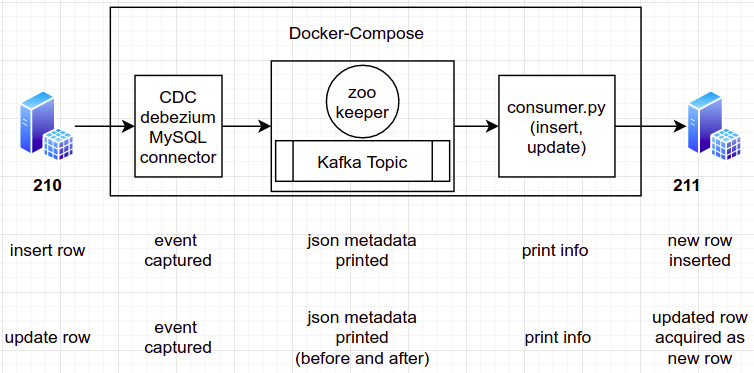
Kafka Streaming Test

(simulation, mysql to mysql, use all files in ‘*simulation*’ directory)



1. Grab the docker-compose.yml file. Run docker-compose up -d mysql destination zookeeper kafka
2. Wait for a while and after few seconds, run docker exec -it mysqldbz /bin/bash
3. Run apt-get update && apt-get install nano
4. Create new configuration file with nano /etc/mysql/conf.d/my.cnf, and set the followings;

[mysqld]

server-id = 223344 (or anything else)

log-bin = mysql-bin

expire\_logs\_days = 1

binlog\_format = row

1. Login to mysql, run mysql - u root -p debezium
2. Make user db\_source user can be accessed from anywhere, which mean the host from select user, host from mysql.user identified as %
3. Grant privileges to db\_source user, run grant all privileges on \* . \* to ‘db\_source’@’%’;
4. Exit root user, login mysql with db\_source, mysql -u db\_source -p test;
5. Create a database and table for testing, script already documented in *table.sql*. Insert some records to the table as a sample. Exit db\_source user and bash.
6. Repeat step 5 until 9 for the *destinationdbz* service, but without the insertion of some records. Grant privileges of *db\_dest* user with *root*. Login again with *db\_dest* user. Create a database and table without a primary key. Make sure the destination schema is already identical with the source schema. Script already documented in *table.sql*.
7. Run connector service with docker-compose up -d connect. Wait for a while until the status is running.
8. Create a Debezium source connection with debezium.json (*make-connection.sh*). Run;

curl -i -X POST \

-H "Accept:application/json" \

-H "Content-Type:application/json" \

127.0.0.1:8083/connectors/ \

--data "@debezium.json"

1. If no error in response, try check list of connections;

curl -i -X GET localhost:8083/connectors

1. Check connector, if all states conditions are “RUNNING” then we are ready.

curl -i -X GET localhost:8083/connectors/database-connector/status

1. Check all topics available, run;

docker exec -it \

$(docker ps | grep kafkadbz | awk '{ print $1 }') \

./bin/kafka-topics.sh \

--bootstrap-server localhost:9092 \

--list

Make sure **connect\_test** topic is available.

1. Create a consumer console to track the changes within the topic.

docker exec -it \

$(docker ps | grep kafkadbz | awk '{ print $1 }') \

./bin/kafka-console-consumer.sh \

--bootstrap-server localhost:9092 \

--topic connect\_test

1. Next is run the consumer node with, docker-compose up -d consumer\_node
2. If all the status is ok, then start the event (Insert, Update, Delete) in Source DB, captured event will be automatically updated on the Sink DB. Script for event trigger already documented on *table.sql*. To check the log run docker logs -f consumerdbz.
3. If try to make connection with remote db still faces some problem, for example mysql remote needs add on file, which is my.cnf.
4. For network connection issue with vpn tunnel, please do docker network prune, the start from the beginning again.