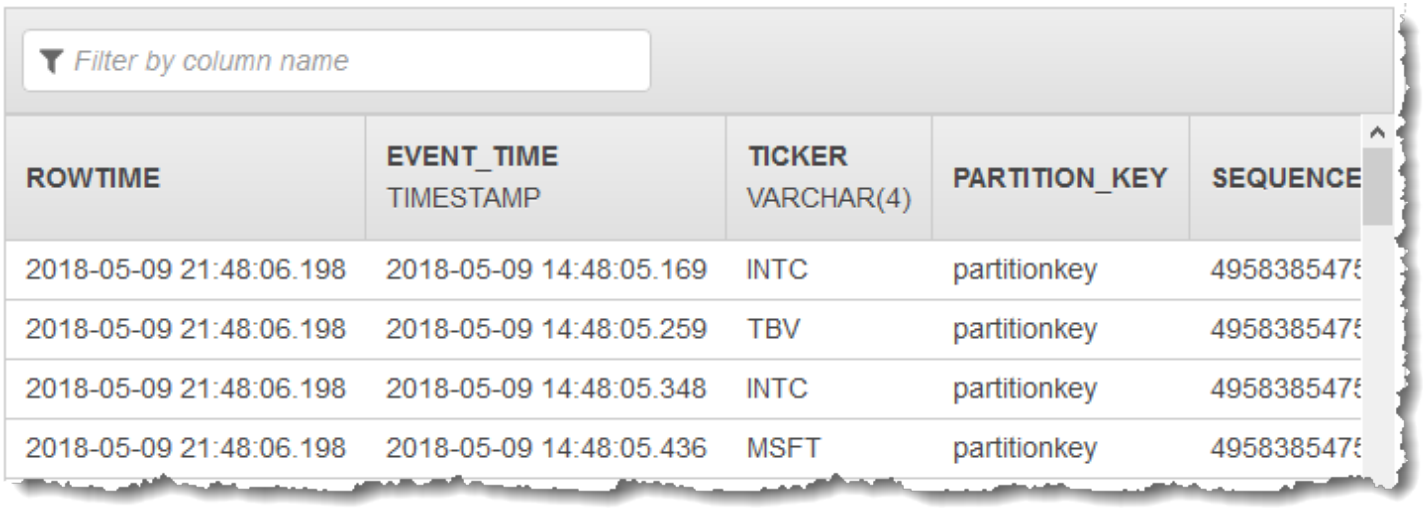
La mayoría de las funciones de SQL anteriores utilizan un formato para convertir las columnas. El formato es flexible. Por ejemplo, puede especificar el formato `yyyy-MM-dd hh:mm:ss` para convertir una cadena de entrada `2009-09-16 03:15:24` en una marca temporal. Para obtener más información, consulte Char To Timestamp(Sys) en la Referencia de SQL de Amazon Managed Service para Apache Flink.

Ejemplo: Transformación de fechas

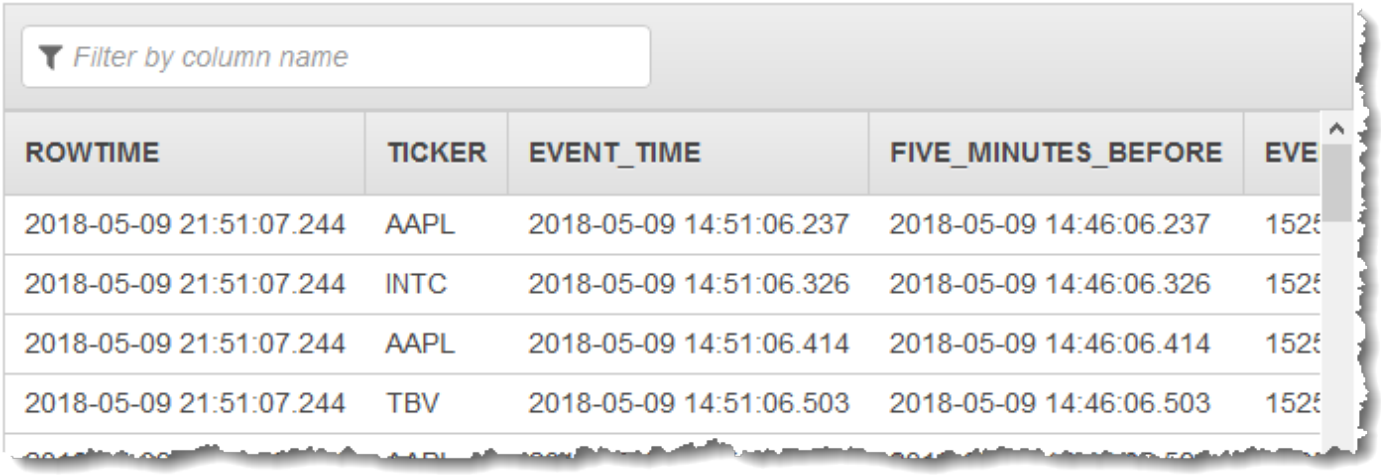
En este ejemplo, escribirá los siguientes registros en un flujo de datos de Amazon Kinesis.

```
{"EVENT_TIME": "2018-05-09T12:50:41.337510", "TICKER": "AAPL"}
{"EVENT_TIME": "2018-05-09T12:50:41.427227", "TICKER": "MSFT"}
{"EVENT_TIME": "2018-05-09T12:50:41.520549", "TICKER": "INTC"}
{"EVENT_TIME": "2018-05-09T12:50:41.610145", "TICKER": "MSFT"}
{"EVENT_TIME": "2018-05-09T12:50:41.704395", "TICKER": "AAPL"}
...
```

A continuación, creará una aplicación de análisis de datos de Kinesis Data Analytics en la consola, con el flujo de datos de Kinesis como origen de streaming. El proceso de detección lee los registros de muestra en el origen de streaming e infiere un esquema en la aplicación con dos columnas (`EVENT_TIME` y `TICKER`), tal como se muestra a continuación.



A continuación, utilizará el código de la aplicación con funciones SQL para convertir el campo de marca temporal `EVENT_TIME` de diversas maneras. Por último, insertará los datos resultantes en otra secuencia en la aplicación, tal y como se muestra en la siguiente captura de pantalla:



Paso 1: crear un flujo de datos de Kinesis

Cree un flujo de datos de Amazon Kinesis y rellénela con registros event time y ticker de la siguiente manera:

1. Inicie sesión en la consola de Kinesis Consola de administración de AWS y ábrala en https://console.aws.amazon.com /kinesis.
2. Elija Flujos de datos en el panel de navegación.
3. Elija Crear secuencia de Kinesis y a continuación cree una secuencia con una partición.
4. Ejecute el siguiente código de Python para rellenar la secuencia con datos de muestra. Este código de ejemplo escribe continuamente un registro con un símbolo de cotización aleatorio y la marca temporal actual en la secuencia.

```
import datetime
import json
import random
import boto3

STREAM_NAME = "ExampleInputStream"


def get_data():
    return {
        "EVENT_TIME": datetime.datetime.now().isoformat(),
        "TICKER": random.choice(["AAPL", "AMZN", "MSFT", "INTC", "TBV"]),
```

```
        "PRICE": round(random.random() * 100, 2),
    }


def generate(stream_name, kinesis_client):
    while True:
        data = get_data()
        print(data)
        kinesis_client.put_record(
            StreamName=stream_name, Data=json.dumps(data),
 PartitionKey="partitionkey"
        )


if __name__ == "__main__":
    generate(STREAM_NAME, boto3.client("kinesis"))
```

Paso 2: creación de una aplicación de Amazon Kinesis Data Analytics

Cree una aplicación de la siguiente manera:

1. Abra la consola de Managed Service for Apache Flink en /kinesisanalytics. https://console.aws.amazon.com
2. Elija Create application (Crear aplicación), escriba el nombre de la aplicación y elija Create application (Crear aplicación).
3. En la página de detalles de la aplicación, elija Connect streaming data (Conectar datos de streaming) para conectarse al origen.
4. En la página Connect to source (Conectarse al origen), haga lo siguiente:
    a. Elija la secuencia que ha creado en la sección anterior.
    b. Elija crear un rol de IAM.
    c. Seleccione Detectar esquema. Espere a que la consola muestre el esquema inferido y los registros de muestra utilizados para inferir en el esquema de la secuencia en la aplicación que ha creado. El esquema inferido cuenta con dos columnas.
    d. Elija Edit Schema (Editar esquema). Cambie el Column type (Tipo de columna) de la columna EVENT_TIME a `TIMESTAMP`.
    e. Elija Save schema and update stream samples. Después de que la consola guarde el esquema, elija Exit (Salir).

f. Elija Guardar y continuar.

5. En la página de detalles de la aplicación, elija Go to SQL editor (Ir al editor de SQL). Para iniciar la aplicación, elija Yes, start application (Sí, iniciar la aplicación) en el cuadro de diálogo que aparece.
6. En el editor de SQL, escriba el código de la aplicación y verifique los resultados como se indica a continuación:

   a. Copie el siguiente código de la aplicación y péguelo en el editor.

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
    TICKER VARCHAR(4),
    event_time TIMESTAMP,
    five_minutes_before TIMESTAMP,
    event_unix_timestamp BIGINT,
    event_timestamp_as_char VARCHAR(50),
    event_second INTEGER);

CREATE OR REPLACE PUMP "STREAM_PUMP" AS INSERT INTO "DESTINATION_SQL_STREAM"

SELECT STREAM
    TICKER,
    EVENT_TIME,
    EVENT_TIME - INTERVAL '5' MINUTE,
    UNIX_TIMESTAMP(EVENT_TIME),
    TIMESTAMP_TO_CHAR('yyyy-MM-dd hh:mm:ss', EVENT_TIME),
    EXTRACT(SECOND FROM EVENT_TIME)
FROM "SOURCE_SQL_STREAM_001"
```

   b. Elija Save and run SQL. En la pestaña Real-time analytics (Análisis en tiempo real), puede ver todas las secuencias en la aplicación creadas por esta y comprobar los datos.

Ejemplo: Transformación de varios tipos de datos

Un requisito común de las aplicaciones de extracción, transformación y carga (ETL) es procesar varios tipos de registros en un origen de streaming. Puede crear aplicaciones de análisis de datos de Kinesis Data Analytics para procesar estos tipos de orígenes de streaming. El proceso es el siguiente:

1. En primer lugar, mapee el origen de streaming a una secuencia de entrada en la aplicación similar a la de las demás aplicaciones de análisis de datos de Kinesis Data Analytics.
2. A continuación, en el código de la aplicación, escriba instrucciones SQL para recuperar filas de determinados tipos de la secuencia de entrada en la aplicación. Acto seguido, insértelas en secuencias en la aplicación independientes. (Puede crear secuencias en la aplicación adicionales en el código de la aplicación.)

En este ejercicio, dispone de un origen de streaming que recibe registros de dos tipos (`Order` y `Trade`). Se trata de órdenes bursátiles y sus operaciones correspondientes. Para cada orden puede haber cero o más operaciones. Los ejemplos de registros de cada tipo se muestran a continuación:

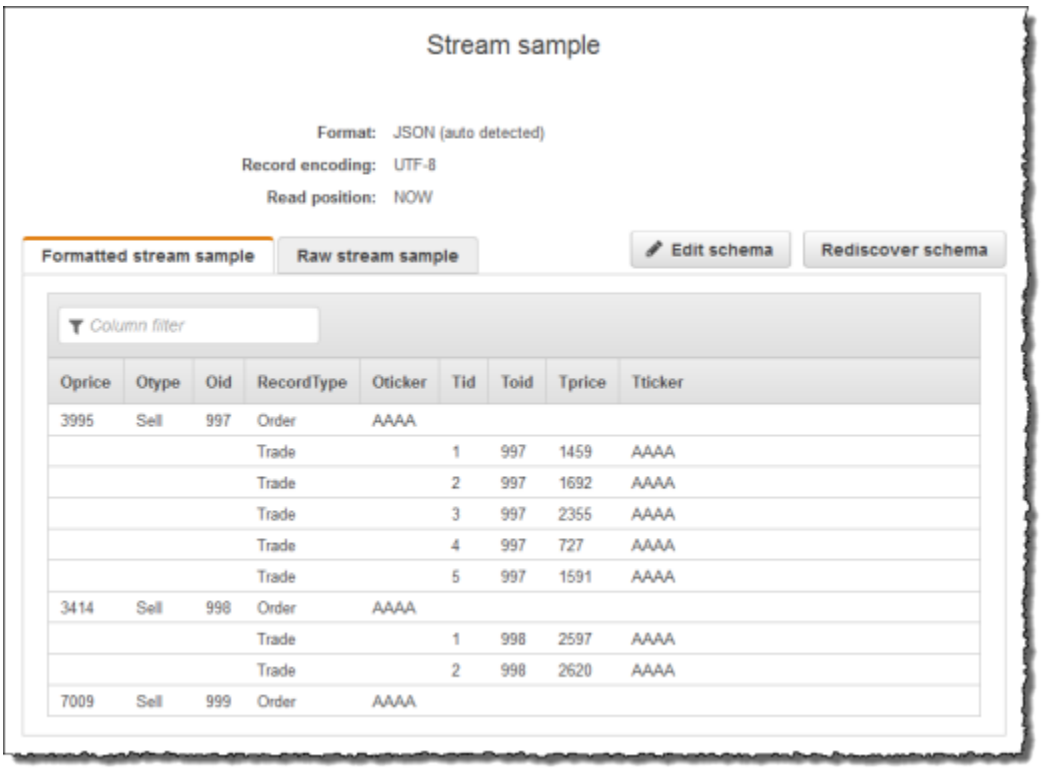
Registro de orden

```
{"RecordType": "Order", "Oprice": 9047, "Otype": "Sell", "Oid": 3811, "Oticker":
 "AAAA"}
```

Registro de operación

```
{"RecordType": "Trade", "Tid": 1, "Toid": 3812, "Tprice": 2089, "Tticker": "BBBB"}
```

Al crear una aplicación mediante el Consola de administración de AWS, la consola muestra el siguiente esquema inferido para el flujo de entrada creado en la aplicación. De forma predeterminada, la consola asigna el nombre `SOURCE_SQL_STREAM_001` a esta secuencia en la aplicación.



Cuando se guarda la configuración, Amazon Kinesis Data Analytics lee continuamente los datos del origen de streaming e introduce filas en la secuencia en la aplicación. A partir de ese momento se pueden realizar análisis de datos en la secuencia en la aplicación.

En el código de la aplicación de este ejemplo, primero se crean dos secuencias en la aplicación adicionales, `Order_Stream` y `Trade_Stream`. A continuación, se filtran las filas de la secuencia `SOURCE_SQL_STREAM_001` en función del tipo de registro y se insertan en las secuencias que se acaban de crear mediante bombas. Si desea más información sobre este patrón de codificación, consulte Código de la aplicación.

1. Filtre las filas de órdenes y operaciones en sus respectivas secuencias en la aplicación:

   a. Filtre los registros de las órdenes en `SOURCE_SQL_STREAM_001` y guarde las órdenes en `Order_Stream`.

```
--Create Order_Stream.
CREATE OR REPLACE STREAM "Order_Stream"
           (
            order_id     integer,
            order_type   varchar(10),
            ticker       varchar(4),
            order_price  DOUBLE,
```

```
            record_type  varchar(10)
            );

CREATE OR REPLACE PUMP "Order_Pump" AS
   INSERT INTO "Order_Stream"
      SELECT STREAM oid, otype,oticker, oprice, recordtype
      FROM   "SOURCE_SQL_STREAM_001"
      WHERE  recordtype = 'Order';
```

b. Filtre los registros de operaciones en `SOURCE_SQL_STREAM_001` y guarde las órdenes en `Trade_Stream`.

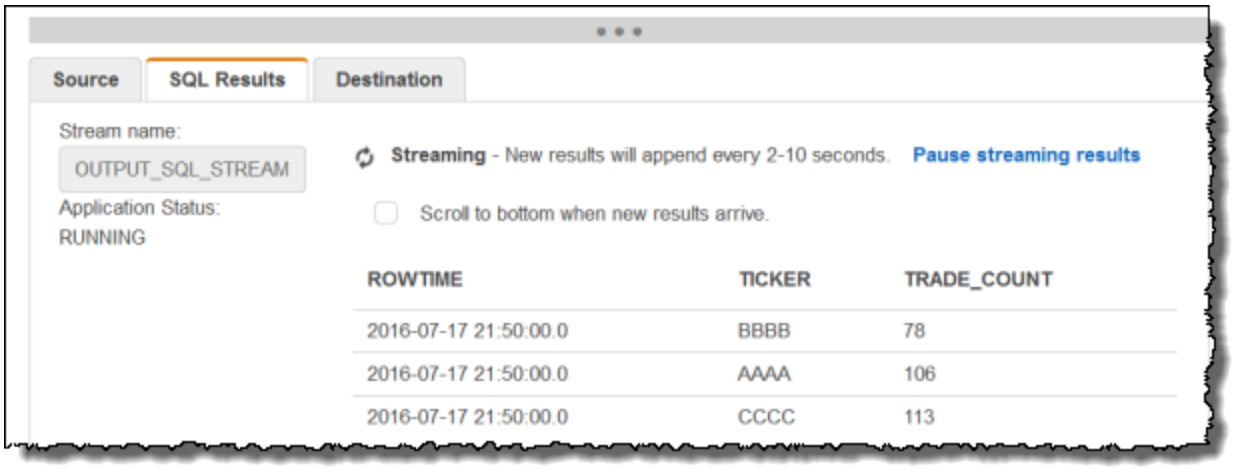
```
--Create Trade_Stream.
CREATE OR REPLACE STREAM "Trade_Stream"
           (trade_id     integer,
            order_id     integer,
            trade_price  DOUBLE,
            ticker       varchar(4),
            record_type  varchar(10)
            );

CREATE OR REPLACE PUMP "Trade_Pump" AS
   INSERT INTO "Trade_Stream"
      SELECT STREAM tid, toid, tprice, tticker, recordtype
      FROM   "SOURCE_SQL_STREAM_001"
      WHERE  recordtype = 'Trade';
```

2. Ahora ya puede realizar análisis adicionales de estas secuencias. En este ejemplo, va a contar la cantidad de operaciones por símbolo en una ventana de saltos de un minuto y va a guardar los resultados en una nueva secuencia, `DESTINATION_SQL_STREAM`.

```
--do some analytics on the Trade_Stream and Order_Stream.
-- To see results in console you must write to OPUT_SQL_STREAM.

CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
            ticker  varchar(4),
            trade_count   integer
            );

CREATE OR REPLACE PUMP "Output_Pump" AS
   INSERT INTO "DESTINATION_SQL_STREAM"
      SELECT STREAM ticker, count(*) as trade_count
      FROM   "Trade_Stream"
```

```
    GROUP BY ticker,
             FLOOR("Trade_Stream".ROWTIME TO MINUTE);
```

Verá un resultado como el que se muestra a continuación:



Temas

- Paso 1: Preparar los datos
- Paso 2: Crear la aplicación

Paso siguiente

Paso 1: Preparar los datos

Paso 1: Preparar los datos

En esta sección, creará un flujo de datos de Kinesis y, a continuación, rellenará los registros de órdenes y operaciones en la secuencia. Este es el origen de streaming para la aplicación que va a crear en el siguiente paso.

Temas

- Paso 1.1: Crear un origen de streaming
- Paso 1.2: Rellenar el origen de streaming

Paso 1.1: Crear un origen de streaming

Puede crear un flujo de datos de Kinesis utilizando la consola o la AWS CLI. El ejemplo adopta `OrdersAndTradesStream` como nombre de secuencia.

- Uso de la consola: inicie sesión en la consola de Kinesis Consola de administración de AWS y ábrala en https://console.aws.amazon.com /kinesis. Elija Data Streams y, a continuación, cree una secuencia con un fragmento. Para obtener más información, consulte Crear secuencia en la Guía para desarrolladores de Amazon Kinesis Data Streams.
- Uso de AWS CLI: utilice el siguiente `create-stream` AWS CLI comando de Kinesis para crear la transmisión:

```
$ aws kinesis create-stream \
--stream-name OrdersAndTradesStream \
--shard-count 1 \
--region us-east-1 \
--profile adminuser
```

Paso 1.2: Rellenar el origen de streaming

Ejecute el siguiente script de Python para rellenar los registros de muestra en `OrdersAndTradesStream`. Si creó la secuencia con otro nombre, actualice el código de Python según sea necesario.

1. Instale Python y `pip`.

   Para obtener más información sobre la instalación de Python, consulte la página web de Python.

   Puede instalar dependencias con pip. Para obtener más información sobre la instalación de pip, consulte la sección Installation en la página web de pip.

2. Ejecute el siguiente código de Python. El comando `put-record` en el código escribe los registros JSON en la secuencia.

```
import json
import random
import boto3

STREAM_NAME = "OrdersAndTradesStream"
```

```
PARTITION_KEY = "partition_key"


def get_order(order_id, ticker):
    return {
        "RecordType": "Order",
        "Oid": order_id,
        "Oticker": ticker,
        "Oprice": random.randint(500, 10000),
        "Otype": "Sell",
    }


def get_trade(order_id, trade_id, ticker):
    return {
        "RecordType": "Trade",
        "Tid": trade_id,
        "Toid": order_id,
        "Tticker": ticker,
        "Tprice": random.randint(0, 3000),
    }


def generate(stream_name, kinesis_client):
    order_id = 1
    while True:
        ticker = random.choice(["AAAA", "BBBB", "CCCC"])
        order = get_order(order_id, ticker)
        print(order)
        kinesis_client.put_record(
            StreamName=stream_name, Data=json.dumps(order),
 PartitionKey=PARTITION_KEY
        )
        for trade_id in range(1, random.randint(0, 6)):
            trade = get_trade(order_id, trade_id, ticker)
            print(trade)
            kinesis_client.put_record(
                StreamName=stream_name,
                Data=json.dumps(trade),
                PartitionKey=PARTITION_KEY,
            )
        order_id += 1
```

```
if __name__ == "__main__":
    generate(STREAM_NAME, boto3.client("kinesis"))
```

Paso siguiente

Paso 2: Crear la aplicación

Paso 2: Crear la aplicación

En esta sección, va a crear una aplicación de análisis de datos de Kinesis Data Analytics. A continuación, deberá actualizar la aplicación mediante la adición de configuración de entrada que asigna el origen de streaming que creó en la sección anterior a una secuencia de entrada en la aplicación.

1. Abra la consola de Managed Service for Apache Flink en https://console.aws.amazon.com / kinesisanalytics.
2. Elija Creación de aplicación. En este ejemplo se utiliza el nombre de la aplicación **`ProcessMultipleRecordTypes`**.
3. En la página de detalles de la aplicación, elija Connect streaming data (Conectar datos de streaming) para conectarse al origen.
4. En la página Connect to source (Conectarse al origen), haga lo siguiente:
   a. Elija la secuencia que ha creado en Paso 1: Preparar los datos.
   b. Elija crear un rol de IAM.
   c. Espere a que la consola muestre el esquema inferido y los registros de muestra utilizados para inferir en el esquema de la secuencia en la aplicación que ha creado.
   d. Elija Guardar y continuar.
5. En el centro de la aplicación, elija Go to SQL editor. Para iniciar la aplicación, elija Yes, start application (Sí, iniciar la aplicación) en el cuadro de diálogo que aparece.
6. En el editor de SQL, escriba el código de la aplicación y verifique los resultados:
   a. Copie el siguiente código de la aplicación y péguelo en el editor.

```
--Create Order_Stream.
CREATE OR REPLACE STREAM "Order_Stream"
           (
            "order_id"     integer,
```

```
            "order_type"   varchar(10),
            "ticker"       varchar(4),
            "order_price"  DOUBLE,
            "record_type"  varchar(10)
            );

CREATE OR REPLACE PUMP "Order_Pump" AS
   INSERT INTO "Order_Stream"
      SELECT STREAM "Oid", "Otype","Oticker", "Oprice", "RecordType"
      FROM   "SOURCE_SQL_STREAM_001"
      WHERE  "RecordType" = 'Order';
--*****************************************
--Create Trade_Stream.
CREATE OR REPLACE STREAM "Trade_Stream"
          ("trade_id"     integer,
           "order_id"     integer,
           "trade_price"  DOUBLE,
           "ticker"       varchar(4),
           "record_type"  varchar(10)
           );

CREATE OR REPLACE PUMP "Trade_Pump" AS
   INSERT INTO "Trade_Stream"
      SELECT STREAM "Tid", "Toid", "Tprice", "Tticker", "RecordType"
      FROM   "SOURCE_SQL_STREAM_001"
      WHERE  "RecordType" = 'Trade';
--*************************************************************
--do some analytics on the Trade_Stream and Order_Stream.
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
            "ticker"  varchar(4),
            "trade_count"   integer
            );

CREATE OR REPLACE PUMP "Output_Pump" AS
   INSERT INTO "DESTINATION_SQL_STREAM"
      SELECT STREAM "ticker", count(*) as trade_count
      FROM   "Trade_Stream"
      GROUP BY "ticker",
               FLOOR("Trade_Stream".ROWTIME TO MINUTE);
```

b. Elija Save and run SQL. Elija la pestaña Real-time analytics (Análisis en tiempo real) para ver todas las secuencias en la aplicación que esta creó y comprobar los datos.

Paso siguiente

Puede configurar la salida de la aplicación para que conserve resultados en un destino externo, como otro flujo de Kinesis o un flujo de entrega de datos de Firehose.

Ejemplos: ventanas y agregación

En esta sección se proporcionan ejemplos de aplicaciones de Amazon Kinesis Data Analytics que utilizan consultas en ventana y agregación. (Para obtener más información, consulte Consultas en ventana). Cada ejemplo proporciona step-by-step instrucciones y código de ejemplo para configurar la aplicación Kinesis Data Analytics.

Temas

- Ejemplo: Ventana escalonada
- Ejemplo: ventana de saltos con ROWTIME
- Ejemplo: ventana de saltos mediante una marca temporal de evento
- Ejemplo: recuperación de los valores más frecuentes (TOP_K_ITEMS_TUMBLING)
- Ejemplo: agregar resultados parciales de una consulta

Ejemplo: Ventana escalonada

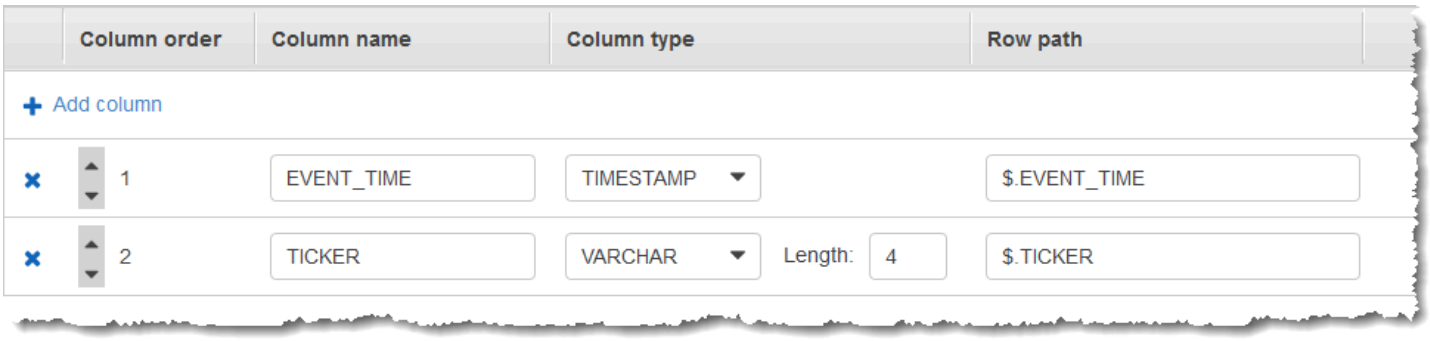
Cuando una consulta de ventanas procesa ventanas distintas para cada clave de partición única, empezando cuando se reciben datos con la clave coincidente, esta ventana recibe el nombre de ventana escalonada. Para obtener más información, consulte Ventanas escalonadas. En este ejemplo de Amazon Kinesis Data Analytics se usan las columnas EVENT_TIME y TICKER para crear ventanas escalonadas. La secuencia de origen contiene grupos de seis registros con idénticos valores de EVENT_TIME y TICKER que llegan en un periodo de un minuto, pero no necesariamente con el mismo valor de minuto (por ejemplo, `18:41:xx`).

En este ejemplo, escribirá los siguientes registros en un flujo de datos de Kinesis a las horas siguientes. El script no escribe las horas en la secuencia, sino la hora a la que la aplicación introduce el registro que se escribe en el campo `ROWTIME`:

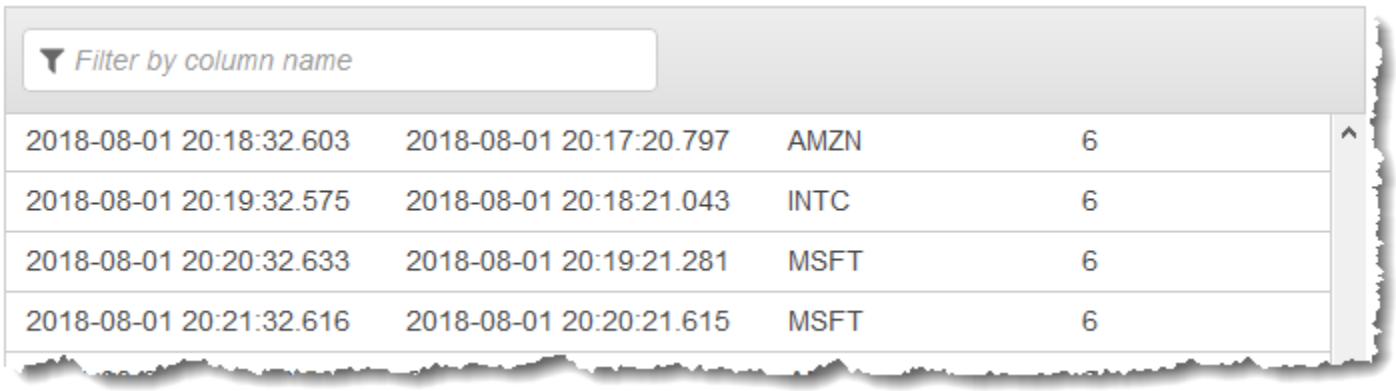
```
{"EVENT_TIME": "2018-08-01T20:17:20.797945", "TICKER": "AMZN"}   20:17:30
{"EVENT_TIME": "2018-08-01T20:17:20.797945", "TICKER": "AMZN"}   20:17:40
{"EVENT_TIME": "2018-08-01T20:17:20.797945", "TICKER": "AMZN"}   20:17:50
{"EVENT_TIME": "2018-08-01T20:17:20.797945", "TICKER": "AMZN"}   20:18:00
```

```
{"EVENT_TIME": "2018-08-01T20:17:20.797945", "TICKER": "AMZN"}   20:18:10
{"EVENT_TIME": "2018-08-01T20:17:20.797945", "TICKER": "AMZN"}   20:18:21
{"EVENT_TIME": "2018-08-01T20:18:21.043084", "TICKER": "INTC"}   20:18:31
{"EVENT_TIME": "2018-08-01T20:18:21.043084", "TICKER": "INTC"}   20:18:41
{"EVENT_TIME": "2018-08-01T20:18:21.043084", "TICKER": "INTC"}   20:18:51
{"EVENT_TIME": "2018-08-01T20:18:21.043084", "TICKER": "INTC"}   20:19:01
{"EVENT_TIME": "2018-08-01T20:18:21.043084", "TICKER": "INTC"}   20:19:11
{"EVENT_TIME": "2018-08-01T20:18:21.043084", "TICKER": "INTC"}   20:19:21
...
```

A continuación, cree una aplicación de Kinesis Data Analytics en, con Consola de administración de AWS la transmisión de datos de Kinesis como fuente de transmisión. El proceso de detección lee los registros de muestra en el origen de streaming e infiere un esquema en la aplicación con dos columnas (`EVENT_TIME` y `TICKER`), tal como se muestra a continuación.



Utilice el código de aplicación con la función `COUNT` para crear una agregación en ventana de los datos. A continuación, inserte los datos resultantes en otra secuencia en la aplicación, tal y como se muestra en la siguiente captura de pantalla:



En el siguiente procedimiento se crea una aplicación de Kinesis Data Analytics que agrupa los valores de la secuencia de entrada en una ventana escalonada basada en EVENT_TIME y TICKER.

Temas

Paso 1: crear un flujo de datos de Kinesis

Cree un flujo de datos de Amazon Kinesis y rellene los registros como se indica a continuación:

1. Inicie sesión en la consola de Kinesis Consola de administración de AWS y ábrala en https://console.aws.amazon.com /kinesis.
2. Elija Flujos de datos en el panel de navegación.
3. Elija Create Kinesis Stream (Crear secuencia de Kinesis) y, a continuación, cree una secuencia con un fragmento. Para obtener más información, consulte Crear secuencia en la Guía para desarrolladores de Amazon Kinesis Data Streams.
4. Para escribir registros en un flujo de datos de Kinesis en un entorno de producción, recomendamos utilizar Kinesis Producer Library o la API de Kinesis Data Streams. Para simplificar, en este ejemplo se utiliza el siguiente script Python para generar registros. Ejecute el código para rellenar los registros de ticker de muestra. Este código sencillo escribe continuamente un grupo de seis registros con el mismo valor aleatorio de `EVENT_TIME` y clave de cotización en la secuencia, en el transcurso de un minuto. Mantenga el script ejecutándose para poder generar el esquema de la aplicación en un paso posterior.

```
import datetime
import json
import random
import time
import boto3

STREAM_NAME = "ExampleInputStream"


def get_data():
    event_time = datetime.datetime.utcnow() - datetime.timedelta(seconds=10)
    return {
        "EVENT_TIME": event_time.isoformat(),
        "TICKER": random.choice(["AAPL", "AMZN", "MSFT", "INTC", "TBV"]),
    }
```

```
def generate(stream_name, kinesis_client):
    while True:
        data = get_data()
        # Send six records, ten seconds apart, with the same event time and ticker
        for _ in range(6):
            print(data)
            kinesis_client.put_record(
                StreamName=stream_name,
                Data=json.dumps(data),
                PartitionKey="partitionkey",
            )
            time.sleep(10)


if __name__ == "__main__":
    generate(STREAM_NAME, boto3.client("kinesis"))
```

Paso 2: creación de una aplicación de Kinesis Data Analytics

Cree una aplicación de análisis de datos de Kinesis Data Analytics de la siguiente manera:

1. Abra la consola de Managed Service for Apache Flink en /kinesisanalytics. https://console.aws.amazon.com
2. Elija Create application (Crear aplicación), escriba el nombre de la aplicación y elija Create application (Crear aplicación).
3. En la página de detalles de la aplicación, elija Connect streaming data (Conectar datos de streaming) para conectarse al origen.
4. En la página Connect to source (Conectarse al origen), haga lo siguiente:

    a. Elija la secuencia que ha creado en la sección anterior.
    b. Elija Discover Schema (Detectar esquema). Espere a que la consola muestre el esquema inferido y los registros de muestra utilizados para inferir en el esquema de la secuencia en la aplicación que ha creado. El esquema inferido cuenta con dos columnas.
    c. Elija Edit Schema (Editar esquema). Cambie el Column type (Tipo de columna) de la columna EVENT_TIME a `TIMESTAMP`.
    d. Elija Save schema and update stream samples. Después de que la consola guarde el esquema, elija Exit (Salir).

e. Elija Guardar y continuar.

5. En la página de detalles de la aplicación, elija Go to SQL editor (Ir al editor de SQL). Para iniciar la aplicación, elija Yes, start application (Sí, iniciar la aplicación) en el cuadro de diálogo que aparece.
6. En el editor de SQL, escriba el código de la aplicación y verifique los resultados como se indica a continuación:

   a. Copie el siguiente código de la aplicación y péguelo en el editor.

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
    event_time TIMESTAMP,
    ticker_symbol    VARCHAR(4),
    ticker_count     INTEGER);

CREATE OR REPLACE PUMP "STREAM_PUMP" AS
  INSERT INTO "DESTINATION_SQL_STREAM"
    SELECT STREAM
        EVENT_TIME,
        TICKER,
        COUNT(TICKER) AS ticker_count
    FROM "SOURCE_SQL_STREAM_001"
    WINDOWED BY STAGGER (
            PARTITION BY TICKER, EVENT_TIME RANGE INTERVAL '1' MINUTE);
```

   b. Elija Save and run SQL.

      En la pestaña Real-time analytics (Análisis en tiempo real), puede ver todas las secuencias en la aplicación creadas por esta y comprobar los datos.

Ejemplo: ventana de saltos con ROWTIME

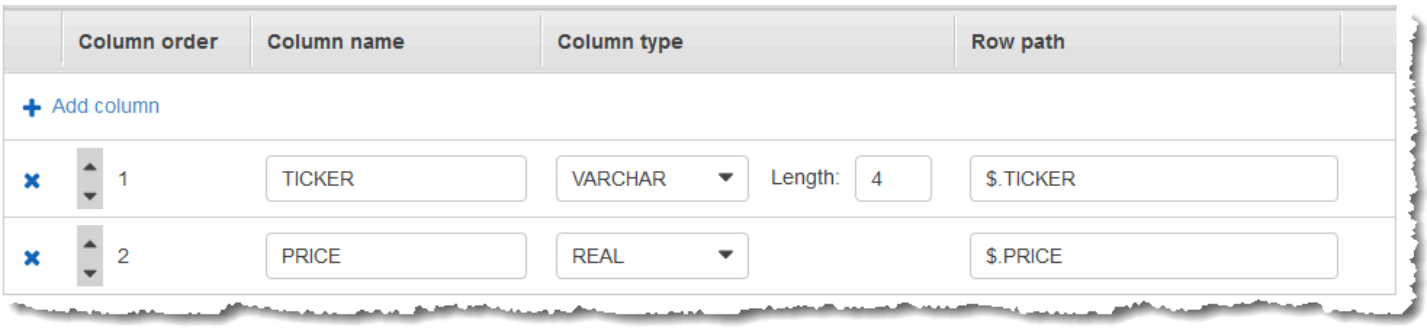
Cuando una consulta en ventana procesa cada ventana de forma que no se superpongan, la ventana se denomina ventana de saltos. Para obtener más información, consulte Ventanas de saltos de tamaño constante (Agregados utilizando GROUP BY). Este ejemplo de Amazon Kinesis Data Analytics se utiliza la columna `ROWTIME` para crear ventanas de saltos. La columna `ROWTIME` representa la hora en que la aplicación leyó el registro.

En este ejemplo, escribirá los siguientes registros en un flujo de datos de Kinesis.

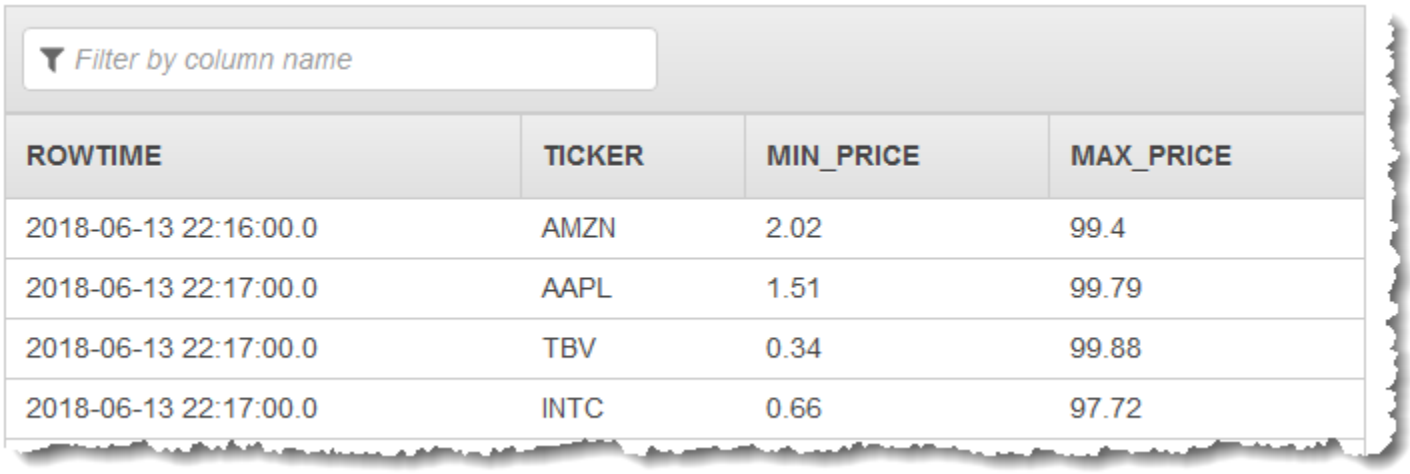
```
{"TICKER": "TBV", "PRICE": 33.11}
```

```
{"TICKER": "INTC", "PRICE": 62.04}
{"TICKER": "MSFT", "PRICE": 40.97}
{"TICKER": "AMZN", "PRICE": 27.9}
...
```

A continuación, cree una aplicación de Kinesis Data Analytics en, con Consola de administración de AWS la transmisión de datos de Kinesis como fuente de transmisión. El proceso de detección lee los registros de muestra en el origen de streaming e infiere un esquema en la aplicación con dos columnas (`TICKER` y `PRICE`), tal como se muestra a continuación.



Utilice el código de aplicación con las funciones `MIN` y `MAX` para crear una agregación en ventana de los datos. A continuación, inserte los datos resultantes en otra secuencia en la aplicación, tal y como se muestra en la siguiente captura de pantalla:



En el siguiente procedimiento, se crea una aplicación de Kinesis Data Analytics que agrega valores en el flujo de entrada en una ventana de saltos basada en ROWTIME.

Temas

- Paso 1: crear un flujo de datos de Kinesis

- Paso 2: creación de una aplicación de Kinesis Data Analytics

Paso 1: crear un flujo de datos de Kinesis

Cree un flujo de datos de Amazon Kinesis y rellene los registros como se indica a continuación:

1. Inicie sesión en la consola de Kinesis Consola de administración de AWS y ábrala en https://console.aws.amazon.com /kinesis.
2. Elija Flujos de datos en el panel de navegación.
3. Elija Create Kinesis Stream (Crear secuencia de Kinesis) y, a continuación, cree una secuencia con un fragmento. Para obtener más información, consulte Crear secuencia en la Guía para desarrolladores de Amazon Kinesis Data Streams.
4. Para escribir registros en un flujo de datos de Kinesis en un entorno de producción, recomendamos utilizar Kinesis Client Library o la API de Kinesis Data Streams. Para simplificar, en este ejemplo se utiliza el siguiente script Python para generar registros. Ejecute el código para rellenar los registros de ticker de muestra. Este código simple escribe continuamente un registro de ticker aleatorio en el flujo. Mantenga el script ejecutándose para poder generar el esquema de la aplicación en un paso posterior.

```
import datetime
import json
import random
import boto3

STREAM_NAME = "ExampleInputStream"


def get_data():
    return {
        "EVENT_TIME": datetime.datetime.now().isoformat(),
        "TICKER": random.choice(["AAPL", "AMZN", "MSFT", "INTC", "TBV"]),
        "PRICE": round(random.random() * 100, 2),
    }


def generate(stream_name, kinesis_client):
    while True:
        data = get_data()
        print(data)
```

```
        kinesis_client.put_record(
            StreamName=stream_name, Data=json.dumps(data),
 PartitionKey="partitionkey"
        )


if __name__ == "__main__":
    generate(STREAM_NAME, boto3.client("kinesis"))
```

Paso 2: creación de una aplicación de Kinesis Data Analytics

Cree una aplicación de análisis de datos de Kinesis Data Analytics de la siguiente manera:

1. Abra la consola de Managed Service for Apache Flink en /kinesisanalytics. https://console.aws.amazon.com
2. Elija Create application (Crear aplicación), introduzca el nombre de la aplicación y elija Create application (Crear aplicación).
3. En la página de detalles de la aplicación, elija Connect streaming data (Conectar datos de streaming) para conectarse al origen.
4. En la página Connect to source (Conectarse al origen), haga lo siguiente:

   a. Elija la secuencia que ha creado en la sección anterior.

   b. Elija Discover Schema (Detectar esquema). Espere a que la consola muestre el esquema inferido y los registros de muestra utilizados para inferir en el esquema de la secuencia en la aplicación que ha creado. El esquema inferido cuenta con dos columnas.

   c. Elija Save schema and update stream samples. Después de que la consola guarde el esquema, elija Exit (Salir).

   d. Elija Guardar y continuar.
5. En la página de detalles de la aplicación, elija Go to SQL editor (Ir al editor de SQL). Para iniciar la aplicación, elija Yes, start application (Sí, iniciar la aplicación) en el cuadro de diálogo que aparece.
6. En el editor de SQL, escriba el código de la aplicación y verifique los resultados como se indica a continuación:

   a. Copie el siguiente código de la aplicación y péguelo en el editor.

