DEBEZIUM WITH KAFKA AND MYSQL FOR CDC

- What is Debezium?

Debezium is a set of distributed services that capture row-level changes in your databases so that your applications can see and respond to those changes. Debezium records in a transaction log all row-level changes committed to each database table. Each application simply reads the transaction logs they're interested in, and they see all of the events in the same order in which they occurred.

- What is Change Data Capture (CDC)?

Change Data Capture, or CDC, is an older term for a system that monitors and captures the changes in data so that other software can respond to those changes.

Debezium is essentially a modern, distributed open source *change data capture platform* that will eventually support monitoring a variety of database systems.

—---

Prerequisites:

- Kafka Installed
- Mysql Database Install
- Debezium Connector For MySqlConnector

===

Install Mysql

Commands As Follows:

1.Open Terminal and paste the following:

- yum install mysql-server*
- systemctl enable mysqld
- systemctl start mysqld
- mysql -u root

- \q
- mysql_secure_installation   ---> (for setup password)
- mysql -u root -p   ----> (ask for password)

===

Install kafka In Redhat

Download the latest Kafka release and extract it:

1.Open Terminal and paste it:

wget https://dlcdn.apache.org/kafka/3.5.0/kafka_2.13-3.5.0.tgz

$ tar -xzf kafka_2.13-3.5.0.tgz

$ cd kafka_2.13-3.5.0

2.Run the following commands in order to start all services in the correct order:

# Start the ZooKeeper service

$ bin/zookeeper-server-start.sh config/zookeeper.properties

3.Open another terminal session and run:

# Start the Kafka broker service

$ bin/kafka-server-start.sh config/server.properties

#Once all services have successfully launched, you will have a basic Kafka environment running and ready to use.

4.For kafka connect

./bin/connect-distributed.sh config/connect-distributed.properties

Following are some useful commands:

1.For create Topic

$ bin/kafka-topics.sh --create --topic myTopic --partitions 3 --replication-factor 1 --bootstrap-server localhost:9092

2. For Details Of Topic

$ bin/kafka-topics.sh --describe --topic myTopic --bootstrap-server localhost:9092

3. For produce the msg from producer

$ bin/kafka-console-producer.sh --topic myTopic --bootstrap-server localhost:9092

4. For consume the msg from consumer

$ bin/kafka-console-consumer.sh --topic myTopic --from-beginning --bootstrap-server localhost:9092

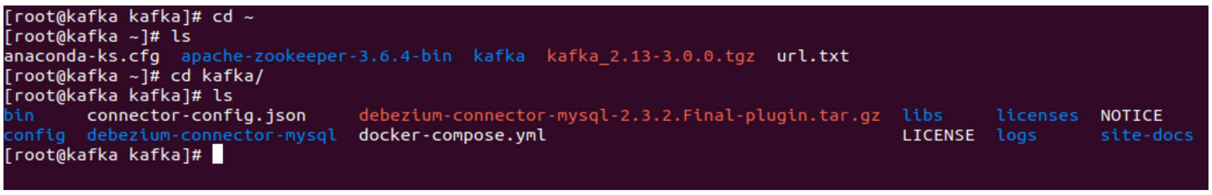
===

Install Debezium

1.Download the Debezium [MySQL connector plug-in](#).

2.Extract the files into your Kafka Connect environment.

Eg. I have downloaded Kafka in my root folder and extracted there. So in the kafka folder i put debezium connector and extracted there.



