Connecting MySQL and Snowflake using Kafka and Debezium

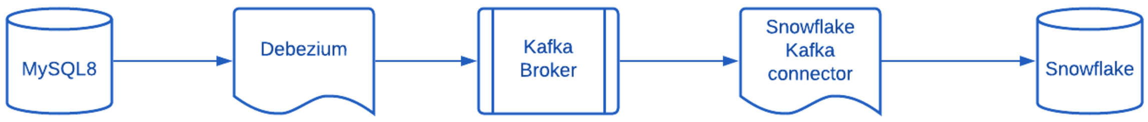
Abstract:

The goal is to create a real-time data pipeline between MySQL and Snowflake databases using Kafka and Debezium.

Tools:

OS: Ubuntu 20.04.2 LTS
MySQL: Ver 8.0.24 for Linux on x86_64 (MySQL Community Server – GPL)
ZooKeeper: Apache ZooKeeper
Kafka: 2.8.0
Debezium: 2.13
Snowflake Kafka connector: 1.5.2 (Maven)
Snowflake: Standard or Enterprise edition (AWS)

Diagram:



The given diagram is a high-level overview of what we are trying to achieve. As we can see in the diagram we will be using debezium to connect MySQL to the Kafka broker. Kafka Broker should be connected to the snowflake Database using the snowflake Kafka connector.

The Zookeeper is used for service synchronization and as a naming registry. When working with Apache Kafka, ZooKeeper is primarily used to track the status of nodes in the Kafka cluster and maintain a list of Kafka topics and messages. Debezium Kafka connector is used to connect MySQL Database to Kafka so that we can transfer the Data to Kafka. Also, by using the Snowflake Kafka connector we can move those Data from the Kafka broker to Snowflake Database.

Configuration:

For the sake of simplicity, We will consider the MySQL server as the Standalone server. The very first step is to install all the required software. We will be installing and Configuring Kafka, Zookeeper, and Debezium on the machine where MySQL is installed and Set Up. We will assume that MySQL is already running. Zookeeper needs JDK 1.8 or higher.

After installing and configuring the zookeeper we will be installing the Kafka. After successful installation of Kafka we need to configure **zookeeper.connect parameter.** We need to pass the IP and port of the zookeeper on which it's running. Now, We need to configure the Debezium, After installing the debezium we will configure the Kafka and Debezium the same way we configure zookeeper.

After completing all this configuration, We need to prepare MySQL Database for replication by creating the replication user in MySQL. Also, Binary Logging must be enabled for debezium to work as it relies on it to capture the data changes.

Some others such as

- Enabling Global Transaction Identifiers (GTDs). This would be necessary for a complex cluster MySQL setup.
- Configuring session timeouts. This may be necessary if the initial snapshot times out due to the amount of data to replicate.
- Enabling query log events. This is useful for troubleshooting.

If we are dealing with different time zones, we also need to update MySQL Timezone to UTC to make sure to avoid all the conflicts.

Now coming to the Snowflake, we can use standard or enterprise as per our usage. We need to create a database and warehouse as well as a SysAdmin user role with required access. Here we do not need to create the table as that will be created by the connector. However, we can explicitly create a table if business need requires.

This POC will fully run on a single virtual machine but a real production scenario may require a different configuration to deal with the higher amount of data to process. A Snowflake Kafka Connector and JDK, these two elements need to install on a server.

We can use an encrypted private key for authentication, so the Bouncy Castle plugin, available in Maven as well. By using MD5 based private key we can protect data from threats. The connector uses key pair authentication to connect to Snowflake so we are going to need a 2048-bit (minimum) RSA key pair.

A connector configuration file specifies the source tables and corresponding Kafka topics. These tables must reside in the same database and schema in the source system. While we are using a single Kafka installation for both the Debezium (CDC) and the Snowflake connectors we need different configuration files to avoid port collisions.

After all these setup and configuration steps, There will be references to the initial snapshot being made if a new table has been added to the replication. In the Snowflake Connector, we should see how the new connector is created, along with the pipes, stages and landing tables corresponding to each of the source tables.

At this point, if we enter the dummy data into MySQL, the MySQL connector will immediately capture that data and send that to Kafka, Then Kafka will be responsible for moving data to the snowflake database using the connector.

Reference Links:

- ETL Works Kafka MySQL Snowflake connection
- Setting up MySQL and Debezium
- Snowflake Kafka Connector
- ZooKeeper installation Doc
- Kafka
- Debezium MySQL Connetor