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (TICKER VARCHAR(4), MIN_PRICE
 REAL, MAX_PRICE REAL);

CREATE OR REPLACE PUMP "STREAM_PUMP" AS INSERT INTO "DESTINATION_SQL_STREAM"
    SELECT STREAM TICKER, MIN(PRICE), MAX(PRICE)
        FROM "SOURCE_SQL_STREAM_001"
        GROUP BY TICKER,
            STEP("SOURCE_SQL_STREAM_001".ROWTIME BY INTERVAL '60' SECOND);
```

b. Elija Save and run SQL.

   En la pestaña Real-time analytics (Análisis en tiempo real), puede ver todas las secuencias en la aplicación creadas por esta y comprobar los datos.

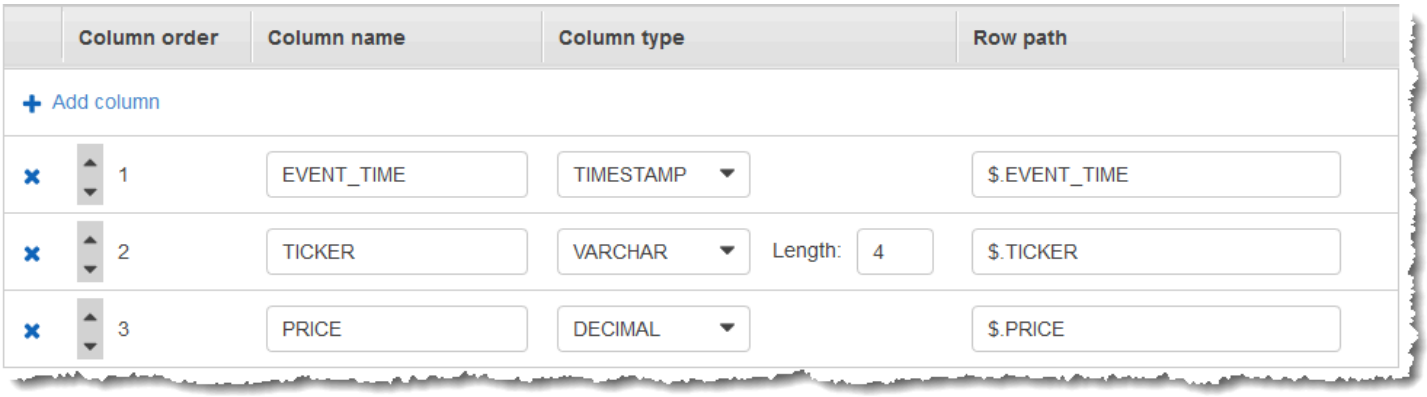
Ejemplo: ventana de saltos mediante una marca temporal de evento

Cuando una consulta en ventana procesa cada ventana de forma que no se superpongan, la ventana se denomina ventana de saltos. Para obtener más información, consulte Ventanas de saltos de tamaño constante (Agregados utilizando GROUP BY). Este ejemplo de Amazon Kinesis Data Analytics muestra una ventana de saltos que utiliza una marca temporal de evento, que es una marca temporal creada por el usuario que se incluye en los datos de streaming. Utiliza este enfoque en lugar de solo usar ROWTIME, que es una marca temporal que crea Kinesis Data Analytics cuando la aplicación recibe el registro. Se usa una marca temporal de evento en los datos de streaming si se desea crear una agregación basada en el momento en que se produjo un evento, en lugar del momento en que lo recibió la aplicación. En este ejemplo, el valor de `ROWTIME` dispara la agregación cada minuto y los registros se agregan tanto por `ROWTIME` como por el tiempo de evento incluido.

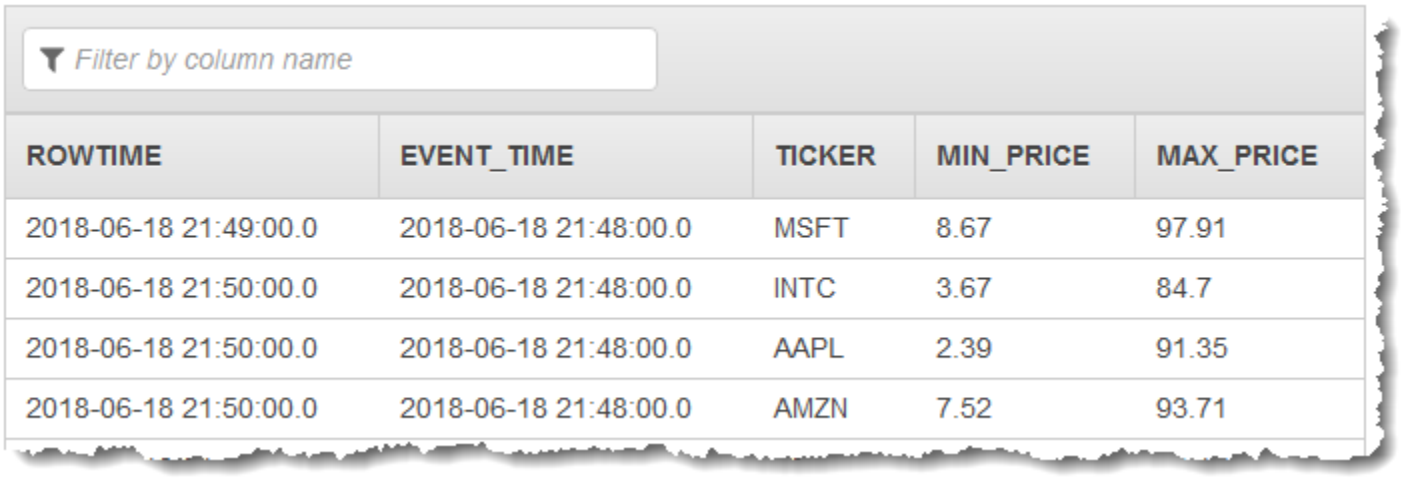
En este ejemplo escribirá los siguientes registros en una secuencia de Amazon Kinesis. El valor de `EVENT_TIME` se establece en 5 segundos en el pasado, para simular el retraso de procesamiento y flujo que podría crear un retardo desde el momento en que se produjo el evento hasta el momento en que el registro se ingiere en Kinesis Data Analytics.

```
{"EVENT_TIME": "2018-06-13T14:11:05.766191", "TICKER": "TBV", "PRICE": 43.65}
{"EVENT_TIME": "2018-06-13T14:11:05.848967", "TICKER": "AMZN", "PRICE": 35.61}
{"EVENT_TIME": "2018-06-13T14:11:05.931871", "TICKER": "MSFT", "PRICE": 73.48}
{"EVENT_TIME": "2018-06-13T14:11:06.014845", "TICKER": "AMZN", "PRICE": 18.64}
...
```

A continuación, cree una aplicación de Kinesis Data Analytics en, con Consola de administración de AWS la transmisión de datos de Kinesis como fuente de transmisión. El proceso de detección lee los registros de muestra en el origen de streaming e infiere un esquema en la aplicación con tres columnas (`EVENT_TIME`, `TICKER` y `PRICE`), tal como se muestra a continuación.



Utilice el código de aplicación con las funciones `MIN` y `MAX` para crear una agregación en ventana de los datos. A continuación, inserte los datos resultantes en otra secuencia en la aplicación, tal y como se muestra en la siguiente captura de pantalla:



ROWTIME	EVENT_TIME	TICKER	MIN_PRICE	MAX_PRICE
2018-06-18 21:49:00.0	2018-06-18 21:48:00.0	MSFT	8.67	97.91
2018-06-18 21:50:00.0	2018-06-18 21:48:00.0	INTC	3.67	84.7
2018-06-18 21:50:00.0	2018-06-18 21:48:00.0	AAPL	2.39	91.35
2018-06-18 21:50:00.0	2018-06-18 21:48:00.0	AMZN	7.52	93.71

En el siguiente procedimiento se crea una aplicación de Kinesis Data Analytics que agrega valores en el flujo de entrada en una ventana de saltos basada en una hora de evento.

Temas

- Paso 1: crear un flujo de datos de Kinesis
- Paso 2: creación de una aplicación de Kinesis Data Analytics

Paso 1: crear un flujo de datos de Kinesis

Cree un flujo de datos de Amazon Kinesis y rellene los registros como se indica a continuación:

1. Inicie sesión en la consola de Kinesis Consola de administración de AWS y ábrala en https://console.aws.amazon.com /kinesis.
2. Elija Flujos de datos en el panel de navegación.
3. Elija Create Kinesis Stream (Crear secuencia de Kinesis) y, a continuación, cree una secuencia con un fragmento. Para obtener más información, consulte Crear secuencia en la Guía para desarrolladores de Amazon Kinesis Data Streams.
4. Para escribir registros en un flujo de datos de Kinesis en un entorno de producción, recomendamos utilizar Kinesis Client Library o la API de Kinesis Data Streams. Para simplificar, en este ejemplo se utiliza el siguiente script Python para generar registros. Ejecute el código para rellenar los registros de ticker de muestra. Este código simple escribe continuamente un registro de ticker aleatorio en el flujo. Mantenga el script ejecutándose para poder generar el esquema de la aplicación en un paso posterior.

```
import datetime
import json
import random
import boto3

STREAM_NAME = "ExampleInputStream"


def get_data():
    return {
        "EVENT_TIME": datetime.datetime.now().isoformat(),
        "TICKER": random.choice(["AAPL", "AMZN", "MSFT", "INTC", "TBV"]),
        "PRICE": round(random.random() * 100, 2),
    }


def generate(stream_name, kinesis_client):
    while True:
        data = get_data()
        print(data)
        kinesis_client.put_record(
```

```
            StreamName=stream_name, Data=json.dumps(data),
 PartitionKey="partitionkey"
        )


if __name__ == "__main__":
    generate(STREAM_NAME, boto3.client("kinesis"))
```

Paso 2: creación de una aplicación de Kinesis Data Analytics

Cree una aplicación de análisis de datos de Kinesis Data Analytics de la siguiente manera:

1. Abra la consola de Managed Service for Apache Flink en /kinesisanalytics. https://console.aws.amazon.com
2. Elija Create application (Crear aplicación), introduzca el nombre de la aplicación y elija Create application (Crear aplicación).
3. En la página de detalles de la aplicación, elija Connect streaming data (Conectar datos de streaming) para conectarse al origen.
4. En la página Connect to source (Conectarse al origen), haga lo siguiente:

   a. Elija la secuencia que ha creado en la sección anterior.
   b. Elija Discover Schema (Detectar esquema). Espere a que la consola muestre el esquema inferido y los registros de muestra utilizados para inferir en el esquema de la secuencia en la aplicación que ha creado. El esquema inferido cuenta con tres columnas.
   c. Elija Edit Schema (Editar esquema). Cambie el Column type (Tipo de columna) de la columna EVENT_TIME a `TIMESTAMP`.
   d. Elija Save schema and update stream samples. Después de que la consola guarde el esquema, elija Exit (Salir).
   e. Elija Guardar y continuar.
5. En la página de detalles de la aplicación, elija Go to SQL editor (Ir al editor de SQL). Para iniciar la aplicación, elija Yes, start application (Sí, iniciar la aplicación) en el cuadro de diálogo que aparece.
6. En el editor de SQL, escriba el código de la aplicación y verifique los resultados como se indica a continuación:

   a. Copie el siguiente código de la aplicación y péguelo en el editor.

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (EVENT_TIME timestamp, TICKER
 VARCHAR(4), min_price REAL, max_price REAL);

CREATE OR REPLACE PUMP "STREAM_PUMP" AS
  INSERT INTO "DESTINATION_SQL_STREAM"
    SELECT STREAM STEP("SOURCE_SQL_STREAM_001".EVENT_TIME BY INTERVAL '60'
 SECOND),
        TICKER,
         MIN(PRICE) AS MIN_PRICE,
         MAX(PRICE) AS MAX_PRICE
    FROM    "SOURCE_SQL_STREAM_001"
    GROUP BY TICKER,
             STEP("SOURCE_SQL_STREAM_001".ROWTIME BY INTERVAL '60' SECOND),
             STEP("SOURCE_SQL_STREAM_001".EVENT_TIME BY INTERVAL '60' SECOND);
```

b. Elija Save and run SQL.

   En la pestaña Real-time analytics (Análisis en tiempo real), puede ver todas las secuencias en la aplicación creadas por esta y comprobar los datos.

Ejemplo: recuperación de los valores más frecuentes (TOP_K_ITEMS_TUMBLING)

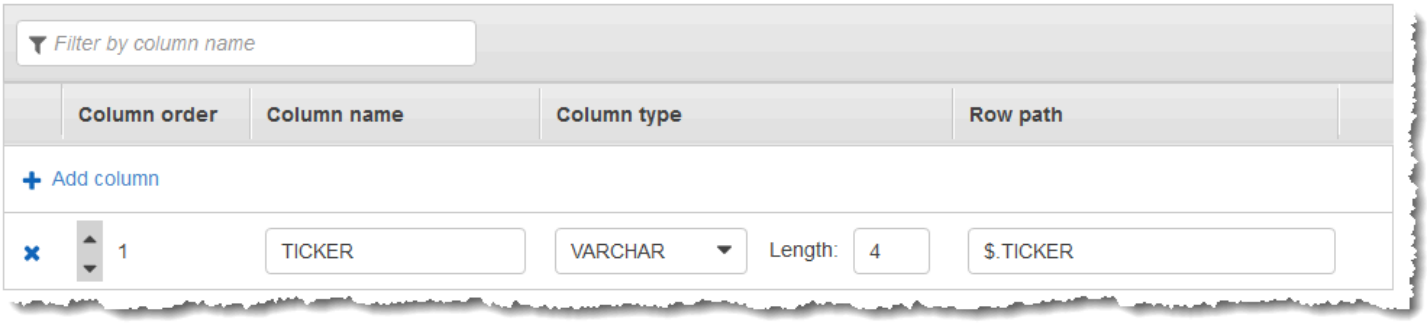
Este ejemplo de Amazon Kinesis Data Analytics demuestra cómo usar la función `TOP_K_ITEMS_TUMBLING` para recuperar los valores más frecuentes en una ventana de saltos. Para obtener más información, consulte Función `TOP_K_ITEMS_TUMBLING` en la Referencia de SQL de Amazon Managed Service para Apache Flink.

La función `TOP_K_ITEMS_TUMBLING` es útil cuando se agregan decenas o cientos de miles de claves y se desea reducir el uso de recursos. La función produce el mismo resultado que la agregación con cláusulas `GROUP BY` y `ORDER BY`.

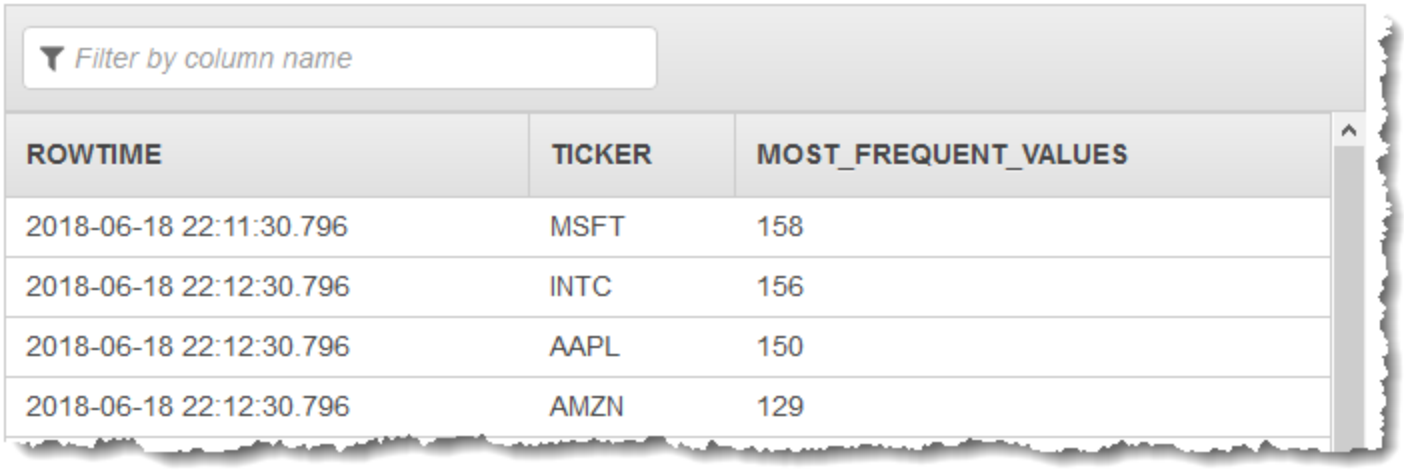
En este ejemplo, escribirá los siguientes registros en un flujo de datos de Amazon Kinesis.

```
{"TICKER": "TBV"}
{"TICKER": "INTC"}
{"TICKER": "MSFT"}
{"TICKER": "AMZN"}
...
```

A continuación, cree una aplicación de Kinesis Data Analytics en, con Consola de administración de AWS la transmisión de datos de Kinesis como fuente de transmisión. El proceso de detección lee los registros de muestra en el origen de streaming e infiere un esquema en la aplicación con una columna (`TICKER`), tal como se muestra a continuación.



Utilice el código de aplicación con la función `TOP_K_VALUES_TUMBLING` para crear una agregación en ventana de los datos. A continuación, inserte los datos resultantes en otra secuencia en la aplicación, tal y como se muestra en la siguiente captura de pantalla:



En el siguiente procedimiento, se crea una aplicación de Kinesis Data Analytics que recupera los valores más frecuentes en el flujo de entrada.

Temas

- Paso 1: crear un flujo de datos de Kinesis
- Paso 2: creación de una aplicación de Kinesis Data Analytics

Paso 1: crear un flujo de datos de Kinesis

Cree un flujo de datos de Amazon Kinesis y rellene los registros como se indica a continuación:

1. Inicie sesión en la consola de Kinesis Consola de administración de AWS y ábrala en https://console.aws.amazon.com /kinesis.
2. Elija Flujos de datos en el panel de navegación.
3. Elija Create Kinesis Stream (Crear secuencia de Kinesis) y, a continuación, cree una secuencia con un fragmento. Para obtener más información, consulte Crear secuencia en la Guía para desarrolladores de Amazon Kinesis Data Streams.
4. Para escribir registros en un flujo de datos de Kinesis en un entorno de producción, recomendamos utilizar Kinesis Client Library o la API de Kinesis Data Streams. Para simplificar, en este ejemplo se utiliza el siguiente script Python para generar registros. Ejecute el código para rellenar los registros de ticker de muestra. Este código simple escribe continuamente un registro de ticker aleatorio en el flujo. Deje el script ejecutándose para poder generar el esquema de la aplicación en un paso posterior.

```
import datetime
import json
import random
import boto3

STREAM_NAME = "ExampleInputStream"


def get_data():
    return {
        "EVENT_TIME": datetime.datetime.now().isoformat(),
        "TICKER": random.choice(["AAPL", "AMZN", "MSFT", "INTC", "TBV"]),
        "PRICE": round(random.random() * 100, 2),
    }


def generate(stream_name, kinesis_client):
    while True:
        data = get_data()
        print(data)
        kinesis_client.put_record(
            StreamName=stream_name, Data=json.dumps(data),
 PartitionKey="partitionkey"
        )


if __name__ == "__main__":
```

```
    generate(STREAM_NAME, boto3.client("kinesis"))
```

Paso 2: creación de una aplicación de Kinesis Data Analytics

Cree una aplicación de análisis de datos de Kinesis Data Analytics de la siguiente manera:

1. Abra la consola de Managed Service for Apache Flink en /kinesisanalytics. https://console.aws.amazon.com
2. Elija Create application (Crear aplicación), escriba el nombre de la aplicación y elija Create application (Crear aplicación).
3. En la página de detalles de la aplicación, elija Connect streaming data (Conectar datos de streaming) para conectarse al origen.
4. En la página Connect to source (Conectarse al origen), haga lo siguiente:

   a. Elija la secuencia que ha creado en la sección anterior.
   b. Elija Discover Schema (Detectar esquema). Espere a que la consola muestre el esquema inferido y los registros de muestra utilizados para inferir en el esquema de la secuencia en la aplicación que ha creado. El esquema inferido tiene una columna.
   c. Elija Save schema and update stream samples. Después de que la consola guarde el esquema, elija Exit (Salir).
   d. Elija Guardar y continuar.
5. En la página de detalles de la aplicación, elija Go to SQL editor (Ir al editor de SQL). Para iniciar la aplicación, elija Yes, start application (Sí, iniciar la aplicación) en el cuadro de diálogo que aparece.
6. En el editor de SQL, escriba el código de la aplicación y verifique los resultados como se indica a continuación:

   a. Copie el siguiente código de la aplicación y péguelo en el editor:

```
CREATE OR REPLACE STREAM DESTINATION_SQL_STREAM (
  "TICKER" VARCHAR(4),
  "MOST_FREQUENT_VALUES" BIGINT
);

CREATE OR REPLACE PUMP "STREAM_PUMP" AS
    INSERT INTO "DESTINATION_SQL_STREAM"
```

```
    SELECT STREAM *
        FROM TABLE (TOP_K_ITEMS_TUMBLING(
            CURSOR(SELECT STREAM * FROM "SOURCE_SQL_STREAM_001"),
            'TICKER',         -- name of column in single quotes
            5,                      -- number of the most frequently occurring
values
            60                      -- tumbling window size in seconds
            )
        );
```

b. Elija Save and run SQL.

   En la pestaña Real-time analytics (Análisis en tiempo real), puede ver todas las secuencias en la aplicación creadas por esta y comprobar los datos.

Ejemplo: agregar resultados parciales de una consulta

Si un flujo de datos de Amazon Kinesis contiene registros con una hora de evento que no coincide exactamente con la hora de ingestión, una selección de resultados en una ventana de saltos contiene los registros que llegaron, pero que no se produjeron necesariamente, dentro de la ventana. En este caso, la ventana de saltos solo contiene un conjunto parcial de los resultados que desea. Existen varios métodos que puede utilizar para corregir este problema:

- Utilizar solo una ventana de saltos y agregar los resultados parciales en el posprocesamiento a través de una base de datos o un almacén de datos mediante upserts. Este enfoque es eficaz en el procesamiento de una aplicación. Gestiona los datos con retraso de forma indefinida para operadores de agregación (`sum`, `min`, `max`, etc.). La desventaja de este enfoque es que debe desarrollar y mantener lógica de aplicación adicional en la capa de la base de datos.
- Utilizar una ventana de saltos y deslizante que produzca resultados parciales de forma anticipada, pero que también siga produciendo resultados completos durante el periodo de la ventana deslizante. Este enfoque trata datos tardíos con una sobrescritura en lugar de un upsert, de modo que no es necesario añadir ninguna lógica de aplicación adicional en la capa de la base de datos. La desventaja de este enfoque es que utiliza más unidades de procesamiento de Kinesis KPUs () y sigue produciendo dos resultados, lo que podría no funcionar en algunos casos de uso.

Para obtener más información sobre las ventanas de saltos y deslizantes, consulte Consultas en ventana.

En el siguiente procedimiento, la agregación de la ventana de saltos produce dos resultados parciales (enviados al flujo de la aplicación `CALC_COUNT_SQL_STREAM`) que deben combinarse para producir un resultado final. A continuación, la aplicación produce una segunda agregación (enviada al flujo de la aplicación `DESTINATION_SQL_STREAM`) que combina los dos resultados parciales.

Para crear una aplicación que agregue resultados parciales utilizando una hora de evento

1. Inicie sesión en la consola de Kinesis Consola de administración de AWS y ábrala en https://console.aws.amazon.com /kinesis.
2. Elija Análisis de datos en el panel de navegación. Cree una aplicación de Kinesis Data Analytics como se describe en el tutorial Introducción a aplicaciones de Amazon Kinesis Data Analytics para SQL.
3. En el editor de SQL, sustituya el código de la aplicación por el siguiente:

```
CREATE OR REPLACE STREAM "CALC_COUNT_SQL_STREAM"
    (TICKER      VARCHAR(4),
    TRADETIME   TIMESTAMP,
    TICKERCOUNT       DOUBLE);

CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM"
    (TICKER      VARCHAR(4),
    TRADETIME   TIMESTAMP,
    TICKERCOUNT       DOUBLE);

CREATE PUMP "CALC_COUNT_SQL_PUMP_001" AS
    INSERT INTO "CALC_COUNT_SQL_STREAM" ("TICKER","TRADETIME", "TICKERCOUNT")
    SELECT STREAM
        "TICKER_SYMBOL",
        STEP("SOURCE_SQL_STREAM_001"."ROWTIME" BY INTERVAL '1' MINUTE) as
 "TradeTime",
        COUNT(*) AS "TickerCount"
    FROM "SOURCE_SQL_STREAM_001"
    GROUP BY
        STEP("SOURCE_SQL_STREAM_001".ROWTIME BY INTERVAL '1' MINUTE),
        STEP("SOURCE_SQL_STREAM_001"."APPROXIMATE_ARRIVAL_TIME" BY INTERVAL '1'
 MINUTE),
        TICKER_SYMBOL;

CREATE PUMP "AGGREGATED_SQL_PUMP" AS
    INSERT INTO "DESTINATION_SQL_STREAM" ("TICKER","TRADETIME", "TICKERCOUNT")
    SELECT STREAM
```

```
    "TICKER",
    "TRADETIME",
    SUM("TICKERCOUNT") OVER W1 AS "TICKERCOUNT"
FROM "CALC_COUNT_SQL_STREAM"
WINDOW W1 AS (PARTITION BY "TRADETIME" RANGE INTERVAL '10' MINUTE PRECEDING);
```

La instrucción `SELECT` del código de la aplicación filtra las filas de `SOURCE_SQL_STREAM_001` para ver cambios en las cotizaciones de los valores superiores al 1 por ciento e introduce las filas en otra secuencia en la aplicación `CHANGE_STREAM` mediante una bomba.

4. Elija Save and run SQL.

La primera bomba envía un flujo a `CALC_COUNT_SQL_STREAM` similar al siguiente. Tenga en cuenta que el conjunto de resultados está incompleto:

Filter by column name

ROWTIME	TICKER	TRADETIME	TICKERCOUNT
2018-01-30 22:57:00.0	BAC	2018-01-30 22:56:00.0	2.0
2018-01-30 22:57:00.0	ALY	2018-01-30 22:56:00.0	2.0
2018-01-30 22:57:00.0	DFG	2018-01-30 22:56:00.0	5.0
2018-01-30 22:57:00.0	CVB	2018-01-30 22:56:00.0	6.0

A continuación, la segunda bomba envía un flujo a `DESTINATION_SQL_STREAM` que contiene el conjunto completo de resultados:

Filter by column name

ROWTIME	TICKER	TRADETIME	TICKERCOUNT
2018-01-30 23:17:00.0	PPL	2018-01-30 23:16:00.0	4.0
2018-01-30 23:18:00.0	TGT	2018-01-30 23:17:00.0	8.0
2018-01-30 23:18:00.0	DFT	2018-01-30 23:17:00.0	6.0
2018-01-30 23:18:00.0	KFU	2018-01-30 23:17:00.0	5.0

Ejemplos: Combinaciones

En esta sección, se proporcionan ejemplos de aplicaciones de análisis de datos de Kinesis Data Analytics que utilizan consultas de combinación. Cada ejemplo proporciona step-by-step instrucciones para configurar y probar la aplicación Kinesis Data Analytics.

Temas

Ejemplo: Agregar datos de referencia a una aplicación de Kinesis Data Analytics

En este ejercicio añadirá datos de referencia a una aplicación de Kinesis Data Analytics existente. Para obtener información sobre los datos de referencia, consulte los siguientes temas:

- Aplicaciones de Amazon Kinesis Data Analytics para SQL: cómo funciona
- Configuración de entrada de la aplicación

En este ejercicio, añadirá los datos de referencia a la aplicación que creó en el ejercicio de Introducción de Kinesis Data Analytics. Los datos de referencia proporcionan el nombre de la empresa para cada símbolo de cotización (ticker); por ejemplo:

```
Ticker, Company
AMZN,Amazon
ASD, SomeCompanyA
MMB, SomeCompanyB
WAS,  SomeCompanyC
```

En primer lugar, complete los pasos del ejercicio de introducción para crear una aplicación inicial. A continuación, siga estos pasos para configurar y añadir datos de referencia a la aplicación:

1. Prepare los datos.
   - Guarde los datos de referencia anteriores como un objeto en Amazon Simple Storage Service (Amazon S3).
   - Cree un rol de IAM, que Kinesis Data Analytics pueda asumir para leer el objeto de Amazon S3 en su nombre.

2. Añada la fuente de los datos de referencia a su aplicación.

   Kinesis Data Analytics lee el objeto de Amazon S3 y crea una tabla de referencia en la aplicación que se puede consultar en el código de la aplicación.

3. Pruebe el código.

   En el código de la aplicación, escriba una consulta de combinación para unir la secuencia en la aplicación con la tabla de referencia en la aplicación con el fin de obtener el nombre de la empresa para cada símbolo de cotización.

Temas

Paso 1: Preparación

En esta sección, almacenará los datos de referencia de muestra como un objeto en un bucket de Amazon S3. También, creará un rol de IAM que Kinesis Data Analytics pueda asumir para leer el objeto en su nombre.

Almacenamiento de datos de referencia como un objeto de Amazon S3

En este paso, almacenará los datos de referencia de muestra como un objeto de Amazon S3.

1. Abra un editor de texto, añada los siguientes datos y guarde el archivo como `TickerReference.csv`.

   ```
   Ticker, Company
   AMZN,Amazon
   ASD, SomeCompanyA
   MMB, SomeCompanyB
   WAS,  SomeCompanyC
   ```

2. Cargue el archivo `TickerReference.csv` en el bucket de S3. Para ver las instrucciones, consulte Carga de objetos en Amazon S3 en la Guía del usuario de Amazon Simple Storage Service.

Creación de un rol de IAM

A continuación, cree un rol de IAM que Kinesis Data Analytics pueda asumir y lea el objeto de Amazon S3.

1. En AWS Identity and Access Management (IAM), cree un rol de IAM denominado. **`KinesisAnalytics-ReadS3Object`** Para crear el rol, siga las instrucciones que se describen en Creación de un rol para un servicio de Amazon (Consola de administración de AWS) en la Guía del usuario de IAM.

   En la consola de IAM, especifique lo siguiente:

   - En Seleccionar tipo de rol, elija AWS Lambda. Tras crear el rol, cambiará la política de confianza para permitir que Kinesis Data Analytics (AWS Lambda no) asuma el rol.
   - No asigne ninguna política en la página Attach Policy.
2. Actualice las políticas de roles de IAM:

   a. En la consola de IAM, elija el rol que ha creado.

   b. En la pestaña Relaciones de confianza, actualice la política de confianza para conceder permisos a Kinesis Data Analytics para asumir el rol. La política de confianza se muestra tras:

      JSON

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

c. En la pestaña Permisos, adjunta una política gestionada por Amazon llamada AmazonS3. ReadOnlyAccess Esto concede permisos al rol para leer un objeto de Amazon S3. Esta política se muestra a continuación:

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:Get*",
        "s3:List*"
      ],
      "Resource": "*"
    }
  ]
}
```

Paso 2: Añadir el origen de datos de referencia a la configuración de la aplicación

En este paso, añadirá un origen de datos de referencia a la configuración de la aplicación. Para comenzar, necesitará la siguiente información:

- El nombre del bucket de S3 y el nombre de la clave de objeto
- El nombre de recurso de Amazon (ARN) del rol de IAM

1. En página principal de la aplicación, elija Connect reference data (Conectar datos de referencia).
2. En la página Conectar el origen de datos de referencia, elija el bucket de Amazon S3 que contiene el objeto de datos de referencia y escriba el nombre de la clave del objeto.
3. Escriba **`CompanyName`** como In-application reference table name (Nombre de tabla de referencia en la aplicación).
4. En la sección Acceso a los recursos elegidos, elija entre las funciones de IAM que Kinesis Analytics puede asumir y elija la función de IAM -READS3Object que creó en KinesisAnalyticsla sección anterior.
5. Seleccione Detectar esquema. La consola detecta dos columnas en los datos de referencia.

6. Elija Save and close.

Paso 3: Prueba: Consultar la tabla de referencia en la aplicación

Ahora, puede consultar la tabla de referencia en la aplicación, `CompanyName`. Puede utilizar la información de referencia para mejorar la aplicación uniendo los datos de precio de cotización con la tabla de referencia. El resultado muestra el nombre de la compañía.

1. Reemplace el código de la aplicación por lo siguiente. La consulta une la secuencia de entrada en la aplicación con la tabla de referencia en la aplicación. El código de la aplicación escribe los resultados en otra secuencia en la aplicación, `DESTINATION_SQL_STREAM`.

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (ticker_symbol VARCHAR(4),
 "Company" varchar(20), sector VARCHAR(12), change DOUBLE, price DOUBLE);

CREATE OR REPLACE PUMP "STREAM_PUMP" AS INSERT INTO "DESTINATION_SQL_STREAM"
   SELECT STREAM ticker_symbol, "c"."Company", sector, change, price
   FROM "SOURCE_SQL_STREAM_001" LEFT JOIN "CompanyName" as "c"
   ON "SOURCE_SQL_STREAM_001".ticker_symbol = "c"."Ticker";
```

2. Compruebe que el resultado de la aplicación aparezca en la pestaña. SQLResults Asegúrese de que algunas de las filas muestren nombres de empresas (no todos los datos de referencia de muestra tienen nombres de empresas).

Ejemplos: Aprendizaje automático

En esta sección, se proporcionan ejemplos de aplicaciones de Amazon Kinesis Data Analytics que utilizan consultas de machine learning. Las consultas de aprendizaje automático realizan un análisis complejo de los datos, basándose en el historial de los datos de la secuencia para descubrir patrones inusuales. Los ejemplos proporcionan step-by-step instrucciones para configurar y probar la aplicación Kinesis Data Analytics.

Temas

- Ejemplo: Detección de anomalías de datos en una secuencia (función RANDOM_CUT_FOREST)
- Ejemplo: Detección de anomalías de datos y obtención de una explicación (función RANDOM_CUT_FOREST_WITH_EXPLANATION)
- Ejemplo: Detección de puntos calientes en una secuencia (función HOTSPOSTS)

Ejemplo: Detección de anomalías de datos en una secuencia (función RANDOM_CUT_FOREST)

Amazon Kinesis Data Analytics proporciona una función (`RANDOM_CUT_FOREST`) que puede asignar una puntuación de anomalías para cada registro en función de los valores en las columnas numéricas. Para obtener más información, consulte Función `RANDOM_CUT_FOREST` en la Referencia de SQL de Amazon Managed Service para Apache Flink.

En este ejercicio, escribirá el código de la aplicación para asignar una puntuación de anomalías a los registros en el origen de streaming de la aplicación. Haga lo siguiente para configurar la aplicación:

1. Configure un origen de streaming: configure un flujo de datos de Kinesis y escriba datos de `heartRate` de muestra, tal y como se muestra a continuación:

   ```
   {"heartRate": 60, "rateType":"NORMAL"}
   ...
   {"heartRate": 180, "rateType":"HIGH"}
   ```

   El procedimiento proporciona un script de Python para que pueda rellenar la secuencia. Los valores de `heartRate` se generan de forma aleatoria. El 99 por ciento de los registros tienen valores de `heartRate` entre 60 y 100, y solo un 1 por ciento de ellos tienen valores de `heartRate` entre 150 y 200. Por lo tanto, los registros que tienen valores de `heartRate` comprendidos entre 150 y 200 son anomalías.
2. Configure la entrada: use la consola para crear una aplicación de Kinesis Data Analytics y configure la entrada de la aplicación mapeando el origen de streaming a una secuencia en la aplicación (`SOURCE_SQL_STREAM_001`). Cuando se inicia la aplicación, Kinesis Data Analytics lee continuamente el origen de streaming e introduce los registros en la secuencia en la aplicación.
3. Especifique el código de la aplicación: el ejemplo utiliza el siguiente código de la aplicación:

   ```
   --Creates a temporary stream.
   CREATE OR REPLACE STREAM "TEMP_STREAM" (
            "heartRate"        INTEGER,
            "rateType"         varchar(20),
            "ANOMALY_SCORE"    DOUBLE);

   --Creates another stream for application output.
   CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
   ```

```
        "heartRate"        INTEGER,
        "rateType"         varchar(20),
        "ANOMALY_SCORE"    DOUBLE);

-- Compute an anomaly score for each record in the input stream
-- using Random Cut Forest
CREATE OR REPLACE PUMP "STREAM_PUMP" AS
   INSERT INTO "TEMP_STREAM"
      SELECT STREAM "heartRate", "rateType", ANOMALY_SCORE
      FROM TABLE(RANDOM_CUT_FOREST(
              CURSOR(SELECT STREAM * FROM "SOURCE_SQL_STREAM_001")));

-- Sort records by descending anomaly score, insert into output stream
CREATE OR REPLACE PUMP "OUTPUT_PUMP" AS
   INSERT INTO "DESTINATION_SQL_STREAM"
      SELECT STREAM * FROM "TEMP_STREAM"
      ORDER BY FLOOR("TEMP_STREAM".ROWTIME TO SECOND), ANOMALY_SCORE DESC;
```

El código lee filas de la puntuación `SOURCE_SQL_STREAM_001`, asigna un puntaje de anomalía, y escribe las filas resultante a otra secuencia en la aplicación (`TEMP_STREAM`). El código de la aplicación luego clasifica los registros en `TEMP_STREAM` y guarda los resultados en otra secuencia en la aplicación (`DESTINATION_SQL_STREAM`). Utilice bombas para insertar filas en secuencias en la aplicación. Para obtener más información, consulte Secuencias y bombeos en la aplicación.

4. Configure la salida: configure la salida de la aplicación para conservar los datos de `DESTINATION_SQL_STREAM` en un destino externo, que es otro flujo de datos de Kinesis. La revisión de las puntuaciones de anomalías que se han asignado a cada registro y la determinación de qué puntuación indica que se ha producido una anomalía (y que es necesario que le avisen) es algo ajeno a la aplicación. Puede utilizar una AWS Lambda función para procesar estas puntuaciones de anomalías y configurar las alertas.

El ejercicio utiliza la región de Este de EE. UU. (Norte de Virginia) (`us-east-1`) para crear estas secuencias y su aplicación. Si utiliza cualquier otra región, debe actualizar el código en consecuencia.

Temas

- Paso 1: Preparación
- Paso 2: Cree una aplicación

- Paso 3: Configurar la salida de la aplicación
- Paso 4: Verificar la salida

Paso siguiente

Paso 1: Preparación

Paso 1: Preparación

Antes de crear una aplicación de análisis de datos de Amazon Kinesis Data Analytics para este ejercicio, deberá crear dos flujos de datos de Kinesis. Configure una de las secuencias como el origen de streaming de su aplicación y otra secuencia como el destino, donde Kinesis Data Analytics sigue siendo la salida de su aplicación.

Temas

- Paso 1.1: crear los flujos de datos de entrada y de salida
- Paso 1.2: Escribir registros de muestra en la secuencia de entrada

Paso 1.1: crear los flujos de datos de entrada y de salida

En esta sección, creará dos secuencias de Kinesis: `ExampleInputStream` y `ExampleOutputStream`. Puede crear estas secuencias mediante la Consola de administración de AWS o la AWS CLI.

- Para utilizar la consola de

  1. Inicie sesión en la consola de Kinesis Consola de administración de AWS y ábrala en https://console.aws.amazon.com /kinesis.
  2. Elija Create data stream (Crear flujo de datos). Cree una secuencia con un fragmento denominado `ExampleInputStream`. Para obtener más información, consulte Crear secuencia en la Guía para desarrolladores de Amazon Kinesis Data Streams.
  3. Repita el paso anterior y cree una secuencia con un fragmento denominada `ExampleOutputStream`.

- Para usar el AWS CLI

  1. Utilice el siguiente `create-stream` AWS CLI comando de Kinesis para crear la primera transmisión ()`ExampleInputStream`.

```
$ aws kinesis create-stream \
--stream-name ExampleInputStream \
--shard-count 1 \
--region us-east-1 \
--profile adminuser
```

2. Ejecute el mismo comando, pero cambie el nombre de la secuencia por `ExampleOutputStream`. Este comando crea la segunda secuencia que usa la aplicación para escribir la salida.

Paso 1.2: Escribir registros de muestra en la secuencia de entrada

En este paso, ejecute el código de Python para generar registros de muestra y escribirlos en la secuencia `ExampleInputStream`.

```
{"heartRate": 60, "rateType":"NORMAL"}
...
{"heartRate": 180, "rateType":"HIGH"}
```

1. Instale Python y `pip`.

   Para obtener más información sobre la instalación de Python, consulte la página web de Python.

   Puede instalar dependencias con pip. Para obtener más información sobre la instalación de pip, consulte la sección Installation en la página web de pip.
2. Ejecute el siguiente código de Python. El comando `put-record` en el código escribe los registros JSON en la secuencia.

```
from enum import Enum
import json
import random
import boto3

STREAM_NAME = "ExampleInputStream"


class RateType(Enum):
    normal = "NORMAL"
    high = "HIGH"
```

```
def get_heart_rate(rate_type):
    if rate_type == RateType.normal:
        rate = random.randint(60, 100)
    elif rate_type == RateType.high:
        rate = random.randint(150, 200)
    else:
        raise TypeError
    return {"heartRate": rate, "rateType": rate_type.value}


def generate(stream_name, kinesis_client, output=True):
    while True:
        rnd = random.random()
        rate_type = RateType.high if rnd < 0.01 else RateType.normal
        heart_rate = get_heart_rate(rate_type)
        if output:
            print(heart_rate)
        kinesis_client.put_record(
            StreamName=stream_name,
            Data=json.dumps(heart_rate),
            PartitionKey="partitionkey",
        )


if __name__ == "__main__":
    generate(STREAM_NAME, boto3.client("kinesis"))
```

Paso siguiente

Paso 2: Cree una aplicación

Paso 2: Cree una aplicación

En esta sección va a crear una aplicación de análisis de datos de Amazon Kinesis Data Analytics de la siguiente manera:

- Configure la entrada de la aplicación para que utilice el flujo de datos de Kinesis que ha creado en the section called “Paso 1: Preparación” como origen de streaming.
- Utilice la plantilla Anomaly Detection (Detección de anomalías) en la consola.

Cómo crear una aplicación de

1. Siga los pasos 1, 2 y 3 del ejercicio de Introducción a Kinesis Data Analytics (consulte Paso 3.1: Cree una aplicación).

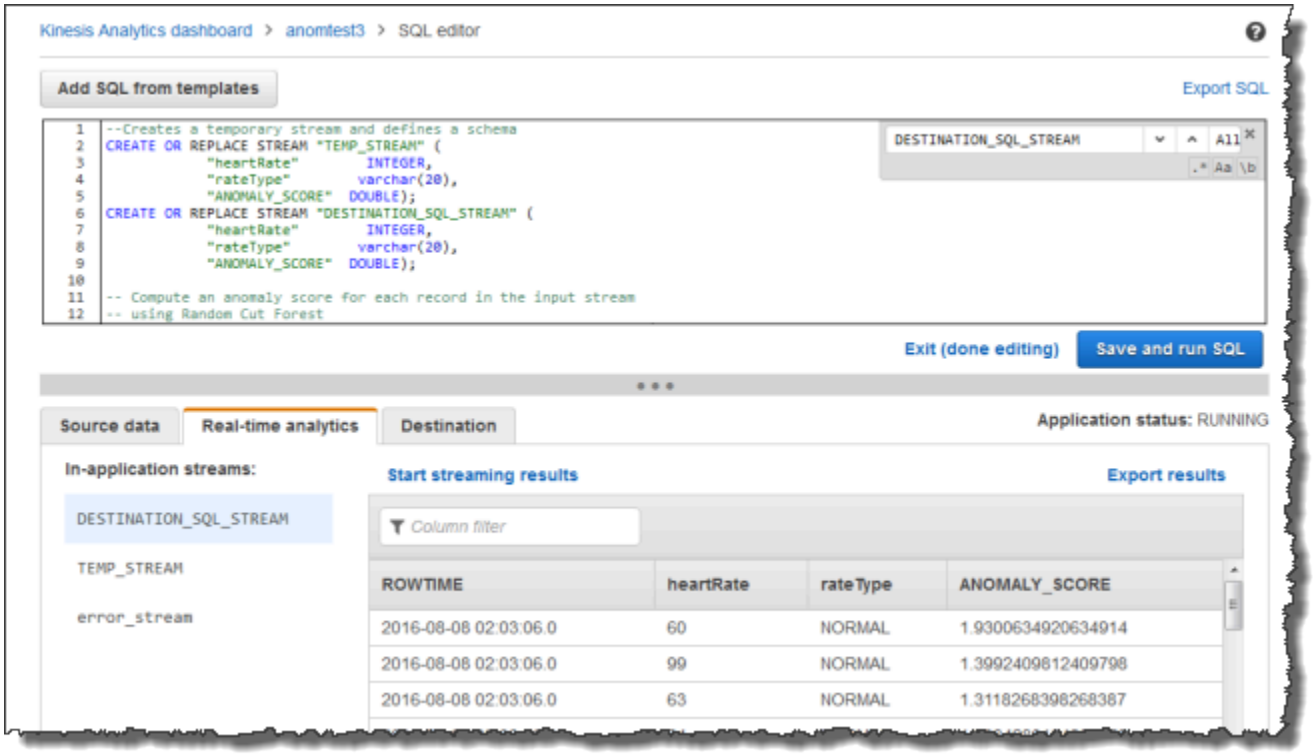
   - En la configuración fuente, haga lo siguiente:
     - Especifique el origen de streaming que ha creado en la sección anterior.
     - Después de que la consola infiera el esquema, edite este y establezca el tipo de columna de `heartRate` en `INTEGER`.