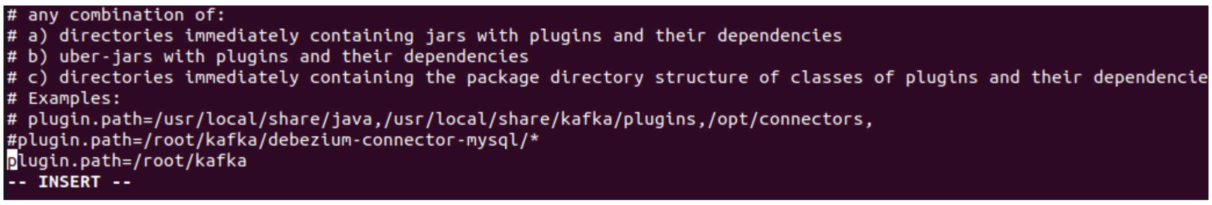
```
[root@kafka kafka]# cd ~
[root@kafka ~]# ls
anaconda-ks.cfg  apache-zookeeper-3.6.4-bin  kafka  kafka_2.13-3.0.0.tgz  url.txt
[root@kafka ~]# cd kafka/
[root@kafka kafka]# ls
bin      connector-config.json     debezium-connector-mysql-2.3.2.Final-plugin.tar.gz  libs     licenses  NOTICE
config  debezium-connector-mysql  docker-compose.yml                                  LICENSE  logs      site-docs
[root@kafka kafka]# 
```

img.1.1

3.Add the directory with the JAR files to [Kafka Connect's `plugin.path`](#).

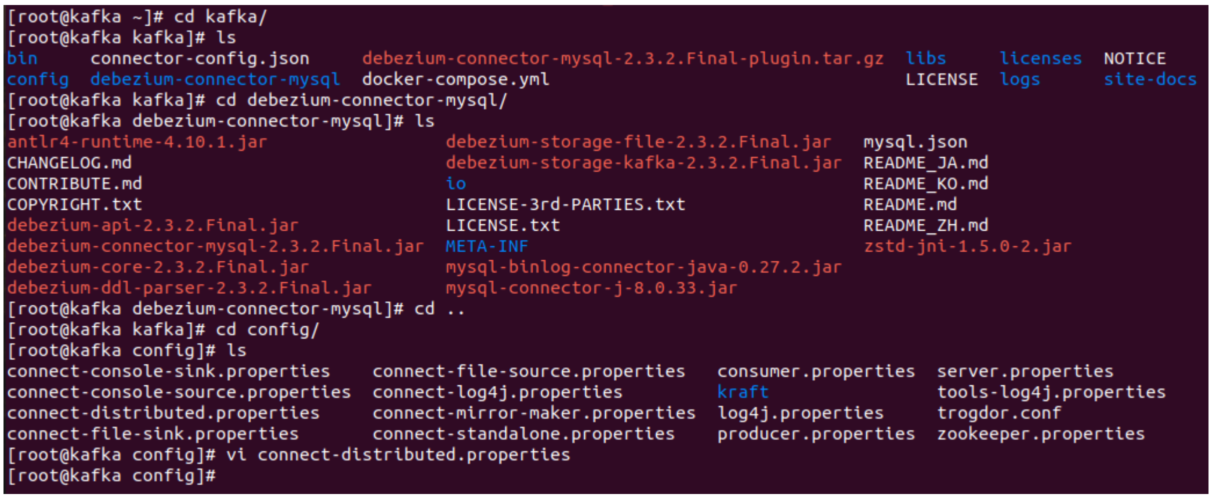
Eg.In img 1.1 we can see the extracted folder debezium. So we need that path as plugin path. In kafka/config there is file name “connect-distributed.properties”, in last you get plugin-path properties , add there path. Like- /root/kafka.

And uncomment some #.eg. listeners=HTTP://:8083



```
# any combination of:
# a) directories immediately containing jars with plugins and their dependencies
# b) uber-jars with plugins and their dependencies
# c) directories immediately containing the package directory structure of classes of plugins and their dependencie
# Examples:
# plugin.path=/usr/local/share/java,/usr/local/share/kafka/plugins,/opt/connectors,
#plugin.path=/root/kafka/debezium-connector-mysql/*
plugin.path=/root/kafka
-- INSERT --
```