       La mayor parte de los valores de la frecuencia cardíaca son normales y el proceso de detección probablemente asignará el tipo `TINYINT` a esta columna. Pero un pequeño porcentaje de valores muestran una frecuencia cardíaca alta. Si estos valores no encajan en el tipo `TINYINT`, Kinesis Data Analytics envía estas filas a una secuencia de errores. Actualice el tipo de datos a `INTEGER` para que pueda adaptarse a todos los datos de frecuencia cardíaca que se han generado.
   - Utilice la plantilla Anomaly Detection (Detección de anomalías) en la consola. A continuación, actualice el código de la plantilla para proporcionar el nombre de columna apropiado.
2. Actualice el código de la aplicación al proporcionar los nombres de columnas. El código de la aplicación resultante se muestra a continuación (pegue este código en el editor de SQL):

```
--Creates a temporary stream.
CREATE OR REPLACE STREAM "TEMP_STREAM" (
         "heartRate"        INTEGER,
         "rateType"         varchar(20),
         "ANOMALY_SCORE"    DOUBLE);

--Creates another stream for application output.
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
         "heartRate"        INTEGER,
         "rateType"         varchar(20),
         "ANOMALY_SCORE"    DOUBLE);

-- Compute an anomaly score for each record in the input stream
-- using Random Cut Forest
CREATE OR REPLACE PUMP "STREAM_PUMP" AS
   INSERT INTO "TEMP_STREAM"
      SELECT STREAM "heartRate", "rateType", ANOMALY_SCORE
      FROM TABLE(RANDOM_CUT_FOREST(
              CURSOR(SELECT STREAM * FROM "SOURCE_SQL_STREAM_001")));
```

```
-- Sort records by descending anomaly score, insert into output stream
CREATE OR REPLACE PUMP "OUTPUT_PUMP" AS
   INSERT INTO "DESTINATION_SQL_STREAM"
      SELECT STREAM * FROM "TEMP_STREAM"
      ORDER BY FLOOR("TEMP_STREAM".ROWTIME TO SECOND), ANOMALY_SCORE DESC;
```

3. Ejecute el código SQL y revise los resultados en la consola de Kinesis Data Analytics:



Paso siguiente

[Paso 3: Configurar la salida de la aplicación](#)

Paso 3: Configurar la salida de la aplicación

Una vez completado el the section called “Paso 2: Cree una aplicación”, dispone de código de la aplicación que lee datos de frecuencia cardíaca de un origen de streaming y asigna una puntuación de anomalías a cada uno de ellos.

Ahora puede enviar el resultado de la aplicación desde la secuencia en la aplicación hasta un destino externo, que es otro flujo de datos de Kinesis (`OutputStreamTestingAnomalyScores`). Puede

analizar las puntuaciones de anomalías y determinar qué frecuencia cardíaca es anómala. Puede ampliar aún más esta aplicación para generar alertas.

Siga estos pasos para configurar la salida de la aplicación:

1. Abra la consola de Amazon Kinesis Data Analytics. En el editor de SQL, puede elegir tanto Destination como Add a destination en el panel de la aplicación.
2. En la página Connect to destination (Conectarse a un destino), elija la secuencia `OutputStreamTestingAnomalyScores` que ha creado en la sección anterior.

   Ahora tiene un destino externo, donde Amazon Kinesis Data Analytics persiste en cualquier registro que escriba su aplicación en la secuencia en la aplicación `DESTINATION_SQL_STREAM`.
3. Si lo desea, puede configurarlo AWS Lambda para que supervise la `OutputStreamTestingAnomalyScores` transmisión y le envíe alertas. Para obtener instrucciones, consulte Procesamiento previo de registros con una función de Lambda. Si no configura alertas, puede revisar los registros que Kinesis Data Analytics escribe en el destino externo, que es un flujo de datos Kinesis de `OutputStreamTestingAnomalyScores`, tal como se describe en Paso 4: Verificar la salida.

Paso siguiente

Paso 4: Verificar la salida

Paso 4: Verificar la salida

Después de configurar la salida de la aplicación en the section called “Paso 3: Configurar la salida de la aplicación”, utilice los siguientes comandos de la AWS CLI para leer los registros de la secuencia de destino que escribe la aplicación:

1. Ejecute el comando `get-shard-iterator` para obtener un puntero a los datos en la secuencia de salida.

```
aws kinesis get-shard-iterator \
--shard-id shardId-000000000000 \
--shard-iterator-type TRIM_HORIZON \
--stream-name OutputStreamTestingAnomalyScores \
--region us-east-1 \
--profile adminuser
```

Puede obtener una respuesta con un valor iterador de fragmento, tal y como se muestra en el ejemplo siguiente respuesta:

```
{
    "ShardIterator":
    "shard-iterator-value"   }
```

Copie el valor de iterador de fragmento.

2. Ejecute el comando AWS CLI `get-records`.

```
aws kinesis get-records \
--shard-iterator shared-iterator-value \
--region us-east-1 \
--profile adminuser
```

El comando devuelve una página de registros y otro iterador de fragmento que puede utilizar en el comando `get-records` posterior para recuperar el siguiente conjunto de registros.

Ejemplo: Detección de anomalías de datos y obtención de una explicación (función RANDOM_CUT_FOREST_WITH_EXPLANATION)

Amazon Kinesis Data Analytics proporciona la función `RANDOM_CUT_FOREST_WITH_EXPLANATION`, que asigna una puntuación de anomalías a cada registro en función de los valores de las columnas numéricas. La función también ofrece una explicación de la anomalía. Para obtener más información, consulte RANDOM_CUT_FOREST_WITH_EXPLANATION en la Referencia de SQL de Amazon Managed Service para Apache Flink.

En este ejercicio, escribirá el código de una aplicación para obtener las puntuaciones de anomalías para los registros del origen de streaming de la aplicación. También obtendrá una explicación para cada anomalía.

Temas

- Paso 1: Preparar los datos
- Paso 2: Crear una aplicación de análisis
- Paso 3: Examinar los resultados

Primer paso

Paso 1: Preparar los datos

Paso 1: Preparar los datos

Antes de crear una aplicación de Amazon Kinesis Data Analytics para este ejemplo, debe crear un flujo de datos de Kinesis para utilizarla como origen de streaming para su aplicación. También debe ejecutar código de Python para escribir datos simulados de tensión arterial en la secuencia.

Temas

- Paso 1.1: crear un flujo de datos de Kinesis
- Paso 1.2: Escribir registros de muestra en la secuencia de entrada

Paso 1.1: crear un flujo de datos de Kinesis

En esta sección creará un flujo de datos de Kinesis denominada `ExampleInputStream`. Puede crear este flujo de datos utilizando el Consola de administración de AWS o el AWS CLI.

- Para utilizar la consola:
  1. Inicie sesión en la consola de Kinesis Consola de administración de AWS y ábrala en https://console.aws.amazon.com /kinesis.
  2. Elija Flujos de datos en el panel de navegación. A continuación, elija Create Kinesis stream (Crear secuencia de Kinesis).
  3. Escriba **`ExampleInputStream`** como nombre del parámetro. Para el número de fragmentos, escriba **`1`**.
- Como alternativa, para usar el AWS CLI para crear el flujo de datos, ejecute el siguiente comando:

```
$ aws kinesis create-stream --stream-name ExampleInputStream --shard-count 1
```

Paso 1.2: Escribir registros de muestra en la secuencia de entrada

En este paso, ejecutará código de Python para generar continuamente registros de muestra y escribirlos en el flujo de datos que ha creado.

1. Instale Python y pip.

Para obtener información sobre la instalación de Python, consulte Python.

Puede instalar dependencias con pip. Para obtener información sobre la instalación de pip, consulte Installation en la documentación de pip.

2. Ejecute el siguiente código de Python. Puede cambiar la región por la que desee utilizar en este ejemplo. El comando `put-record` en el código escribe los registros JSON en la secuencia.

```
from enum import Enum
import json
import random
import boto3

STREAM_NAME = "ExampleInputStream"


class PressureType(Enum):
    low = "LOW"
    normal = "NORMAL"
    high = "HIGH"


def get_blood_pressure(pressure_type):
    pressure = {"BloodPressureLevel": pressure_type.value}
    if pressure_type == PressureType.low:
        pressure["Systolic"] = random.randint(50, 80)
        pressure["Diastolic"] = random.randint(30, 50)
    elif pressure_type == PressureType.normal:
        pressure["Systolic"] = random.randint(90, 120)
        pressure["Diastolic"] = random.randint(60, 80)
    elif pressure_type == PressureType.high:
        pressure["Systolic"] = random.randint(130, 200)
        pressure["Diastolic"] = random.randint(90, 150)
    else:
        raise TypeError
    return pressure


def generate(stream_name, kinesis_client):
    while True:
        rnd = random.random()
        pressure_type = (
```

```
            PressureType.low
            if rnd < 0.005
            else PressureType.high
            if rnd > 0.995
            else PressureType.normal
        )
        blood_pressure = get_blood_pressure(pressure_type)
        print(blood_pressure)
        kinesis_client.put_record(
            StreamName=stream_name,
            Data=json.dumps(blood_pressure),
            PartitionKey="partitionkey",
        )


if __name__ == "__main__":
    generate(STREAM_NAME, boto3.client("kinesis"))
```

Paso siguiente

Paso 2: Crear una aplicación de análisis

Paso 2: Crear una aplicación de análisis

En esta sección creará una aplicación de análisis de datos de Amazon Kinesis Data Analytics y la configurará para que utilice el flujo de datos de Kinesis que ha creado como origen de streaming en the section called “Paso 1: Preparar los datos”. A continuación, ejecutará código de la aplicación que utiliza la función `RANDOM_CUT_FOREST_WITH_EXPLANATION`.

Cómo crear una aplicación de

1. Abra la consola de Kinesis en https://console.aws.amazon.com /kinesis.
2. Elija Data Analytics (Análisis de datos) en el panel de navegación y, a continuación, elija Create application (Crear aplicación).
3. Proporcione un nombre y una descripción (opcional) para la aplicación y elija Create application.
4. Selecciona Conectar datos de streaming y, a continuación, selecciona una opción ExampleInputStreamde la lista.
5. Elija Discover esquema y asegúrese de que `Systolic` y `Diastolic` aparecen como columnas de tipo `INTEGER`. Si son de otro tipo, seleccione Edit schema y asigne el tipo `INTEGER` a cada una de ellas.

6. En Real time analytics, elija Go to SQL editor. Cuando se le pregunte, elija la opción de ejecutar la aplicación.

7. Pegue el código siguiente en el editor de SQL y, a continuación, elija Save and run SQL.

```
--Creates a temporary stream.
CREATE OR REPLACE STREAM "TEMP_STREAM" (
         "Systolic"                 INTEGER,
         "Diastolic"                INTEGER,
         "BloodPressureLevel"       varchar(20),
         "ANOMALY_SCORE"            DOUBLE,
         "ANOMALY_EXPLANATION"      varchar(512));

--Creates another stream for application output.
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
         "Systolic"                 INTEGER,
         "Diastolic"                INTEGER,
         "BloodPressureLevel"       varchar(20),
         "ANOMALY_SCORE"            DOUBLE,
         "ANOMALY_EXPLANATION"      varchar(512));

-- Compute an anomaly score with explanation for each record in the input stream
-- using RANDOM_CUT_FOREST_WITH_EXPLANATION
CREATE OR REPLACE PUMP "STREAM_PUMP" AS
   INSERT INTO "TEMP_STREAM"
      SELECT STREAM "Systolic", "Diastolic", "BloodPressureLevel", ANOMALY_SCORE,
 ANOMALY_EXPLANATION
      FROM TABLE(RANDOM_CUT_FOREST_WITH_EXPLANATION(
              CURSOR(SELECT STREAM * FROM "SOURCE_SQL_STREAM_001"), 100, 256,
 100000, 1, true));

-- Sort records by descending anomaly score, insert into output stream
CREATE OR REPLACE PUMP "OUTPUT_PUMP" AS
   INSERT INTO "DESTINATION_SQL_STREAM"
      SELECT STREAM * FROM "TEMP_STREAM"
      ORDER BY FLOOR("TEMP_STREAM".ROWTIME TO SECOND), ANOMALY_SCORE DESC;
```

Paso siguiente

Paso 3: Examinar los resultados

Paso 3: Examinar los resultados

Cuando ejecute el código SQL de este ejemplo, primero verá filas con una puntuación de anomalías igual a cero. Esto ocurre durante la fase de aprendizaje inicial. A continuación, obtendrá resultados similares a los siguientes:

```
ROWTIME SYSTOLIC DIASTOLIC BLOODPRESSURELEVEL ANOMALY_SCORE ANOMALY_EXPLANATION
27:49.0 101      66        NORMAL             0.711460417   {"Systolic":
{"DIRECTION":"LOW","STRENGTH":"0.0922","ATTRIBUTION_SCORE":"0.3792"},"Diastolic":
{"DIRECTION":"HIGH","STRENGTH":"0.0210","ATTRIBUTION_SCORE":"0.3323"}}
27:50.0 144      123       HIGH               3.855851061   {"Systolic":
{"DIRECTION":"HIGH","STRENGTH":"0.8567","ATTRIBUTION_SCORE":"1.7447"},"Diastolic":
{"DIRECTION":"HIGH","STRENGTH":"7.0982","ATTRIBUTION_SCORE":"2.1111"}}
27:50.0 113      69        NORMAL             0.740069409   {"Systolic":
{"DIRECTION":"LOW","STRENGTH":"0.0549","ATTRIBUTION_SCORE":"0.3750"},"Diastolic":
{"DIRECTION":"LOW","STRENGTH":"0.0394","ATTRIBUTION_SCORE":"0.3650"}}
27:50.0 105      64        NORMAL             0.739644157   {"Systolic":
{"DIRECTION":"HIGH","STRENGTH":"0.0245","ATTRIBUTION_SCORE":"0.3667"},"Diastolic":
{"DIRECTION":"LOW","STRENGTH":"0.0524","ATTRIBUTION_SCORE":"0.3729"}}
27:50.0 100      65        NORMAL             0.736993425   {"Systolic":
{"DIRECTION":"HIGH","STRENGTH":"0.0203","ATTRIBUTION_SCORE":"0.3516"},"Diastolic":
{"DIRECTION":"LOW","STRENGTH":"0.0454","ATTRIBUTION_SCORE":"0.3854"}}
27:50.0 108      69        NORMAL             0.733767202   {"Systolic":
{"DIRECTION":"LOW","STRENGTH":"0.0974","ATTRIBUTION_SCORE":"0.3961"},"Diastolic":
{"DIRECTION":"LOW","STRENGTH":"0.0189","ATTRIBUTION_SCORE":"0.3377"}}
```

- El algoritmo de la función `RANDOM_CUT_FOREST_WITH_EXPLANATION` considera que las columnas `Systolic` y `Diastolic` son numéricas y las utiliza como entrada.
- La columna `BloodPressureLevel` contiene datos de texto, por lo que el algoritmo no la tiene en cuenta. Esta columna simplemente es una ayuda visual para ayudarle a reconocer rápidamente los niveles normales, altos y bajos de tensión arterial de este ejemplo.
- En la columna `ANOMALY_SCORE`, los registros con puntuaciones más altas son los que presentan una mayor anomalía. El segundo registro de este conjunto de resultados de ejemplo el más anómalo, con una puntuación de anomalías de 3,855851061.
- Para comprender en qué medida contribuye cada una de las columnas numéricas analizadas por el algoritmo a la puntuación de anomalías, consulte el campo JSON denominado `ATTRIBUTION_SCORE` en la columna `ANOMALY_SCORE`. En el caso de la segunda fila de este conjunto de resultados de muestra, las columnas `Systolic` y `Diastolic` contribuyen a la

anomalía en la proporción de 1.7447:2.1111. En otras palabras, el 45 por ciento de la explicación de la puntuación de anomalías puede atribuirse al valor sistólico y el resto se debe al valor diastólico.

- Para determinar en qué dirección es anómalo el punto representado por la segunda fila de este ejemplo, consulte el campo JSON denominado `DIRECTION`. En este caso, tanto el valor diastólico como el valor sistólico están marcados como `HIGH`. Para determinar la confianza con la que estas direcciones son correctas, consulte el campo JSON denominado `STRENGTH`. En este ejemplo, el algoritmo está más seguro de que el valor diastólico es alto. En efecto, el valor normal para la lectura diastólica suele estar entre 60 y 80, y 123 es mucho mayor de lo esperado.

Ejemplo: Detección de puntos calientes en una secuencia (función HOTSPOSTS)

Amazon Kinesis Data Analytics dispone de la función `HOTSPOTS` que localiza y devuelve información sobre las regiones relativamente densas en los datos. Para obtener más información, consulte HOTSPOTS en la Referencia de SQL de Amazon Managed Service para Apache Flink.

En este ejercicio, puede escribir código de la aplicación para localizar puntos calientes en el origen de streaming de su aplicación. Realice los pasos siguientes para configurar la aplicación:

1. Configure un origen de streaming: configure una secuencia de Kinesis y escriba datos coordinados de muestra, tal y como se muestra a continuación:

   ```
   {"x": 7.921782426109737, "y": 8.746265312709893, "is_hot": "N"}
   {"x": 0.722248626528026, "y": 4.648868803193405, "is_hot": "Y"}
   ```

   El ejemplo proporciona un script de Python para que pueda rellenar la secuencia. Los valores `x` e `y` se generan de forma aleatoria. Algunos registros se agrupan en torno a determinadas ubicaciones.

   El campo `is_hot` se suministra como indicador si el script ha generado el valor de forma intencionada como parte de un punto caliente. Esto puede ayudarle a evaluar si la función de detección de puntos calientes funciona correctamente.

2. Cree la aplicación: utilice la Consola de administración de AWS para crear la aplicación de análisis de datos de Kinesis Data Analytics. Configure la entrada de la aplicación mapeando el origen de streaming a una secuencia en la aplicación (`SOURCE_SQL_STREAM_001`). Cuando se

inicia la aplicación, Kinesis Data Analytics lee continuamente el origen de streaming e introduce los registros en la secuencia en la aplicación.

En este ejercicio, utilizará el siguiente código para la aplicación:

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
    "x" DOUBLE,
    "y" DOUBLE,
    "is_hot" VARCHAR(4),
    HOTSPOTS_RESULT VARCHAR(10000)
);
CREATE OR REPLACE PUMP "STREAM_PUMP" AS
    INSERT INTO "DESTINATION_SQL_STREAM"
    SELECT "x", "y", "is_hot", "HOTSPOTS_RESULT"
    FROM TABLE (
        HOTSPOTS(
            CURSOR(SELECT STREAM "x", "y", "is_hot" FROM "SOURCE_SQL_STREAM_001"),
            1000,
            0.2,
            17)
    );
```

El código lee filas en `SOURCE_SQL_STREAM_001`, las analiza en busca de puntos calientes relevantes y escribe los datos resultantes en otra secuencia en la aplicación (`DESTINATION_SQL_STREAM`). Utilice bombas para insertar filas en secuencias en la aplicación. Para obtener más información, consulte Secuencias y bombeos en la aplicación.

3. Configure la salida: configure la salida de la aplicación para que envíe los datos desde la aplicación a un destino externo, que es otro flujo de datos de Kinesis. Revise las puntuaciones de los puntos calientes y determine qué puntuaciones indican que se ha producido un punto caliente (del cual desea que se le avise). Puede utilizar una AWS Lambda función para seguir procesando la información de los puntos de acceso y configurar las alertas.
4. Verifique el resultado: el ejemplo incluye una JavaScript aplicación que lee los datos del flujo de salida y los muestra gráficamente, de modo que pueda ver los puntos de acceso que genera la aplicación en tiempo real.

El ejercicio utiliza la región de Oeste de EE. UU. (Oregón) (`us-west-2`) para crear estas secuencias y su aplicación. Si utiliza otra región, deberá actualizar el código en consecuencia.

Temas

Paso 1: Crear las secuencias de entrada y de salida

Antes de crear una aplicación de análisis de datos de Amazon Kinesis Data Analytics para el Ejemplo de puntos de acceso, deberá crear dos flujos de datos de Kinesis. Configure una de las secuencias como el origen de streaming de su aplicación y otra secuencia como el destino, donde Kinesis Data Analytics sigue siendo la salida de su aplicación.

Temas

Paso 1.1: crear los flujos de datos de Kinesis

En esta sección, creará dos secuencias de Kinesis: `ExampleInputStream` y `ExampleOutputStream`.

Cree estas flujos de datos con la consola o la AWS CLI.

- Para crear estas flujos de datos con la consola:

  1. Inicie sesión en la consola de Kinesis Consola de administración de AWS y ábrala en https://console.aws.amazon.com /kinesis.
  2. Elija Flujos de datos en el panel de navegación.
  3. Elija Create Kinesis Stream (Crear secuencia de Kinesis) y, a continuación, cree una secuencia con un fragmento denominado `ExampleInputStream`.
  4. Repita el paso anterior y cree una secuencia con un fragmento denominada `ExampleOutputStream`.

- Para crear flujos de datos con la AWS CLI:

- Cree transmisiones (`ExampleInputStreamyExampleOutputStream`) mediante el siguiente comando de Kinesis `create-stream` AWS CLI . Para crear la segunda secuencia, que la aplicación utilizará para escribir la salida, ejecute el mismo comando, cambiando el nombre de la secuencia a `ExampleOutputStream`.

```
$ aws kinesis create-stream \
--stream-name ExampleInputStream \
--shard-count 1 \
--region us-west-2 \
--profile adminuser

$ aws kinesis create-stream \
--stream-name ExampleOutputStream \
--shard-count 1 \
--region us-west-2 \
--profile adminuser
```

Paso 1.2: Escribir registros de muestra en la secuencia de entrada

En este paso, ejecute el código de Python para generar registros de muestra y escribir en la secuencia `ExampleInputStream`.

```
{"x": 7.921782426109737, "y": 8.746265312709893, "is_hot": "N"}
{"x": 0.722248626580026, "y": 4.648868803193405, "is_hot": "Y"}
```

1. Instale Python y `pip`.

   Para obtener más información sobre la instalación de Python, consulte la página web de Python.

   Puede instalar dependencias con pip. Para obtener más información sobre la instalación de pip, consulte la sección Installation en la página web de pip.
2. Ejecute el siguiente código de Python. Este código hace lo siguiente:

   - Genera un punto caliente potencial en algún lugar del plano (X, Y).
   - Genera un conjunto de 1000 puntos para cada punto caliente. De estos puntos, el 20 % se agrupa en torno a un punto caliente. El resto se genera de forma aleatoria en todo el espacio.
   - El comando `put-record` escribe los registros JSON en la secuencia.

> **Important**
>
> No cargue este archivo en un servidor web, ya que contiene sus credenciales de AWS .

```
import json
from pprint import pprint
import random
import time
import boto3

STREAM_NAME = "ExampleInputStream"


def get_hotspot(field, spot_size):
    hotspot = {
        "left": field["left"] + random.random() * (field["width"] - spot_size),
        "width": spot_size,
        "top": field["top"] + random.random() * (field["height"] - spot_size),
        "height": spot_size,
    }
    return hotspot


def get_record(field, hotspot, hotspot_weight):
    rectangle = hotspot if random.random() < hotspot_weight else field
    point = {
        "x": rectangle["left"] + random.random() * rectangle["width"],
        "y": rectangle["top"] + random.random() * rectangle["height"],
        "is_hot": "Y" if rectangle is hotspot else "N",
    }
    return {"Data": json.dumps(point), "PartitionKey": "partition_key"}


def generate(
    stream_name, field, hotspot_size, hotspot_weight, batch_size, kinesis_client
):
    """
    Generates points used as input to a hotspot detection algorithm.
    With probability hotspot_weight (20%), a point is drawn from the hotspot;
```

```
    otherwise, it is drawn from the base field. The location of the hotspot
    changes for every 1000 points generated.
    """
    points_generated = 0
    hotspot = None
    while True:
        if points_generated % 1000 == 0:
            hotspot = get_hotspot(field, hotspot_size)
        records = [
            get_record(field, hotspot, hotspot_weight) for _ in range(batch_size)
        ]
        points_generated += len(records)
        pprint(records)
        kinesis_client.put_records(StreamName=stream_name, Records=records)

        time.sleep(0.1)


if __name__ == "__main__":
    generate(
        stream_name=STREAM_NAME,
        field={"left": 0, "width": 10, "top": 0, "height": 10},
        hotspot_size=1,
        hotspot_weight=0.2,
        batch_size=10,
        kinesis_client=boto3.client("kinesis"),
    )
```

Paso siguiente

Paso 2: creación de una aplicación de Kinesis Data Analytics

Paso 2: creación de una aplicación de Kinesis Data Analytics

En esta sección del Ejemplo de puntos de acceso, crea una aplicación de análisis de datos de Kinesis Data Analytics tal como se indica a continuación:

- Configure la entrada de la aplicación para que utilice el flujo de datos de Kinesis que ha creado como origen de streaming en el Paso 1.
- Utilice el código de la aplicación proporcionado en la Consola de administración de AWS.

Cómo crear una aplicación de

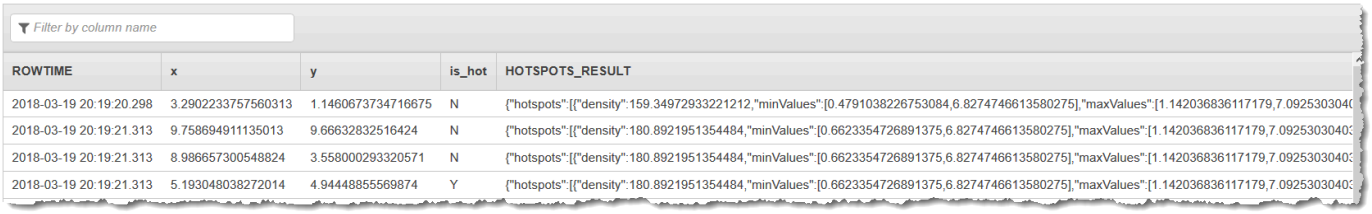
1. Cree una aplicación de análisis de datos de Kinesis Data Analytics; para ello, siga los pasos 1, 2 y 3 del ejercicio de Introducción (consulte Paso 3.1: Cree una aplicación).

   En la configuración fuente, haga lo siguiente:

   - Especifique el origen de streaming que creó en the section called “Paso 1: Crear secuencias”.
   - Después de que la consola deduzca el esquema, edítelo. Asegúrese de que los tipos de columna `x` e `y` estén establecidos en `DOUBLE` y que el tipo de columna `IS_HOT` esté establecido en `VARCHAR`.
2. Utilice el siguiente código de la aplicación (puede pegar este código en el editor de SQL):

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
    "x" DOUBLE,
    "y" DOUBLE,
    "is_hot" VARCHAR(4),
    HOTSPOTS_RESULT VARCHAR(10000)
);
CREATE OR REPLACE PUMP "STREAM_PUMP" AS
    INSERT INTO "DESTINATION_SQL_STREAM"
    SELECT "x", "y", "is_hot", "HOTSPOTS_RESULT"
    FROM TABLE (
        HOTSPOTS(
            CURSOR(SELECT STREAM "x", "y", "is_hot" FROM "SOURCE_SQL_STREAM_001"),
            1000,
            0.2,
            17)
    );
```

3. Ejecute el código SQL y revise los resultados.



Filter by column name

ROWTIME	x	y	is_hot	HOTSPOTS_RESULT
2018-03-19 20:19:20.298	3.2902233757560313	1.1460673734716675	N	{"hotspots":[{"density":159.34972933221212,"minValues":[0.4791038226753084,6.8274746613580275],"maxValues":[1.142036836117179,7.0925303040
2018-03-19 20:19:21.313	9.758694911135013	9.66632832516424	N	{"hotspots":[{"density":180.8921951354484,"minValues":[0.6623354726891375,6.8274746613580275],"maxValues":[1.142036836117179,7.0925303040
2018-03-19 20:19:21.313	8.986657300548824	3.558000293320571	N	{"hotspots":[{"density":180.8921951354484,"minValues":[0.6623354726891375,6.8274746613580275],"maxValues":[1.142036836117179,7.0925303040
2018-03-19 20:19:21.313	5.193048038272014	4.94448855569874	Y	{"hotspots":[{"density":180.8921951354484,"minValues":[0.6623354726891375,6.8274746613580275],"maxValues":[1.142036836117179,7.0925303040

Paso siguiente

Paso 3: Configurar la salida de la aplicación

Paso 3: Configurar la salida de la aplicación

Llegados a este punto del Ejemplo de puntos de acceso, el código de la aplicación de Amazon Kinesis Data Analytics ha descubierto ya numerosos puntos de acceso de un origen de streaming y les ha asignado una puntuación a cada uno.

Ahora puede enviar el resultado de la aplicación desde la secuencia en la aplicación hasta un destino externo, que es otro flujo de datos de Kinesis (`ExampleOutputStream`). A continuación, puede analizar las puntuaciones de los puntos calientes y determinar cuál podría ser un umbral adecuado de calor de los puntos calientes. Puede ampliar esta aplicación más para generar alertas.

Para configurar la salida de la aplicación

1. Abra la consola de Kinesis Data Analytics https://console.aws.amazon.comen /kinesisanalytics.
2. En el editor de SQL, puede elegir tanto Destination como Add a destination en el panel de la aplicación.
3. En la página Add a destination (Añadir un destino), seleccione Select from your streams (Seleccionar de sus secuencias). A continuación, elija la secuencia `ExampleOutputStream` que ha creado en la sección anterior.

   Ahora tiene un destino externo, donde Amazon Kinesis Data Analytics persiste en cualquier registro que escriba su aplicación en la secuencia en la aplicación `DESTINATION_SQL_STREAM`.
4. Si lo desea, puede configurarla para AWS Lambda que supervise la transmisión y le envíe alertas`ExampleOutputStream`. Para obtener más información, consulte Uso de una función de Lambda como salida. También puede revisar los registros que escribe Kinesis Data Analytics en el destino externo, que es la secuencia `ExampleOutputStream` de Kinesis, tal como se describe en Paso 4: Verificar la salida de la aplicación.

Paso siguiente

Paso 4: Verificar la salida de la aplicación

Paso 4: Verificar la salida de la aplicación

En esta sección del ejemplo de puntos calientes, debe configurar una aplicación web que muestre la información de puntos calientes en un control de gráficos vectoriales escalables (SVG).

1. Cree un archivo denominado `index.html` con el siguiente contenido:

```
<!doctype html>
<html lang=en>
<head>
    <meta charset=utf-8>
    <title>hotspots viewer</title>

    <style>
    #visualization {
      display: block;
      margin: auto;
    }

    .point {
      opacity: 0.2;
    }

    .hot {
      fill: red;
    }

    .cold {
      fill: blue;
    }

    .hotspot {
      stroke: black;
      stroke-opacity: 0.8;
      stroke-width: 1;
      fill: none;
    }
    </style>
    <script src="https://sdk.amazonaws.com/js/aws-sdk-2.202.0.min.js"></script>
    <script src="https://d3js.org/d3.v4.min.js"></script>
</head>
<body>
<svg id="visualization" width="600" height="600"></svg>
<script src="hotspots_viewer.js"></script>
</body>
</html>
```

2. Cree un archivo en el mismo directorio denominado `hotspots_viewer.js` con el siguiente contenido. Proporcione su región, las credenciales y el nombre de la secuencia de salida en las variables suministradas.

```
// Visualize example output from the Kinesis Analytics hotspot detection algorithm.
// This script assumes that the output stream has a single shard.

// Modify this section to reflect your AWS configuration
var awsRegion = "",        // The  where your Kinesis Analytics application is
 configured.
    accessKeyId = "",      // Your Access Key ID
    secretAccessKey = "",  // Your Secret Access Key
    outputStream = "";     // The name of the Kinesis Stream where the output from
 the HOTSPOTS function is being written

// The variables in this section should reflect way input data was generated and
 the parameters that the HOTSPOTS
// function was called with.
var windowSize = 1000, // The window size used for hotspot detection
    minimumDensity = 40,  // A filter applied to returned hotspots before
 visualization
    xRange = [0, 10],  // The range of values to display on the x-axis
    yRange = [0, 10];  // The range of values to display on the y-axis

////////////////////////////////////////////////////////////////////////////////
// D3 setup
////////////////////////////////////////////////////////////////////////////////

var svg = d3.select("svg"),
    margin = {"top": 20, "right": 20, "bottom": 20, "left": 20},
    graphWidth = +svg.attr("width") - margin.left - margin.right,
    graphHeight = +svg.attr("height") - margin.top - margin.bottom;

// Return the linear function that maps the segment [a, b] to the segment [c, d].
function linearScale(a, b, c, d) {
    var m = (d - c) / (b - a);
    return function(x) {
        return c + m * (x - a);
    };
}

// helper functions to extract the x-value from a stream record and scale it for
 output
```

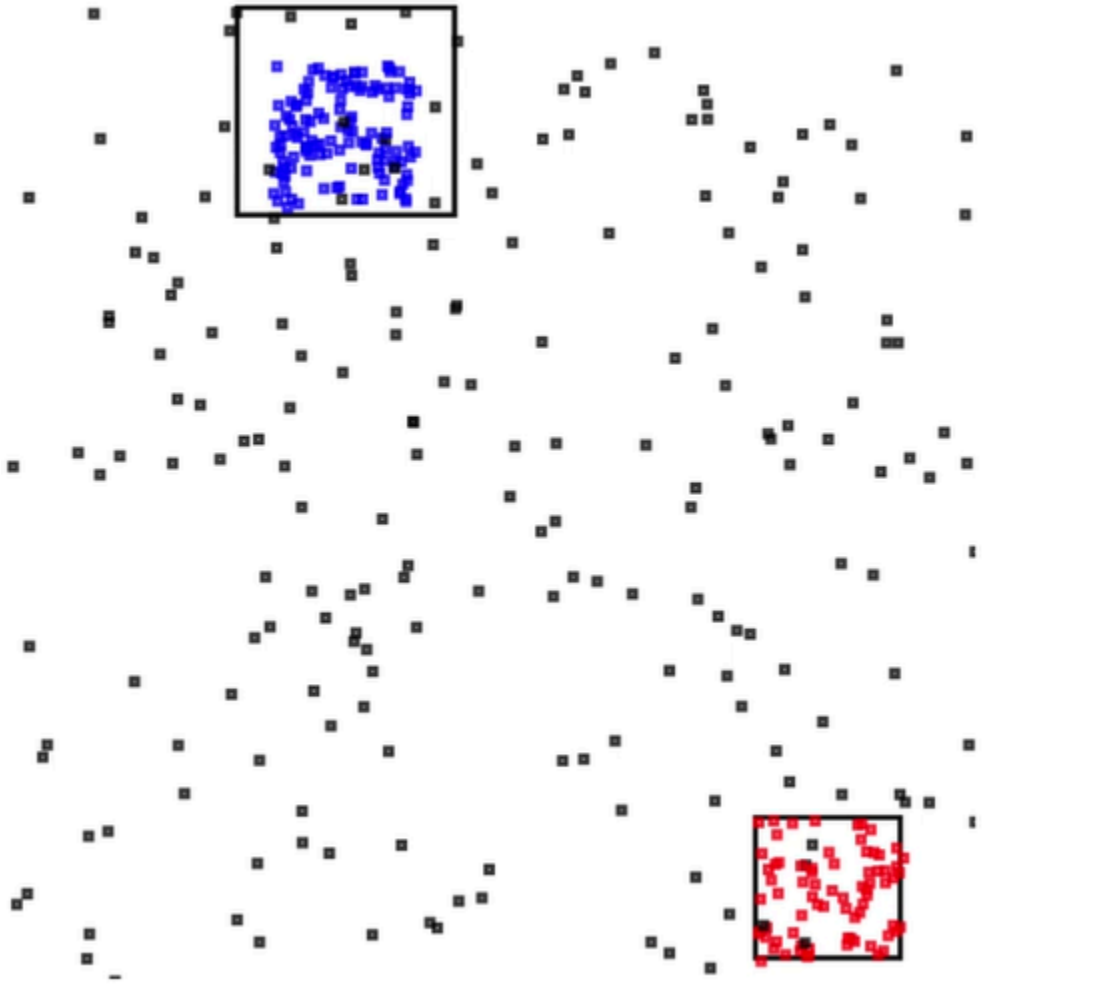
```
var xValue = function(r) { return r.x; },
    xScale = linearScale(xRange[0], xRange[1], 0, graphWidth),
    xMap = function(r) { return xScale(xValue(r)); };

// helper functions to extract the y-value from a stream record and scale it for
 output
var yValue = function(r) { return r.y; },
    yScale = linearScale(yRange[0], yRange[1], 0, graphHeight),
    yMap = function(r) { return yScale(yValue(r)); };

// a helper function that assigns a CSS class to a point based on whether it was
 generated as part of a hotspot
var classMap = function(r) { return r.is_hot == "Y" ? "point hot" : "point
 cold"; };

var g = svg.append("g")
    .attr("transform", "translate(" + margin.left + "," + margin.top + ")");

function update(records, hotspots) {

    var points = g.selectAll("circle")
        .data(records, function(r) { return r.dataIndex; });

    points.enter().append("circle")
        .attr("class", classMap)
        .attr("r", 3)
        .attr("cx", xMap)
        .attr("cy", yMap);

    points.exit().remove();

    if (hotspots) {
        var boxes = g.selectAll("rect").data(hotspots);

        boxes.enter().append("rect")
            .merge(boxes)
            .attr("class", "hotspot")
            .attr("x", function(h) { return xScale(h.minValues[0]); })
            .attr("y", function(h) { return yScale(h.minValues[1]); })
            .attr("width", function(h) { return xScale(h.maxValues[0]) -
 xScale(h.minValues[0]); })
            .attr("height", function(h) { return yScale(h.maxValues[1]) -
 yScale(h.minValues[1]); });
```

```
        boxes.exit().remove();
    }
}

////////////////////////////////////////////////////////////////////////////////
// Use the AWS SDK to pull output records from Kinesis and update the visualization
////////////////////////////////////////////////////////////////////////////////

var kinesis = new AWS.Kinesis({
    "region": awsRegion,
    "accessKeyId": accessKeyId,
    "secretAccessKey": secretAccessKey
});

var textDecoder = new TextDecoder("utf-8");

// Decode an output record into an object and assign it an index value
function decodeRecord(record, recordIndex) {
    var record = JSON.parse(textDecoder.decode(record.Data));
    var hotspots_result = JSON.parse(record.HOTSPOTS_RESULT);
    record.hotspots = hotspots_result.hotspots
        .filter(function(hotspot) { return hotspot.density >= minimumDensity});
    record.index = recordIndex
    return record;
}

// Fetch a new records from the shard iterator, append them to records, and update
 the visualization
function getRecordsAndUpdateVisualization(shardIterator, records, lastRecordIndex)
 {
    kinesis.getRecords({
        "ShardIterator": shardIterator
    }, function(err, data) {
        if (err) {
            console.log(err, err.stack);
            return;
        }

        var newRecords = data.Records.map(function(raw) { return decodeRecord(raw,
 ++lastRecordIndex); });
        newRecords.forEach(function(record) { records.push(record); });

        var hotspots = null;
        if (newRecords.length > 0) {
```

```
            hotspots = newRecords[newRecords.length - 1].hotspots;
        }

        while (records.length > windowSize) {
            records.shift();
        }

        update(records, hotspots);

        getRecordsAndUpdateVisualization(data.NextShardIterator, records,
 lastRecordIndex);
    });
}

// Get a shard iterator for the output stream and begin updating the visualization.
 Note that this script will only
// read records from the first shard in the stream.
function init() {
    kinesis.describeStream({
        "StreamName": outputStream
    }, function(err, data) {
        if (err) {
            console.log(err, err.stack);
            return;
        }

        var shardId = data.StreamDescription.Shards[0].ShardId;

        kinesis.getShardIterator({
            "StreamName": outputStream,
            "ShardId": shardId,
            "ShardIteratorType": "LATEST"
        }, function(err, data) {
            if (err) {
                console.log(err, err.stack);
                return;
            }
            getRecordsAndUpdateVisualization(data.ShardIterator, [], 0);
        })
    });
}

// Start the visualization
```

```
init();
```

3. Cuando el código de Python de la primera sección se esté ejecutando, abra `index.html` en un navegador web. La información de punto caliente aparece en la página, como se muestra a continuación.



Ejemplos: Alertas y errores

En esta sección, se proporcionan ejemplos de aplicaciones de análisis de datos de Kinesis Data Analytics que utilizan alertas y errores. Cada ejemplo proporciona step-by-step instrucciones y códigos para ayudarle a configurar y probar la aplicación Kinesis Data Analytics.

Temas

Ejemplo: Creación de alertas simples

En esta aplicación de Kinesis Data Analytics, la consulta se ejecuta de forma continua en la secuencia en la aplicación creada sobre la secuencia de demostración. Para obtener más información, consulte Consultas continuas.

Si hay filas cuya cotización cambia más del 1 por ciento, dichas filas se insertan en otra secuencia en la aplicación. En el ejercicio, puede configurar la salida de la aplicación para conservar los resultados en un destino externo. A continuación, puede continuar investigando los resultados. Por ejemplo, puede usar una AWS Lambda función para procesar registros y enviarle alertas.

Para crear una aplicación con alertas simples

1. Cree una aplicación de Kinesis Data Analytics, tal y como se describe en el ejercicio de Introducción de Kinesis Data Analytics.
2. En el editor de SQL de Kinesis Data Analytics, sustituya el código de la aplicación por el siguiente:

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM"
           (ticker_symbol VARCHAR(4),
            sector        VARCHAR(12),
            change        DOUBLE,
            price         DOUBLE);

CREATE OR REPLACE PUMP "STREAM_PUMP" AS
   INSERT INTO "DESTINATION_SQL_STREAM"
      SELECT STREAM ticker_symbol, sector, change, price
      FROM   "SOURCE_SQL_STREAM_001"
      WHERE  (ABS(Change / (Price - Change)) * 100) > 1;
```

   La instrucción SELECT del código de la aplicación filtra las filas de SOURCE_SQL_STREAM_001 para seleccionar aquellas en las que los cambios en las cotizaciones de los valores sean superiores al 1 por ciento. A continuación, inserta esas filas en otra secuencia en la aplicación

`DESTINATION_SQL_STREAM` mediante una bomba. Para obtener más información sobre el patrón de codificación que explica la utilización de bombas para insertar filas en las secuencias en la aplicación, consulte Código de la aplicación.

3. Elija Save and run SQL.
4. Añada un destino. Para ello, elija la pestaña Destination (Destino) en el editor de SQL o Add a destination (Añadir un destino) en la página de detalles de la aplicación.

   a. En el editor de SQL, elija la pestaña Destination (Destino) y, a continuación, elija Connect to a destination (Conectarse a un destino).

      En la página Connect to destination (Conectarse a un destino), elija Create New (Crear uno nuevo).

   b. Elija Go to Kinesis Streams.

   c. En la consola de Amazon Kinesis Data Streams, cree una nueva secuencia de Kinesis (por ejemplo, `gs-destination`) con una partición. Espere hasta que el estado de secuencia sea ACTIVE

   d. Vuelva a la consola de Kinesis Data Analytics. En la página Connect to destination (Conectarse a un destino), elija la secuencia que ha creado.

      Si la secuencia no aparece, actualice la página.

   e. Elija Guardar y continuar.

   Ahora tiene un destino externo, un flujo de datos de Kinesis donde Kinesis Data Analytics conserva la salida de la aplicación en la secuencia en la aplicación `DESTINATION_SQL_STREAM`.
5. Configure AWS Lambda para supervisar la transmisión de Kinesis que ha creado e invocar una función Lambda.

   Para obtener instrucciones, consulte Procesamiento previo de registros con una función de Lambda.

Ejemplo: Creación de alertas limitadas

En esta aplicación de Kinesis Data Analytics, la consulta se ejecuta de forma continua en la secuencia en la aplicación creada sobre la secuencia de demostración. Para obtener más información, consulte Consultas continuas. Si hay filas cuya cotización cambia más del 1 por ciento,

dichas filas se insertan en otra secuencia en la aplicación. La aplicación limita las alertas de forma que se envíe una alerta inmediatamente cuando cambia la cotización. Sin embargo, no se envía a la secuencia en la aplicación más de una alerta por minuto por cada símbolo de cotización.

Para crear una aplicación con alertas limitadas

1. Cree una aplicación de Kinesis Data Analytics, tal y como se describe en el ejercicio de Introducción de Kinesis Data Analytics.
2. En el editor de SQL de Kinesis Data Analytics, sustituya el código de la aplicación por el siguiente:

```
CREATE OR REPLACE STREAM "CHANGE_STREAM"
           (ticker_symbol VARCHAR(4),
            sector        VARCHAR(12),
            change        DOUBLE,
            price         DOUBLE);

CREATE OR REPLACE PUMP "change_pump" AS
   INSERT INTO "CHANGE_STREAM"
      SELECT STREAM ticker_symbol, sector, change, price
      FROM   "SOURCE_SQL_STREAM_001"
      WHERE  (ABS(Change / (Price - Change)) * 100) > 1;

-- ** Trigger Count and Limit **
-- Counts "triggers" or those values that evaluated true against the previous where
 clause
-- Then provides its own limit on the number of triggers per hour per ticker symbol
 to what
-- is specified in the WHERE clause

CREATE OR REPLACE STREAM TRIGGER_COUNT_STREAM (
   ticker_symbol VARCHAR(4),
   change REAL,
   trigger_count INTEGER);

CREATE OR REPLACE PUMP trigger_count_pump AS INSERT INTO TRIGGER_COUNT_STREAM
SELECT STREAM ticker_symbol, change, trigger_count
FROM (
    SELECT STREAM ticker_symbol, change, COUNT(*) OVER W1 as trigger_count
    FROM "CHANGE_STREAM"
    --window to perform aggregations over last minute to keep track of triggers
    WINDOW W1 AS (PARTITION BY ticker_symbol RANGE INTERVAL '1' MINUTE PRECEDING)
```

```
)
WHERE trigger_count >= 1;
```

La instrucción SELECT del código de la aplicación filtra las filas de SOURCE_SQL_STREAM_001 para ver cambios en las cotizaciones de los valores superiores al 1 por ciento e introduce las filas en otra secuencia en la aplicación CHANGE_STREAM mediante una bomba.

A continuación, la aplicación crea una segunda secuencia denominada TRIGGER_COUNT_STREAM para las alertas limitadas. Una segunda consulta selecciona registros de una ventana que se desplaza hacia delante cada vez que se admite un registro en ella, de forma que solo se escriba en la secuencia un registro por cada símbolo de cotización cada minuto.

3. Elija Save and run SQL.

El ejemplo envía a TRIGGER_COUNT_STREAM una secuencia similar a la siguiente:

Filter by column name

ROWTIME	TICKER_SYMBOL	CHANGE	TRIGGER_COUNT
2018-01-08 22:59:15.742	ASD	-1.77	1
2018-01-08 22:59:15.742	ASD	-1.77	1
2018-01-08 22:59:20.752	DFT	-3.16	1
2018-01-08 22:59:35.775	IOP	-1.88	1

Ejemplo: Exploración de la secuencia de errores en la aplicación

Amazon Kinesis Data Analytics proporciona una secuencia de errores en la aplicación para cada aplicación que crea. Cualquier fila que la aplicación no pueda procesar se envía a esta secuencia de errores. Podría plantearse conservar los datos de la secuencia de errores a un destino externo para poder investigarlos.

Realizará los siguientes ejercicios en la consola. En estos ejemplos, introducirá errores en la configuración de entrada editando el esquema inferido por el proceso de detección y, a continuación, verificará las filas enviadas a la secuencia de errores.

Temas

Introducción de un error de análisis

En este ejercicio, usted introducirá un error de análisis.

1. Cree una aplicación de Kinesis Data Analytics, tal y como se describe en el ejercicio de introducción de Kinesis Data Analytics.
2. En la página de detalles de la aplicación, elija Conectar datos de streaming.
3. Si ha seguido el ejercicio de introducción, tendrá una secuencia de demostración (`kinesis-analytics-demo-stream`) en la cuenta. En la página Connect to source (Conectarse al origen), elija esta secuencia de demostración.
4. Kinesis Data Analytics toma una muestra de la secuencia de demostración con el fin de inferir un esquema para la secuencia de entrada en la aplicación que crea. La consola muestra el esquema inferido y los datos de muestra en la pestaña Formatted stream sample.
5. A continuación, edite el esquema y modifique el tipo de columna para introducir el error de análisis. Elija Edit schema (Editar esquema).
6. Cambie el `TICKER_SYMBOL` tipo de columna de `VARCHAR(4)` a `INTEGER`.

   Ahora que no es válido el tipo de columna del esquema en la aplicación que se ha creado, Kinesis Data Analytics no puede introducir datos en la secuencia en la aplicación. En su lugar, envía las filas a la secuencia de errores.
7. Elija Save schema.
8. Elija Refresh schema samples.

   Observe que no haya filas en el ejemplo de Formatted stream. Sin embargo, la pestaña Error stream muestra los datos que contengan un mensaje de error. La pestaña Error stream muestra los datos enviados a la secuencia de errores en la aplicación.

   Dado que se ha cambiado el tipo de datos de las columnas, Kinesis Data Analytics no ha podido llevar los datos a la secuencia de entrada en la aplicación. En su lugar, envía los datos a la secuencia de errores.

Introducción de un error de división por cero

En este ejercicio, va a actualizar el código de la aplicación para introducir un error de tiempo de ejecución (una división por cero). Observe que Amazon Kinesis Data Analytics envía las filas resultantes a la secuencia de errores en la aplicación, no a la secuencia en la aplicación `DESTINATION_SQL_STREAM`, en la que se deben escribir los resultados.

1. Cree una aplicación de Kinesis Data Analytics, tal y como se describe en el ejercicio de introducción de Kinesis Data Analytics.

   Verifique los resultados en la pestaña Real-time analytics de la siguiente manera:

   Sour

2. Actualice la instrucción `SELECT` en el código de la aplicación para introducir una división por cero; por ejemplo:

   ```
   SELECT STREAM ticker_symbol, sector, change, (price / 0) as ProblemColumn
   FROM "SOURCE_SQL_STREAM_001"
   WHERE sector SIMILAR TO '%TECH%';
   ```

3. Ejecute la aplicación .

   Debido a que se produce el error de división por cero en tiempo de ejecución, en lugar de escribir los resultados en `DESTINATION_SQL_STREAM`, Kinesis Data Analytics envía las filas a la secuencia de errores en la aplicación. En la pestaña Real-time analytics (Análisis en tiempo real), elija la secuencia de errores y podrá ver las filas en la secuencia de errores en la aplicación.

Ejemplos: Aceleradores de soluciones

El sitio de AWS soluciones tiene AWS CloudFormation plantillas disponibles que puede utilizar para crear rápidamente soluciones completas de transmisión de datos.

Están disponibles las siguientes plantillas:

Información sobre la Cuenta de AWS actividad en tiempo real

Esta solución registra y visualiza las métricas de acceso y uso de los recursos para sus Cuenta de AWS usuarios en tiempo real. Para obtener más información, consulte Información sobre la Cuenta de AWS actividad en tiempo real.

Supervisión de AWS IoT dispositivos en tiempo real con Kinesis Data Analytics

Esta solución recopila, procesa, analiza y visualiza la conectividad y los datos de actividad de los dispositivos IoT en tiempo real. Para obtener más información, consulte Supervisión de AWS IoT dispositivos en tiempo real con Kinesis Data Analytics.

Análisis web con Kinesis Data Analytics en tiempo real

Esta solución recopila, procesa, analiza y visualiza secuencias de clics en sitios web en tiempo real. Para obtener más información, consulte Análisis web en tiempo real con Kinesis Data Analytics.

Solución Amazon Connected Vehicle

Esta solución recopila, procesa, analiza y visualiza datos de IoT de vehículos en tiempo real. Para obtener más información, consulte Solución Amazon Connected Vehicle.

Seguridad en Amazon Kinesis Data Analytics

La seguridad en la nube AWS es la máxima prioridad. Como AWS cliente, se beneficiará 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 acerca de los programas de conformidad que se aplican a Kinesis Data Analytics, consulte Servicios de AWS en el ámbito del programa de conformidad.
- 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 ayuda a comprender cómo aplicar el modelo de responsabilidad compartida cuando se utiliza Kinesis Data Analytics. En los siguientes temas, se le mostrará cómo configurar Kinesis Data Analytics para satisfacer sus objetivos de seguridad y conformidad. También aprenderá a utilizar otros servicios de Amazon que le ayudarán a supervisar y proteger sus recursos de Kinesis Data Analytics.

Temas

Protección de datos en aplicaciones de Amazon Kinesis Data Analytics para SQL

Puede proteger sus datos mediante las herramientas que proporciona AWS. Kinesis Data Analytics puede funcionar con servicios que admiten el cifrado de datos, como Kinesis Data Streams, Firehose y Amazon S3.

Cifrado de datos en Kinesis Data Analytics

Cifrado en reposo

Tenga en cuenta lo siguiente sobre el cifrado de datos en reposo con Kinesis Data Analytics::

- Puede cifrar los datos de la transmisión de datos entrante de Kinesis mediante. StartStreamEncryption Para obtener más información, consulte ¿Qué es el cifrado del lado del servidor para Kinesis Data Streams?
- Los datos de salida se pueden cifrar en reposo con Firehose, que permite almacenar los datos en un bucket de Amazon S3 cifrado. Se puede especificar la clave de cifrado que el bucket de Amazon S3 va a utilizar. Para obtener más información, consulte Protección de los datos mediante el cifrado del servidor con claves administradas por KMS (SSE-KMS).
- El código de la aplicación se cifra en reposo.
- Los datos de referencia de la aplicación se cifran en reposo.

Cifrado en tránsito

Kinesis Data Analytics cifra todos los datos en tránsito. El cifrado en tránsito está habilitado para todas las aplicaciones de Kinesis Data Analytics y no se puede deshabilitar.

Kinesis Data Analytics cifra los datos en tránsito en los siguientes casos:

- Datos en tránsito de Kinesis Data Streams para Kinesis Data Analytics.
- Datos en tránsito entre los componentes internos de Kinesis Data Analytics.
- Datos en tránsito entre Kinesis Data Analytics y Firehose.

Administración de claves

El cifrado de datos de Kinesis Data Analytics utiliza claves administradas por los servicios. No se admiten las claves administradas por el cliente.

Gestión de identidad y acceso en Kinesis Data Analytics

Amazon Kinesis Data Analytics necesita permisos para leer registros de un origen de streaming que especifique en la configuración de entrada de su aplicación. Amazon Kinesis Data Analytics también necesita permisos para escribir la salida de su aplicación en las secuencias que especifique en la configuración de salida de la aplicación.

Para conceder estos permisos puede crear un rol de IAM que Amazon Kinesis Data Analytics pueda admitir. Los permisos que conceda a un rol determinarán lo que Amazon Kinesis Data Analytics podrá hacer cuando el servicio asuma dicho rol.

> **Note**
>
> La información de esta sección resulta útil si desea crear un rol de IAM usted mismo. Al crear una aplicación en la consola de Amazon Kinesis Data Analytics, esta puede crear un rol de IAM en ese momento. La consola utiliza la siguiente convención de nomenclatura para los roles de IAM que crea:
>
> ```
> kinesis-analytics-ApplicationName
> ```
>
> Una vez creado el rol, puede revisar el rol y las políticas asociadas en la consola de IAM.

Cada rol de IAM tiene dos políticas asociadas. En la política de confianza, debe especificar quién puede asumir el rol. En la política de permisos (puede haber una o más), debe especificar los permisos que desea conceder a este rol. En las siguientes secciones se describen estas políticas, que puede utilizar al crear un rol de IAM.

Política de confianza

Para conceder permisos a Amazon Kinesis Data Analytics para asumir un rol de acceso a un origen de streaming o de referencia, asocie la siguiente política de confianza a un rol de IAM:

JSON

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

Política de permisos

Si está creando un rol de IAM para permitir que Amazon Kinesis Data Analytics lea desde un origen de streaming de una aplicación, debe conceder permisos para las acciones de lectura correspondientes. Según el origen (por ejemplo, un flujo de Kinesis, un flujo de entrega de Firehose o un origen de referencia en un bucket de Amazon S3), puede asociar la siguiente política de permisos.

Política de permisos para leer una secuencia de Kinesis

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ReadInputKinesis",
            "Effect": "Allow",
            "Action": [
                "kinesis:DescribeStream",
                "kinesis:GetShardIterator",
                "kinesis:GetRecords",
                "kinesis:ListShards"
```

```
            ],
            "Resource": [
                "arn:aws:kinesis:us-east-1:123456789012:stream/inputStreamName"
            ]
        }
    ]
}
```

Política de permisos para leer un flujo de entrega de Firehose

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ReadInputFirehose",
            "Effect": "Allow",
            "Action": [
                "firehose:DescribeDeliveryStream",
                "firehose:Get*"
            ],
            "Resource": [
                "arn:aws:firehose:us-
east-1:123456789012:deliverystream/inputFirehoseName"
            ]
        }
    ]
}
```

> **Note**
>
> El permiso `firehose:Get*` hace referencia a un acceso interno que Kinesis Data Analytics utiliza para acceder al flujo. No hay acceso público de un flujo de entrega de Firehose.

Si indica que Amazon Kinesis Data Analytics escriba directamente en destinos externos en la configuración de salida de su aplicación, debe conceder el siguiente permiso al rol de IAM.

Política de permisos para escribir en una secuencia de Kinesis

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "WriteOutputKinesis",
            "Effect": "Allow",
            "Action": [
                "kinesis:DescribeStream",
                "kinesis:PutRecord",
                "kinesis:PutRecords"
            ],
            "Resource": [
                "arn:aws:kinesis:us-east-1:123456789012:stream/output-stream-
name"
            ]
        }
    ]
}
```

Política de permisos para escribir en una secuencia de entrega de Firehose

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "WriteOutputFirehose",
            "Effect": "Allow",
            "Action": [
                "firehose:DescribeDeliveryStream",
                "firehose:PutRecord",
                "firehose:PutRecordBatch"
            ],
            "Resource": [
```

```
                "arn:aws:firehose:us-east-1:123456789012:deliverystream/output-
firehose-name"
            ]
        }
    ]
}
```

Política de permisos para leer un origen de datos de referencia de un bucket de Amazon S3

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:Get*",
        "s3:List*"
      ],
      "Resource": "*"
    }
  ]
}
```

Prevención de la sustitución confusa entre servicios

En AWS, la suplantación entre servicios puede producirse cuando un servicio (el servicio de llamadas) llama a otro servicio (el servicio llamado). El servicio que lleva a cabo las llamadas se puede manipular para actuar en función de los recursos de otro cliente a pesar de que no debe tener los permisos adecuados, lo que da como resultado un problema de suplente confuso.

Para evitar que los agentes confusos, AWS proporciona herramientas que lo ayudan a proteger sus datos en todos los servicios utilizando los directores de servicio a los que se les ha dado acceso a los recursos de su cuenta. Esta sección se centra en la prevención de problemas de suplentes confusos entre servicios específica de Kinesis Data Analytics; sin embargo, puede obtener más información sobre este tema en la sección El problema del suplente confuso de la Guía del usuario de IAM.

En el contexto de Kinesis Data Analytics for SQL, le recomendamos que utilice las claves de contexto aws SourceArn: y aws SourceAccount: global condition en su política de confianza de roles para limitar el acceso al rol únicamente a las solicitudes generadas por los recursos esperados.

Utiliza `aws:SourceArn` si desea que solo se asocie un recurso al acceso entre servicios. Utiliza `aws:SourceAccount` si quiere permitir que cualquier recurso de esa cuenta se asocie al uso entre servicios.

El valor de `aws:SourceArn` debe ser el ARN del recurso utilizado por Kinesis Data Analytics, que se especifica con el siguiente formato: `arn:aws:kinesisanalytics:region:account:resource`.

La forma más eficaz de protegerse contra el problema del suplente confuso es utilizar la clave de contexto de condición global de `aws:SourceArn` con el ARN completo del recurso.

Si no conoce el ARN completo del recurso o si está especificando varios recursos, utilice la clave `aws:SourceArn` con caracteres comodines (*) para las partes desconocidas del ARN. Por ejemplo: `arn:aws:kinesisanalytics::111122223333:*`.

Si bien la mayoría de las acciones de la API de Kinesis Data Analytics for SQL AddApplicationInput, DeleteApplicationpor ejemplo CreateApplication, se realizan en el contexto de aplicaciones específicas, DiscoverInputSchemala acción no se ejecuta en el contexto de ninguna aplicación. Esto significa que la función utilizada en esta acción no debe especificar completamente un recurso en la clave de condición `SourceArn`. A continuación, se muestra un ejemplo en el que se utiliza un ARN comodín:

```
{
   ...
   "ArnLike":{
      "aws:SourceArn":"arn:aws:kinesisanalytics:us-east-1:123456789012:*"
   }
   ...
}
```

El rol predeterminado generado por Kinesis Data Analytics para SQL usa este comodín. Esto garantiza que la detección del esquema de entrada funcione sin problemas en la experiencia de la consola. Sin embargo, recomendamos editar la política de confianza para utilizar un ARN completo después de descubrir el esquema e implementar una completa mitigación de suplentes confusos.

Las políticas de funciones que proporcione a Kinesis Data Analytics, así como las políticas de confianza de las funciones generadas para usted, pueden utilizar las claves de condición aws SourceArn: y aws SourceAccount:.

Para protegerse contra el problema de suplente confuso, lleve a cabo los siguientes pasos:

Cómo protegerse contra el problema del suplente confuso

1. Inicie sesión en la consola AWS de administración y abra la consola de IAM en. https://console.aws.amazon.com/iam/
2. Elija Roles y, a continuación, seleccione el rol que desee modificar.
3. Elija Editar la política de confianza.
4. En la página Editar política de confianza, sustituya la política JSON predeterminada por una política que utilice una o ambas claves contextuales `aws:SourceArn` y `aws:SourceAccount` de condición global. Consulte el siguiente ejemplo de política:
5. Elija Actualizar política.

   JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "Service":"kinesisanalytics.amazonaws.com"
         },
         "Action":"sts:AssumeRole",
         "Condition":{
            "StringEquals":{
               "aws:SourceAccount":"Account ID"
            },
            "ArnEquals":{
               "aws:SourceArn":"arn:aws:kinesisanalytics:us-
east-1:123456789012:application/my-app"
            }
         }
      }
   ]
```

```
}
```

Autenticación y control de acceso para

El acceso a requiere credenciales. Esas credenciales deben tener permisos para acceder a AWS los recursos, como una aplicación o una instancia de Amazon Elastic Compute Cloud (Amazon EC2). En las secciones siguientes se incluye información detallada sobre cómo usar AWS Identity and Access Management (IAM) y para proteger el acceso a sus recursos.

Control de acceso

Aunque disponga de credenciales válidas para autenticar las solicitudes, si no tiene permisos, no podrá crear recursos de ni obtener acceso a ellos. Por ejemplo, debe tener permiso para crear una aplicación de .

En las secciones siguientes, se describe cómo administrar los permisos de . Recomendamos que lea primero la información general.

- Información general sobre la administración de los permisos de acceso a los recursos de
- Uso de políticas basadas en identidad (políticas de IAM) para
- Permisos de la API: referencia de acciones, permisos y recursos

Autenticación con identidades

La autenticación es la forma de iniciar sesión AWS con sus credenciales de identidad. Debe autenticarse como usuario de Usuario raíz de la cuenta de AWS IAM o asumir una función de IAM.

Puede iniciar sesión como una identidad federada con las credenciales de una fuente de identidad, como AWS IAM Identity Center (IAM Identity Center), la autenticación de inicio de sesión único o las credenciales. Google/Facebook Para obtener más información sobre el inicio de sesión, consulte Cómo iniciar sesión en la Cuenta de AWS en la Guía del usuario de AWS Sign-In .

Para el acceso programático, AWS proporciona un SDK y una CLI para firmar criptográficamente las solicitudes. Para obtener más información, consulte AWS Signature Version 4 para solicitudes de API en la Guía del usuario de IAM.

Cuenta de AWS usuario root

Al crear un Cuenta de AWS, se comienza con una identidad de inicio de sesión denominada usuario Cuenta de AWS raíz que tiene acceso completo a todos Servicios de AWS los recursos. Se recomiendaencarecidamente que no utilice el usuario raíz para las tareas diarias. Para ver las tareas que requieren credenciales de usuario raíz, consulte Tareas que requieren credenciales de usuario raíz en la Guía del usuario de IAM.

Identidad federada

Como práctica recomendada, exija a los usuarios humanos que utilicen la federación con un proveedor de identidades para acceder Servicios de AWS mediante credenciales temporales.

Una identidad federada es un usuario del directorio empresarial, del proveedor de identidades web o al Directory Service que se accede Servicios de AWS mediante credenciales de una fuente de identidad. Las identidades federadas asumen roles que proporcionan credenciales temporales.

Para una administración de acceso centralizada, se recomienda AWS IAM Identity Center. Para obtener más información, consulte ¿Qué es el Centro de identidades de IAM? en la Guía del usuario de AWS IAM Identity Center .

Usuarios y grupos de IAM

Un usuario de IAM es una identidad con permisos específicos para una sola persona o aplicación. Recomendamos el uso de credenciales temporales en lugar de usuarios de IAM con credenciales de larga duración. Para obtener más información, consulte Exigir a los usuarios humanos que utilicen la federación con un proveedor de identidad para acceder AWS mediante credenciales temporales en la Guía del usuario de IAM.

Un grupo de IAM especifica un conjunto de usuarios de IAM y facilita la administración de los permisos para grupos grandes de usuarios. Para obtener más información, consulte Casos de uso para usuarios de IAM en la Guía del usuario de IAM.

Roles de IAM

Un Rol de IAM es una identidad con permisos específicos que proporciona credenciales temporales. Puede asumir un rol cambiando de un rol de usuario a uno de IAM (consola) o llamando a una AWS CLI operación de AWS API. Para obtener más información, consulte Métodos para asumir un rol en la Guía del usuario de IAM.

Los roles de IAM son útiles para el acceso de usuario federado, los permisos de usuario de IAM temporales, el acceso entre cuentas, el acceso entre servicios y las aplicaciones que se ejecutan en Amazon EC2. Para obtener más información, consulte Acceso a recursos entre cuentas en IAM en la Guía del usuario de IAM.

Información general sobre la administración de los permisos de acceso a los recursos de

> **Warning**
>
> Para proyectos nuevos, le recomendamos que utilice el nuevo servicio gestionado para Apache Flink Studio en lugar de Kinesis Data Analytics para SQL Applications. El servicio gestionado para Apache Flink Studio combina la facilidad de uso con capacidades analíticas avanzadas, lo que le permite crear aplicaciones sofisticadas de procesamiento de flujos en cuestión de minutos.

Para dar acceso, agregue permisos a los usuarios, grupos o roles:

- Usuarios y grupos en AWS IAM Identity Center:

  Cree un conjunto de permisos. Siga las instrucciones de Creación de un conjunto de permisos en la Guía del usuario de AWS IAM Identity Center .
- Usuarios gestionados en IAM a través de un proveedor de identidades:

  Cree un rol para la federación de identidades. Siga las instrucciones descritas en Creación de un rol para un proveedor de identidad de terceros (federación) en la Guía del usuario de IAM.
- Usuarios de IAM:
  - Cree un rol que el usuario pueda aceptar. Siga las instrucciones descritas en Creación de un rol para un usuario de IAM en la Guía del usuario de IAM.
  - (No recomendado) Adjunte una política directamente a un usuario o agregue un usuario a un grupo de usuarios. Siga las instrucciones descritas en Adición de permisos a un usuario (consola) de la Guía del usuario de IAM.

> **Note**
>
> Un administrador de cuentas (o usuario administrador) es un usuario que tiene privilegios de administrador. Para obtener más información, consulte Prácticas recomendadas de IAM en la Guía del usuario de IAM.

Temas

- Recursos y operaciones
- Titularidad de los recursos
- Administración del acceso a los recursos
- Especificar elementos de política: acciones, efectos y entidades principales
- Especificación de las condiciones de una política

Recursos y operaciones

En , el recurso principal es una aplicación. En las políticas se emplean nombres de recurso de Amazon (ARN) para identificar los recursos a los que se aplican las políticas.

Estos recursos tienen nombres de recursos de Amazon (ARNs) exclusivos asociados a ellos, como se muestra en la siguiente tabla.

Tipo de recurso	Formato de ARN
Aplicación	`arn:aws:kinesisanalytics:` *`region`*`:`*`account-id`*`:application/`*`application-name`*

proporciona un conjunto de operaciones para trabajar con recursos. Para ver la lista de las operaciones disponibles, consulte Acciones.

Titularidad de los recursos

Cuenta de AWS Es propietario de los recursos que se crean en la cuenta, independientemente de quién los haya creado. En concreto, el propietario del recurso es el Cuenta de AWS de la entidad principal (es decir, la cuenta raíz, un usuario o un rol de IAM) que autentica la solicitud de creación del recurso. Los siguientes ejemplos ilustran cómo funciona:

- Si utiliza las credenciales de su cuenta raíz Cuenta de AWS para crear una aplicación, Cuenta de AWS es el propietario del recurso. (En , el recurso es una aplicación).
- Si crea un usuario en su Cuenta de AWS cuenta y le concede permisos para crear una aplicación, el usuario podrá crear una aplicación. Sin embargo, usted Cuenta de AWS, al que pertenece el usuario, es el propietario del recurso de la aplicación. Le recomendamos encarecidamente que conceda permisos a los roles y no a los usuarios.
- Si crea una función de IAM Cuenta de AWS con permisos para crear una aplicación, cualquier persona que pueda asumir esa función podrá crear una aplicación. La suya Cuenta de AWS, a la que pertenece el usuario, es la propietaria del recurso de la aplicación.

Administración del acceso a los recursos

Una política de permisos describe quién tiene acceso a qué. En la siguiente sección se explican las opciones disponibles para crear políticas de permisos.

> **Note**
>
> En esta sección se explica el uso de IAM en el contexto de . No se proporciona información detallada sobre el servicio de IAM. Para ver la documentación completa de IAM, consulte ¿Qué es IAM? en la Guía del usuario de IAM. Para obtener más información acerca de la sintaxis y las descripciones de las políticas de IAM, consulte Referencia de políticas JSON de IAM en la Guía del usuario de IAM.

Las políticas que se asocian a una identidad de IAM se denominan políticas basadas en la identidad (políticas de IAM). Las políticas que se asocian a un recurso se denominan políticas basadas en recursos. Solamente admite las políticas basadas en identidades (políticas de IAM).

Temas

- Políticas basadas en identidad (políticas de IAM)
- Políticas basadas en recursos

Políticas basadas en identidad (políticas de IAM)

Puede asociar políticas a identidades de IAM. Por ejemplo, puede hacer lo siguiente:

- Asociar una política de permisos a un usuario o un grupo de su cuenta: para conceder a permisos de usuario para crear un recurso, como una aplicación, puede asociar una política de permisos a un usuario o a un grupo al que pertenezca el usuario.
- Adjuntar una política de permisos a un rol (conceder permisos para cuentas cruzadas): puede adjuntar una política de permisos basada en identidades a un rol de IAM para conceder permisos para cuentas cruzadas. Por ejemplo, el administrador de la cuenta A puede crear un rol para conceder permisos entre cuentas a otro Cuenta de AWS (por ejemplo, la cuenta B) o a un servicio de Amazon de la siguiente manera:
  1. El administrador de la Cuenta A crea un rol de IAM y adjunta una política de permisos al rol que concede permisos sobre los recursos de la Cuenta A.
  2. El administrador de la cuenta A asocia una política de confianza al rol que identifica la cuenta B como la entidad principal que puede asumir el rol.
  3. A continuación, el administrador de la cuenta B puede delegar permisos para asumir el rol a cualquier usuario de la cuenta B. De este modo, los usuarios de la cuenta B podrán crear recursos y obtener acceso a ellos en la cuenta A. La entidad principal de la política de confianza también puede ser la entidad principal de un servicio de si desea conceder permisos para asumir el rol a un servicio de Amazon.

  Para obtener más información sobre el uso de IAM para delegar permisos, consulte Administración de accesos en la Guía del usuario de IAM.

A continuación, se ofrece una política de ejemplo que concede permiso para la acción `kinesisanalytics:CreateApplication` , que es necesario para crear una aplicación.

> **Note**
>
> Esta es una política de ejemplo introductorio. Al adjuntar la política al usuario, este podrá crear una aplicación mediante el AWS SDK AWS CLI o el SDK. Sin embargo, el usuario necesitará más permisos para configurar la entrada y la salida. Además, el usuario deberá obtener más permisos al utilizar la consola. En las secciones posteriores se proporciona más información al respecto.

JSON

```
{
```

```
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "Stmt1473028104000",
            "Effect": "Allow",
            "Action": [
                "kinesisanalytics:CreateApplication"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

Para obtener más información acerca del uso de políticas basadas en identidades con , consulte Uso de políticas basadas en identidad (políticas de IAM) para. Para obtener más información sobre usuarios, grupos, roles y permisos, consulte Identidades (usuarios, grupos y roles) en la Guía del usuario de IAM.

Políticas basadas en recursos

Otros servicios, como Amazon S3, también admiten políticas de permisos basadas en recursos. Por ejemplo, puede asociar una política a un bucket de S3 para administrar los permisos de acceso a dicho bucket. no admite políticas basadas en recursos.

Especificar elementos de política: acciones, efectos y entidades principales

Para cada recurso de , el servicio define un conjunto de operaciones de API. Para conceder permisos para estas operaciones de API, define un conjunto de acciones que usted puede especificar en una política. Algunas operaciones de API pueden requerir permisos para más de una acción para poder realizar la operación de API. Para obtener más información sobre los recursos y las operaciones de API, consulte Recursos y operaciones y Acciones.

A continuación se indican los elementos más básicos de la política:

- Recurso: use un Nombre de recurso de Amazon (ARN) para identificar el recurso al que se aplica la política. Para obtener más información, consulte Recursos y operaciones.

- Acción: utilice palabras clave de acción para identificar las operaciones del recurso que desea permitir o denegar. Por ejemplo, puede utilizar `create` para permitir a los usuarios crear una aplicación.
- Efecto: especifique el efecto, permitir o denegar, cuando el usuario solicite la acción específica. Si no concede acceso de forma explícita (permitir) a un recurso, el acceso se deniega implícitamente. También puede denegar explícitamente el acceso a un recurso para asegurarse de que un usuario no pueda obtener acceso a él, aunque otra política le conceda acceso.
- Entidad principal: en las políticas basadas en identidades (políticas de IAM), el usuario al que se asocia esta política es la entidad principal implícita. Para las políticas basadas en recursos, debe especificar el usuario, la cuenta, el servicio u otra entidad que desee que reciba permisos (se aplica solo a las políticas basadas en recursos). no admite políticas basadas en recursos.

Para obtener más información sobre la sintaxis y descripciones de las políticas de IAM, consulte IAM JSON Policy Reference (Referencia de la política JSON de IAM) en la Guía del usuario de IAM.

Para ver una de tabla con todas las operaciones de la API y los recursos a los que se aplican, consulte Permisos de la API: referencia de acciones, permisos y recursos.

Especificación de las condiciones de una política

Al conceder permisos, puede utilizar el lenguaje de la política de acceso para especificar las condiciones en las que se debe aplicar una política. Por ejemplo, es posible que desee que solo se aplique una política después de una fecha específica. Para obtener más información sobre cómo especificar condiciones en un lenguaje de política, consulte Condition en la Guía del usuario de IAM.

Para expresar condiciones, se usan claves de condición predefinidas. No hay claves de condición específicas para . Sin embargo, hay claves AWS de condición generales que puede utilizar según convenga. Para obtener una lista completa de las claves AWS de ancho, consulte las claves disponibles para las condiciones en la Guía del usuario de IAM.

Uso de políticas basadas en identidad (políticas de IAM) para

Los siguientes ejemplos de políticas basadas en identidad muestran cómo un administrador de la cuenta puede asociar políticas de permisos a identidades de IAM (es decir, usuarios, grupos y roles) y, por lo tanto, conceder permisos para realizar operaciones en recursos.

> **Important**
>
> Le recomendamos que consulte primero los temas de introducción en los que se explican los conceptos básicos y las opciones disponibles para administrar el acceso a sus recursos de . Para obtener más información, consulte Información general sobre la administración de los permisos de acceso a los recursos de.

Temas

- Permisos necesarios para usar la consola de
- Políticas administradas (predefinidas) por Amazon para
- Ejemplos de políticas administradas por el cliente

A continuación se muestra un ejemplo de una política de permisos.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "Stmt1473028104000",
            "Effect": "Allow",
            "Action": [
                "kinesisanalytics:CreateApplication"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

La política tiene una declaración:

- La primera declaración concede permisos para una acción (`kinesisanalytics:CreateApplication`) en un recurso que utiliza el nombre de recurso de

Amazon (ARN) para la aplicación. El ARN en este caso especifica un comodín (*) para indicar que el permiso se concede para cualquier recurso.

Para ver una tabla con todas las operaciones de la API y los recursos a los que se aplican, consulte Permisos de la API: referencia de acciones, permisos y recursos.

Permisos necesarios para usar la consola de

En el caso de un usuario que trabaja con la consola de , debe conceder los permisos necesarios. Por ejemplo, si desea conceder que un usuario tenga permisos para crear una aplicación, conceda permisos que le mostrarán los orígenes de streaming en la cuenta para que este pueda configurar la entrada y la salida de la consola.

Le recomendamos lo siguiente:

- Utilice políticas administradas de Amazon para conceder los permisos de usuario. Para conocer las políticas disponibles, consulte Políticas administradas (predefinidas) por Amazon para.
- Cree políticas personalizadas. En este caso, le recomendamos que revise los ejemplos proporcionados en esta sección. Para obtener más información, consulte Ejemplos de políticas administradas por el cliente.

Políticas administradas (predefinidas) por Amazon para

AWS aborda muchos casos de uso comunes al proporcionar políticas de IAM independientes que son creadas y administradas por. AWS Estas políticas administradas por Amazon conceden los permisos necesarios para casos de uso comunes, lo que le evita tener que investigar los permisos necesarios. Para más información, consulte Políticas administradas de Amazon en la Guía del usuario de IAM.

Las siguientes políticas administradas por Amazon, que se pueden adjuntar a los usuarios de la cuenta, son específicas de:

- **`AmazonKinesisAnalyticsReadOnly`**— Otorga permisos para realizar acciones que permiten al usuario enumerar las aplicaciones y revisar input/output la configuración. También concede permisos para que un usuario pueda ver una lista de flujos de Kinesis y de flujos de entrega de Firehose. Dado que aplicación está en ejecución, el usuario puede ver los datos de origen y los resultados de análisis en tiempo real en la consola.

- **`AmazonKinesisAnalyticsFullAccess`**: otorga permisos para todas las acciones y todos los demás permisos que permiten a un usuario crear y administrar aplicaciones. Sin embargo, tenga en cuenta lo siguiente:

  - Estos permisos no son suficientes si el usuario quiere crear un rol de IAM en la consola (con estos permisos el usuario puede seleccionar un rol existente). Si desea que el usuario pueda crear un rol de IAM en la consola, añada la política administrada de Amazon `IAMFullAccess`.

  - Un usuario debe tener permiso para que acción `iam:PassRole` pueda especificar un rol de IAM al configurar la aplicación. Esta política administrada de Amazon concede al usuario permiso para la acción `iam:PassRole` solo en los roles de IAM que empiecen con el prefijo `service-role/kinesis-analytics`.

    Si el usuario quiere configurar la aplicación con un rol que no tenga este prefijo, en primer lugar deberá conceder permisos de forma explícita al usuario para la acción `iam:PassRole` del rol específico.

También puede crear sus propias políticas de IAM personalizadas para conceder permisos a las acciones y recursos de . Puede asociar estas políticas personalizadas a los usuarios o grupos de que requieran esos permisos.

Ejemplos de políticas administradas por el cliente

Los ejemplos que aparecen en esta sección muestran un grupo de políticas de ejemplo que puede asociar a un usuario. Si es la primera vez que crea una política, le recomendamos que cree primer lugar un usuario en su cuenta. A continuación, asocie las políticas al usuario por orden, según se detalla en los pasos de esta sección. Luego, podrá utilizar la consola para comprobar los efectos de cada política a medida que la asigna al usuario.

En un primer momento como el usuario no tiene permisos no puede hacer nada en la consola. Al asociar políticas al usuario, podrá verificar que este pueda realizar diversas acciones en la consola.

Le recomendamos que utilice dos ventanas de navegador. En una ventana, cree el usuario y conceda permisos. En la otra, inicie sesión Consola de administración de AWS con las credenciales del usuario y verifique los permisos a medida que los concede.

Para ver ejemplos que ilustran cómo crear un rol de IAM que puede utilizar como rol de ejecución en la aplicación, consulte Creación de roles de IAM en la Guía del usuario de IAM.

Pasos de ejemplo

- Paso 1: Crear un usuario de IAM
- Paso 2: conceder permisos al usuario para acciones que no son específicas de
- Paso 3: Permitir que el usuario vea una lista de aplicaciones y los detalles
- Paso 4: Permitir que el usuario inicie aplicaciones específicas
- Paso 5: Permitir que el usuario cree una aplicación de
- Paso 6: permitir a la aplicación utilizar el procesamiento previo de Lambda

Paso 1: Crear un usuario de IAM

En primer lugar, cree un usuario de IAM, añádalo a un grupo de IAM con permisos administrativos y, a continuación, conceda permisos administrativos al usuario que ha creado. A continuación, podrás acceder AWS mediante una URL especial y las credenciales de ese usuario.

Para obtener instrucciones, consulte Creación de su primer grupo de administradores y usuarios de IAM en la Guía del usuario de IAM.

Paso 2: conceder permisos al usuario para acciones que no son específicas de

En primer lugar, conceda permiso a un usuario para todas las acciones que no son específicas de y que el usuario necesitará para trabajar con aplicaciones de . Estos incluyen permisos para trabajar con transmisiones (acciones de Amazon Kinesis Data Streams, acciones de Amazon Data Firehose) y permisos para acciones. CloudWatch Asigne la siguiente política al usuario.

Es necesario actualizar la política proporcionando un nombre de rol de IAM para el que desea conceder el permiso `iam:PassRole` o especificar un carácter comodín (*) para indicar todos los roles de IAM. Esto no es una práctica segura, pero no puede crear un rol de IAM específico durante esta prueba.

JSON

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

```
        {
            "Effect": "Allow",
            "Action": [
                "kinesis:CreateStream",
                "kinesis:DeleteStream",
                "kinesis:DescribeStream",
                "kinesis:ListStreams",
                "kinesis:PutRecord",
                "kinesis:PutRecords"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "firehose:DescribeDeliveryStream",
                "firehose:ListDeliveryStreams"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:ListMetrics"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "logs:GetLogEvents",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:ListPolicyVersions",
                "iam:ListRoles"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
```

```
            "Resource": "arn:aws:iam::*:role/service-role/role-name"
        }
    ]
}
```

Paso 3: Permitir que el usuario vea una lista de aplicaciones y los detalles

La siguiente política concede a un usuario los siguientes permisos:

- Permiso de la acción `kinesisanalytics:ListApplications` para que el usuario pueda ver una lista de aplicaciones. Se trata de una llamada a la API en el nivel de servicio y, por tanto, especifique"*" como valor de `Resource`.
- Permiso de la acción `kinesisanalytics:DescribeApplication` para que pueda obtener información acerca de cualquiera de las aplicaciones.

Añada esta política para el usuario.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kinesisanalytics:ListApplications"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "kinesisanalytics:DescribeApplication"
            ],
            "Resource": "arn:aws:kinesisanalytics:us-
east-1:123456789012:application/*"
        }
    ]
```

```
}
```

Verifique estos permisos iniciando sesión en la consola con las credenciales de usuario.

Paso 4: Permitir que el usuario inicie aplicaciones específicas

Si desea que el usuario pueda iniciar una de las aplicaciones de existentes, asocie la siguiente política al usuario. La política proporciona el permiso para la acción `kinesisanalytics:StartApplication`. Debe actualizar la política proporcionando su ID de cuenta, AWS región y nombre de la aplicación.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kinesisanalytics:StartApplication"
            ],
            "Resource": "arn:aws:kinesisanalytics:us-
east-1:123456789012:application/application-name"
        }
    ]
}
```

Paso 5: Permitir que el usuario cree una aplicación de

Si desea que el usuario cree una aplicación de , a continuación puede asociar la política siguiente al usuario. Debe actualizar la política y proporcionar una AWS región, su ID de cuenta y un nombre de aplicación específico que desee que cree el usuario o un «*» para que el usuario pueda especificar cualquier nombre de aplicación (y así crear varias aplicaciones).

JSON

```
{
    "Version":"2012-10-17",
```

```
    "Statement": [
        {
            "Sid": "Stmt1473028104000",
            "Effect": "Allow",
            "Action": [
                "kinesisanalytics:CreateApplication"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "kinesisanalytics:StartApplication",
                "kinesisanalytics:UpdateApplication",
                "kinesisanalytics:AddApplicationInput",
                "kinesisanalytics:AddApplicationOutput"
            ],
            "Resource": "arn:aws:kinesisanalytics:us-
east-1:123456789012:application/application-name"
        }
    ]
}
```

Paso 6: permitir a la aplicación utilizar el procesamiento previo de Lambda

Si desea que la aplicación pueda utilizar el procesamiento previo de Lambda, asocie la siguiente política al rol.

```
    {
      "Sid": "UseLambdaFunction",
      "Effect": "Allow",
      "Action": [
          "lambda:InvokeFunction",
          "lambda:GetFunctionConfiguration"
      ],
      "Resource": "<FunctionARN>"
  }
```

Permisos de la API: referencia de acciones, permisos y recursos

Cuando configure Control de acceso y escriba una política de permisos que se pueda asociar a una identidad de IAM (políticas basadas en identidad), puede utilizar la siguiente lista como referencia. se incluye cada operación de la API, las acciones correspondientes para las que puede conceder permisos para realizar la acción y el AWS recurso para el que puede conceder los permisos. Las acciones se especifican en el campo `Action` de la política y el valor del recurso se especifica en el campo `Resource` de la política.

Puedes usar claves AWS de condición generales en tus políticas para expresar las condiciones. Para obtener una lista completa de las claves AWS anchas, consulta las claves disponibles en la Guía del usuario de IAM.

> **Note**
>
> Para especificar una acción, use el prefijo `kinesisanalytics` seguido del nombre de operación de la API (por ejemplo, `kinesisanalytics:AddApplicationInput`).

API y permisos necesarios para las acciones

Operación de API:

Permisos necesarios (acción de la API):

Recursos:

API y permisos necesarios para las acciones

API de Amazon RDS y permisos necesarios para las acciones

Operación de API:AddApplicationInput

Acción: `kinesisanalytics:AddApplicationInput`

Recursos:

`arn:aws:kinesisanalytics:` *`region`*`:`*`accountId`*`:application/`*`application-name`*

GetApplicationState

La consola utiliza un método interno llamado `GetApplicationState` para crear muestras o acceder a los datos de la aplicación. La aplicación de servicio debe tener permisos para que la API interna de `kinesisanalytics:GetApplicationState` pueda crear muestras o acceder a los datos de la aplicación a través de Consola de administración de AWS.

Supervisión de Amazon Kinesis Data Analytics

Kinesis Data Analytics proporciona funciones de supervisión para sus aplicaciones. Para obtener más información, consulte Supervisión.

Validación de la conformidad de aplicaciones de Amazon Kinesis Data Analytics para SQL

Los auditores externos evalúan la seguridad y la conformidad de Amazon Kinesis Data Analytics como parte de AWS varios programas de conformidad. Esto incluye SOC, PCI, HIPAA y otros.

Para ver una lista de AWS los servicios incluidos en el ámbito de los programas de conformidad específicos, consulta Amazon Services in Scope by Compliance Program. Para obtener información general, consulte Programas de conformidad de AWS.

Puede descargar informes de auditoría de terceros utilizando AWS Artifact. Para obtener más información, consulte Descarga de informes en AWS Artifact.

Su responsabilidad con la conformidad al utilizar Kinesis Data Analytics se determina en función de la confidencialidad de los datos, los objetivos de conformidad de su empresa y la legislación y los reglamentos aplicables. Si su uso de Kinesis Data Analytics está sujeto a conformidad con normas tales como HIPAA o PCI, AWS proporciona recursos para ayudarle:

- Guías de inicio rápido sobre seguridad y conformidad: estas guías de implementación analizan las consideraciones arquitectónicas y proporcionan los pasos para implementar entornos básicos centrados en la seguridad y el cumplimiento. AWS
- Documento técnico sobre cómo diseñar una arquitectura para la seguridad y el cumplimiento de la HIPAA: este documento técnico describe cómo las empresas pueden utilizar para crear aplicaciones que cumplan con la HIPAA. AWS
- AWS Recursos de cumplimiento: esta colección de libros de trabajo y guías puede aplicarse a su sector y ubicación.

- AWS Config— Este AWS servicio evalúa en qué medida las configuraciones de sus recursos cumplen con las prácticas internas, las directrices del sector y las normativas.
- AWS Security Hub CSPM— Este AWS servicio proporciona una visión integral del estado de su seguridad AWS que le ayuda a comprobar el cumplimiento de los estándares y las mejores prácticas del sector de la seguridad.

Resiliencia en Amazon Kinesis Data 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 zonas de disponibilidad sin interrupciones. Las zonas de disponibilidad tienen una mayor disponibilidad, tolerancia a errores y escalabilidad que las infraestructuras tradicionales de centros de datos únicos o múltiples.

Para obtener más información sobre AWS las regiones y las zonas de disponibilidad, consulte Infraestructura global.AWS

Además de la infraestructura AWS global, Kinesis Data Analytics ofrece varias funciones que le ayudan a satisfacer sus necesidades de respaldo y resiliencia de datos.

Recuperación ante desastres

Kinesis Data Analytics se ejecuta en un modo sin servidor y se ocupa de las reducciones del rendimiento del host, la disponibilidad de las zonas de disponibilidad y otros problemas relacionados con la infraestructura al llevar a cabo una migración automática. Cuando esto sucede, Kinesis Data Analytics se asegura de que la aplicación se procesa sin que se pierdan datos. Para obtener más información, consulte Modelo de entrega para conservar la salida de las aplicaciones en destinos externos.

Seguridad de la infraestructura en aplicaciones de Kinesis Data Analytics para SQL

Como servicio gestionado, Amazon Kinesis Data Analytics está protegido por AWS los procedimientos de seguridad de red global que se describen en el documento técnico Amazon Web Services: Overview of Security Processes.

Utilice las llamadas a la API AWS publicadas para acceder a Kinesis Data Analytics a través de la red. Los clientes deben ser compatibles con Transport Layer Security (TLS) 1.2 o una versión posterior. Los clientes también deben ser compatibles con conjuntos de cifrado con confidencialidad directa total (PFS) tales como Ephemeral Diffie-Hellman (DHE) o Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). La mayoría de los sistemas modernos como Java 7 y posteriores son compatibles con estos modos.

Además, las solicitudes deben estar firmadas mediante un ID de clave de acceso y una clave de acceso secreta que esté asociada a una entidad principal de IAM. También puedes utilizar AWS Security Token Service (AWS STS) para generar credenciales de seguridad temporales para firmar solicitudes.

Prácticas recomendadas de seguridad para Kinesis Data Analytics

Amazon Kinesis Data Analytics 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.

Uso de los roles de IAM para obtener acceso a otros servicios de Amazon

La aplicación de Kinesis Data Analytics debe tener credenciales válidas para poder obtener acceso a otros recursos, como los flujos de datos de Kinesis, los flujos de entrega de Firehose o los buckets de Amazon S3. No debe almacenar AWS las credenciales directamente en la aplicación ni 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, debería utilizar un rol de IAM para administrar las credenciales temporales que la aplicación utiliza para obtener acceso a otros recursos. Al utilizar un rol, no tiene que utilizar credenciales a largo plazo para acceder a otros recursos.

Para obtener más información, consulte los siguientes temas de la guía del usuario de IAM:

- Roles de IAM
- Situaciones habituales con los roles: usuarios, aplicaciones y servicios

Implementación del cifrado en el servidor en recursos dependientes

Los datos en reposo y los datos en tránsito se cifran en Kinesis Data Analytics y este cifrado no se puede deshabilitar. Debe implementar el cifrado del lado del servidor en los recursos dependientes, como los flujos de datos de Kinesis, los flujos de entrega de Firehose y los buckets de Amazon S3. Para obtener más información acerca de la implementación del cifrado del lado del servidor en recursos dependientes, consulte Protección de los datos.

Úselo CloudTrail para monitorear las llamadas a la API

Kinesis Data Analytics está integrado AWS CloudTrail con un servicio que proporciona un registro de las acciones realizadas por un usuario, un rol o un servicio de Amazon en Kinesis Data Analytics.

Con la información recopilada por CloudTrail, puede determinar la solicitud que se realizó a Kinesis Data Analytics, la dirección IP desde la que se realizó la solicitud, quién la realizó, cuándo se realizó y detalles adicionales.

Para obtener más información, consulte the section called “Uso AWS CloudTrail”.

Monitorización de for SQL Applications

El monitoreo es una parte importante del mantenimiento de la fiabilidad, la disponibilidad y el desempeño de y su aplicación de . Debe recopilar datos de supervisión de todas las partes de la AWS solución para poder depurar más fácilmente un error multipunto si se produce uno. No obstante, antes de comenzar a monitorear , debe crear un plan 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 supervisión va a utilizar?
- ¿Quién se encargará de realizar las tareas de supervisión?
- ¿Quién debería recibir una notificación cuando surjan problemas?

El siguiente paso consiste en establecer un punto de referencia del desempeño de normal en su entorno. Para ello se mide el desempeño en distintos momentos y bajo distintas condiciones de carga. A medida que supervise , puede almacenar datos de monitoreo históricos. A medida que lo haga, puede compararlos con los datos de desempeño actuales, identificar patrones de desempeño normal y anomalías en el desempeño, así como desarrollar métodos para la resolución de problemas.

Con , usted monitorea la aplicación. La aplicación procesa los flujos de datos (de entrada o salida), los cuales incluyen identificadores que puede utilizar para restringir la búsqueda en los registros. CloudWatch Para obtener información sobre cómo procesa los flujos de datos, consulte Aplicaciones de Amazon Kinesis Data Analytics para SQL: cómo funciona.

La métrica más importante es `millisBehindLatest`, que indica a qué distancia detrás una aplicación está leyendo desde el origen de streaming. En un caso típico, los milisegundos detrás deben ser en o cerca de cero. Es frecuente que aparezcan picos breves, que aparecen como un aumento en `millisBehindLatest`.

Te recomendamos configurar una CloudWatch alarma que se active cuando la aplicación tenga más de una hora de retraso en la lectura de la fuente de streaming. Para algunos casos de uso que requieren procesamiento muy cerca del tiempo real, como la emisión de datos procesados en una

aplicación en directo, podría decidir configurar la alarma en un valor inferior, como, por ejemplo, cinco minutos.

Temas

Herramientas de monitorización

AWS proporciona varias herramientas que puede utilizar para supervisar. 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.

Herramientas de monitorización automatizadas

Puede utilizar las siguientes herramientas de monitorización automatizado para vigilar e informar cuando haya algún problema:

- 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 del Servicio de Notificación Simple (Amazon SNS) o a una política de Amazon EC2 Auto Scaling. CloudWatch las alarmas no invocan acciones simplemente porque se encuentren 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.
- Amazon CloudWatch Logs: supervise, almacene y acceda a sus archivos de registro desde AWS CloudTrail u otras fuentes. Para obtener más información, consulte Supervisión de archivos de registro en la Guía del CloudWatch usuario de Amazon.
- Amazon CloudWatch Events: haga coincidir los eventos y diríjalos a una o más funciones o transmisiones de destino para realizar cambios, capturar información de estado y tomar medidas correctivas. Para obtener más información, consulta Qué es Amazon CloudWatch Events en la Guía del CloudWatch usuario de Amazon.

- AWS CloudTrail Supervisión de registros: comparta archivos de registro entre cuentas, supervise los archivos de CloudTrail registro en tiempo real enviándolos a CloudWatch Logs, cree aplicaciones de procesamiento de registros en Java y valide que sus archivos de registro no hayan cambiado después de su entrega CloudTrail. Para obtener más información, consulte Trabajar con archivos de CloudTrail registro en la Guía del AWS CloudTrail usuario.

Herramientas de monitorización manual

Otra parte importante del monitoreo consiste en monitorear manualmente los elementos que las CloudWatch alarmas no cubren. Los Consola de administración de AWS paneles CloudWatch, Trusted Advisor, y otros proporcionan una at-a-glance vista del estado de su AWS entorno.

- 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
  - Buscar y examinar todas sus métricas
  - Crear y editar las alarmas de notificación de problemas
- AWS Trusted Advisor puede ayudarle a supervisar su rendimiento, fiabilidad, seguridad y rentabilidad. Cuatro instancias Trusted Advisor están disponibles para todos los usuarios. Hay más de 50 comprobaciones disponibles para usuarios con un plan de soporte Business o Enterprise. Para obtener más información, consulte AWS Trusted Advisor.

Monitorización con Amazon CloudWatch

Puedes monitorizar las aplicaciones con Amazon CloudWatch. CloudWatch recopila y procesa datos sin procesar para convertirlos en métricas legibles y casi en tiempo real. Estas estadísticas se conservan durante un periodo de dos semanas. Puede acceder a información histórica y disponer de una mejor perspectiva sobre el desempeño de su aplicación web o servicio. De forma predeterminada, los datos métricos se envían automáticamente a CloudWatch. Para obtener

más información, consulta ¿Qué es Amazon CloudWatch? en la Guía del CloudWatch usuario de Amazon.

Temas

- Métricas y dimensiones
- Consulta de dimensiones y métricas de
- Creación de CloudWatch alarmas para monitorizar
- Trabajar con Amazon CloudWatch Logs

Métricas y dimensiones

El espacio de nombres de `AWS/KinesisAnalytics` incluye las siguientes métricas.

Métrica	Description (Descripción)
`Bytes`	El número de bytes leídos (por secuencia de entrada) o escritos (por secuencia de salida). Niveles: por secuencia de entrada y por secuencia de salida.
`KPUs`	Cantidad de unidades de procesamiento de Kinesis Processing Units que se utilizan para ejecutar su aplicación de procesamiento de flujos. El número medio de horas KPUs utilizadas por hora determina la facturación de tu aplicación. Niveles: nivel de aplicación
`MillisBehindLatest`	Indica el retraso de lectura de una fuente de streaming por parte de una aplicación con respecto a la hora actual. Niveles: nivel de aplicación
`Records`	El número de registros leídos (por secuencia de entrada) o escritos (por secuencia de salida).

Métrica	Description (Descripción)
	Niveles: por secuencia de entrada y por secuencia de salida.
`Success`	1 para cada intento de entrega correcto al destino configurado para la aplicación; 0 para cada intento de entrega que ha fallado. El valor medio de esta métrica indica cuántas entregas correctas se han realizado. Niveles: por destino.
`InputProcessing.Duration`	El tiempo necesario para cada invocación de AWS Lambda función realizada por. Niveles: por secuencia de entrada
`InputProcessing.OkRecords`	El número de registros devueltos por una función de Lambda que se marcaron con el estado `Ok`. Niveles: por secuencia de entrada
`InputProcessing.OkBytes`	La suma de bytes de los registros devueltos por una función de Lambda que se marcaron con el estado `Ok`. Niveles: por secuencia de entrada
`InputProcessing.DroppedRecords`	El número de registros devueltos por una función de Lambda que se marcaron con el estado `Dropped`. Niveles: por secuencia de entrada
`InputProcessing.ProcessingFailedRecords`	El número de registros devueltos por una función de Lambda que se marcaron con el estado `ProcessingFailed` . Niveles: por secuencia de entrada

Métrica	Description (Descripción)
`InputProcessing.Success`	El número de invocaciones de funciones de Lambda correctas realizadas por . Niveles: por secuencia de entrada
`LambdaDelivery.OkRecords`	El número de registros devueltos por una función de Lambda que se marcaron con el estado `Ok`. Niveles: Por destino de Lambda
`LambdaDelivery.DeliveryFailedRecords`	El número de registros devueltos por una función de Lambda que se marcaron con el estado `DeliveryFailed` . Niveles: Por destino de Lambda
`LambdaDelivery.Duration`	Tiempo que se tarda en invocar cada función de Lambda. Niveles: Por destino de Lambda

proporciona métricas para las siguientes dimensiones.

Dimensión	Description (Descripción)
`Flow`	Por secuencia de entrada: entrada Por secuencia de salida: salida
`Id`	Por secuencia de entrada: ID de entrada Por secuencia de salida: ID de salida

Consulta de dimensiones y métricas de

Cuando su aplicación procesa los flujos de datos, envía las siguientes métricas y dimensiones a CloudWatch. Puede utilizar los siguientes procedimientos para consultar las métricas de .

En la consola, se agrupan las métricas en primer lugar por el espacio de nombres de servicio y, luego, por las combinaciones de dimensiones dentro de cada espacio de nombres.

Para ver las métricas mediante la CloudWatch consola

1. Abra la CloudWatch consola en https://console.aws.amazon.com/cloudwatch/.
2. En el panel de navegación, seleccione Métricas.
3. En el panel CloudWatch Métricas por categoría de, elija una categoría de métricas.
4. En el panel superior, desplácese para ver la lista completa de métricas.

Para ver las métricas mediante el AWS CLI

- En el símbolo del sistema, ejecute el siguiente comando.

```
aws cloudwatch list-metrics --namespace "AWS/KinesisAnalytics" --region region
```

Se recopilan las métricas en los siguientes niveles:

- Aplicación
- Secuencia de entrada
- Secuencia de salida

Creación de CloudWatch alarmas para monitorizar

Puede crear una CloudWatch alarma de Amazon que envíe un mensaje de Amazon SNS cuando la alarma cambie de estado. Una alarma vigila una métrica individual durante un periodo de tiempo que usted especifica. 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 que se envía a un tema de Amazon SNS o a una política de escalado automático.

Las alarmas invocan acciones únicamente para los cambios de estado prolongados. Para que una CloudWatch alarma invoque una acción, el estado debe haber cambiado y mantenerse durante un período de tiempo específico.

Puede configurar las alarmas mediante la Consola de administración de AWS, o la CloudWatch API CloudWatch AWS CLI, tal y como se describe a continuación.

Para configurar una alarma mediante la CloudWatch consola

1. Inicie sesión en Consola de administración de AWS y abra la CloudWatch consola en https://console.aws.amazon.com/cloudwatch/.
2. Elija Crear alarma. Se inicia el Create Alarm Wizard (Asistente de creación de alarmas).
3. Seleccione Kinesis Analytics Metrics (Métricas de Kinesis Analytics). A continuación, desplácese por las métricas de para localizar la métrica en la que desea colocar una alarma.

   Para mostrar solo las métricas de ; busque el ID del sistema de archivos de su sistema de archivos. Seleccione la métrica para crear una alarma y, luego, elija Next (Siguiente).
4. Escriba valores para Name (Nombre), Description (Descripción) y Whenever (Siempre que) para la métrica.
5. Si CloudWatch quieres enviarte un correo electrónico cuando se alcance el estado de alarma, en el campo Siempre que aparezca esta alarma:, selecciona State is ALARM. En el campo Send notification to: (Enviar notificación a:), elija un tema de SNS. Si selecciona Crear tema, puede definir el nombre y las direcciones de correo electrónico de una nueva lista de suscripción de correo electrónico. Esta lista se guarda y aparece en el campo para futuras alarmas.

   > **Note**
   >
   > Si utiliza Crear tema para crear un nuevo tema de Amazon SNS, debe verificar las direcciones de correo electrónico para que reciban notificaciones. Los correos electrónicos solo se envían cuando la alarma entra en estado de alarma. Si este cambio en el estado de la alarma se produce antes de que se verifiquen las direcciones de correo electrónico, no reciben una notificación.
6. En la sección Alarm Preview, previsualice la alarma que está a punto de crear.
7. Elija Create Alarm para crear la alarma.

Para configurar una alarma mediante la CloudWatch CLI

- Llamar a `mon-put-metric-alarm`. Para obtener más información, consulte la referencia de Amazon CloudWatch CLI.

Para configurar una alarma mediante la CloudWatch API

- Llamar a `PutMetricAlarm`. Para obtener más información, consulta la referencia de la CloudWatch API de Amazon.

Trabajar con Amazon CloudWatch Logs

Si una aplicación de está mal configurado, puede pasar a un estado de ejecución durante el inicio de la aplicación. O bien puede actualizar pero no procesar los datos en la secuencia de entrada en la aplicación. Al añadir una opción de CloudWatch registro a la aplicación, puede supervisar los problemas de configuración de la aplicación.

puede generar errores de configuración en las siguientes condiciones:

- El flujo de datos de Kinesis utilizado para la entrada no existe.
- El flujo de entrega de Amazon Data Firehose utilizado para la entrada no existe.
- El bucket de Amazon S3 utilizado como origen de datos de referencia no existe.
- El archivo especificado en el origen de datos de referencia en el bucket de S3 no existe.
- El recurso correcto no está definido en la función AWS Identity and Access Management (IAM) que administra los permisos relacionados.
- No se define el permiso correcto en el rol de IAM que administra los permisos relacionados.
- no tiene permiso para asumir el rol de IAM que administra los permisos relacionados.

Para obtener más información sobre Amazon CloudWatch, consulta la Guía del CloudWatch usuario de Amazon.

Añadir la acción PutLogEvents política

necesita permisos para escribir errores de configuración. CloudWatch Puede añadir estos permisos a la función de IAM que asume, tal y como se describe a continuación. Para obtener más información acerca del uso de un rol de IAM, consulte Gestión de identidad y acceso en Kinesis Data Analytics.

Política de confianza

Para conceder permisos a para asumir una función de IAM, puede asignar la siguiente política de confianza al rol.

JSON

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

Política de permisos

Para conceder a una aplicación permisos para escribir eventos de registro CloudWatch desde un recurso, puede utilizar la siguiente política de permisos de IAM.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "Stmt0123456789000",
            "Effect": "Allow",
            "Action": [
                "logs:PutLogEvents"
            ],
            "Resource": [
                "arn:aws:logs:us-east-1:123456789012:log-group:my-log-group:log-
stream:my-log-stream*"
            ]
        }
    ]
}
```

Incorporación de la monitorización de errores de configuración

Utilice las siguientes acciones de la API para añadir una opción de CloudWatch registro a una aplicación nueva o existente o cambiar una opción de registro para una aplicación existente.

> **Note**
>
> Actualmente, solo puedes añadir una opción de CloudWatch registro a una aplicación mediante las acciones de la API. No puedes añadir opciones de CloudWatch registro mediante la consola.

Añadir una opción de CloudWatch registro al crear una aplicación

El siguiente ejemplo de código muestra cómo utilizar la `CreateApplication` acción para añadir una opción de CloudWatch registro al crear una aplicación. Para obtener más información sobre `Create_Application`, consulte CreateApplication.

```
{
   "ApplicationCode": "<The SQL code the new application will run on the input
 stream>",
   "ApplicationDescription": "<A friendly description for the new application>",
   "ApplicationName": "<The name for the new application>",
   "Inputs": [ ... ],
   "Outputs": [ ... ],
   "CloudWatchLoggingOptions": [{
      "LogStreamARN": "<Amazon Resource Name (ARN) of the CloudWatch log stream to add
 to the new application>",
      "RoleARN": "<ARN of the role to use to access the log>"
 }]
}
```

Añadir una opción de CloudWatch registro a una aplicación existente

El siguiente ejemplo de código muestra cómo utilizar la acción `AddApplicationCloudWatchLoggingOption` para añadir una opción de registro de CloudWatch a una aplicación. Para obtener más información acerca de `AddApplicationCloudWatchLoggingOption`, consulte AddApplicationCloudWatchLoggingOption.

```
{
```

```
   "ApplicationName": "<Name of the application to add the log option to>",
   "CloudWatchLoggingOption": {
      "LogStreamARN": "<ARN of the log stream to add to the application>",
      "RoleARN": "<ARN of the role to use to access the log>"
   },
   "CurrentApplicationVersionId": <Version of the application to add the log to>
}
```

Actualización de una opción de CloudWatch registro existente

El siguiente ejemplo de código muestra cómo utilizar la `UpdateApplication` acción para modificar una opción de CloudWatch registro existente. Para obtener más información acerca de `UpdateApplication`, consulte UpdateApplication.

```
{
   "ApplicationName": "<Name of the application to update the log option for>",
   "ApplicationUpdate": {
      "CloudWatchLoggingOptionUpdates": [
         {
            "CloudWatchLoggingOptionId": "<ID of the logging option to modify>",
            "LogStreamARNUpdate": "<ARN of the new log stream to use>",
            "RoleARNUpdate": "<ARN of the new role to use to access the log stream>"
         }
      ],
   },
   "CurrentApplicationVersionId": <ID of the application version to modify>
}
```

Eliminar una opción de CloudWatch registro de una aplicación

El siguiente ejemplo de código muestra cómo utilizar la `DeleteApplicationCloudWatchLoggingOption` acción para eliminar una opción de CloudWatch registro existente. Para obtener más información acerca de `DeleteApplicationCloudWatchLoggingOption`, consulte DeleteApplicationCloudWatchLoggingOption.

```
{
   "ApplicationName": "<Name of application to delete log option from>",
   "CloudWatchLoggingOptionId": "<ID of the application log option to delete>",
```

```
    "CurrentApplicationVersionId": <Version of the application to delete the log option
 from>
}
```

Errores de configuración

Las siguientes secciones contienen detalles sobre los errores que puede ver en Amazon CloudWatch Logs debido a una aplicación mal configurada.

Formato del mensaje con error

Los mensajes de error generados por la configuración inadecuada de la aplicación aparecen en el siguiente formato.

```
{
"applicationARN": "string",
"applicationVersionId": integer,
"messageType": "ERROR",
"message": "string",
"inputId": "string",
"referenceId": "string",
"errorCode": "string"
"messageSchemaVersion": "integer",
}
```

Los campos de un mensaje de error contienen la siguiente información:

- `applicationARN`: El nombre de recurso de Amazon (ARN) de la generación de aplicación, por ejemplo: `arn:aws:kinesisanalytics:us-east-1:112233445566:application/sampleApp`
- `applicationVersionId`: La versión de la aplicación al momento en que se encontró el error. Para obtener más información, consulte ApplicationDetail.
- `messageType`: El tipo de mensaje. En la actualidad, este tipo puede ser únicamente `ERROR`.
- `message`: Los detalles del error, por ejemplo:

```
There is a problem related to the configuration of your input. Please check that the
 resource exists, the role has the correct permissions to access the resource and
 that Kinesis Analytics can assume the role provided.
```

- `inputId`: La ID asociada con la entrada de la aplicación. Este valor solo está presente si esta entrada es la causa del error. Este valor no está presente si `referenceId` está presente. Para obtener más información, consulte DescribeApplication.
- `referenceId`: El ID asociado con el origen de datos de referencia de la aplicación. Este valor solo está presente si este origen es la causa del error. Este valor no está presente si `inputId` está presente. Para obtener más información, consulte DescribeApplication.
- `errorCode`: El identificador del error. Este ID es `InputError` o `ReferenceDataError`.
- `messageSchemaVersion`: Un valor que especifica el mensaje actual de la versión del esquema, actualmente 1. Puede comprobar este valor para ver si se actualizó el esquema de mensajes de error.

Errores

Entre los errores que pueden aparecer en CloudWatch los registros se incluyen los siguientes.

El recurso no existe

Si se especifica un ARN para una secuencia de entrada de Kinesis que no existe, pero el ARN es sintácticamente correcto, se genera un error como el siguiente.

```
{
"applicationARN": "arn:aws:kinesisanalytics:us-east-1:112233445566:application/
sampleApp",
"applicationVersionId": "5",
 "messageType": "ERROR",
 "message": "There is a problem related to the configuration of your input. Please
 check that the resource exists, the role has the correct permissions to access the
 resource and that Kinesis Analytics can assume the role provided.",
 "inputId":"1.1",
 "errorCode": "InputError",
 "messageSchemaVersion": "1"
}
```

Si se usa una clave de archivo de Amazon S3 para los datos de referencia, se genera un error como el siguiente.

```
{
 "applicationARN": "arn:aws:kinesisanalytics:us-east-1:112233445566:application/
sampleApp",
```

```
 "applicationVersionId": "5",
 "messageType": "ERROR",
 "message": "There is a problem related to the configuration of your reference data.
 Please check that the bucket and the file exist, the role has the correct permissions
 to access these resources and that Kinesis Analytics can assume the role provided.",
 "referenceId":"1.1",
 "errorCode": "ReferenceDataError",
 "messageSchemaVersion": "1"
}
```

El rol no existe

Si se especifica un ARN para un rol de entrada de IAM que no existe, pero el ARN es sintácticamente correcto, se generará un error como el siguiente.

```
{
 "applicationARN": "arn:aws:kinesisanalytics:us-east-1:112233445566:application/
sampleApp",
 "applicationVersionId": "5",
 "messageType": "ERROR",
 "message": "There is a problem related to the configuration of your input. Please
 check that the resource exists, the role has the correct permissions to access the
 resource and that Kinesis Analytics can assume the role provided.",
 "inputId":null,
 "errorCode": "InputError",
 "messageSchemaVersion": "1"
}
```

El rol no tiene permisos para obtener acceso al recurso

Si se utiliza un rol de entrada que no tiene permiso para obtener acceso a los recursos de entrada, como, por ejemplo, una secuencia de origen de Kinesis, se genera un error como el siguiente.

```
{
 "applicationARN": "arn:aws:kinesisanalytics:us-east-1:112233445566:application/
sampleApp",
 "applicationVersionId": "5",
 "messageType": "ERROR",
 "message": "There is a problem related to the configuration of your input. Please
 check that the resource exists, the role has the correct permissions to access the
 resource and that Kinesis Analytics can assume the role provided.",
 "inputId":null,
```

```
  "errorCode": "InputError",
  "messageSchemaVersion": "1"
}
```

Registro de llamadas a la API de AWS CloudTrail con

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. CloudTrail captura todas las llamadas a la API como eventos. Las llamadas capturadas incluyen las llamadas desde la consola de y las llamadas desde el código a las operaciones de la API de . Si crea una ruta, puede habilitar la entrega continua de CloudTrail eventos a un bucket de Amazon S3, incluidos los eventos para. 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 a qué dirección IP se realizó la solicitud, quién la realizó, cuándo se realizó y detalles adicionales.

Para obtener más información CloudTrail, consulte la Guía AWS CloudTrail del usuario.

Información en CloudTrail

CloudTrail está habilitada en su AWS cuenta al crear la cuenta. Cuando se produce una actividad en, esa actividad se registra en un CloudTrail evento junto con otros eventos de AWS servicio en el historial de eventos. Puedes ver, buscar y descargar los eventos recientes en tu AWS cuenta. Para obtener más información, consulte Visualización de eventos con el historial de CloudTrail eventos.

Para tener un registro continuo de los eventos de tu AWS cuenta, incluidos los eventos de tu cuenta, crea una ruta. Un rastro permite CloudTrail entregar archivos de registro a un bucket de Amazon S3. De forma predeterminada, cuando se crea un registro de seguimiento en la consola, el registro de seguimiento se aplica a todas las . 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 más información, consulte los siguientes temas:

- Introducción a la creación de registros de seguimiento
- CloudTrail Integraciones y servicios compatibles
- Configuración de las notificaciones de Amazon SNS para CloudTrail
- Recibir archivos de CloudTrail registro de varias regiones y recibir archivos de CloudTrail registro de varias cuentas

Todas las acciones se registran CloudTrail y se documentan en la referencia de la API. Por ejemplo, las llamadas a las `UpdateApplication` acciones `CreateApplication` y las acciones generan entradas en los archivos de CloudTrail registro.

Cada entrada de registro o evento contiene información sobre quién generó la solicitud. La información de identidad del usuario le ayuda a determinar lo siguiente:

- Si la solicitud se realizó con credenciales de usuario Usuario raíz de la cuenta de AWS o con ellas.
- 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 Elemento userIdentity de CloudTrail .

Descripción de las entradas de archivos de registro de

Un rastro es una configuración que permite la entrega de eventos como archivos de registro a un bucket de Amazon S3 que usted especifique. CloudTrail Los archivos de registro contienen una o más entradas de registro. Un evento representa una solicitud única de cualquier fuente e incluye información sobre la acción solicitada, la fecha y la hora de la acción, los parámetros de la solicitud, etc. CloudTrail Los archivos de registro no son un registro ordenado de las llamadas a la API pública, por lo que no aparecen en ningún orden específico.

En el siguiente ejemplo, se muestra una entrada de CloudTrail registro que muestra las DescribeApplicationacciones AddApplicationCloudWatchLoggingOptiony.

```
{
    "Records": [
        {
            "eventVersion": "1.05",
            "userIdentity": {
                "type": "IAMUser",
                "principalId": "EX_PRINCIPAL_ID",
                "arn": "arn:aws:iam::012345678910:user/Alice",
                "accountId": "012345678910",
                "accessKeyId": "EXAMPLE_KEY_ID",
                "userName": "Alice"
            },
            "eventTime": "2019-03-14T01:03:00Z",
            "eventSource": "kinesisanalytics.amazonaws.com",
```

```
            "eventName": "AddApplicationCloudWatchLoggingOption",
            "awsRegion": "us-east-1",
            "sourceIPAddress": "127.0.0.1",
            "userAgent": "aws-sdk-java/unknown-version Linux/x.xx",
            "requestParameters": {
                "currentApplicationVersionId": 1,
                "cloudWatchLoggingOption": {
                    "roleARN": "arn:aws:iam::012345678910:role/cloudtrail_test",
                    "logStreamARN": "arn:aws:logs:us-east-1:012345678910:log-
group:cloudtrail-test:log-stream:sql-cloudwatch"
                },
                "applicationName": "cloudtrail-test"
            },
            "responseElements": null,
            "requestID": "e897cd34-45f4-11e9-8912-e52573a36cd9",
            "eventID": "57fe50e9-c764-47c3-a0aa-d0c271fa1cbb",
            "eventType": "AwsApiCall",
            "apiVersion": "2015-08-14",
            "recipientAccountId": "303967445486"
        },
        {
            "eventVersion": "1.05",
            "userIdentity": {
                "type": "IAMUser",
                "principalId": "EX_PRINCIPAL_ID",
                "arn": "arn:aws:iam::012345678910:user/Alice",
                "accountId": "012345678910",
                "accessKeyId": "EXAMPLE_KEY_ID",
                "userName": "Alice"
            },
            "eventTime": "2019-03-14T05:37:20Z",
            "eventSource": "kinesisanalytics.amazonaws.com",
            "eventName": "DescribeApplication",
            "awsRegion": "us-east-1",
            "sourceIPAddress": "127.0.0.1",
            "userAgent": "aws-sdk-java/unknown-version Linux/x.xx",
            "requestParameters": {
                "applicationName": "cloudtrail-test"
            },
            "responseElements": null,
            "requestID": "3b74eb29-461b-11e9-a645-fb677e53d147",
            "eventID": "750d0def-17b6-4c20-ba45-06d9d45e87ee",
            "eventType": "AwsApiCall",
            "apiVersion": "2015-08-14",
```

```
            "recipientAccountId": "012345678910"
        }
    ]
}
```

Límites

Fechas de retirada

Tras considerarlo detenidamente, hemos decidido retirar las aplicaciones de Amazon Kinesis Data Analytics para SQL. Para ayudarle a planificar y migrar aplicaciones de Amazon Kinesis Data Analytics para SQL, retiraremos la oferta gradualmente a lo largo de 15 meses. Hay que tener en cuenta dos fechas importantes: el 15 de octubre de 2025 y el 27 de enero de 2026.

1. El 15 de octubre de 2025, suspenderemos sus aplicaciones y las pondremos en el estado `READY`. Podrá reiniciar sus aplicaciones en ese momento y seguir usándolas de forma normal con sujeción a los límites de servicio.
2. A partir del 15 de octubre de 2025, no podrá crear nuevas aplicaciones de Amazon Kinesis Data Analytics para SQL. Podrá ejecutar las aplicaciones existentes de forma normal con sujeción a los límites de servicio.
3. Eliminaremos sus aplicaciones a partir del 27 de enero de 2026. No podrá iniciar ni utilizar sus aplicaciones de Amazon Kinesis Data Analytics para SQL. A partir de ese momento, las aplicaciones de Amazon Kinesis Data Analytics para SQL dejarán de estar disponibles.

Le recomendamos que migre sus aplicaciones a Amazon Managed Service para Apache Flink o Amazon Managed Service para Apache Flink Studio antes del 15 de octubre de 2025. Para obtener recursos que le ayuden con la migración, consulte Ejemplos de migración a Managed Service para Apache Flink. Para obtener más información sobre Amazon Managed Service for Apache Flink o Amazon Managed Service para Apache Flink Studio, consulte la guía para desarrolladores de Amazon Managed Service para Apache Flink.

Límites

Cuando trabaje con aplicaciones de Amazon Kinesis Data Analytics para SQL, tenga en cuenta los siguientes límites:

- Kinesis Data Analytics for SQL está disponible en las AWS siguientes regiones: EE.UU. Este (Ohio), EE.UU. Este (Norte de Virginia), EE.UU. Oeste (Oregón), Canadá (Central), Europa (París), Europa (Irlanda), Europa (Fráncfort), Europa (Londres), Asia Pacífico (Hong Kong), Asia Pacífico (Bombay), Asia Pacífico (Sídney), Asia Pacífico (Singapur), Asia Pacífico (Seúl), Asia Pacífico

(Tokio), Sudamérica (São Paulo), (EEUU-Este) AWS GovCloud , (EEUU-Oeste). AWS GovCloud No tenemos previsto lanzar Kinesis Data Analytics for SQL en regiones AWS adicionales.

- Después del 28 de junio de 2023, no podrá crear nuevas aplicaciones de Kinesis Data Analytics for SQL mediante AWS la consola de administración si aún no utiliza Kinesis Data Analytics for SQL. Para obtener información sobre las fechas de retirada de Kinesis Data Analytics para SQL, consulte Fechas de retirada. Si creó una aplicación de Kinesis Data Analytics for SQL antes del 28 de junio de 2023, no habrá cambios en la forma en que crea y ejecuta las aplicaciones en la actualidad en AWS una región en la que ya utiliza Kinesis Data Analytics for SQL. Sin embargo, ya no podrá crear nuevas aplicaciones con la AWS consola en una región en la que no utilice Kinesis Data Analytics for SQL.
- Después del 12 de septiembre de 2023, no podrá crear nuevas aplicaciones con Kinesis Data Firehose como origen si aún no utiliza Kinesis Data Analytics para SQL. Para obtener información sobre las fechas de retirada de Kinesis Data Analytics para SQL, consulte Fechas de retirada. Los clientes actuales que utilizan aplicaciones de Kinesis Data Analytics para SQL con `KinesisFirehoseInput` pueden seguir añadiendo aplicaciones con `KinesisFirehoseInput` dentro de una cuenta existente mediante Kinesis Data Analytics. Si ya es cliente y desea crear una nueva cuenta con aplicaciones de Kinesis Data Analytics para SQL con `KinesisFirehoseInput`, puede abrir un caso de soporte. Para obtener más información, consulte el Centro de AWS Support.
- El tamaño de una fila de una secuencia en la aplicación se limita a 512 KB. Kinesis Data Analytics utiliza hasta 1 KB para almacenar metadatos. Los metadatos se tienen en cuenta para el límite de filas. Si el tamaño del registro del origen de flujo es superior a 50 KB, puede dividir el registro en varias filas en el flujo de la aplicación proporcionando el esquema adecuado en la configuración de entrada mediante el delimitador de filas.
- El código SQL de una aplicación se limita a 100 KB.
- El tiempo máximo que recomendamos para una consulta en ventana es de una hora. Las secuencias en la aplicación se guardan en un almacenamiento volátil. Si se producen interrupciones inesperadas de la aplicación, esta volverá a crear la secuencia a partir de los datos de origen del almacenamiento volátil.
- El rendimiento máximo que recomendamos para una sola secuencia en la aplicación es de entre 2 y 20 MB/segundo, dependiendo de la complejidad de la consulta de la aplicación.
- Puede crear hasta 50 aplicaciones de Kinesis Data Analytics AWS por región en su cuenta. Puede crear un caso para solicitar en aplicaciones adicionales a través de la solicitud de incremento del límite de servicio. Para obtener más información, consulte el Centro de AWS Support.

- El rendimiento máximo de transmisión que puede procesar una sola aplicación de Kinesis Data Analytics for SQL es de aproximadamente el MB/sec. This assumes that you have increased the number of in-application streams to the maximum value of 64, and you have increased your KPU limit beyond 8 (see the following limit for details). If your application needs to process more than 100 MB/sec 100% de la entrada. Siga uno de estos procedimientos:
  - Utilice varias instancias de aplicaciones de Kinesis Data Analytics para SQL para procesar la entrada
  - Utilice Managed Service for Apache Flink for Java Applications si desea seguir usando una sola secuencia y aplicación.

  > **Note**
  >
  > Le recomendamos revisar periódicamente la métrica `InputProcessing.0kBytes` de su aplicación para poder planificar con antelación el uso de varias aplicaciones SQL o migrar a Amazon Managed Service para Apache Flink en el caso de las aplicaciones Java si el rendimiento de entrada proyectado de su aplicación superará los 100 MB/seg. También le recomendamos que active una CloudWatch alarma `InputProcessing.0kBytes` para que se le notifique cuando su aplicación se acerque al límite de rendimiento de entrada. Esto puede resultar útil, ya que puede actualizar la consulta de la aplicación para compensar por un mayor rendimiento y, de este modo, evitar la contrapresión y los retrasos en los análisis. Para más información, consulte Solución de problemas. Esto también puede resultar útil si dispone de un mecanismo para reducir el rendimiento en las fases iniciales.

- El número de unidades de procesamiento de Kinesis (KPU) está limitado a ocho. Para obtener instrucciones sobre cómo solicitar un aumento de este límite, consulte Solicitar un aumento de límite en Amazon Service Limits.

  Con Kinesis Data Analytics, paga solo por los recursos que usa. Se le cobrará una tarifa por hora en función del número medio de horas KPUs que se utilice para ejecutar su aplicación de procesamiento de flujos. Una sola KPU le proporciona 1 vCPU y 4 GB de memoria.
- Cada aplicación puede tener un origen de streaming y hasta un origen de datos de referencia.
- Puede configurar hasta tres destinos para su aplicación de Kinesis Data Analytics. Le recomendamos que utilice uno de estos destinos para conservar los datos de la secuencia de errores en la aplicación.
- El objeto de Amazon S3 que almacena datos de referencia puede tener un tamaño de hasta 1 GB.

- Si cambias los datos de referencia que están almacenados en el depósito de S3 después de cargar los datos de referencia en una tabla de la aplicación, tendrás que utilizar la UpdateApplication operación (mediante la API o AWS CLI) para actualizar los datos de la tabla de la aplicación. Actualmente, Consola de administración de AWS no admite la actualización de los datos de referencia de tu aplicación.
- Actualmente, Kinesis Data Analytics no admite los datos generados por Amazon Kinesis Producer Library (KPL).
- Puede asignar hasta 50 etiquetas por aplicación.

Prácticas recomendadas

En esta sección se describen las prácticas recomendadas para trabajar con aplicaciones de Amazon Kinesis Data Analytics.

Temas

Administración de aplicaciones

Cuando administra aplicaciones de Amazon Kinesis Data Analytics, siga estas prácticas recomendadas:

- Configure CloudWatch las alarmas de Amazon: puede utilizar las CloudWatch métricas que proporciona Kinesis Data Analytics para supervisar lo siguiente:
  - Bytes de entrada y registros de entrada (cantidad de bytes y registros que se introducen en la aplicación)
  - Bytes de salida y registros de salida
  - `MillisBehindLatest` (el retraso que lleva la aplicación al leer en el origen de streaming)

  Le recomendamos que configure al menos dos CloudWatch alarmas en las siguientes métricas para sus aplicaciones en producción:
  - `MillisBehindLatest`: en la mayoría de los casos, le recomendamos que establezca esta alarma para que se dispare cuando su aplicación se encuentre 1 hora atrás de los datos más recientes, con una media de 1 minuto. Para las aplicaciones con menores necesidades end-to-end de procesamiento, puede ajustarlo a una tolerancia más baja. Esta alarma puede ayudar a garantizar que su aplicación está leyendo los datos más recientes.

- Para evitar la excepción `ReadProvisionedThroughputException`, limite la cantidad de aplicaciones de producción que leen de un mismo flujo de datos de Kinesis a dos aplicaciones.

  > **Note**
  >
  > En este caso, la aplicación se refiere a cualquier aplicación que pueda leer desde el origen de streaming. Solo una aplicación de Kinesis Data Analytics puede leer desde un flujo de entrega de Firehose. Sin embargo, muchas aplicaciones pueden leer desde una transmisión de datos de Kinesis, como una aplicación de Kinesis Data Analytics o. AWS Lambda El límite de aplicaciones recomendado se refiere a todas las aplicaciones que usted configura para que lean desde un origen de streaming.

  Amazon Kinesis Data Analytics lee un origen de streaming aproximadamente una vez por segundo por aplicación. Sin embargo, una aplicación que cae detrás podría leer datos más rápidamente para ponerse al corriente. Para permitir el rendimiento suficiente para que las aplicaciones se pongan al corriente, limite la cantidad de aplicaciones que leen el mismo origen de datos.

- Limite la cantidad de aplicaciones de producción que leen de un mismo flujo de entrega de Firehose a una aplicación.

  Un flujo de entrega de Firehose puede escribir en destinos como Amazon S3 y Amazon Redshift. También puede ser un origen de streaming para su aplicación de . Por lo tanto, le recomendamos que no configure más de una aplicación de Kinesis Data Analytics por flujo de entrega de Firehose. Esto ayuda a garantizar que la secuencia de entrega también se pueda entregar a otros destinos.

Escalado de aplicaciones

Configure la aplicación para sus necesidades de escalado futuras aumentando por adelantado el número de secuencias en la aplicación de entrada en un valor superior a uno (el valor predeterminado). Le recomendamos las siguientes opciones del lenguaje en función del rendimiento de la aplicación:

- Utilice varias secuencias e instancias de aplicaciones de Kinesis Data Analytics para SQL si su aplicación tiene necesidades de escalado superiores a 100 MB/segundo.
- Utilice Managed Service para Apache Flink Applications si desea utilizar un solo flujo y una sola aplicación.

> **Note**
>
> Le recomendamos revisar periódicamente la `InputProcessing.0kBytes` métrica de su aplicación para poder planificar con antelación el uso de varias aplicaciones de SQL o la migración a aplicaciones gestionadas. flink/latest/java/ if your application's projected input throughput exceeds 100 MB/sec

Monitorización de aplicaciones

Le recomendamos que active una CloudWatch alarma `InputProcessing.0kBytes` para que se le notifique cuando su aplicación se acerque al límite de rendimiento de entrada. Esto puede resultar útil, ya que puede actualizar la consulta de la aplicación para compensar por un mayor rendimiento y, de este modo, evitar la contrapresión y los retrasos en los análisis. Para más información, consulte Solución de problemas. Esto también puede resultar útil si dispone de un mecanismo para reducir el rendimiento en las fases iniciales.

- El rendimiento máximo que recomendamos para una sola secuencia en la aplicación es de entre 2 y 20 MB/segundo, dependiendo de la complejidad de la consulta de la aplicación.
- El rendimiento máximo de flujo que puede procesar una sola aplicación de Kinesis Data Analytics para SQL es de aproximadamente 100 MB/seg. Esto supone que ha aumentado el número de flujos en la aplicación hasta un valor máximo de 64 y que ha aumentado su límite de KPU más allá de 8. Para obtener más información, consulte Límites.

> **Note**
>
> Le recomendamos revisar periódicamente la `InputProcessing.0kBytes` métrica de su aplicación para poder planificar con antelación el uso de varias aplicaciones SQL o la migración a aplicaciones gestionadas. flink/latest/java/ if your application's projected input throughput exceeds 100 MB/sec

Definición de esquemas de entrada

Cuando configure la entrada de aplicaciones en la consola, primero especifique un origen de streaming. Entonces, la consola utiliza la API de detección (consulte DiscoverInputSchema) para inferir un esquema por registros de muestreo en el origen de streaming. El esquema define, entre otras cosas, los nombres y los tipos de datos de las columnas en la secuencia en la aplicación resultante. La consola muestra el esquema. Le recomendamos que haga lo siguiente con este esquema inferido:

- Pruebe el esquema inferido cuanto sea necesario. El proceso de detección solo utiliza una muestra de registros en el origen de streaming para inferir en un esquema. Si el origen de streaming tiene muchos tipos de registro, la API de detección podría no haber atendido uno o varios tipos de registro. Esta situación puede dar lugar a un esquema que no refleja los datos sobre el origen de streaming de manera precisa.

  Cuando se inicia la aplicación, estos tipos de registro perdidos podrían dar lugar a errores de análisis. Amazon Kinesis Data Analytics envía estos registros a la secuencia de errores en la aplicación. Para reducir estos errores de análisis, le recomendamos que pruebe el esquema inferido de forma interactiva en la consola y monitoree la secuencia en la aplicación para registros no atendidos.

- La API de Kinesis Data Analytics no permite especificar la restricción de `NOT NULL` en columnas en la configuración de entrada. Si desea restricciones `NOT NULL` en columnas en su secuencia en la aplicación, cree estas secuencias en la aplicación utilizando el código de su aplicación. A continuación, puede copiar los datos de una secuencia en la aplicación en otra y, luego, se aplica la restricción.

  Todo intento de insertar filas con valores `NULL` cuando se requiere un valor, da lugar a un error. Kinesis Data Analytics envía estos errores a la secuencia de errores en la aplicación.

- Relaje los tipos de datos inferidos por el proceso de detección. El proceso de detección recomienda columnas y tipos de datos basados en una muestra aleatoria de registros en el origen de streaming. Le recomendamos que los revise detenidamente y considere flexibilizar estos tipos de datos para cubrir todos los posibles casos de registros en la entrada. Esto garantiza menos errores de análisis a través de la aplicación mientras se está ejecutando. Por ejemplo, si un esquema inferido tiene un `SMALLINT` como tipo de columna, plantéese cambiarlo a `INTEGER`.

- Utilice las funciones de SQL en el código de su aplicación para administrar los datos o las columnas que no tienen estructura. Es posible que haya datos columnas que no tengan estructura, como datos de registro, en la entrada. Para ver ejemplos, consulte Ejemplo: transformación DateTime de valores. Un enfoque de la gestión de este tipo de datos para definir el esquema con una sola columna del tipo `VARCHAR(N)`, donde `N` es la fila más alta posible que esperaría ver en su secuencia. En el código de aplicación puede leer los registros de entrada y utilizar las funciones `String` y `Date Time` para analizar y esquematizar los datos sin procesar.

- Asegúrese de gestionar completamente los datos de origen de streaming que contengan anidaciones de más de dos niveles de profundidad. Cuando los datos de origen son JSON, puede realizar el anidamiento. La API de detección infiere un esquema que aplana un nivel de anidamiento. Para dos niveles del anidamiento, la API de detección también intenta aplanarlos. Además de dos niveles del anidamiento, no hay soporte limitado para aplanamiento. Para gestionar las anidaciones por completo, tiene que modificar el esquema inferido manualmente para adaptarlo a sus necesidades. Utilice una de las siguientes estrategias para hacerlo:

  - Utilice la ruta de la fila JSON para extraer, de manera selectiva, únicamente los pares de valores de claves necesarios para su aplicación. Una ruta de la fila JSON proporciona un puntero al par de valores clave específicos que desee utilizar en su aplicación. Puede hacerlo para cualquier nivel de anidamiento.
  - Utilice la ruta de la fila JSON para extraer, de manera selectiva, objetos JSON complejos y, a continuación, utilizar funciones de manipulación de cadenas en el código de su aplicación para extraer los datos específicos que necesita.

Conexión a salidas

Le recomendamos que cada aplicación tenga al menos dos salidas:

- Utilice el primer destino para insertar los resultados de sus consultas SQL.
- Utilice el segundo destino para insertar todo el flujo de error y enviarlo a un bucket de S3 a través de un flujo de entrega de Firehose.

Creación del código de la aplicación

Le recomendamos lo siguiente:

- En la instrucción SQL, no especifique una ventana basada en tiempo que sea superior a una hora por las siguientes razones:
  - A veces, una aplicación tiene que reiniciarse, ya sea porque actualizó la aplicación o por razones internas de . Cuando se reinicia, se deben volver a leer todos los datos incluidos en la ventana desde el origen de datos de streaming. Esto lleva tiempo hasta que Kinesis Data Analytics pueda emitir la salida de dicha ventana.
  - Kinesis Data Analytics debe mantener todo lo relacionado con el estado de la aplicación, incluidos los datos relevantes, todo el tiempo. Esto consume una cantidad significativa de unidades de procesamiento de Kinesis Data Analytics.
- Durante el desarrollo, mantenga el tamaño de la ventana pequeño en sus instrucciones SQL para poder ver los resultados con mayor rapidez. Al implementar la aplicación en su entorno de producción, puede establecer el tamaño de la ventana, según corresponda.
- En lugar de una única instrucción SQL compleja, plantéese dividirla en varias y guardar los resultados de cada paso en secuencias en la aplicación intermedias. Esto puede ayudarle a realizar la depuración con mayor rapidez.
- Cuando se utilizan ventanas de saltos de tamaño constante, le recomendamos que utilice dos ventanas: una para el tiempo de procesamiento y otra para el tiempo lógico (tiempo de ingestión o de evento). Para obtener más información, consulte Marcas temporales y la comuna ROWTIME.

Prueba de aplicaciones

Cuando cambie el esquema o código de aplicación de su aplicación de , le recomendamos que utilice una aplicación de prueba para verificar los cambios antes de implementarlos en producción.

Configuración de una aplicación de prueba

Es posible configurar una aplicación de prueba bien a través de la consola o utilizando una plantilla de CloudFormation . El uso de una CloudFormation plantilla ayuda a garantizar que los cambios de código que realice en la aplicación de prueba y en la aplicación activa sean coherentes.

Al configurar una aplicación de prueba, puede conectar la aplicación a sus datos en vivo o puede rellenar una secuencia con datos simulados para probarlos. Le recomendamos dos métodos para rellenar una secuencia con datos simulados:

- Utilice el Kinesis Data Generator (KDG). El KDG utiliza una plantilla de datos para enviar datos aleatorios a una secuencia de Kinesis. El KDG es fácil de utilizar, pero no es adecuado para probar relaciones complejas entre elementos de datos como, por ejemplo, para aplicaciones que detectan puntos calientes de datos o anomalías.
- Utilice una aplicación Python personalizada para enviar los datos más complejos a un flujo de datos de . Una aplicación Python puede generar relaciones complejas entre elementos de datos, como, por ejemplo, puntos calientes o anomalías. Para ver un ejemplo de una aplicación Python que envía datos agrupados en un punto caliente de datos, consulte Ejemplo: Detección de puntos calientes en una secuencia (función HOTSPOSTS).

Al ejecutar la aplicación de prueba, vea los resultados utilizando un destino (como, por ejemplo, un flujo de entrega de Firehose a una base de datos de Amazon Redshift) en lugar de ver el flujo de la aplicación en la consola. Los datos que se muestran en la consola son una muestra de la secuencia y no contienen todos los registros.

Prueba de cambios de esquema

Al cambiar el esquema de la secuencia de entrada de una aplicación, utilice su aplicación de prueba para comprobar que lo siguiente es verdadero:

- Los datos de su secuencia se están convirtiendo al tipo de datos correcto. Por ejemplo, asegúrese de que los datos de fecha y hora no se introduzcan en la aplicación como cadena.
- Los datos se están analizando y se convierten al tipo de datos que desea. Si se producen errores de introducción o conversión, puede verlos en la consola o asignar un destino a la secuencia de error y ver los errores en el almacén de destino.
- Los campos de datos para los datos de caracteres tienen la longitud suficiente y la aplicación no trunca los datos de caracteres. Puede comprobar los registros de datos en su almacén de destino para comprobar que los datos de la aplicación no se truncan.

Prueba de cambios de código

La prueba de cambios en su código SQL requiere algunos conocimientos de dominio de su aplicación. Debe poder determinar qué salida se tiene que probar y cuál sería la salida correcta. Para

ver las posibles áreas problemáticas que verificar la hora de modificar el código SQL de la aplicación, consulte Solución de problemas de aplicaciones de Amazon Kinesis Data Analytics para SQL.

Solución de problemas de aplicaciones de Amazon Kinesis Data Analytics para SQL

La información siguiente le puede ayudar a solucionar los problemas que puedan presentarse con aplicaciones de Amazon Kinesis Data Analytics para SQL.

Temas

Aplicaciones detenidas

- ¿Qué es una aplicación de Kinesis Data Analytics para SQL detenida?

  Una aplicación detenida es una aplicación que hemos observado que no procesa ningún registro durante un mínimo de tres meses. Esto significa que los clientes pagan por los recursos de Kinesis Data Analytics para SQL que no están utilizando.

- ¿Cuándo AWS empezará a detener las aplicaciones inactivas?

  AWS empezará a detener las aplicaciones inactivas el 14 de noviembre de 2023 y finalizará el 21 de noviembre de 2023. Detendremos las aplicaciones inactivas en el horario de oficina de la zona horaria de la región.

- ¿Se pueden reiniciar las aplicaciones de Kinesis Data Analytics para SQL detenidas?

  Sí. Si necesita reiniciar la aplicación, puede hacerlo de la forma habitual. No es necesario que envíe un ticket de soporte.

- Cuando AWS se detenga una aplicación inactiva, ¿se eliminará también alguno de los resultados de mi consulta?

  No. En primer lugar, dado que la aplicación está inactiva, no procesa las consultas. En segundo lugar, los resultados de la consulta no se almacenan en Kinesis Data Analytics para SQL. Configure la aplicación de Kinesis Data Analytics para SQL con un destino receptor al que se envíen los resultados de sus cálculos (por ejemplo, en Amazon S3 u otro flujo de datos). Por lo tanto, usted conserva la plena propiedad de sus datos y seguirán siendo recuperables según las condiciones de ese servicio de almacenamiento.
- ¿Qué sucede si no quiero que se detenga mi solicitud?

  Puede enviar un correo electrónico al equipo de servicio (kda-sql-questions@amazon .com) solicitando que las solicitudes no se detengan en ningún momento antes del 10 de noviembre de 2023. El correo electrónico debe incluir su ID de cuenta y el ARN de la aplicación.

No se puede ejecutar el código SQL

Si quiere saber cómo conseguir que una instrucción SQL concreta funcione correctamente, dispone de varios recursos cuando utilice Kinesis Data Analytics::

- Para obtener más información sobre las instrucciones SQL, consulte Ejemplos de Kinesis Data Analytics para SQL. En esta sección se proporcionan varios ejemplos de SQL que pueden ser de utilidad.
- La Referencia de SQL de Amazon Kinesis Data Analytics proporciona una guía detallada para la creación de instrucciones SQL de streaming.
- Si sigue teniendo problemas, le aconsejamos que publique su pregunta en los foros de Kinesis Data Analytics.

No se puede detectar mi esquema

En algunos casos, Kinesis Data Analytics no detecta ni descubre ningún esquema. En muchos de estos casos, puede seguir utilizando Kinesis Data Analytics.

Supongamos que tiene datos codificados en UTF-8 sin delimitador o datos que utilizan un formato diferente a los valores separados por comas (CSV), o que la API de detección no ha determinado el esquema. En estos casos, puede definir un esquema de forma manual o utilizar funciones de manipulación de cadenas para estructurar los datos.

Para descubrir el esquema de la secuencia, Kinesis Data Analytics muestra de forma aleatoria los datos más recientes de la secuencia. Si no envía de forma coherente datos a su secuencia, Kinesis Data Analytics no podrá recuperar un fragmento ni detectar un esquema. Para obtener más información, consulte Uso de la función de detección de esquema en datos de streaming.

Los datos de referencia no están actualizados

Los datos de referencia se cargan desde el objeto de Amazon Simple Storage Service (Amazon S3) en la aplicación al iniciarla o actualizarla, o bien durante las interrupciones en la aplicación debidas a problemas con los servicios.

Los datos de referencia no se cargan en la aplicación cuando se realizan actualizaciones en el objeto de Amazon S3 subyacente.

Si los datos de referencia de la aplicación no están actualizados, puede volver a cargarlos siguiendo estos pasos:

1. En la consola de Kinesis Data Analytics, elija el nombre de la aplicación en la lista y, a continuación, elija Detalles de la aplicación.
2. Elija Go to SQL editor (Ir al editor de SQL) para abrir la página Real-time analytics (Análisis en tiempo real) de la aplicación.
3. En la vista Source Data (Datos de origen), elija el nombre de la tabla de datos de referencia.
4. Elija Actions (Acciones), Synchronize reference data table (Sincronizar tabla de datos de referencia).

La aplicación no escribe en el destino

Si los datos no se escriben en el destino, compruebe lo siguiente:

- Verifique que el rol de la aplicación tenga suficientes permisos para obtener acceso al destino. Para obtener más información, consulte Política de permisos para escribir en una secuencia de Kinesis o Política de permisos para escribir en una secuencia de entrega de Firehose.
- Compruebe que el destino de la aplicación está configurado correctamente y que la aplicación utiliza el nombre correcto para el flujo de salida.
- Comprueba las CloudWatch métricas de Amazon de tu flujo de salida para ver si se están grabando datos. Para obtener información sobre el uso de CloudWatch las métricas, consultaMonitorización con Amazon CloudWatch.

- Agregue un flujo de CloudWatch registro mediantethe section called "AddApplicationCloudWatchLoggingOption". La aplicación escribirá los errores de configuración en la secuencia de registros.

Si la configuración de roles y destinos parece correcta, intente reiniciar la aplicación, especificando `LAST_STOPPED_POINT` para InputStartingPositionConfiguration.

Parámetros de salud de aplicaciones importantes para supervisar

Para asegurarse de que su aplicación se ejecuta adecuadamente, le recomendamos que supervise determinados parámetros importantes.

El parámetro más importante a monitorizar es la CloudWatch métrica de Amazon`MillisBehindLatest`. Esta métrica representa el retraso del horario actual que está leyendo de la secuencia. Esta métrica le ayuda a determinar si está procesando registros de la secuencia de origen lo suficientemente rápido.

Como regla general, debes configurar una CloudWatch alarma para que se active si te retrasas más de una hora. No obstante, la cantidad de tiempo depende del caso de uso. Puede ajustarlo como sea necesario.

Para obtener más información, consulte Prácticas recomendadas.

Errores de códigos no válidos cuando se ejecuta una aplicación

Si no puede guardar y ejecutar el código SQL de la aplicación de Amazon Kinesis Data Analytics estas suelen ser las causas habituales:

- La secuencia se ha redefinido en el código SQL: después de crear una secuencia y la bomba asociada a esta, no se puede redefinir la misma secuencia en el código. Para obtener más información sobre la creación de una secuencia, consulte CREATE STREAM en la Referencia de SQL de Amazon Kinesis Data Analytics. Para obtener más información sobre la creación de una bomba, consulte CREATE PUMP.
- Una cláusula GROUP BY utiliza varias columnas ROWTIME: puede especificar una única columna ROWTIME en la cláusula GROUP BY. Para obtener más información, consulte GROUP BY y ROWTIME en la Referencia de SQL de Amazon Kinesis Data Analytics.

- Uno o varios tipos de datos tienen una conversión no válida: en este caso, el código ha emitido una conversión implícita que no es válida. Por ejemplo, es posible que esté emitiendo una `timestamp` a un `bigint` en el código.
- Una secuencia tiene el mismo nombre que la secuencia reservada por el servicio: una secuencia no puede tener el mismo nombre que la secuencia `error_stream` reservada por el servicio.

La aplicación escribe errores en la secuencia de errores

Si su aplicación escribe errores en el flujo de errores en la aplicación, puede descodificar el valor del campo `DATA_ROW` utilizando las bibliotecas estándar. Para obtener más información acerca de la secuencia de errores, consulte Gestión de errores.

Rendimiento insuficiente o alto MillisBehindLatest

Si la MillisBehindLatestmétrica de su aplicación aumenta constantemente o está por encima de 1000 (un segundo), puede deberse a las siguientes razones:

- Compruebe la InputBytes CloudWatch métrica de su aplicación. Si está adquiriendo más de 4 MB/s, esto puede provocar un aumento de `MillisBehindLatest`. Para mejorar el desempeño de la aplicación, aumente el valor del parámetro `InputParallelism`. Para obtener más información, consulte Paralelizar secuencias de entrada para mejorar el desempeño.
- Compruebe la métrica Success de la entrega de salida de la aplicación para ver si hay errores de entrega en el destino. Compruebe que ha configurado correctamente la salida, y que la secuencia de salida tiene capacidad suficiente.
- Si la aplicación utiliza una AWS Lambda función para el preprocesamiento o como salida, compruebe la métrica InputProcessing.Duration o LambdaDelivery CloudWatch .Duration de la aplicación. Si la duración de invocación de la función de Lambda es superior a 5 segundos, considere la posibilidad de hacer lo siguiente:
  - Aumentar la asignación de Memoria de la función de Lambda. Puede hacerlo en la consola de AWS Lambda , en la página Configuration (Configuración), en Basic settings (Configuración básica). Para obtener más información, consulte Configuración de funciones de Lambda en la Guía para desarrolladores de AWS Lambda .
  - Aumentar el número de fragmentos de la secuencia de entrada de la aplicación. Esto aumenta el número de funciones que la aplicación invocará en paralelo, lo que podría aumentar el desempeño.

  - Comprobar que la función no está realizando llamadas de bloqueo que afecten al desempeño, como, por ejemplo, solicitudes síncronas de recursos externos.
  - Examine su AWS Lambda función para ver si hay otras áreas en las que pueda mejorar el rendimiento. Compruebe los CloudWatch registros de la función Lambda de la aplicación. Para obtener más información, consulte Acceder a Amazon CloudWatch Metrics en la Guía para AWS Lambda desarrolladores.
- Compruebe que la aplicación no llega al límite predeterminado de unidades de procesamiento de Kinesis (KPU). Si la aplicación alcanza este límite, puede solicitar un aumento del límite. Para obtener más información, consulte Escalado automático de aplicaciones para incrementar el desempeño.
- Si su aplicación sigue teniendo problemas después de aumentar el límite de KPU, compruebe que el rendimiento de entrada de la aplicación no supere los 100MB/sec. If it exceeds 100MB/sec. Le recomendamos implementar cambios para reducir el rendimiento general a fin de estabilizar la aplicación, por ejemplo, reduciendo la cantidad de datos que se envían a la fuente de datos de la que lee la aplicación Sql de Kinesis Data Analytics. También recomendamos otros enfoques, como aumentar el paralelismo de la aplicación, reducir el período de tiempo de los cálculos, cambiar los tipos de datos en columnas de VARCHAR a tipos de datos con tamaños más pequeños (por ejemplo, INTEGER, LONG, etc.) y reducir los datos procesados mediante muestreo o filtrado.

> **Note**
>
> Le recomendamos revisar periódicamente la `InputProcessing.0kBytes` métrica de su aplicación para poder planificar con antelación el uso de varias aplicaciones SQL o la migración a aplicaciones gestionadas. flink/latest/java/ if your application's projected input throughput will exceed 100 MB/sec

Referencia de SQL de Amazon Kinesis Data Analytics

Para obtener información sobre los elementos del lenguaje SQL compatibles con Kinesis Data Analytics, consulte Referencia de SQL de Kinesis Data Analytics.

referencia de la API

> **Note**
>
> Esta documentación es para la versión 1 de la API de Amazon Kinesis Data Analytics, que solo admite aplicaciones SQL. La versión 2 de la API admite aplicaciones SQL y Java. Para obtener más información sobre la versión 2, consulte la documentación de Amazon Managed Service para Apache Flink API V2.

Puede utilizarla AWS CLI para explorar la API de Amazon Kinesis Data Analytics. En esta guía se proporcionan ejercicios Introducción a aplicaciones de Amazon Kinesis Data Analytics para SQL que utilizan la AWS CLI.

Temas

Acciones

Se admiten las siguientes acciones:

- DescribeApplication
- DiscoverInputSchema
- ListApplications
- ListTagsForResource
- StartApplication
- StopApplication
- TagResource
- UntagResource
- UpdateApplication

AddApplicationCloudWatchLoggingOption

> **Note**
>
> Esta documentación es para la versión 1 de la API de Amazon Kinesis Data Analytics, que solo admite aplicaciones SQL. La versión 2 de la API admite aplicaciones SQL y Java. Para obtener más información acerca de la versión 2, consulte la documentación de la API V2 de Amazon Kinesis Data Analytics.

Añade un flujo de CloudWatch registro para supervisar los errores de configuración de la aplicación. Para obtener más información sobre el uso de transmisiones de CloudWatch registros con las aplicaciones de Amazon Kinesis Analytics, consulte Trabajar con CloudWatch Amazon Logs.

Sintaxis de la solicitud

```
{
   "ApplicationName": "string",
   "CloudWatchLoggingOption": {
      "LogStreamARN": "string",
      "RoleARN": "string"
   },
   "CurrentApplicationVersionId": number
}
```

Parámetros de la solicitud

La solicitud acepta los siguientes datos en formato JSON.

ApplicationName

El nombre de la aplicación de Kinesis Analytics.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. Longitud máxima de 128.

Patrón: `[a-zA-Z0-9_.-]+`

Obligatorio: sí

CloudWatchLoggingOption

Proporciona el nombre de recurso de Amazon (ARN) del flujo de CloudWatch registro y el ARN del rol de IAM. Nota: Para escribir los mensajes de la aplicación CloudWatch, la función de IAM que se utilice debe tener habilitada la `PutLogEvents` acción política.

Tipo: objeto CloudWatchLoggingOption

Obligatorio: sí

CurrentApplicationVersionId

El ID de versión de la aplicación Kinesis Analytics.

Tipo: largo

Rango válido: valor mínimo de 1. Valor máximo de 999999999.

Obligatorio: sí

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta HTTP 200 con un cuerpo HTTP vacío.

Errores

ConcurrentModificationException

Se produce una excepción como resultado de una modificación simultánea de una aplicación. Por ejemplo, dos personas que intentan editar la misma aplicación al mismo tiempo.

message

Código de estado HTTP: 400

InvalidArgumentException

El valor del parámetro de entrada especificado no es válido.

message

Código de estado HTTP: 400

ResourceInUseException

La aplicación no está disponible para esta operación.

message

Código de estado HTTP: 400

ResourceNotFoundException

No se encuentra la aplicación especificada.

message

Código de estado HTTP: 400

UnsupportedOperationException

La solicitud se rechazó porque no se admite un parámetro especificado o porque un recurso especificado no es válido para esta operación.

Código de estado HTTP: 400

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS Interfaz de línea de comandos V2
- AWS SDK para .NET V4
- AWS SDK para C++
- AWS SDK para Go v2
- AWS SDK para Java V2
- AWS SDK para JavaScript V3
- AWS SDK para Kotlin
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK para Ruby V3

AddApplicationInput

> **Note**
>
> Esta documentación es para la versión 1 de la API de Amazon Kinesis Data Analytics, que solo admite aplicaciones SQL. La versión 2 de la API admite aplicaciones SQL y Java. Para obtener más información acerca de la versión 2, consulte la documentación de la API V2 de Amazon Kinesis Data Analytics.

Añade un origen de flujo a la aplicación Amazon Kinesis. Para obtener información conceptual, consulte Configuración de entrada de la aplicación.

Puede agregar un origen de flujo al crear una aplicación o puede usar esta operación para agregar un origen de flujo después de crear una aplicación. Para obtener más información, consulte CreateApplication.

Cualquier actualización de la configuración, incluida la adición de un origen de transmisión con esta operación, da lugar a una nueva versión de la aplicación. Puede utilizar la operación DescribeApplication para buscar la versión actual de la aplicación.

Esto operación requiere permisos para realizar la acción `kinesisanalytics:AddApplicationInput`.

Sintaxis de la solicitud

```
{
   "ApplicationName": "string",
   "CurrentApplicationVersionId": number,
   "Input": {
      "InputParallelism": {
         "Count": number
      },
      "InputProcessingConfiguration": {
         "InputLambdaProcessor": {
            "ResourceARN": "string",
            "RoleARN": "string"
         }
      },
      "InputSchema": {
         "RecordColumns": [
```

```
            {
               "Mapping": "string",
               "Name": "string",
               "SqlType": "string"
            }
         ],
         "RecordEncoding": "string",
         "RecordFormat": {
            "MappingParameters": {
               "CSVMappingParameters": {
                  "RecordColumnDelimiter": "string",
                  "RecordRowDelimiter": "string"
               },
               "JSONMappingParameters": {
                  "RecordRowPath": "string"
               }
            },
            "RecordFormatType": "string"
         }
      },
      "KinesisFirehoseInput": {
         "ResourceARN": "string",
         "RoleARN": "string"
      },
      "KinesisStreamsInput": {
         "ResourceARN": "string",
         "RoleARN": "string"
      },
      "NamePrefix": "string"
   }
}
```

Parámetros de la solicitud

La solicitud acepta los siguientes datos en formato JSON.

ApplicationName

Nombre de la aplicación Amazon Kinesis Analytics existente a la que desea añadir el origen de flujo.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. Longitud máxima de 128.

Patrón: `[a-zA-Z0-9_.-]+`

Obligatorio: sí

CurrentApplicationVersionId

Versión actual de la aplicación Amazon Kinesis Analytics. Puede utilizar la operación DescribeApplication para buscar la versión actual de la aplicación.

Tipo: largo

Rango válido: valor mínimo de 1. Valor máximo de 999999999.

Obligatorio: sí

Input

La entrada que se va a añadir.

Tipo: objeto Input

Obligatorio: sí

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta HTTP 200 con un cuerpo HTTP vacío.

Errores

CodeValidationException

El código de aplicación (consulta) proporcionado por el usuario no es válido. Puede tratarse de un simple error de sintaxis.

message

Test

Código de estado HTTP: 400

ConcurrentModificationException

Se produce una excepción como resultado de una modificación simultánea de una aplicación. Por ejemplo, dos personas que intentan editar la misma aplicación al mismo tiempo.

message

Código de estado HTTP: 400

InvalidArgumentException

El valor del parámetro de entrada especificado no es válido.

message

Código de estado HTTP: 400

ResourceInUseException

La aplicación no está disponible para esta operación.

message

Código de estado HTTP: 400

ResourceNotFoundException

No se encuentra la aplicación especificada.

message

Código de estado HTTP: 400

UnsupportedOperationException

La solicitud se rechazó porque no se admite un parámetro especificado o porque un recurso especificado no es válido para esta operación.

Código de estado HTTP: 400

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS Interfaz de línea de comandos V2
- AWS SDK para .NET V4
- AWS SDK para C++

- AWS SDK para Go v2
- AWS SDK para Java V2
- AWS SDK para JavaScript V3
- AWS SDK para Kotlin
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK para Ruby V3

AddApplicationInputProcessingConfiguration

> **Note**
>
> Esta documentación es para la versión 1 de la API de Amazon Kinesis Data Analytics, que solo admite aplicaciones SQL. La versión 2 de la API admite aplicaciones SQL y Java. Para obtener más información acerca de la versión 2, consulte la documentación de la API V2 de Amazon Kinesis Data Analytics.

Añade un InputProcessingConfigurationa una aplicación. Un procesador de entrada preprocesa los registros en el flujo de entrada antes de la ejecución del código SQL de la aplicación. Actualmente, el único procesador de entrada disponible es AWS Lambda.

Sintaxis de la solicitud

```
{
   "ApplicationName": "string",
   "CurrentApplicationVersionId": number,
   "InputId": "string",
   "InputProcessingConfiguration": {
      "InputLambdaProcessor": {
         "ResourceARN": "string",
         "RoleARN": "string"
      }
   }
}
```

Parámetros de la solicitud

La solicitud acepta los siguientes datos en formato JSON.

ApplicationName

Nombre de la aplicación a la que desea añadir la configuración de procesamiento de salida.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. Longitud máxima de 128.

Patrón: `[a-zA-Z0-9_.-]+`

Obligatorio: sí

CurrentApplicationVersionId

Versión de la aplicación a la que desea añadir la configuración de procesamiento de salida. Puede utilizar la operación DescribeApplication para obtener la versión actual de la aplicación. Si la versión especificada no es la actual, se devuelve la `ConcurrentModificationException`.

Tipo: largo

Rango válido: valor mínimo de 1. Valor máximo de 999999999.

Obligatorio: sí

InputId

El ID de la configuración de entrada a la que se va a añadir la configuración de procesamiento de entrada. Puede obtener una lista de las entradas IDs de una aplicación mediante la DescribeApplicationoperación.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 50 caracteres.

Patrón: `[a-zA-Z0-9_.-]+`

Obligatorio: sí

InputProcessingConfiguration

La InputProcessingConfiguration que se va a agregar a la aplicación.

Tipo: objeto InputProcessingConfiguration

Obligatorio: sí

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta HTTP 200 con un cuerpo HTTP vacío.

Errores

ConcurrentModificationException

Se produce una excepción como resultado de una modificación simultánea de una aplicación. Por ejemplo, dos personas que intentan editar la misma aplicación al mismo tiempo.

message

Código de estado HTTP: 400

InvalidArgumentException

El valor del parámetro de entrada especificado no es válido.

message

Código de estado HTTP: 400

ResourceInUseException

La aplicación no está disponible para esta operación.

message

Código de estado HTTP: 400

ResourceNotFoundException

No se encuentra la aplicación especificada.

message

Código de estado HTTP: 400

UnsupportedOperationException

La solicitud se rechazó porque no se admite un parámetro especificado o porque un recurso especificado no es válido para esta operación.

Código de estado HTTP: 400

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS Interfaz de línea de comandos V2
- AWS SDK para .NET V4
- AWS SDK para C++
- AWS SDK para Go v2
- AWS SDK para Java V2
- AWS SDK para JavaScript V3
- AWS SDK para Kotlin
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK para Ruby V3

AddApplicationOutput

> **Note**
>
> Esta documentación es para la versión 1 de la API de Amazon Kinesis Data Analytics, que solo admite aplicaciones SQL. La versión 2 de la API admite aplicaciones SQL y Java. Para obtener más información acerca de la versión 2, consulte la documentación de la API V2 de Amazon Kinesis Data Analytics.

Añade un destino externo a su aplicación de Amazon Kinesis Analytics.

Si desea que Amazon Kinesis Analytics entregue datos desde una transmisión dentro de la aplicación a un destino externo (como una transmisión de Amazon Kinesis, una transmisión de entrega de Amazon Kinesis Firehose o una función de AWS Lambda), añada la configuración correspondiente a su aplicación mediante esta operación. Puede configurar una o más salidas para su aplicación. Cada configuración de salida asigna un flujo en la aplicación y un destino externo.

Puede utilizar una de las configuraciones de salida para entregar datos desde su flujo de errores en la aplicación a un destino externo para poder analizar los errores. Para obtener más información, consulte la sección sobre Descripción de salida de la aplicación (destino).

Cualquier actualización de la configuración, incluida la adición de un origen de transmisión con esta operación, da lugar a una nueva versión de la aplicación. Puede utilizar la operación DescribeApplication para buscar la versión actual de la aplicación.

Para conocer los límites en relación con el número de entradas y salidas de la aplicación que puede configurar, consulte Límites.

Esto operación requiere permisos para realizar la acción `kinesisanalytics:AddApplicationOutput`.

Sintaxis de la solicitud

```
{
   "ApplicationName": "string",
   "CurrentApplicationVersionId": number,
   "Output": {
      "DestinationSchema": {
         "RecordFormatType": "string"
```

```
      },
      "KinesisFirehoseOutput": {
         "ResourceARN": "string",
         "RoleARN": "string"
      },
      "KinesisStreamsOutput": {
         "ResourceARN": "string",
         "RoleARN": "string"
      },
      "LambdaOutput": {
         "ResourceARN": "string",
         "RoleARN": "string"
      },
      "Name": "string"
   }
}
```

Parámetros de la solicitud

La solicitud acepta los siguientes datos en formato JSON.

ApplicationName

Nombre de la aplicación a la que desea añadir la configuración de salida.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. Longitud máxima de 128.

Patrón: `[a-zA-Z0-9_.-]+`

Obligatorio: sí

CurrentApplicationVersionId

Versión de la aplicación a la que desea añadir la configuración de salida. Puede utilizar la operación DescribeApplication para obtener la versión actual de la aplicación. Si la versión especificada no es la actual, se devuelve la `ConcurrentModificationException`.

Tipo: largo

Rango válido: valor mínimo de 1. Valor máximo de 999999999.

Obligatorio: sí

Output

Una gama de objetos, cada una describe una configuración de salida. En la configuración de salida, debe especificar el nombre de una transmisión dentro de la aplicación, un destino (es decir, una transmisión de Amazon Kinesis, una transmisión de entrega de Amazon Kinesis Firehose o AWS una función de Lambda) y registrar la formación que se utilizará al escribir en el destino.

Tipo: objeto Output

Obligatorio: sí

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta HTTP 200 con un cuerpo HTTP vacío.

Errores

ConcurrentModificationException

Se produce una excepción como resultado de una modificación simultánea de una aplicación. Por ejemplo, dos personas que intentan editar la misma aplicación al mismo tiempo.

message

Código de estado HTTP: 400

InvalidArgumentException

El valor del parámetro de entrada especificado no es válido.

message

Código de estado HTTP: 400

ResourceInUseException

La aplicación no está disponible para esta operación.

message

Código de estado HTTP: 400

ResourceNotFoundException

No se encuentra la aplicación especificada.

message

Código de estado HTTP: 400

UnsupportedOperationException

La solicitud se rechazó porque no se admite un parámetro especificado o porque un recurso especificado no es válido para esta operación.

Código de estado HTTP: 400

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas AWS SDKs específicos, consulte lo siguiente:

- AWS Interfaz de línea de comandos V2
- AWS SDK para .NET V4
- AWS SDK para C++
- AWS SDK para Go v2
- AWS SDK para Java V2
- AWS SDK para JavaScript V3
- AWS SDK para Kotlin
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK para Ruby V3

AddApplicationReferenceDataSource

> **Note**
>
> Esta documentación es para la versión 1 de la API de Amazon Kinesis Data Analytics, que solo admite aplicaciones SQL. La versión 2 de la API admite aplicaciones SQL y Java. Para obtener más información acerca de la versión 2, consulte la documentación de la API V2 de Amazon Kinesis Data Analytics.

Añade un origen de datos de referencia a la aplicación existente.

Amazon Kinesis Analytics lee los datos de referencia (es decir, un objeto de Amazon S3) y crea una tabla en la aplicación dentro de la aplicación. En la solicitud, debe proporcionar el origen (el nombre de la clave de objeto y el nombre del bucket de S3), el nombre de la tabla en la aplicación que se va a crear y la información de mapeo necesaria que describe cómo los datos en el objeto de Amazon S3 se correlacionan con las columnas de la tabla en la aplicación resultante.

Para obtener información conceptual, consulte Configuración de entrada de la aplicación. Para conocer los límites respecto a los orígenes de datos que puede añadir a su aplicación, consulte Límites.

Esto operación requiere permisos para realizar la acción `kinesisanalytics:AddApplicationOutput`.

Sintaxis de la solicitud

```
{
   "ApplicationName": "string",
   "CurrentApplicationVersionId": number,
   "ReferenceDataSource": {
      "ReferenceSchema": {
         "RecordColumns": [
            {
               "Mapping": "string",
               "Name": "string",
               "SqlType": "string"
            }
         ],
         "RecordEncoding": "string",
```

```
            "RecordFormat": {
               "MappingParameters": {
                  "CSVMappingParameters": {
                     "RecordColumnDelimiter": "string",
                     "RecordRowDelimiter": "string"
                  },
                  "JSONMappingParameters": {
                     "RecordRowPath": "string"
                  }
               },
               "RecordFormatType": "string"
            }
         },
         "S3ReferenceDataSource": {
            "BucketARN": "string",
            "FileKey": "string",
            "ReferenceRoleARN": "string"
         },
         "TableName": "string"
      }
}
```

Parámetros de la solicitud

La solicitud acepta los siguientes datos en formato JSON.

ApplicationName

Nombre de una aplicación existente.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. Longitud máxima de 128.

Patrón: `[a-zA-Z0-9_.-]+`

Obligatorio: sí

CurrentApplicationVersionId

Versión de la aplicación para la que va a añadir el origen de datos de referencia. Puede utilizar la operación DescribeApplication para obtener la versión actual de la aplicación. Si la versión especificada no es la actual, se devuelve la `ConcurrentModificationException`.

Tipo: largo

Rango válido: valor mínimo de 1. Valor máximo de 999999999.

Obligatorio: sí

ReferenceDataSource

El origen de datos de referencia puede ser un objeto en el bucket de Amazon S3. Amazon Kinesis Analytics lee el objeto y copia los datos en la tabla en la aplicación que se crea. Debe proporcionar el nombre de la clave de objeto, bucket de S3 y la tabla en la aplicación resultante que se crea. Asimismo, debe proporcionar un rol de IAM con los permisos necesarios que Amazon Kinesis Analytics puede adoptar para leer el objeto del bucket de S3 en su nombre.

Tipo: objeto ReferenceDataSource

Obligatorio: sí

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta HTTP 200 con un cuerpo HTTP vacío.

Errores

ConcurrentModificationException

Se produce una excepción como resultado de una modificación simultánea de una aplicación. Por ejemplo, dos personas que intentan editar la misma aplicación al mismo tiempo.

message

Código de estado HTTP: 400

InvalidArgumentException

El valor del parámetro de entrada especificado no es válido.

message

Código de estado HTTP: 400

ResourceInUseException

La aplicación no está disponible para esta operación.

message

Código de estado HTTP: 400

ResourceNotFoundException

No se encuentra la aplicación especificada.

message

Código de estado HTTP: 400

UnsupportedOperationException

La solicitud se rechazó porque no se admite un parámetro especificado o porque un recurso especificado no es válido para esta operación.

Código de estado HTTP: 400

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS Interfaz de línea de comandos V2
- AWS SDK para .NET V4
- AWS SDK para C++
- AWS SDK para Go v2
- AWS SDK para Java V2
- AWS SDK para JavaScript V3
- AWS SDK para Kotlin
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK para Ruby V3

CreateApplication

> **Note**
>
> Esta documentación es para la versión 1 de la API de Amazon Kinesis Data Analytics, que solo admite aplicaciones SQL. La versión 2 de la API admite aplicaciones SQL y Java. Para obtener más información acerca de la versión 2, consulte la documentación de la API V2 de Amazon Kinesis Data Analytics.

Crea una aplicación de Amazon Kinesis Data Analytics. Puede configurar cada aplicación con una fuente de streaming como entrada, un código de aplicación para procesar la entrada y hasta tres destinos en los que desee que Amazon Kinesis Analytics escriba los datos de salida de su aplicación. Para obtener una descripción general, consulte Cómo funciona.

En la configuración de entrada, se asigna el origen de flujo a un flujo integrado en la aplicación, lo que puede considerarse una tabla que se actualiza constantemente. En la asignación, debe proporcionar un esquema para el flujo en la aplicación y asignar cada columna de datos en el flujo en la aplicación a un elemento de datos en el origen de flujo.

Una o varias instrucciones SQL que leen datos de entrada, los transforman y generan la salida. El código de su aplicación puede crear uno o más artefactos de SQL, como flujos o bombas de SQL.

En la configuración de salida, puede configurar la aplicación para que escriba datos de flujos integrados en la aplicación creados en sus aplicaciones en un máximo de tres destinos.

Para leer los datos del flujo de origen o escribirlos en los flujos de destino, Amazon Kinesis Analytics necesita sus permisos. Puede conceder estos permisos mediante la creación de roles de IAM. Esto operación requiere permisos para realizar la acción `kinesisanalytics:CreateApplication`.

Para ver los ejercicios introductorios sobre cómo crear una aplicación de Amazon Kinesis Analytics, consulte Introducción.

Sintaxis de la solicitud

```
{
   "ApplicationCode": "string",
   "ApplicationDescription": "string",
   "ApplicationName": "string",
   "CloudWatchLoggingOptions": [
```

```
      {
         "LogStreamARN": "string",
         "RoleARN": "string"
      }
   ],
   "Inputs": [
      {
         "InputParallelism": {
            "Count": number
         },
         "InputProcessingConfiguration": {
            "InputLambdaProcessor": {
               "ResourceARN": "string",
               "RoleARN": "string"
            }
         },
         "InputSchema": {
            "RecordColumns": [
               {
                  "Mapping": "string",
                  "Name": "string",
                  "SqlType": "string"
               }
            ],
            "RecordEncoding": "string",
            "RecordFormat": {
               "MappingParameters": {
                  "CSVMappingParameters": {
                     "RecordColumnDelimiter": "string",
                     "RecordRowDelimiter": "string"
                  },
                  "JSONMappingParameters": {
                     "RecordRowPath": "string"
                  }
               },
               "RecordFormatType": "string"
            }
         },
         "KinesisFirehoseInput": {
            "ResourceARN": "string",
            "RoleARN": "string"
         },
         "KinesisStreamsInput": {
            "ResourceARN": "string",
```

```
            "RoleARN": "string"
         },
         "NamePrefix": "string"
      }
   ],
   "Outputs": [
      {
         "DestinationSchema": {
            "RecordFormatType": "string"
         },
         "KinesisFirehoseOutput": {
            "ResourceARN": "string",
            "RoleARN": "string"
         },
         "KinesisStreamsOutput": {
            "ResourceARN": "string",
            "RoleARN": "string"
         },
         "LambdaOutput": {
            "ResourceARN": "string",
            "RoleARN": "string"
         },
         "Name": "string"
      }
   ],
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ]
}
```

Parámetros de la solicitud

La solicitud acepta los siguientes datos en formato JSON.

ApplicationCode

Una o varias instrucciones SQL que leen datos de entrada, los transforman y generan la salida. Por ejemplo, puede escribir una instrucción SQL que lea los datos de un flujo en la aplicación, genere un promedio acumulado del número de clics de anuncios por proveedor e inserte las

filas resultantes en otro flujo en la aplicación mediante bombeos. Para obtener más información acerca del patrón típico, consulte Código de la aplicación.

Puede proporcionar una serie de instrucciones SQL, en la que la salida de una instrucción puede utilizarse como la entrada de la siguiente. Puede almacenar los resultados intermedios mediante la creación de flujos y bombeos en la aplicación.

Tenga en cuenta que el código de la aplicación debe crear flujos con los nombres especificados en las `Outputs`. Por ejemplo, si las `Outputs` definen flujos de salida llamados `ExampleOutputStream1` y `ExampleOutputStream2`, el código de la aplicación debe crear estos flujos.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 0. Longitud máxima de 102400.

Obligatorio: no

ApplicationDescription

Descripción resumida de la aplicación.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 0. La longitud máxima es 1024.

Obligatorio: no

ApplicationName

Nombre de su aplicación de Amazon Kinesis Analytics (por ejemplo, `sample-app`).

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. Longitud máxima de 128.

Patrón: `[a-zA-Z0-9_.-]+`

Obligatorio: sí

CloudWatchLoggingOptions

Utilice este parámetro para configurar un flujo de CloudWatch registro para supervisar los errores de configuración de la aplicación. Para obtener más información, consulta Cómo trabajar con Amazon CloudWatch Logs.

Tipo: matriz de objetos CloudWatchLoggingOption

Obligatorio: no

Inputs

Utilice este parámetro para configurar la entrada de la aplicación.

Puede configurar su aplicación para que reciba entradas de un solo origen de streaming. En esta configuración, asigna el origen de streaming a un flujo en la aplicación que se crea. El código de la aplicación puede entonces consultar el flujo en la aplicación como una tabla (es como un tabla que se actualiza constantemente).

Para el origen de streaming, debe proporcionar el nombre de recurso de Amazon (ARN) y el formato de datos en el flujo (por ejemplo, JSON, CSV, etc.). También debe proporcionar un rol de IAM que Amazon Kinesis Analytics pueda adoptar para leer este flujo en su nombre.

Para crear la secuencia en la aplicación, debe especificar un esquema para transformar sus datos en una versión esquematizada utilizada en SQL. En el esquema, debe proporcionar el necesario mapeo de los elementos de datos en el origen de streaming para registrar columnas en el flujo en la aplicación.

Tipo: matriz de objetos Input

Obligatorio: no

Outputs

Puede configurar la aplicación para que escriba datos de flujos integrados en la aplicación en un máximo de tres destinos.

Estos destinos pueden ser flujos de Amazon Kinesis, flujos de entrega de Amazon Kinesis Firehose, destinos de AWS Lambda o cualquier combinación de los tres.

En la configuración, debe especificar el nombre del flujo en la aplicación, el flujo de destino o el nombre de recurso de Amazon (ARN) de la función de Lambda y el formato que se utilizará al escribir los datos. Además proporciona un rol de IAM que Amazon Kinesis Analytics puede asumir para escribir la secuencia o función de Lambda en su nombre.

En la configuración de salida, también proporciona el flujo de salida o el ARN de la función de Lambda. Para los destinos del flujo, debe proporcionar el formato de los datos del flujo (por

ejemplo, JSON o CSV). Además proporcione un rol de IAM que Amazon Kinesis Analytics puede asumir para escribir la secuencia o función de Lambda en su nombre.

Tipo: matriz de objetos Output

Obligatorio: no

Tags

Lista de una o más etiquetas para asignar a la aplicación. Una etiqueta es un par clave-valor que identifica una aplicación. Tenga en cuenta que el número máximo de etiquetas incluye las etiquetas del sistema. El número máximo de etiquetas de la aplicación definidas por el usuario es 50. Para obtener más información, consulte Uso de etiquetas.

Tipo: matriz de objetos Tag

Miembros de la matriz: número mínimo de 1 artículo. La cantidad máxima es de 200 elementos.

Obligatorio: no

Sintaxis de la respuesta

```
{
   "ApplicationSummary": {
      "ApplicationARN": "string",
      "ApplicationName": "string",
      "ApplicationStatus": "string"
   }
}
```

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta HTTP 200.

El servicio devuelve los datos siguientes en formato JSON.

ApplicationSummary

En respuesta a su solicitud `CreateApplication`, Amazon Kinesis Analytics devuelve una respuesta con un resumen de la aplicación que creó, incluidos el nombre, el nombre y el estado de la aplicación nombre de recurso de Amazon (ARN).

Tipo: objeto ApplicationSummary

Errores

CodeValidationException

El código de aplicación (consulta) proporcionado por el usuario no es válido. Puede tratarse de un simple error de sintaxis.

message

Test

Código de estado HTTP: 400

ConcurrentModificationException

Se produce una excepción como resultado de una modificación simultánea de una aplicación. Por ejemplo, dos personas que intentan editar la misma aplicación al mismo tiempo.

message

Código de estado HTTP: 400

InvalidArgumentException

El valor del parámetro de entrada especificado no es válido.

message

Código de estado HTTP: 400

LimitExceededException

Se ha superado el número de aplicaciones permitidas.

message

Código de estado HTTP: 400

ResourceInUseException

La aplicación no está disponible para esta operación.

message

Código de estado HTTP: 400

TooManyTagsException

Aplicación creada con demasiadas etiquetas o se han añadido demasiadas etiquetas a una aplicación. Tenga en cuenta que el número máximo de etiquetas incluye las etiquetas del sistema. El número máximo de etiquetas de la aplicación definidas por el usuario es 50.

Código de estado HTTP: 400

UnsupportedOperationException

La solicitud se rechazó porque no se admite un parámetro especificado o porque un recurso especificado no es válido para esta operación.

Código de estado HTTP: 400

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS Interfaz de línea de comandos V2
- AWS SDK para .NET V4
- AWS SDK para C++
- AWS SDK para Go v2
- AWS SDK para Java V2
- AWS SDK para JavaScript V3
- AWS SDK para Kotlin
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK para Ruby V3

DeleteApplication

> **Note**
>
> Esta documentación es para la versión 1 de la API de Amazon Kinesis Data Analytics, que solo admite aplicaciones SQL. La versión 2 de la API admite aplicaciones SQL y Java. Para obtener más información acerca de la versión 2, consulte la documentación de la API V2 de Amazon Kinesis Data Analytics.

Elimina la aplicación especificada. Amazon Kinesis Analytics detiene la ejecución de la aplicación y la elimina, incluidos los artefactos de la aplicación (como los flujos dentro de la aplicación, la tabla de referencia y el código de la aplicación).

Esto operación requiere permisos para realizar la acción `kinesisanalytics:DeleteApplication`.

Sintaxis de la solicitud

```
{
   "ApplicationName": "string",
   "CreateTimestamp": number
}
```

Parámetros de la solicitud

La solicitud acepta los siguientes datos en formato JSON.

ApplicationName

Nombre de la aplicación Amazon Kinesis Analytics que se va a eliminar.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. Longitud máxima de 128.

Patrón: `[a-zA-Z0-9_.-]+`

Obligatorio: sí

CreateTimestamp

Para obtener este valor, puede usar la operación `DescribeApplication`.

Tipo: marca temporal

Obligatorio: sí

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta HTTP 200 con un cuerpo HTTP vacío.

Errores

ConcurrentModificationException

Se produce una excepción como resultado de una modificación simultánea de una aplicación. Por ejemplo, dos personas que intentan editar la misma aplicación al mismo tiempo.

message

Código de estado HTTP: 400

ResourceInUseException

La aplicación no está disponible para esta operación.

message

Código de estado HTTP: 400

ResourceNotFoundException

No se encuentra la aplicación especificada.

message

Código de estado HTTP: 400

UnsupportedOperationException

La solicitud se rechazó porque no se admite un parámetro especificado o porque un recurso especificado no es válido para esta operación.

Código de estado HTTP: 400

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS Interfaz de línea de comandos V2
- AWS SDK para .NET V4
- AWS SDK para C++
- AWS SDK para Go v2
- AWS SDK para Java V2
- AWS SDK para JavaScript V3
- AWS SDK para Kotlin
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK para Ruby V3

DeleteApplicationCloudWatchLoggingOption

> **Note**
>
> Esta documentación es para la versión 1 de la API de Amazon Kinesis Data Analytics, que solo admite aplicaciones SQL. La versión 2 de la API admite aplicaciones SQL y Java. Para obtener más información acerca de la versión 2, consulte la documentación de la API V2 de Amazon Kinesis Data Analytics.

Elimina un flujo de CloudWatch registro de una aplicación. Para obtener más información sobre el uso de transmisiones de CloudWatch registros con las aplicaciones de Amazon Kinesis Analytics, consulte Trabajar con CloudWatch Amazon Logs.

Sintaxis de la solicitud

```
{
   "ApplicationName": "string",
   "CloudWatchLoggingOptionId": "string",
   "CurrentApplicationVersionId": number
}
```

Parámetros de la solicitud

La solicitud acepta los siguientes datos en formato JSON.

ApplicationName

El nombre de la aplicación de Kinesis Analytics.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. Longitud máxima de 128.

Patrón: `[a-zA-Z0-9_.-]+`

Obligatorio: sí

CloudWatchLoggingOptionId

La `CloudWatchLoggingOptionId` de la opción de CloudWatch registro que se va a eliminar. Puede obtener el `CloudWatchLoggingOptionId` mediante la operación DescribeApplication.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 50 caracteres.

Patrón: `[a-zA-Z0-9_.-]+`

Obligatorio: sí

CurrentApplicationVersionId

El ID de versión de la aplicación Kinesis Analytics.

Tipo: largo

Rango válido: valor mínimo de 1. Valor máximo de 999999999.

Obligatorio: sí

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta HTTP 200 con un cuerpo HTTP vacío.

Errores

ConcurrentModificationException

Se produce una excepción como resultado de una modificación simultánea de una aplicación. Por ejemplo, dos personas que intentan editar la misma aplicación al mismo tiempo.

message

Código de estado HTTP: 400

InvalidArgumentException

El valor del parámetro de entrada especificado no es válido.

message

Código de estado HTTP: 400

ResourceInUseException

La aplicación no está disponible para esta operación.

message

Código de estado HTTP: 400

ResourceNotFoundException

No se encuentra la aplicación especificada.

message

Código de estado HTTP: 400

UnsupportedOperationException

La solicitud se rechazó porque no se admite un parámetro especificado o porque un recurso especificado no es válido para esta operación.

Código de estado HTTP: 400

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS Interfaz de línea de comandos V2
- AWS SDK para .NET V4
- AWS SDK para C++
- AWS SDK para Go v2
- AWS SDK para Java V2
- AWS SDK para JavaScript V3
- AWS SDK para Kotlin
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK para Ruby V3

DeleteApplicationInputProcessingConfiguration

> **Note**
>
> Esta documentación es para la versión 1 de la API de Amazon Kinesis Data Analytics, que solo admite aplicaciones SQL. La versión 2 de la API admite aplicaciones SQL y Java. Para obtener más información acerca de la versión 2, consulte la documentación de la API V2 de Amazon Kinesis Data Analytics.

Elimina una InputProcessingConfiguration de una entrada.

Sintaxis de la solicitud

```
{
   "ApplicationName": "string",
   "CurrentApplicationVersionId": number,
   "InputId": "string"
}
```

Parámetros de la solicitud

La solicitud acepta los siguientes datos en formato JSON.

ApplicationName

El nombre de la aplicación de Kinesis Analytics.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. Longitud máxima de 128.

Patrón: `[a-zA-Z0-9_.-]+`

Obligatorio: sí

CurrentApplicationVersionId

El ID de versión de la aplicación Kinesis Analytics.

Tipo: largo

Rango válido: valor mínimo de 1. Valor máximo de 999999999.

Obligatorio: sí

InputId

El ID de la configuración de entrada de la que se va a eliminar la configuración de procesamiento de entrada. Puede obtener una lista de las entradas IDs de una aplicación mediante la DescribeApplicationoperación.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 50 caracteres.

Patrón: `[a-zA-Z0-9_.-]+`

Obligatorio: sí

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta HTTP 200 con un cuerpo HTTP vacío.

Errores

ConcurrentModificationException

Se produce una excepción como resultado de una modificación simultánea de una aplicación. Por ejemplo, dos personas que intentan editar la misma aplicación al mismo tiempo.

message

Código de estado HTTP: 400

InvalidArgumentException

El valor del parámetro de entrada especificado no es válido.

message

Código de estado HTTP: 400

ResourceInUseException

La aplicación no está disponible para esta operación.

message

Código de estado HTTP: 400

ResourceNotFoundException

No se encuentra la aplicación especificada.

message

Código de estado HTTP: 400

UnsupportedOperationException

La solicitud se rechazó porque no se admite un parámetro especificado o porque un recurso especificado no es válido para esta operación.

Código de estado HTTP: 400

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS Interfaz de línea de comandos V2
- AWS SDK para .NET V4
- AWS SDK para C++
- AWS SDK para Go v2
- AWS SDK para Java V2
- AWS SDK para JavaScript V3
- AWS SDK para Kotlin
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK para Ruby V3

DeleteApplicationOutput

> **Note**
>
> Esta documentación es para la versión 1 de la API de Amazon Kinesis Data Analytics, que solo admite aplicaciones SQL. La versión 2 de la API admite aplicaciones SQL y Java. Para obtener más información acerca de la versión 2, consulte la documentación de la API V2 de Amazon Kinesis Data Analytics.

Elimina la configuración de destino de salida de la configuración de la aplicación. Amazon Kinesis Analytics dejará de escribir datos del flujo integrado en la aplicación correspondiente, en el destino de salida externo.

Esto operación requiere permisos para realizar la acción `kinesisanalytics:DeleteApplicationOutput`.

Sintaxis de la solicitud

```
{
   "ApplicationName": "string",
   "CurrentApplicationVersionId": number,
   "OutputId": "string"
}
```

Parámetros de la solicitud

La solicitud acepta los siguientes datos en formato JSON.

ApplicationName

Nombre de la aplicación Amazon Kinesis Analytics

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. Longitud máxima de 128.

Patrón: `[a-zA-Z0-9_.-]+`

Obligatorio: sí

CurrentApplicationVersionId

Versión de la aplicación Amazon Kinesis Analytics. Puede utilizar la operación DescribeApplication para obtener la versión actual de la aplicación. Si la versión especificada no es la actual, se devuelve la `ConcurrentModificationException`.

Tipo: largo

Rango válido: valor mínimo de 1. Valor máximo de 999999999.

Obligatorio: sí

OutputId

ID de la configuración que se eliminará. Cada configuración de salida que se agrega a la aplicación, ya sea cuando se crea la aplicación o posteriormente mediante la AddApplicationOutputoperación, tiene un identificador único. Debe proporcionar el ID para identificar de forma exclusiva la configuración de salida que desea eliminar de la configuración de la aplicación. Puede usar la operación DescribeApplication para obtener el `OutputId` específico.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 50 caracteres.

Patrón: `[a-zA-Z0-9_.-]+`

Obligatorio: sí

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta HTTP 200 con un cuerpo HTTP vacío.

Errores

ConcurrentModificationException

Se produce una excepción como resultado de una modificación simultánea de una aplicación. Por ejemplo, dos personas que intentan editar la misma aplicación al mismo tiempo.

message

Código de estado HTTP: 400

InvalidArgumentException

El valor del parámetro de entrada especificado no es válido.

message

Código de estado HTTP: 400

ResourceInUseException

La aplicación no está disponible para esta operación.

message

Código de estado HTTP: 400

ResourceNotFoundException

No se encuentra la aplicación especificada.

message

Código de estado HTTP: 400

UnsupportedOperationException

La solicitud se rechazó porque no se admite un parámetro especificado o porque un recurso especificado no es válido para esta operación.

Código de estado HTTP: 400

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS Interfaz de línea de comandos V2
- AWS SDK para .NET V4
- AWS SDK para C++
- AWS SDK para Go v2
- AWS SDK para Java V2

- AWS SDK para JavaScript V3
- AWS SDK para Kotlin
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK para Ruby V3

DeleteApplicationReferenceDataSource

> **Note**
>
> Esta documentación es para la versión 1 de la API de Amazon Kinesis Data Analytics, que solo admite aplicaciones SQL. La versión 2 de la API admite aplicaciones SQL y Java. Para obtener más información acerca de la versión 2, consulte la documentación de la API V2 de Amazon Kinesis Data Analytics.

Elimina una configuración de origen de datos de referencia de la configuración de la aplicación especificada.

Si la aplicación está en ejecución, Amazon Kinesis Analytics elimina inmediatamente la tabla de la aplicación que creó mediante AddApplicationReferenceDataSourcela operación.

Esto operación requiere permisos para realizar la acción `kinesisanalytics.DeleteApplicationReferenceDataSource`.

Sintaxis de la solicitud

```
{
   "ApplicationName": "string",
   "CurrentApplicationVersionId": number,
   "ReferenceId": "string"
}
```

Parámetros de la solicitud

La solicitud acepta los siguientes datos en formato JSON.

ApplicationName

Nombre de una aplicación existente.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. Longitud máxima de 128.

Patrón: `[a-zA-Z0-9_.-]+`

Obligatorio: sí

CurrentApplicationVersionId

La versión de la aplicación. Puede utilizar la operación DescribeApplication para obtener la versión actual de la aplicación. Si la versión especificada no es la actual, se devuelve la `ConcurrentModificationException`.

Tipo: largo

Rango válido: valor mínimo de 1. Valor máximo de 999999999.

Obligatorio: sí

ReferenceId

El ID del origen de datos de referencia. Al añadir una fuente de datos de referencia a la aplicación mediante el AddApplicationReferenceDataSource, Amazon Kinesis Analytics le asigna un ID. Puede utilizar la operación DescribeApplication para obtener el ID de referencia.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 50 caracteres.

Patrón: `[a-zA-Z0-9_.-]+`

Obligatorio: sí

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta HTTP 200 con un cuerpo HTTP vacío.

Errores

ConcurrentModificationException

Se produce una excepción como resultado de una modificación simultánea de una aplicación. Por ejemplo, dos personas que intentan editar la misma aplicación al mismo tiempo.

message

Código de estado HTTP: 400

InvalidArgumentException

El valor del parámetro de entrada especificado no es válido.

message

Código de estado HTTP: 400

ResourceInUseException

La aplicación no está disponible para esta operación.

message

Código de estado HTTP: 400

ResourceNotFoundException

No se encuentra la aplicación especificada.

message

Código de estado HTTP: 400

UnsupportedOperationException

La solicitud se rechazó porque no se admite un parámetro especificado o porque un recurso especificado no es válido para esta operación.

Código de estado HTTP: 400

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS Interfaz de línea de comandos V2
- AWS SDK para .NET V4
- AWS SDK para C++
- AWS SDK para Go v2
- AWS SDK para Java V2
- AWS SDK para JavaScript V3

- AWS SDK para Kotlin
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK para Ruby V3

DescribeApplication

> **Note**
>
> Esta documentación es para la versión 1 de la API de Amazon Kinesis Data Analytics, que solo admite aplicaciones SQL. La versión 2 de la API admite aplicaciones SQL y Java. Para obtener más información acerca de la versión 2, consulte la documentación de la API V2 de Amazon Kinesis Data Analytics.

Devuelve información sobre una aplicación específica de Amazon Kinesis Analytics.

Si desea enumerar la lista de todas las aplicaciones de su cuenta, utilice la operación ListApplications.

Esto operación requiere permisos para realizar la acción `kinesisanalytics:DescribeApplication`. Puede usar `DescribeApplication` para obtener el VersionID de la aplicación actual, al que necesita para llamar a otras operaciones, como `Update`.

Sintaxis de la solicitud

```
{
   "ApplicationName": "string"
}
```

Parámetros de la solicitud

La solicitud acepta los siguientes datos en formato JSON.

ApplicationName

Nombre de la aplicación.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. Longitud máxima de 128.

Patrón: `[a-zA-Z0-9_.-]+`

Obligatorio: sí

Sintaxis de la respuesta

```
{
   "ApplicationDetail": {
      "ApplicationARN": "string",
      "ApplicationCode": "string",
      "ApplicationDescription": "string",
      "ApplicationName": "string",
      "ApplicationStatus": "string",
      "ApplicationVersionId": number,
      "CloudWatchLoggingOptionDescriptions": [
         {
            "CloudWatchLoggingOptionId": "string",
            "LogStreamARN": "string",
            "RoleARN": "string"
         }
      ],
      "CreateTimestamp": number,
      "InputDescriptions": [
         {
            "InAppStreamNames": [ "string" ],
            "InputId": "string",
            "InputParallelism": {
               "Count": number
            },
            "InputProcessingConfigurationDescription": {
               "InputLambdaProcessorDescription": {
                  "ResourceARN": "string",
                  "RoleARN": "string"
               }
            },
            "InputSchema": {
               "RecordColumns": [
                  {
                     "Mapping": "string",
                     "Name": "string",
                     "SqlType": "string"
                  }
               ],
               "RecordEncoding": "string",
               "RecordFormat": {
                  "MappingParameters": {
                     "CSVMappingParameters": {
```

```
                        "RecordColumnDelimiter": "string",
                        "RecordRowDelimiter": "string"
                     },
                     "JSONMappingParameters": {
                        "RecordRowPath": "string"
                     }
                  },
                  "RecordFormatType": "string"
               }
            },
            "InputStartingPositionConfiguration": {
               "InputStartingPosition": "string"
            },
            "KinesisFirehoseInputDescription": {
               "ResourceARN": "string",
               "RoleARN": "string"
            },
            "KinesisStreamsInputDescription": {
               "ResourceARN": "string",
               "RoleARN": "string"
            },
            "NamePrefix": "string"
         }
      ],
      "LastUpdateTimestamp": number,
      "OutputDescriptions": [
         {
            "DestinationSchema": {
               "RecordFormatType": "string"
            },
            "KinesisFirehoseOutputDescription": {
               "ResourceARN": "string",
               "RoleARN": "string"
            },
            "KinesisStreamsOutputDescription": {
               "ResourceARN": "string",
               "RoleARN": "string"
            },
            "LambdaOutputDescription": {
               "ResourceARN": "string",
               "RoleARN": "string"
            },
            "Name": "string",
            "OutputId": "string"
```

```
            }
         ],
         "ReferenceDataSourceDescriptions": [
            {
               "ReferenceId": "string",
               "ReferenceSchema": {
                  "RecordColumns": [
                     {
                        "Mapping": "string",
                        "Name": "string",
                        "SqlType": "string"
                     }
                  ],
                  "RecordEncoding": "string",
                  "RecordFormat": {
                     "MappingParameters": {
                        "CSVMappingParameters": {
                           "RecordColumnDelimiter": "string",
                           "RecordRowDelimiter": "string"
                        },
                        "JSONMappingParameters": {
                           "RecordRowPath": "string"
                        }
                     },
                     "RecordFormatType": "string"
                  }
               },
               "S3ReferenceDataSourceDescription": {
                  "BucketARN": "string",
                  "FileKey": "string",
                  "ReferenceRoleARN": "string"
               },
               "TableName": "string"
            }
         ]
      }
}
```

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta HTTP 200.

El servicio devuelve los datos siguientes en formato JSON.

[ApplicationDetail](#)

Proporciona una descripción de la aplicación, como el nombre de recurso de Amazon (ARN) de la aplicación, el estado, la última versión y los detalles de configuración de entrada y salida.

Tipo: objeto [ApplicationDetail](#)

Errores

ResourceNotFoundException

No se encuentra la aplicación especificada.

message

Código de estado HTTP: 400

UnsupportedOperationException

La solicitud se rechazó porque no se admite un parámetro especificado o porque un recurso especificado no es válido para esta operación.

Código de estado HTTP: 400

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- [AWS Interfaz de línea de comandos V2](#)
- [AWS SDK para .NET V4](#)
- [AWS SDK para C++](#)
- [AWS SDK para Go v2](#)
- [AWS SDK para Java V2](#)
- [AWS SDK para JavaScript V3](#)
- [AWS SDK para Kotlin](#)
- [AWS SDK para PHP V3](#)
- [AWS SDK para Python](#)

- AWS SDK para Ruby V3

DiscoverInputSchema

> **Note**
>
> Esta documentación es para la versión 1 de la API de Amazon Kinesis Data Analytics, que solo admite aplicaciones SQL. La versión 2 de la API admite aplicaciones SQL y Java. Para obtener más información acerca de la versión 2, consulte la documentación de la API V2 de Amazon Kinesis Data Analytics.

Infiere un esquema mediante la evaluación de registros de muestra en el origen de flujo especificado (flujo de Amazon Kinesis o flujo de entrega de Amazon Kinesis Firehose) o en el objeto de S3 especificado. En la respuesta, la operación devuelve el esquema inferido y también los registros de muestra que la operación utilizó para deducir el esquema.

Puede usar el esquema inferido al configurar un origen de flujo para su aplicación. Para obtener información conceptual, consulte Configuración de entrada de la aplicación. Tenga en cuenta que cuando crea una aplicación mediante la consola de Amazon Kinesis Analytics, la consola utiliza esta operación para deducir un esquema y mostrarlo en la interfaz de usuario de la consola.

Esto operación requiere permisos para realizar la acción `kinesisanalytics:DiscoverInputSchema`.

Sintaxis de la solicitud

```
{
   "InputProcessingConfiguration": {
      "InputLambdaProcessor": {
         "ResourceARN": "string",
         "RoleARN": "string"
      }
   },
   "InputStartingPositionConfiguration": {
      "InputStartingPosition": "string"
   },
   "ResourceARN": "string",
   "RoleARN": "string",
   "S3Configuration": {
      "BucketARN": "string",
      "FileKey": "string",
```

```
        "RoleARN": "string"
    }
}
```

Parámetros de la solicitud

La solicitud acepta los siguientes datos en formato JSON.

InputProcessingConfiguration

Utilice InputProcessingConfiguration para preprocesar los registros antes de detectar el esquema de los registros.

Tipo: objeto InputProcessingConfiguration

Obligatorio: no

InputStartingPositionConfiguration

Punto en el que desea que Amazon Kinesis Analytics comience a leer los registros con los fines de descubrimiento de orígenes de flujo especificados.

Tipo: objeto InputStartingPositionConfiguration

Obligatorio: no

ResourceARN

El nombre de recurso de Amazon (ARN) del origen de streaming.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: no

RoleARN

El ARN del rol de IAM que Amazon Kinesis Analytics puede asumir para acceder al flujo en su nombre.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: no

S3Configuration

Especifique este parámetro para descubrir un esquema a partir de datos en un objeto de Amazon S3.

Tipo: objeto S3Configuration

Obligatorio: no

Sintaxis de la respuesta

```
{
   "InputSchema": {
      "RecordColumns": [
         {
            "Mapping": "string",
            "Name": "string",
            "SqlType": "string"
         }
      ],
      "RecordEncoding": "string",
      "RecordFormat": {
         "MappingParameters": {
            "CSVMappingParameters": {
               "RecordColumnDelimiter": "string",
               "RecordRowDelimiter": "string"
            },
            "JSONMappingParameters": {
               "RecordRowPath": "string"
            }
         },
         "RecordFormatType": "string"
      }
   },
   "ParsedInputRecords": [
      [ "string" ]
   ],
```

```
    "ProcessedInputRecords": [ "string" ],
    "RawInputRecords": [ "string" ]
}
```

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta HTTP 200.

El servicio devuelve los datos siguientes en formato JSON.

InputSchema

Esquema deducido del origen de streaming. Describe el formato de los datos del origen de streaming y cómo se asigna cada elemento de datos a las columnas correspondientes que se crean en la secuencia en la aplicación que puede crear.

Tipo: objeto SourceSchema

ParsedInputRecords

Conjunto de elementos, donde cada elemento corresponde a una fila de un registro de flujo (un registro de flujo puede tener más de una fila).

Tipo: Matriz de matrices de cadenas

ProcessedInputRecords

Transmita datos modificados por el procesador especificado en el parámetro `InputProcessingConfiguration`.

Tipo: matriz de cadenas

RawInputRecords

Datos de flujo sin procesar que se asignaron para deducir el esquema.

Tipo: matriz de cadenas

Errores

InvalidArgumentException

El valor del parámetro de entrada especificado no es válido.

message

Código de estado HTTP: 400

ResourceProvisionedThroughputExceededException

Discovery no pudo obtener un registro de la fuente de streaming debido a Amazon Kinesis ProvisionedThroughputExceededException Streams. Para obtener más información, consulte la referencia GetRecordsde la API de Amazon Kinesis Streams.

Código de estado HTTP: 400

ServiceUnavailableException

El servicio no está disponible. Retroceda y vuelva a ejecutar la operación.

Código de estado HTTP: 500

UnableToDetectSchemaException

El formato de la fecha no es válido. Amazon Kinesis Analytics no puede detectar el esquema del origen de flujo determinado.

Código de estado HTTP: 400

UnsupportedOperationException

La solicitud se rechazó porque no se admite un parámetro especificado o porque un recurso especificado no es válido para esta operación.

Código de estado HTTP: 400

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS Interfaz de línea de comandos V2
- AWS SDK para .NET V4
- AWS SDK para C++
- AWS SDK para Go v2
- AWS SDK para Java V2

- AWS SDK para JavaScript V3
- AWS SDK para Kotlin
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK para Ruby V3

ListApplications

> **Note**
>
> Esta documentación es para la versión 1 de la API de Amazon Kinesis Data Analytics, que solo admite aplicaciones SQL. La versión 2 de la API admite aplicaciones SQL y Java. Para obtener más información acerca de la versión 2, consulte la documentación de la API V2 de Amazon Kinesis Data Analytics.

Devuelve una lista de las aplicaciones de Amazon Kinesis Analytics de su cuenta. Para cada aplicación, la respuesta incluye el nombre de la aplicación, el nombre de recurso de Amazon (ARN) y el estado. Si la respuesta devuelve el valor `HasMoreApplications` como verdadero, puede enviar otra solicitud añadiendo el `ExclusiveStartApplicationName` en el cuerpo de la solicitud y estableciendo el valor en el último nombre de la aplicación de la respuesta anterior.

Si desea obtener información detallada sobre una aplicación específica, utilice DescribeApplication.

Esto operación requiere permisos para realizar la acción `kinesisanalytics:ListApplications`.

Sintaxis de la solicitud

```
{
   "ExclusiveStartApplicationName": "string",
   "Limit": number
}
```

Parámetros de la solicitud

La solicitud acepta los siguientes datos en formato JSON.

ExclusiveStartApplicationName

Nombre de la aplicación con la que se va a iniciar la lista. Cuando se utiliza la paginación para recuperar la lista, no es necesario especificar este parámetro en la primera solicitud. Sin embargo, en las solicitudes posteriores, añada el último nombre de la aplicación de la respuesta anterior para obtener la siguiente página de solicitudes.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. Longitud máxima de 128.

Patrón: `[a-zA-Z0-9_.-]+`

Obligatorio: no

Limit

Número máximo de aplicaciones para enumerar.

Tipo: entero

Rango válido: valor mínimo de 1. Valor máximo de 50.

Obligatorio: no

Sintaxis de la respuesta

```
{
   "ApplicationSummaries": [
      {
         "ApplicationARN": "string",
         "ApplicationName": "string",
         "ApplicationStatus": "string"
      }
   ],
   "HasMoreApplications": boolean
}
```

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta HTTP 200.

El servicio devuelve los datos siguientes en formato JSON.

ApplicationSummaries

Lista de objetos `ApplicationSummary`.

Tipo: matriz de objetos ApplicationSummary

HasMoreApplications

Devuelve el valor True si hay más aplicaciones que recuperar.

Tipo: Booleano

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS Interfaz de línea de comandos V2
- AWS SDK para .NET V4
- AWS SDK para C++
- AWS SDK para Go v2
- AWS SDK para Java V2
- AWS SDK para JavaScript V3
- AWS SDK para Kotlin
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK para Ruby V3

ListTagsForResource

Recupera la lista de las etiquetas de clave-valor asignadas a la aplicación. Para obtener más información, consulte Uso de etiquetas.

Sintaxis de la solicitud

```
{
   "ResourceARN": "string"
}
```

Parámetros de la solicitud

La solicitud acepta los siguientes datos en formato JSON.

ResourceARN

El ARN de la aplicación de para la que se recuperarán las etiquetas.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

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 HTTP 200.

El servicio devuelve los datos siguientes en formato JSON.

Tags

Las etiquetas clave-valor asignadas a la aplicación.

Tipo: matriz de objetos Tag

Miembros de la matriz: número mínimo de 1 artículo. La cantidad máxima es de 200 elementos.

Errores

ConcurrentModificationException

Se produce una excepción como resultado de una modificación simultánea de una aplicación. Por ejemplo, dos personas que intentan editar la misma aplicación al mismo tiempo.

message

Código de estado HTTP: 400

InvalidArgumentException

El valor del parámetro de entrada especificado no es válido.

message

Código de estado HTTP: 400

ResourceNotFoundException

No se encuentra la aplicación especificada.

message

Código de estado HTTP: 400

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS Interfaz de línea de comandos V2
- AWS SDK para .NET V4
- AWS SDK para C++
- AWS SDK para Go v2
- AWS SDK para Java V2
- AWS SDK para JavaScript V3
- AWS SDK para Kotlin
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK para Ruby V3

StartApplication

> **Note**
>
> Esta documentación es para la versión 1 de la API de Amazon Kinesis Data Analytics, que solo admite aplicaciones SQL. La versión 2 de la API admite aplicaciones SQL y Java. Para obtener más información acerca de la versión 2, consulte la documentación de la API V2 de Amazon Kinesis Data Analytics.

Inicia la aplicación Amazon Kinesis Analytics especificada. Tras crear una aplicación, debe llamar exclusivamente a esta operación para iniciar la aplicación.

Una vez iniciada la aplicación, comienza a consumir los datos de entrada, los procesa y graba el resultado en el destino configurado.

El estado de la aplicación debe ser `READY` para que pueda iniciar una aplicación. Puede obtener el estado de la aplicación en la consola o mediante la DescribeApplicationoperación.

Tras iniciar la aplicación, puede impedir que procese la entrada mediante una llamada a la StopApplicationoperación.

Esto operación requiere permisos para realizar la acción `kinesisanalytics:StartApplication`.

Sintaxis de la solicitud

```
{
   "ApplicationName": "string",
   "InputConfigurations": [
      {
         "Id": "string",
         "InputStartingPositionConfiguration": {
            "InputStartingPosition": "string"
         }
      }
   ]
}
```

Parámetros de la solicitud

La solicitud acepta los siguientes datos en formato JSON.

ApplicationName

Nombre de la aplicación.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. Longitud máxima de 128.

Patrón: `[a-zA-Z0-9_.-]+`

Obligatorio: sí

InputConfigurations

Identifica la entrada específica, mediante un ID, que la aplicación comienza a consumir. Amazon Kinesis Analytics comienza a leer el origen de flujo asociado a la entrada. También puede especificar en qué parte del origen de flujo quiere que Amazon Kinesis Analytics comience a leer.

Tipo: matriz de objetos InputConfiguration

Obligatorio: sí

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta HTTP 200 con un cuerpo HTTP vacío.

Errores

InvalidApplicationConfigurationException

La configuración de la aplicación proporcionada por el usuario no es válida.

message

prueba

Código de estado HTTP: 400

InvalidArgumentException

El valor del parámetro de entrada especificado no es válido.

message

Código de estado HTTP: 400

ResourceInUseException

La aplicación no está disponible para esta operación.

message

Código de estado HTTP: 400

ResourceNotFoundException

No se encuentra la aplicación especificada.

message

Código de estado HTTP: 400

UnsupportedOperationException

La solicitud se rechazó porque no se admite un parámetro especificado o porque un recurso especificado no es válido para esta operación.

Código de estado HTTP: 400

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS Interfaz de línea de comandos V2
- AWS SDK para .NET V4
- AWS SDK para C++
- AWS SDK para Go v2
- AWS SDK para Java V2
- AWS SDK para JavaScript V3

- AWS SDK para Kotlin
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK para Ruby V3

StopApplication

> **Note**
>
> Esta documentación es para la versión 1 de la API de Amazon Kinesis Data Analytics, que solo admite aplicaciones SQL. La versión 2 de la API admite aplicaciones SQL y Java. Para obtener más información acerca de la versión 2, consulte la documentación de la API V2 de Amazon Kinesis Data Analytics.

Detiene el procesamiento de los datos de entrada por la aplicación. Puede detener una aplicación solo si está en ejecución. Puede utilizar la DescribeApplicationoperación para buscar el estado de la aplicación. Una vez detenida la aplicación, Amazon Kinesis Analytics deja de leer los datos de la entrada, la aplicación deja de procesar los datos y no se escribe ningún resultado en el destino.

Esto operación requiere permisos para realizar la acción `kinesisanalytics:StopApplication`.

Sintaxis de la solicitud

```
{
   "ApplicationName": "string"
}
```

Parámetros de la solicitud

La solicitud acepta los siguientes datos en formato JSON.

ApplicationName

Nombre de la aplicación en ejecución que se va a detener.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. Longitud máxima de 128.

Patrón: `[a-zA-Z0-9_.-]+`

Obligatorio: sí

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta HTTP 200 con un cuerpo HTTP vacío.

Errores

ResourceInUseException

La aplicación no está disponible para esta operación.

message

Código de estado HTTP: 400

ResourceNotFoundException

No se encuentra la aplicación especificada.

message

Código de estado HTTP: 400

UnsupportedOperationException

La solicitud se rechazó porque no se admite un parámetro especificado o porque un recurso especificado no es válido para esta operación.

Código de estado HTTP: 400

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS Interfaz de línea de comandos V2
- AWS SDK para .NET V4
- AWS SDK para C++
- AWS SDK para Go v2
- AWS SDK para Java V2
- AWS SDK para JavaScript V3

- AWS SDK para Kotlin
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK para Ruby V3

TagResource

Añade una o más etiquetas clave-valor a una aplicación de Kinesis Analytics. Tenga en cuenta que el número máximo de etiquetas incluye las etiquetas del sistema. El número máximo de etiquetas de la aplicación definidas por el usuario es 50. Para obtener más información, consulte Uso de etiquetas.

Sintaxis de la solicitud

```
{
   "ResourceARN": "string",
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ]
}
```

Parámetros de la solicitud

La solicitud acepta los siguientes datos en formato JSON.

ResourceARN

El ARN de la aplicación para asignar las etiquetas.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: sí

Tags

Las etiquetas clave-valor para asignar a la aplicación.

Tipo: matriz de objetos Tag

Miembros de la matriz: número mínimo de 1 artículo. La cantidad máxima es de 200 elementos.

Obligatorio: sí

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta HTTP 200 con un cuerpo HTTP vacío.

Errores

ConcurrentModificationException

Se produce una excepción como resultado de una modificación simultánea de una aplicación. Por ejemplo, dos personas que intentan editar la misma aplicación al mismo tiempo.

message

Código de estado HTTP: 400

InvalidArgumentException

El valor del parámetro de entrada especificado no es válido.

message

Código de estado HTTP: 400

ResourceInUseException

La aplicación no está disponible para esta operación.

message

Código de estado HTTP: 400

ResourceNotFoundException

No se encuentra la aplicación especificada.

message

Código de estado HTTP: 400

TooManyTagsException

Aplicación creada con demasiadas etiquetas o se han añadido demasiadas etiquetas a una aplicación. Tenga en cuenta que el número máximo de etiquetas incluye las etiquetas del sistema. El número máximo de etiquetas de la aplicación definidas por el usuario es 50.

Código de estado HTTP: 400

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS Interfaz de línea de comandos V2
- AWS SDK para .NET V4
- AWS SDK para C++
- AWS SDK para Go v2
- AWS SDK para Java V2
- AWS SDK para JavaScript V3
- AWS SDK para Kotlin
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK para Ruby V3

UntagResource

Elimina una o varias etiquetas de una aplicación de Kinesis Analytics. Para obtener más información, consulte Uso de etiquetas.

Sintaxis de la solicitud

```
{
   "ResourceARN": "string",
   "TagKeys": [ "string" ]
}
```

Parámetros de la solicitud

La solicitud acepta los siguientes datos en formato JSON.

ResourceARN

El ARN de la aplicación de Kinesis Analytics de la que se van a quitar las etiquetas.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: sí

TagKeys

Una lista de claves de etiquetas que se van a eliminar de la aplicación especificada.

Tipo: matriz de cadenas

Miembros de la matriz: número mínimo de 1 artículo. La cantidad máxima es de 200 elementos.

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 HTTP 200 con un cuerpo HTTP vacío.

Errores

ConcurrentModificationException

Se produce una excepción como resultado de una modificación simultánea de una aplicación. Por ejemplo, dos personas que intentan editar la misma aplicación al mismo tiempo.

message

Código de estado HTTP: 400

InvalidArgumentException

El valor del parámetro de entrada especificado no es válido.

message

Código de estado HTTP: 400

ResourceInUseException

La aplicación no está disponible para esta operación.

message

Código de estado HTTP: 400

ResourceNotFoundException

No se encuentra la aplicación especificada.

message

Código de estado HTTP: 400

TooManyTagsException

Aplicación creada con demasiadas etiquetas o se han añadido demasiadas etiquetas a una aplicación. Tenga en cuenta que el número máximo de etiquetas incluye las etiquetas del sistema. El número máximo de etiquetas de la aplicación definidas por el usuario es 50.

Código de estado HTTP: 400

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS Interfaz de línea de comandos V2
- AWS SDK para .NET V4
- AWS SDK para C++
- AWS SDK para Go v2
- AWS SDK para Java V2
- AWS SDK para JavaScript V3
- AWS SDK para Kotlin
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK para Ruby V3

UpdateApplication

> **Note**
>
> Esta documentación es para la versión 1 de la API de Amazon Kinesis Data Analytics, que solo admite aplicaciones SQL. La versión 2 de la API admite aplicaciones SQL y Java. Para obtener más información acerca de la versión 2, consulte la documentación de la API V2 de Amazon Kinesis Data Analytics.

Actualiza una aplicación Amazon Kinesis Analytics existente. Con esta API, puede actualizar el código de la aplicación, la configuración de entrada y la configuración de salida.

Tenga en cuenta que Amazon Kinesis Analytics actualiza `CurrentApplicationVersionId` cada vez que actualiza la aplicación.

Esta operación necesita permiso para la acción `kinesisanalytics:UpdateApplication`.

Sintaxis de la solicitud

```
{
   "ApplicationName": "string",
   "ApplicationUpdate": {
      "ApplicationCodeUpdate": "string",
      "CloudWatchLoggingOptionUpdates": [
         {
            "CloudWatchLoggingOptionId": "string",
            "LogStreamARNUpdate": "string",
            "RoleARNUpdate": "string"
         }
      ],
      "InputUpdates": [
         {
            "InputId": "string",
            "InputParallelismUpdate": {
               "CountUpdate": number
            },
            "InputProcessingConfigurationUpdate": {
               "InputLambdaProcessorUpdate": {
                  "ResourceARNUpdate": "string",
                  "RoleARNUpdate": "string"
               }
```

```
            },
            "InputSchemaUpdate": {
               "RecordColumnUpdates": [
                  {
                     "Mapping": "string",
                     "Name": "string",
                     "SqlType": "string"
                  }
               ],
               "RecordEncodingUpdate": "string",
               "RecordFormatUpdate": {
                  "MappingParameters": {
                     "CSVMappingParameters": {
                        "RecordColumnDelimiter": "string",
                        "RecordRowDelimiter": "string"
                     },
                     "JSONMappingParameters": {
                        "RecordRowPath": "string"
                     }
                  },
                  "RecordFormatType": "string"
               }
            },
            "KinesisFirehoseInputUpdate": {
               "ResourceARNUpdate": "string",
               "RoleARNUpdate": "string"
            },
            "KinesisStreamsInputUpdate": {
               "ResourceARNUpdate": "string",
               "RoleARNUpdate": "string"
            },
            "NamePrefixUpdate": "string"
         }
      ],
      "OutputUpdates": [
         {
            "DestinationSchemaUpdate": {
               "RecordFormatType": "string"
            },
            "KinesisFirehoseOutputUpdate": {
               "ResourceARNUpdate": "string",
               "RoleARNUpdate": "string"
            },
            "KinesisStreamsOutputUpdate": {
```

```
            "ResourceARNUpdate": "string",
            "RoleARNUpdate": "string"
         },
         "LambdaOutputUpdate": {
            "ResourceARNUpdate": "string",
            "RoleARNUpdate": "string"
         },
         "NameUpdate": "string",
         "OutputId": "string"
      }
   ],
   "ReferenceDataSourceUpdates": [
      {
         "ReferenceId": "string",
         "ReferenceSchemaUpdate": {
            "RecordColumns": [
               {
                  "Mapping": "string",
                  "Name": "string",
                  "SqlType": "string"
               }
            ],
            "RecordEncoding": "string",
            "RecordFormat": {
               "MappingParameters": {
                  "CSVMappingParameters": {
                     "RecordColumnDelimiter": "string",
                     "RecordRowDelimiter": "string"
                  },
                  "JSONMappingParameters": {
                     "RecordRowPath": "string"
                  }
               },
               "RecordFormatType": "string"
            }
         },
         "S3ReferenceDataSourceUpdate": {
            "BucketARNUpdate": "string",
            "FileKeyUpdate": "string",
            "ReferenceRoleARNUpdate": "string"
         },
         "TableNameUpdate": "string"
      }
   ]
```

```
    },
    "CurrentApplicationVersionId": number
}
```

Parámetros de la solicitud

La solicitud acepta los siguientes datos en formato JSON.

ApplicationName

Nombre de la aplicación Amazon Kinesis Analytics que se va a actualizar.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. Longitud máxima de 128.

Patrón: `[a-zA-Z0-9_.-]+`

Obligatorio: sí

ApplicationUpdate

Describe las actualizaciones de la aplicación.

Tipo: objeto ApplicationUpdate

Obligatorio: sí

CurrentApplicationVersionId

El ID de la versión de la aplicación actual. Para obtener este valor, puede usar la operación DescribeApplication.

Tipo: largo

Rango válido: valor mínimo de 1. Valor máximo de 999999999.

Obligatorio: sí

Elementos de respuesta

Si la acción se realiza correctamente, el servicio devuelve una respuesta HTTP 200 con un cuerpo HTTP vacío.

Errores

CodeValidationException

El código de aplicación (consulta) proporcionado por el usuario no es válido. Puede tratarse de un simple error de sintaxis.

message

Test

Código de estado HTTP: 400

ConcurrentModificationException

Se produce una excepción como resultado de una modificación simultánea de una aplicación. Por ejemplo, dos personas que intentan editar la misma aplicación al mismo tiempo.

message

Código de estado HTTP: 400

InvalidArgumentException

El valor del parámetro de entrada especificado no es válido.

message

Código de estado HTTP: 400

ResourceInUseException

La aplicación no está disponible para esta operación.

message

Código de estado HTTP: 400

ResourceNotFoundException

No se encuentra la aplicación especificada.

message

Código de estado HTTP: 400

UnsupportedOperationException

La solicitud se rechazó porque no se admite un parámetro especificado o porque un recurso especificado no es válido para esta operación.

Código de estado HTTP: 400

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS Interfaz de línea de comandos V2
- AWS SDK para .NET V4
- AWS SDK para C++
- AWS SDK para Go v2
- AWS SDK para Java V2
- AWS SDK para JavaScript V3
- AWS SDK para Kotlin
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK para Ruby V3

Data Types

Los siguientes tipos de datos son compatibles:

- ApplicationDetail
- ApplicationSummary
- ApplicationUpdate
- CloudWatchLoggingOption
- CloudWatchLoggingOptionDescription
- CloudWatchLoggingOptionUpdate
- CSVMappingParameters

- DestinationSchema
- Input
- InputConfiguration
- InputDescription
- InputLambdaProcessor
- InputLambdaProcessorDescription
- InputLambdaProcessorUpdate
- InputParallelism
- InputParallelismUpdate
- InputProcessingConfiguration
- InputProcessingConfigurationDescription
- InputProcessingConfigurationUpdate
- InputSchemaUpdate
- InputStartingPositionConfiguration
- InputUpdate
- JSONMappingParameters
- KinesisFirehoseInput
- KinesisFirehoseInputDescription
- KinesisFirehoseInputUpdate
- KinesisFirehoseOutput
- KinesisFirehoseOutputDescription
- KinesisFirehoseOutputUpdate
- KinesisStreamsInput
- KinesisStreamsInputDescription
- KinesisStreamsInputUpdate
- KinesisStreamsOutput
- KinesisStreamsOutputDescription
- KinesisStreamsOutputUpdate
- LambdaOutput
- LambdaOutputDescription

- LambdaOutputUpdate
- MappingParameters
- Output
- OutputDescription
- OutputUpdate
- RecordColumn
- RecordFormat
- ReferenceDataSource
- ReferenceDataSourceDescription
- ReferenceDataSourceUpdate
- S3Configuration
- S3ReferenceDataSource
- S3ReferenceDataSourceDescription
- S3ReferenceDataSourceUpdate
- SourceSchema
- Tag

ApplicationDetail

> **Note**
>
> Esta documentación es para la versión 1 de la API de Amazon Kinesis Data Analytics, que solo admite aplicaciones SQL. La versión 2 de la API admite aplicaciones SQL y Java. Para obtener más información acerca de la versión 2, consulte la documentación de la API V2 de Amazon Kinesis Data Analytics.

Proporciona una descripción de la aplicación, como el nombre de recurso de Amazon (ARN) de la aplicación, el estado, la última versión y los detalles de configuración de entrada y salida.

Contenido

ApplicationARN

ARN de la aplicación.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: sí

ApplicationName

Nombre de la aplicación.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. Longitud máxima de 128.

Patrón: `[a-zA-Z0-9_.-]+`

Obligatorio: sí

ApplicationStatus

Estado de una aplicación.

Tipo: cadena

Valores válidos: `DELETING | STARTING | STOPPING | READY | RUNNING | UPDATING | AUTOSCALING`

Obligatorio: sí

ApplicationVersionId

Proporciona la versión de la aplicación actual.

Tipo: largo

Rango válido: valor mínimo de 1. Valor máximo de 999999999.

Obligatorio: sí

ApplicationCode

Devuelve el código de la aplicación que proporcionó para realizar el análisis de datos en cualquiera de los flujos de la aplicación.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 0. Longitud máxima de 102400.

Obligatorio: no

ApplicationDescription

Descripción de la aplicación.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 0. La longitud máxima es 1024.

Obligatorio: no

CloudWatchLoggingOptionDescriptions

Describe los flujos de CloudWatch registro que están configurados para recibir los mensajes de la aplicación. Para obtener más información sobre el uso de transmisiones de CloudWatch registros con las aplicaciones de Amazon Kinesis Analytics, consulte Trabajar con CloudWatch Amazon Logs.

Tipo: matriz de objetos CloudWatchLoggingOptionDescription

Obligatorio: no

CreateTimestamp

La marca de tiempo de creación de la versión de la aplicación.

Tipo: marca temporal

Obligatorio: no

InputDescriptions

Describe la configuración de entrada de la aplicación. Para obtener más información, consulte Configuración de entrada de la aplicación.

Tipo: matriz de objetos InputDescription

Obligatorio: no

LastUpdateTimestamp

Marca de tiempo en que la aplicación se actualizó por última vez.

Tipo: marca temporal

Obligatorio: no

OutputDescriptions

Describe la configuración de salida de la aplicación. Para obtener más información, consulte Configuración de salida de la aplicación.

Tipo: matriz de objetos OutputDescription

Obligatorio: no

ReferenceDataSourceDescriptions

Describe los orígenes de datos de referencia configurados para una aplicación. Para obtener más información, consulte Configuración de entrada de la aplicación.

Tipo: matriz de objetos ReferenceDataSourceDescription

Obligatorio: no

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

ApplicationSummary

> **Note**
>
> Esta documentación es para la versión 1 de la API de Amazon Kinesis Data Analytics, que solo admite aplicaciones SQL. La versión 2 de la API admite aplicaciones SQL y Java. Para obtener más información acerca de la versión 2, consulte la documentación de la API V2 de Amazon Kinesis Data Analytics.

Proporciona información resumida de la aplicación, incluido el nombre de recurso de Amazon (ARN) de la aplicación, el nombre y el estado.

Contenido

ApplicationARN

ARN de la aplicación.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: sí

ApplicationName

Nombre de la aplicación.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. Longitud máxima de 128.

Patrón: `[a-zA-Z0-9_.-]+`

Obligatorio: sí

ApplicationStatus

Estado de una aplicación.

Tipo: cadena

Valores válidos: `DELETING | STARTING | STOPPING | READY | RUNNING | UPDATING | AUTOSCALING`

Obligatorio: sí

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

ApplicationUpdate

Describe las actualizaciones que se deben aplicar a una aplicación Amazon Kinesis Analytics existente.

Contenido

ApplicationCodeUpdate

Describe las actualizaciones del código de la aplicación.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 0. Longitud máxima de 102400.

Obligatorio: no

CloudWatchLoggingOptionUpdates

Describe las actualizaciones de las opciones de CloudWatch registro de aplicaciones.

Tipo: matriz de objetos CloudWatchLoggingOptionUpdate

Obligatorio: no

InputUpdates

Describe las actualizaciones de configuración de entrada de la aplicación.

Tipo: matriz de objetos InputUpdate

Obligatorio: no

OutputUpdates

Describe las actualizaciones de configuración de salida de la aplicación.

Tipo: matriz de objetos OutputUpdate

Obligatorio: no

ReferenceDataSourceUpdates

Describe las actualizaciones de los orígenes de datos de referencia de la aplicación.

Tipo: matriz de objetos ReferenceDataSourceUpdate

Obligatorio: no

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

CloudWatchLoggingOption

Proporciona una descripción de las opciones de CloudWatch registro, incluidos el nombre de recurso de Amazon (ARN) del flujo de registro y el ARN del rol.

Contenido

LogStreamARN

ARN del CloudWatch registro para recibir los mensajes de la aplicación.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: sí

RoleARN

ARN de IAM del rol que se utilizará para enviar los mensajes de la aplicación. Nota: Para escribir los mensajes de la aplicación CloudWatch, la función de IAM que se utilice debe tener habilitada la acción `PutLogEvents` de política.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: sí

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

CloudWatchLoggingOptionDescription

Descripción de la opción de CloudWatch registro.

Contenido

LogStreamARN

ARN del CloudWatch registro para recibir los mensajes de la aplicación.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: sí

RoleARN

ARN de IAM del rol que se utilizará para enviar los mensajes de la aplicación. Nota: Para escribir los mensajes de la aplicación CloudWatch, la función de IAM utilizada debe tener habilitada la acción `PutLogEvents` de política.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: sí

CloudWatchLoggingOptionId

ID de la descripción de la opción de CloudWatch registro.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 50 caracteres.

Patrón: `[a-zA-Z0-9_.-]+`

Obligatorio: no

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

CloudWatchLoggingOptionUpdate

Describe las actualizaciones de las opciones de CloudWatch registro.

Contenido

CloudWatchLoggingOptionId

ID de la opción de CloudWatch registro que se va a actualizar

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 50 caracteres.

Patrón: `[a-zA-Z0-9_.-]+`

Obligatorio: sí

LogStreamARNUpdate

ARN del CloudWatch registro para recibir los mensajes de la aplicación.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: no

RoleARNUpdate

ARN de IAM del rol que se utilizará para enviar los mensajes de la aplicación. Nota: Para escribir los mensajes de la aplicación CloudWatch, la función de IAM utilizada debe tener habilitada la acción `PutLogEvents` de política.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: no

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

CSVMappingParameters

Ofrece información de mapeo adicional cuando el formato de registro utiliza delimitadores, como por ejemplo, CSV. Por ejemplo, los siguientes registros de ejemplo utilizan el formato CSV, donde los registros utilizan '\n' como el delimitador de la fila y una coma (“,”) como el delimitador de la columna:

```
"name1", "address1"

"name2", "address2"
```

Contenido

RecordColumnDelimiter

Delimitador de columnas. Por ejemplo, en un formato CSV, el delimitador de columnas típico es una coma (“,”).

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1.

Obligatorio: sí

RecordRowDelimiter

Delimitador de filas. Por ejemplo, en un formato CSV, el delimitador de filas típico es '\n'.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1.

Obligatorio: sí

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

DestinationSchema

Describe el formato de datos cuando se escriben registros en el destino. Para obtener más información, consulte Configuración de salida de la aplicación.

Contenido

RecordFormatType

Especifica el formato de los registros en el flujo de salida.

Tipo: cadena

Valores válidos: `JSON | CSV`

Obligatorio: sí

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

Input

Al configurar la entrada de la aplicación, debe especificar el origen de streaming, el nombre de la secuencia en la aplicación que se crea y el mapeo entre los dos. Para obtener más información, consulte Configuración de entrada de la aplicación.

Contenido

InputSchema

Describe el formato de los datos del origen de transmisión y cómo se asigna cada elemento de datos a las columnas correspondientes del flujo en la aplicación que se crea.

También se utiliza para describir el formato del origen de datos de referencia.

Tipo: objeto SourceSchema

Obligatorio: sí

NamePrefix

El prefijo del nombre que se utilizará al crear el flujo en la aplicación. Suponga que especifica un prefijo "»MyInApplicationStream. A continuación, Amazon Kinesis Analytics crea una o más transmisiones en `InputParallelism` la aplicación (según el recuento que haya especificado) con los nombres MyInApplicationStream "_001», MyInApplicationStream "_002», etc.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 32 caracteres.

Obligatorio: sí

InputParallelism

Describe el número de flujos en la aplicación que se crearán.

Los datos del origen se dirigen a estas secuencias de entrada en la aplicación.

Consulte Configuración de entrada de la aplicación.

Tipo: objeto InputParallelism

Obligatorio: no

InputProcessingConfiguration

La InputProcessingConfiguration para la entrada. Un procesador de entrada transforma los registros a medida que se reciben desde el flujo, antes de la ejecución del código SQL de la aplicación. Actualmente, la única configuración de procesamiento de entrada disponible es InputLambdaProcessor.

Tipo: objeto InputProcessingConfiguration

Obligatorio: no

KinesisFirehoseInput

Si el origen de streaming es una secuencia de entrega de Amazon Kinesis Firehose, identifica el ARN de la secuencia de entrega y un rol de IAM que permite a Amazon Kinesis Analytics obtener acceso a la secuencia en su nombre.

Nota: `KinesisStreamsInput` o `KinesisFirehoseInput` son necesarios.

Tipo: objeto KinesisFirehoseInput

Obligatorio: no

KinesisStreamsInput

Si el origen de streaming es una secuencia de entrega de Amazon Kinesis Firehose, identifica el Nombre de recurso de Amazon (ARN) de la secuencia y un rol de IAM que permite a Amazon Kinesis Analytics obtener acceso a la secuencia en su nombre.

Nota: `KinesisStreamsInput` o `KinesisFirehoseInput` son necesarios.

Tipo: objeto KinesisStreamsInput

Obligatorio: no

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas AWS SDKs específicos, consulte lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2

- AWS SDK para Ruby V3

InputConfiguration

Al iniciar la aplicación, proporciona esta configuración, que identifica la fuente de entrada y el punto del origen de entrada en el que desea que la aplicación comience a procesar los registros.

Contenido

Id

ID del origen de entrada. Para obtener este ID, llame a la operación DescribeApplication.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 50 caracteres.

Patrón: `[a-zA-Z0-9_.-]+`

Obligatorio: sí

InputStartingPositionConfiguration

Punto en el que desea que la aplicación comience a procesar los registros del origen del flujo.

Tipo: objeto InputStartingPositionConfiguration

Obligatorio: sí

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

InputDescription

Describe la configuración de entrada de la aplicación. Para obtener más información, consulte Configuración de entrada de la aplicación.

Contenido

InAppStreamNames

Devuelve los nombres de los flujos de la aplicación que están asignados al origen de flujo.

Tipo: matriz de cadenas

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 32 caracteres.

Obligatorio: no

InputId

ID de entrada asociada con la entrada de la aplicación. Este es el ID que Amazon Kinesis Analytics asigna a cada configuración de entrada que añada a la aplicación.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 50 caracteres.

Patrón: `[a-zA-Z0-9_.-]+`

Obligatorio: no

InputParallelism

Describe el paralelismo configurado (número de flujos en la aplicación asignados al origen de flujo).

Tipo: objeto InputParallelism

Obligatorio: no

InputProcessingConfigurationDescription

La descripción del preprocesador que se ejecuta en los registros de esta entrada antes de ejecutar el código de la aplicación.

Tipo: objeto InputProcessingConfigurationDescription

Obligatorio: no

InputSchema

Describe el formato de los datos del origen de transmisión y cómo se asigna cada elemento de datos a las columnas correspondientes del flujo en la aplicación que se crea.

Tipo: objeto SourceSchema

Obligatorio: no

InputStartingPositionConfiguration

Punto en el que la aplicación está configurada para leer el flujo de entrada.

Tipo: objeto InputStartingPositionConfiguration

Obligatorio: no

KinesisFirehoseInputDescription

Si el origen de streaming es una secuencia de entrega de Amazon Kinesis Firehose, proporciona el ARN de la secuencia de entrega y un rol de IAM que permite a Amazon Kinesis Analytics obtener acceso a la secuencia en su nombre.

Tipo: objeto KinesisFirehoseInputDescription

Obligatorio: no

KinesisStreamsInputDescription

Si el origen de streaming es una secuencia de entrega de Amazon Kinesis Firehose, proporciona el nombre de recurso de Amazon (ARN) de la secuencia de Amazon Kinesis y un rol de IAM que permite a Amazon Kinesis Analytics obtener acceso a la secuencia en su nombre.

Tipo: objeto KinesisStreamsInputDescription

Obligatorio: no

NamePrefix

Prefijo de nombre en la aplicación.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 32 caracteres.

Obligatorio: no

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

InputLambdaProcessor

Objeto que contiene el nombre de recurso de Amazon (ARN) de la función AWS Lambda que se utiliza para preprocesar los registros de la transmisión y el ARN de la función de IAM que se utiliza para acceder a la función Lambda. AWS

Contenido

ResourceARN

Es el ARN de la función de AWS Lambda que opera en los registros del flujo.

> **Note**
>
> Para especificar una versión anterior de la función de Lambda a la última, incluya la versión de la función de Lambda en el ARN de la función de Lambda. Para obtener más información sobre Lambda ARNs, consulte Ejemplo ARNs: Lambda AWS

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: sí

RoleARN

El ARN de la función de IAM que se utiliza para acceder a la función AWS Lambda.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: sí

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

InputLambdaProcessorDescription

Objeto que contiene el nombre de recurso de Amazon (ARN) de la función AWS Lambda que se utiliza para preprocesar los registros de la transmisión y el ARN de la función de IAM que se utiliza para acceder a la expresión Lambda. AWS

Contenido

ResourceARN

El ARN de la función de AWS Lambda que se utiliza para preprocesar los registros del flujo.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: no

RoleARN

El ARN de la función de IAM que se utiliza para acceder a la función AWS Lambda.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: no

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

InputLambdaProcessorUpdate

Representa una actualización de la InputLambdaProcessorque se utiliza para preprocesar los registros de la transmisión.

Contenido

ResourceARNUpdate

Nombre de recurso de Amazon (ARN) de la nueva función de AWS Lambda que se utiliza para preprocesar registros en el flujo.

> **Note**
>
> Para especificar una versión anterior de la función de Lambda a la última, incluya la versión de la función de Lambda en el ARN de la función de Lambda. Para obtener más información sobre Lambda ARNs, consulte Ejemplo ARNs: Lambda AWS

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: no

RoleARNUpdate

El ARN de la nueva función de IAM que se utiliza para acceder a la función AWS Lambda.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: no

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

InputParallelism

Describe el número de secuencias en la aplicación para crear un origen de streaming determinado. Para obtener más información sobre paralelismos, consulte Configuración de entrada de la aplicación.

Contenido

Count

El número de secuencias en la aplicación que se deben crear. Para obtener más información, consulte Límites.

Tipo: entero

Rango válido: valor mínimo de 1. Valor máximo de 64.

Obligatorio: no

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

InputParallelismUpdate

Proporciona actualizaciones del recuento de paralelismos.

Contenido

CountUpdate

Número de secuencias en la aplicación para crear un origen de streaming específico.

Tipo: entero

Rango válido: valor mínimo de 1. Valor máximo de 64.

Obligatorio: no

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

InputProcessingConfiguration

Proporciona una descripción de un procesador que se utiliza para procesar previamente los registros en el flujo antes de que los procese el código de la aplicación. Actualmente, el único procesador de entrada disponible es AWS Lambda.

Contenido

InputLambdaProcessor

El InputLambdaProcessor que se utiliza para el procesamiento previo de los registros en el flujo antes de que los procese el código de la aplicación.

Tipo: objeto InputLambdaProcessor

Obligatorio: sí

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

InputProcessingConfigurationDescription

Proporciona información de configuración acerca de un procesador de entrada. Actualmente, el único procesador de entrada disponible es AWS Lambda.

Contenido

InputLambdaProcessorDescription

Proporciona información de configuración sobre el asociado InputLambdaProcessorDescription.

Tipo: objeto InputLambdaProcessorDescription

Obligatorio: no

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

InputProcessingConfigurationUpdate

Describe las actualizaciones de una InputProcessingConfiguration.

Contenido

InputLambdaProcessorUpdate

Proporciona información de actualización de un InputLambdaProcessor.

Tipo: objeto InputLambdaProcessorUpdate

Obligatorio: sí

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

InputSchemaUpdate

Describe las actualizaciones del esquema de entrada de la aplicación.

Contenido

RecordColumnUpdates

Una lista de objetos `RecordColumn`. Cada objeto describe la asignación del elemento de origen del flujo y la columna correspondiente en la secuencia en la aplicación.

Tipo: matriz de objetos RecordColumn

Miembros de la matriz: número mínimo de 1 artículo. La cantidad máxima es de 1000 elementos.

Obligatorio: no

RecordEncodingUpdate

Especifica la codificación de los registros en el origen de transmisión. Por ejemplo, UTF-8.

Tipo: cadena

Patrón: `UTF-8`

Obligatorio: no

RecordFormatUpdate

Especifica el formato de los registros en el origen de transmisión.

Tipo: objeto RecordFormat

Obligatorio: no

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

InputStartingPositionConfiguration

Describe el punto en el que la aplicación lee del origen del flujo.

Contenido

InputStartingPosition

La posición de inicio del flujo.

- `NOW` - Empieza a leer justo después del registro más reciente del flujo y comienza con la marca de tiempo de la solicitud que emitió el cliente.
- `TRIM_HORIZON` - Empieza a leer a partir del último registro no sin recortar del flujo, que es el registro más antiguo disponible en el flujo. Esta opción no está disponible para un flujo de entrega de Amazon Kinesis Firehose.
- `LAST_STOPPED_POINT` - Reanuda la lectura desde el punto en que la aplicación dejó de leer por última vez.

Tipo: cadena

Valores válidos: `NOW | TRIM_HORIZON | LAST_STOPPED_POINT`

Obligatorio: no

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

InputUpdate

Describe las actualizaciones de una configuración de entrada específica (identificada por la `InputId` de una aplicación).

Contenido

InputId

ID de entrada de la entrada de la aplicación que se va a actualizar.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 50 caracteres.

Patrón: `[a-zA-Z0-9_.-]+`

Obligatorio: sí

InputParallelismUpdate

Describe las actualizaciones de paralelismo (el número de flujos en la aplicación que Amazon Kinesis Analytics crea para el origen de flujo específico).

Tipo: objeto InputParallelismUpdate

Obligatorio: no

InputProcessingConfigurationUpdate

Describe las actualizaciones de una configuración de procesamiento de entradas.

Tipo: objeto InputProcessingConfigurationUpdate

Obligatorio: no

InputSchemaUpdate

Describe el formato de los datos del origen de streaming y cómo se asigna cada elemento de datos en el origen de streaming a las columnas correspondientes de la secuencia en la aplicación que se crea.

Tipo: objeto InputSchemaUpdate

Obligatorio: no

KinesisFirehoseInputUpdate

Si el origen de flujo que se va a actualizar es un flujo de entrega de Amazon Kinesis Firehose, proporciona un ARN de flujo actualizado y un ARN de rol de IAM.

Tipo: objeto KinesisFirehoseInputUpdate

Obligatorio: no

KinesisStreamsInputUpdate

Si el origen de flujo que se va a actualizar es un flujo de Amazon Kinesis, proporciona un nombre de recurso de Amazon (ARN) de flujo actualizado y un ARN de rol de IAM.

Tipo: objeto KinesisStreamsInputUpdate

Obligatorio: no

NamePrefixUpdate

Prefijo de nombre para los flujos dentro de la aplicación que Amazon Kinesis Analytics crea para el origen de flujo específico.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 32 caracteres.

Obligatorio: no

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

JSONMappingParameters

Proporciona información adicional de la asignación cuando JSON es el formato de registro en el origen de transmisión.

Contenido

RecordRowPath

Ruta del origen de nivel superior que contiene los registros.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1.

Obligatorio: sí

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

KinesisFirehoseInput

Identifica un flujo de entrega de Amazon Kinesis Firehose como el origen de streaming. Debe proporcionar el Nombre de recurso de Amazon (ARN) del flujo de entrega y el ARN del rol de IAM que permite que Amazon Kinesis Analytics acceda al flujo en su nombre.

Contenido

ResourceARN

ARN del flujo de entrega de entrada.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: sí

RoleARN

El ARN del rol de IAM que Amazon Kinesis Analytics puede asumir para acceder al flujo en su nombre. Asegúrese de que el rol tiene los permisos necesarios para obtener acceso al flujo.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: sí

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

KinesisFirehoseInputDescription

Describe el flujo de entrega de Amazon Kinesis Firehose que se configura como origen del flujo en la configuración de entrada de la aplicación.

Contenido

ResourceARN

El nombre de recurso de Amazon (ARN) del flujo de entrega de Amazon Kinesis Firehose.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: no

RoleARN

El ARN del rol de IAM que Amazon Kinesis Analytics asume para acceder al flujo.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: no

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

KinesisFirehoseInputUpdate

Al actualizar la configuración de entrada de la aplicación, proporciona información sobre un flujo de entrega de Amazon Kinesis Firehose como el origen del flujo.

Contenido

ResourceARNUpdate

Nombre de recurso de Amazon (ARN) del flujo de entrada de Amazon Kinesis Firehose que se va a leer.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: no

RoleARNUpdate

El ARN del rol de IAM que Amazon Kinesis Analytics puede asumir para acceder al flujo en su nombre. Debe conceder los permisos necesarios a este rol.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: no

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

KinesisFirehoseOutput

Al configurar la salida de la aplicación, identifica una secuencia de entrega de Amazon Kinesis Firehose como destino. Debe proporcionar el nombre de recurso de Amazon (ARN) y el ARN del rol de IAM de la secuencia que permite que Amazon Kinesis Analytics escriba en la secuencia en su nombre.

Contenido

ResourceARN

El ARN de la secuencia de entrega de Amazon Kinesis Firehose de destino en el que se va a escribir.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: sí

RoleARN

El ARN del rol de IAM que Amazon Kinesis Analytics puede asumir para escribir en el flujo de destino en su nombre. Debe conceder los permisos necesarios a este rol.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: sí

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2

- AWS SDK para Ruby V3

KinesisFirehoseOutputDescription

Para el resultado de una aplicación, describe el flujo de Amazon Kinesis Firehose configurado como destino.

Contenido

ResourceARN

El nombre de recurso de Amazon (ARN) del flujo de entrega de Amazon Kinesis Firehose.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: no

RoleARN

El ARN del rol de IAM que Amazon Kinesis Analytics puede asumir para acceder al flujo.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: no

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

KinesisFirehoseOutputUpdate

Al actualizar una configuración de salida mediante la UpdateApplicationoperación, proporciona información sobre una transmisión de entrega de Amazon Kinesis Firehose configurada como destino.

Contenido

ResourceARNUpdate

El nombre de recurso de Amazon (ARN) del flujo de Amazon Kinesis Firehose de destino en el que se va a escribir.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: no

RoleARNUpdate

El ARN del rol de IAM que Amazon Kinesis Analytics puede asumir para acceder al flujo en su nombre. Debe conceder los permisos necesarios a este rol.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: no

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2

- AWS SDK para Ruby V3

KinesisStreamsInput

Identifica un flujo de Amazon Kinesis como el origen de streaming. Debe proporcionar el Nombre de recurso de Amazon (ARN) del flujo y el ARN del rol de IAM que permite que Amazon Kinesis Analytics acceda al flujo en su nombre.

Contenido

ResourceARN

El ARN del flujo de entrada de Amazon Kinesis que se va a leer.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: sí

RoleARN

El ARN del rol de IAM que Amazon Kinesis Analytics puede asumir para acceder al flujo en su nombre. Debe conceder los permisos necesarios a este rol.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: sí

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

KinesisStreamsInputDescription

Describe el flujo de Amazon Kinesis que se configura como origen de flujo en la configuración de entrada de la aplicación.

Contenido

ResourceARN

El nombre de recurso de Amazon (ARN) del flujo de datos de Amazon Kinesis.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: no

RoleARN

El ARN del rol de IAM que Amazon Kinesis Analytics puede asumir para acceder al flujo.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: no

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

KinesisStreamsInputUpdate

Al actualizar la configuración de entrada de la aplicación, proporciona información sobre un flujo de Amazon Kinesis como el origen del flujo.

Contenido

ResourceARNUpdate

Nombre de recurso de Amazon (ARN) del flujo de entrada de Amazon Kinesis que se va a leer.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: no

RoleARNUpdate

El ARN del rol de IAM que Amazon Kinesis Analytics puede asumir para acceder al flujo en su nombre. Debe conceder los permisos necesarios a este rol.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: no

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

KinesisStreamsOutput

Al configurar la salida de la aplicación, identifica un flujo de Amazon Kinesis como destino. Debe proporcionar el Nombre de recurso de Amazon (ARN) del flujo y el ARN del rol de IAM que Amazon Kinesis Analytics puede utilizar para escribir en el flujo en su nombre.

Contenido

ResourceARN

El ARN del flujo de Amazon Kinesis de destino en el que se va a escribir.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: sí

RoleARN

El ARN del rol de IAM que Amazon Kinesis Analytics puede asumir para escribir en el flujo de destino en su nombre. Debe conceder los permisos necesarios a este rol.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: sí

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

KinesisStreamsOutputDescription

Para el resultado de una aplicación, describe el flujo de Amazon Kinesis configurada como destino.

Contenido

ResourceARN

El nombre de recurso de Amazon (ARN) del flujo de datos de Amazon Kinesis.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: no

RoleARN

El ARN del rol de IAM que Amazon Kinesis Analytics puede asumir para acceder al flujo.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: no

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

KinesisStreamsOutputUpdate

Al actualizar una configuración de salida mediante la UpdateApplicationoperación, proporciona información sobre una transmisión de Amazon Kinesis configurada como destino.

Contenido

ResourceARNUpdate

El nombre de recurso de Amazon (ARN) del flujo de Amazon Kinesis de destino en el que quiera escribir el resultado.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: no

RoleARNUpdate

El ARN del rol de IAM que Amazon Kinesis Analytics puede asumir para acceder al flujo en su nombre. Debe conceder los permisos necesarios a este rol.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: no

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

LambdaOutput

Al configurar la salida de la aplicación, identifica una función AWS Lambda como destino. Proporciona el nombre de recurso de Amazon (ARN) de la función y el ARN del rol de IAM que Amazon Kinesis Analytics puede utilizar para escribir en la función en su nombre.

Contenido

ResourceARN

El nombre de recurso de Amazon (ARN) de la función de Lambda de destino en la que se va a escribir.

> **Note**
>
> Para especificar una versión anterior de la función de Lambda a la última, incluya la versión de la función de Lambda en el ARN de la función de Lambda. Para obtener más información sobre Lambda ARNs, consulte Ejemplo ARNs: Lambda AWS

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: sí

RoleARN

El ARN del rol de IAM que Amazon Kinesis Analytics puede asumir para escribir en la función de destino en su nombre. Debe conceder los permisos necesarios a este rol.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: sí

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

LambdaOutputDescription

Para el resultado de una aplicación, describe la función AWS Lambda configurada como su destino.

Contenido

ResourceARN

El nombre de recurso de Amazon (ARN) de la función de Lambda de destino.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: no

RoleARN

El ARN del rol de IAM que Amazon Kinesis Analytics puede asumir para escribir en la función de destino.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: no

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

LambdaOutputUpdate

Al actualizar una configuración de salida mediante la UpdateApplicationoperación, proporciona información sobre una función AWS Lambda configurada como destino.

Contenido

ResourceARNUpdate

El nombre de recurso de Amazon (ARN) de la función de Lambda de destino.

> **Note**
>
> Para especificar una versión anterior de la función de Lambda a la última, incluya la versión de la función de Lambda en el ARN de la función de Lambda. Para obtener más información sobre Lambda ARNs, consulte Ejemplo ARNs: Lambda AWS

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: no

RoleARNUpdate

El ARN del rol de IAM que Amazon Kinesis Analytics puede asumir para escribir en la función de destino en su nombre. Debe conceder los permisos necesarios a este rol.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: no

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

MappingParameters

Al configurar de entrada de la aplicación en el momento de crear o actualizar una aplicación, proporciona información adicional acerca del mapeo específica para el formato de registro (como JSON, CSV o campos de registro delimitados por algún delimitador) en el origen de streaming.

Contenido

CSVMappingParameters

Ofrece información adicional de la asignación cuando el formato de registro utiliza delimitadores (por ejemplo, CSV).

Tipo: objeto CSVMappingParameters

Obligatorio: no

JSONMappingParameters

Proporciona información adicional de la asignación cuando JSON es el formato de registro en el origen de transmisión.

Tipo: objeto JSONMappingParameters

Obligatorio: no

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

Output

Describe la configuración de salida de la aplicación en la que identifica una secuencia en la aplicación y un destino donde desea que se escriban los datos de la secuencia en la aplicación. El destino puede ser un flujo de Amazon Kinesis o un flujo de entrega de Amazon Kinesis Firehose.

Para los límites sobre la cantidad de destinos que una aplicación puede escribir y otras limitaciones, consulte Límites.

Contenido

DestinationSchema

Describe el formato de datos cuando se escriben registros en el destino. Para obtener más información, consulte Configuración de salida de la aplicación.

Tipo: objeto DestinationSchema

Obligatorio: sí

Name

El nombre del flujo en la aplicación.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 32 caracteres.

Obligatorio: sí

KinesisFirehoseOutput

Identifica un flujo de entrega de Amazon Kinesis Firehose como el destino.

Tipo: objeto KinesisFirehoseOutput

Obligatorio: no

KinesisStreamsOutput

Identifica un flujo de Amazon Kinesis como el destino.

Tipo: objeto KinesisStreamsOutput

Obligatorio: no

LambdaOutput

Identifica una función AWS Lambda como destino.

Tipo: objeto LambdaOutput

Obligatorio: no

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

OutputDescription

Describe la configuración de salida de la aplicación, que incluye el nombre del flujo en la aplicación y el destino en el que se escriben los datos del flujo. El destino puede ser un flujo de Amazon Kinesis o un flujo de entrega de Amazon Kinesis Firehose.

Contenido

DestinationSchema

Formato de datos utilizado para escribir datos en el destino.

Tipo: objeto DestinationSchema

Obligatorio: no

KinesisFirehoseOutputDescription

Describe el flujo de entrega de Amazon Kinesis Firehose como destino en el que se escribe la salida.

Tipo: objeto KinesisFirehoseOutputDescription

Obligatorio: no

KinesisStreamsOutputDescription

Describe el flujo de Amazon Kinesis configurado como destino en el que se escribe la salida.

Tipo: objeto KinesisStreamsOutputDescription

Obligatorio: no

LambdaOutputDescription

Describe la función AWS Lambda configurada como el destino donde se escribe la salida.

Tipo: objeto LambdaOutputDescription

Obligatorio: no

Name

El nombre del flujo en la aplicación configurado como salida.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 32 caracteres.

Obligatorio: no

OutputId

Identificador único para la configuración de salida.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 50 caracteres.

Patrón: `[a-zA-Z0-9_.-]+`

Obligatorio: no

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

OutputUpdate

Describe las actualizaciones de la configuración de salida identificadas por `OutputId`.

Contenido

OutputId

Identifica la configuración de salida específica que desea actualizar.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 50 caracteres.

Patrón: `[a-zA-Z0-9_.-]+`

Obligatorio: sí

DestinationSchemaUpdate

Describe el formato de datos cuando se escriben registros en el destino. Para obtener más información, consulte Configuración de salida de la aplicación.

Tipo: objeto DestinationSchema

Obligatorio: no

KinesisFirehoseOutputUpdate

Describe el flujo de entrega de Amazon Kinesis Firehose configurado como destino para la salida.

Tipo: objeto KinesisFirehoseOutputUpdate

Obligatorio: no

KinesisStreamsOutputUpdate

Describe el flujo de Amazon Kinesis como destino para la salida.

Tipo: objeto KinesisStreamsOutputUpdate

Obligatorio: no

LambdaOutputUpdate

Describe una función AWS Lambda como destino de la salida.

Tipo: objeto LambdaOutputUpdate

Obligatorio: no

NameUpdate

Si desea especificar un flujo diferente en la aplicación para esta configuración de salida, utilice este campo y especifique el nombre del nuevo flujo en la aplicación.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 32 caracteres.

Obligatorio: no

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

RecordColumn

Describe el mapeo de cada elemento de datos en el origen de streaming a la columna correspondiente en la secuencia en la aplicación.

También se utiliza para describir el formato del origen de datos de referencia.

Contenido

Name

El nombre de la columna creado en el flujo de entrada en la aplicación o la tabla de referencia.

Tipo: cadena

Obligatorio: sí

SqlType

El tipo de columna creado en el flujo de entrada en la aplicación o la tabla de referencia.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1.

Obligatorio: sí

Mapping

Consulte el elemento de datos en la entrada de streaming o el origen de datos de referencia. Este elemento es obligatorio si lo RecordFormatTypees`JSON`.

Tipo: cadena

Requerido: no

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2

- AWS SDK para Ruby V3

RecordFormat

Describe el formato de registro y la información de mapeo pertinente que debe aplicarse para esquematizar los registros en la secuencia.

Contenido

RecordFormatType

El tipo de formato de registro.

Tipo: cadena

Valores válidos: `JSON | CSV`

Obligatorio: sí

MappingParameters

Al configurar de entrada de la aplicación en el momento de crear o actualizar una aplicación, proporciona información adicional acerca del mapeo específica para el formato de registro (como JSON, CSV o campos de registro delimitados por algún delimitador) en el origen de streaming.

Tipo: objeto MappingParameters

Obligatorio: no

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

ReferenceDataSource

Describe el origen de datos de referencia proporcionando la información de origen (nombre del bucket de S3 y nombre de la clave de objeto), el nombre de la tabla en la aplicación resultante que se crea y el esquema necesario para asignar los elementos de datos del objeto de Amazon S3 a la tabla en la aplicación.

Contenido

ReferenceSchema

Describe el formato de los datos del origen de transmisión y cómo se asigna cada elemento de datos a las columnas correspondientes creadas en el flujo en la aplicación.

Tipo: objeto SourceSchema

Obligatorio: sí

TableName

El nombre de la tabla en la aplicación que se va a crear.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 32 caracteres.

Obligatorio: sí

S3ReferenceDataSource

Identifica el bucket de S3 y el objeto que contiene los datos de referencia. También identifica el rol de IAM que Amazon Kinesis Analytics puede asumir para leer este objeto en su nombre. Una aplicación de Amazon Kinesis Analytics carga los datos de referencia solo una vez. Si los datos cambian, debe llamar a la operación `UpdateApplication` para activar la recarga de datos en su aplicación.

Tipo: objeto S3ReferenceDataSource

Obligatorio: no

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

ReferenceDataSourceDescription

Describe el origen de datos de referencia configurado para una aplicación.

Contenido

ReferenceId

El ID del origen de datos de referencia. Este es el ID que Amazon Kinesis Analytics asigna al añadir la fuente de datos de referencia a la aplicación mediante AddApplicationReferenceDataSourcela operación.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 50 caracteres.

Patrón: `[a-zA-Z0-9_.-]+`

Obligatorio: sí

S3ReferenceDataSourceDescription

Proporciona el nombre del bucket de S3 y el nombre de clave de objeto que contiene los datos de referencia. También proporciona el nombre de recurso de Amazon (ARN) del rol de IAM que Amazon Kinesis Analytics puede asumir para leer el objeto de Amazon S3 y rellenar la tabla de referencia de la aplicación.

Tipo: objeto S3ReferenceDataSourceDescription

Obligatorio: sí

TableName

El nombre de la tabla de la aplicación creado por la configuración del origen de datos de referencia específica.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 32 caracteres.

Obligatorio: sí

ReferenceSchema

Describe el formato de los datos del origen de transmisión y cómo se asigna cada elemento de datos a las columnas correspondientes creadas en el flujo en la aplicación.

Tipo: objeto SourceSchema

Obligatorio: no

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

ReferenceDataSourceUpdate

Al actualizar la configuración de un origen de datos de referencia para una aplicación, este objeto proporciona todos los valores actualizados (como el nombre clave del objeto y el nombre del bucket de origen), el nombre de la tabla en la aplicación que se crea y la información de asignación actualizada que asigna los datos del objeto de Amazon S3 a la tabla de referencia en la aplicación que se crea.

Contenido

ReferenceId

El ID del origen de datos de referencia que se está actualizando. Para obtener este valor, puede usar la operación DescribeApplication.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 50 caracteres.

Patrón: `[a-zA-Z0-9_.-]+`

Obligatorio: sí

ReferenceSchemaUpdate

Describe el formato de los datos del origen de transmisión y cómo se asigna cada elemento de datos a las columnas correspondientes creadas en el flujo en la aplicación.

Tipo: objeto SourceSchema

Obligatorio: no

S3ReferenceDataSourceUpdate

Describe el nombre del bucket de S3, el nombre clave del objeto y el rol de IAM que Amazon Kinesis Analytics puede asumir para leer el objeto de Amazon S3 en su nombre y rellenar la tabla de referencia de la aplicación.

Tipo: objeto S3ReferenceDataSourceUpdate

Obligatorio: no

TableNameUpdate

Nombre de la tabla en la aplicación que se crea con esta actualización.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 32 caracteres.

Obligatorio: no

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

S3Configuration

Proporciona una descripción de un origen de datos de Amazon S3, incluido el nombre de recurso de Amazon (ARN) del bucket de S3, el ARN del rol de IAM que se utiliza para acceder al bucket y el nombre del objeto de Amazon S3 que contiene los datos.

Contenido

BucketARN

ARN del bucket de S3 que contiene los datos.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: sí

FileKey

El nombre del objeto que contiene los datos.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 1024 caracteres.

Obligatorio: sí

RoleARN

ARN de IAM del rol utilizado para acceder a los datos.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: sí

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

S3ReferenceDataSource

Identifica el bucket de S3 y el objeto que contiene los datos de referencia. También identifica el rol de IAM que Amazon Kinesis Analytics puede asumir para leer este objeto en su nombre.

Una aplicación de Amazon Kinesis Analytics carga los datos de referencia solo una vez. Si los datos cambian, debe llamar a la operación UpdateApplication para activar la recarga de datos en su aplicación.

Contenido

BucketARN

El nombre de recurso de Amazon (ARN) del bucket de S3.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: sí

FileKey

El nombre de la clave de objeto que contiene los datos de referencia.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 1024 caracteres.

Obligatorio: sí

ReferenceRoleARN

El ARN del rol de IAM que el servicio puede asumir para leer datos en su nombre. Este rol debe tener permiso para ejecutar la acción `s3:GetObject` en el objeto y política de confianza que permite a la entidad principal de servicio de Amazon Kinesis Analytics asumir este rol.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: sí

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

S3ReferenceDataSourceDescription

Proporciona el nombre de bucket y el nombre clave de objeto que almacenan los datos de referencia.

Contenido

BucketARN

El nombre de recurso de Amazon (ARN) del bucket de S3.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: sí

FileKey

El nombre de clave del objeto de Amazon S3.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 1024 caracteres.

Obligatorio: sí

ReferenceRoleARN

El ARN del rol de IAM que Amazon Kinesis Analytics puede asumir leer el objeto de Amazon S3 en su nombre para completar la tabla de referencia de la aplicación.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: sí

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

S3ReferenceDataSourceUpdate

Describe el nombre del bucket de S3, el nombre clave del objeto y el rol de IAM que Amazon Kinesis Analytics puede asumir para leer el objeto de Amazon S3 en su nombre y rellenar la tabla de referencia de la aplicación.

Contenido

BucketARNUpdate

El nombre de recurso de Amazon (ARN) del bucket de S3.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: no

FileKeyUpdate

Nombre clave de objeto.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 1024 caracteres.

Obligatorio: no

ReferenceRoleARNUpdate

El ARN del rol de IAM que Amazon Kinesis Analytics puede asumir leer el objeto de Amazon S3 en su nombre para completar la aplicación.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 2048 caracteres.

Patrón: `arn:.*`

Obligatorio: no

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulte lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

SourceSchema

Describe el formato de los datos del origen de transmisión y cómo se asigna cada elemento de datos a las columnas correspondientes creadas en el flujo en la aplicación.

Contenido

RecordColumns

Una lista de objetos `RecordColumn`.

Tipo: matriz de objetos RecordColumn

Miembros de la matriz: número mínimo de 1 artículo. La cantidad máxima es de 1000 elementos.

Obligatorio: sí

RecordFormat

Especifica el formato de los registros en el origen de transmisión.

Tipo: objeto RecordFormat

Obligatorio: sí

RecordEncoding

Especifica la codificación de los registros en el origen de transmisión. Por ejemplo, UTF-8.

Tipo: cadena

Patrón: `UTF-8`

Obligatorio: no

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

Tag

Un par clave-valor (el valor es opcional) que puede definir y asignar a AWS los recursos. Si especifica una etiqueta que ya existe, el valor de la etiqueta se reemplaza por el valor que especifique en la solicitud. Tenga en cuenta que el número máximo de etiquetas incluye las etiquetas del sistema. El número máximo de etiquetas de la aplicación definidas por el usuario es 50. Para obtener más información, consulte Uso de etiquetas.

Contenido

Key

La clave de la etiqueta clave-valor.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 1. La longitud máxima es de 128 caracteres.

Obligatorio: sí

Value

El valor de la etiqueta clave-valor. El valor es opcional.

Tipo: cadena

Limitaciones de longitud: longitud mínima de 0. La longitud máxima es de 256 caracteres.

Obligatorio: no

Véase también

Para obtener más información sobre el uso de esta API en uno de los idiomas específicos AWS SDKs, consulta lo siguiente:

- AWS SDK para C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

Historial de revisión para Amazon Kinesis Data Analytics

En la siguiente tabla se describen los cambios importantes que se han realizado en la documentación desde la última versión de Amazon Kinesis Data Analytics.

- Versión de API: 14-08-2015
- Última actualización de la documentación: 8 de mayo de 2019

Cambio	Descripción	Fecha
Etiquetado de aplicaciones de Kinesis Data Analytics	Utilice las etiquetas de las aplicaciones para determina r los costos de cada aplicació n, para controlar el acceso o para los fines que defina el usuario. Para obtener más información, consulte Uso del etiquetado.	8 de mayo de 2019
Registro de llamadas a la API de Kinesis Data Analytics con AWS CloudTrail	Amazon Kinesis Data Analytics está integrado AWS CloudTrail con un servicio que proporciona un registro de las acciones realizadas por un usuario, un rol o AWS un servicio en Kinesis Data Analytics. Para obtener más información, consulte Uso AWS CloudTrail.	22 de marzo de 2019
Kinesis Data Analytics disponible en la región de Fráncfort	Kinesis Analytics ya está disponible en la región Europa (Fráncfort). Para obtener más información, consulte Regiones y puntos	18 de julio de 2018

Cambio	Descripción	Fecha
	de conexión: Kinesis Data Analytics.	
Utilizar datos de referencia en la consola	Ahora puede trabajar con los datos de referencia de la aplicación en la consola. Para obtener más informaci ón, consulte Ejemplo: Agregar datos de referencia a una aplicación de Kinesis Data Analytics.	13 de julio de 2018
Ejemplos de consultas en ventana	Ejemplos de aplicaciones para ventanas y agregación. Para obtener más información, consulte Ejemplos: ventanas y agregación.	9 de julio de 2018
Prueba de aplicaciones	Orientación sobre pruebas de cambios de código y esquemas de aplicación. Para obtener más información, consulte Prueba de aplicacio nes.	3 de julio de 2018
Ejemplos de aplicaciones para el procesamiento previo de datos	Ejemplos de código adicional es para REGEX_LOG _PARSE, REGEX_REPLACE y los operadores. DateTime Para obtener más información, consulte Ejemplos: Transform ación de datos.	18 de mayo de 2018

Cambio	Descripción	Fecha
Aumentar el tamaño de las filas devueltas y del código SQL	El tamaño máximo de una fila devuelta se aumenta a 512 KB, y el del código SQL de una aplicación, a 100 KB. Para obtener más información, consulte Límites.	2 de mayo de 2018
AWS Lambda ejemplos de funciones en Java y.NET	Muestras de código para crear funciones de Lambda para el procesamiento previo de registros y como destinos de aplicaciones. Para obtener más información, consulte Creación de funciones de Lambda para el procesamiento previo y Creación de funciones de Lambda como destinos de aplicaciones.	22 de marzo de 2018
Nueva función `HOTSPOTS`	Localizar y devolver información sobre las regiones relativamente densas en los datos. Para obtener más información, consulte Ejemplo: Detección de puntos calientes en una secuencia (función HOTSPOSTS).	19 de marzo de 2018
Función de Lambda como destino	Envío de los resultados del análisis a una función de Lambda como destino. Para obtener más información, consulte Uso de una función de Lambda como salida.	20 de diciembre de 2017

Cambio	Descripción	Fecha
Nueva función `RANDOM_CUT_FOREST_WITH_EXPLANATION`	Explicación de qué campos contribuyen a una puntuación de anomalías en un flujo de datos. Para obtener más información, consulte Ejemplo: Detección de anomalías de datos y obtención de una explicación (función RANDOM_CUT_FOREST_WITH_EXPLANATION).	2 de noviembre de 2017
Detección de esquemas en los datos estáticos	Ejecute la detección del esquema en los datos estáticos almacenados en un bucket de Amazon S3. Para obtener más información, consulte Uso de la función de detección de esquema en datos estáticos.	6 de octubre de 2017
Característica de procesamiento previo de Lambda	Preprocese los registros en un flujo de entrada con un análisis AWS Lambda previo. Para obtener más información, consulte Procesamiento previo de registros con una función de Lambda.	6 de octubre de 2017

Cambio	Descripción	Fecha
Escalado automático de aplicaciones	Aumento automático del desempeño de datos de las aplicaciones con el escalado automático. Para obtener más información, consulte Escalado automático de aplicaciones para incrementar el desempeño.	13 de septiembre de 2017
Múltiples secuencias de entrada en la aplicación	Aumento del desempeño de las aplicaciones con varias secuencias en la aplicación. Para obtener más información, consulte Paralelizar secuencias de entrada para mejorar el desempeño.	29 de junio de 2017
Guía de uso de Consola de administración de AWS para Kinesis Data Analytics	Edición de un esquema inferido y de código SQL con el editor de esquemas y el editor de SQL en la consola de Kinesis Data Analytics. Para obtener más información, consulte Paso 4 (opcional): Editar el esquema y el código SQL utilizando la consola.	7 de abril de 2017
Versión pública	Se ha publicado la Guía para desarrolladores de Amazon Kinesis Data Analytics.	11 de agosto de 2016
Versión preliminar de	Vista previa de la Guía para desarrolladores de Amazon Kinesis Data Analytics.	29 de enero de 2016

AWS Glosario

Para obtener la AWS terminología más reciente, consulte el AWS glosario de la Glosario de AWS Referencia.