img.1.2



```
[root@kafka ~]# cd kafka/
[root@kafka kafka]# ls
bin     connector-config.json     debezium-connector-mysql-2.3.2.Final-plugin.tar.gz  libs     licenses  NOTICE
config  debezium-connector-mysql  docker-compose.yml                                  LICENSE  logs      site-docs
[root@kafka kafka]# cd debezium-connector-mysql/
[root@kafka debezium-connector-mysql]# ls
antlr4-runtime-4.10.1.jar                    debezium-storage-file-2.3.2.Final.jar   mysql.json
CHANGELOG.md                                 debezium-storage-kafka-2.3.2.Final.jar  README_JA.md
CONTRIBUTE.md                                io                                      README_KO.md
COPYRIGHT.txt                                LICENSE-3rd-PARTIES.txt                 README.md
debezium-api-2.3.2.Final.jar                 LICENSE.txt                             README_ZH.md
debezium-connector-mysql-2.3.2.Final.jar     META-INF                                zstd-jni-1.5.0-2.jar
debezium-core-2.3.2.Final.jar                mysql-binlog-connector-java-0.27.2.jar
debezium-ddl-parser-2.3.2.Final.jar          mysql-connector-j-8.0.33.jar
[root@kafka debezium-connector-mysql]# cd ..
[root@kafka kafka]# cd config/
[root@kafka config]# ls
connect-console-sink.properties    connect-file-source.properties   consumer.properties  server.properties
connect-console-source.properties  connect-log4j.properties         kraft                tools-log4j.properties
connect-distributed.properties     connect-mirror-maker.properties  log4j.properties     trogdor.conf
connect-file-sink.properties       connect-standalone.properties    producer.properties  zookeeper.properties
[root@kafka config]# vi connect-distributed.properties
[root@kafka config]#
```

img.1.3

Note= dont add debezium mysql folder name in path.

4.Configure the connector and add the configuration to your Kafka Connect cluster.

Make “connector-config.json” in kafka folder. And add some configuration like as follow:

```
{
    "name": "inventory-connector",
   "config": {
      "connector.class": "io.debezium.connector.mysql.MySqlConnector",
      "database.hostname": "192.168.10.183",
      "database.port": "3306",
      "database.user": "ganeshSql",
      "database.password": "Mahity@123",
      "database.server.id": "1",
       "database.server.name": "Mysql",
      "topic.prefix": "MyApp",
       "database.whitelist": "new",
       "database.include.list": "new",
       "table.include.list": "new.Call_Strikes",
       "table.whitelist": "new.Call_Strikes",
       "database.history.kafka.topic": "schema-changes.new",
       "database.history":
"io.debezium.relational.history.MemoryDatabaseHistory",
      "schema.history.internal.kafka.bootstrap.servers": "localhost:9092",
      "schema.history.internal.kafka.topic": "testTopic",
      "include.schema.changes": "false",
       "tasks.max": "1",
       "decimal.handling.mode": "precise",
       "bigint.unsigned.handling.mode": "long",
       "max.batch.size": "2048",
       "max.queue.size":"8192",
       "poll.interval.ms": "500",

       "connect.timeout.ms": "30000",

       "snapshot.mode": "SCHEMA_ONLY_RECOVERY",

       "snapshot.new.tables": "parallel",

       "snapshot.locking.mode": "minimal",

       "snapshot.include.collection.list": "new.Call_Strikes",

       "schema.history.internal.kafka.recovery.poll.interval.ms": "10000",

      "schema.history.internal.kafka.query.timeout.ms": "3000",

      "schema.history.internal.kafka.create.timeout.ms": "30000",

      "schema.history.internal.kafka.recovery.attempts": "1000",

      "schema.history.internal.skip.unparseable.ddl": "false",

      "schema.history.internal.store.only.captured.tables.ddl": "false",

      "schema.history.internal.store.only.captured.databases.ddl": "true",

       "transforms": "unwrap",

      "transforms.unwrap.type":"io.debezium.transforms.ExtractNewRecordState",

"transforms.unwrap.drop.tombstones":"false",

"transforms.unwrap.delete.handling.mode":  "rewrite",

"Transforms.unwrap.add.fields" : "table,lsn",


"plugin.path":"/root/kafka"

   }}
```

And save it.

1. name= you can give any name as Connector name.

2. In config. Connector.class name will be = “io.debezium.connector.mysql.MySqlConnector” bcoz in debezium tar file there is MysqlConnector class present in io/debezium/connector/mysql/MysqlConnector.class.
3. "database.hostname": "192.168.10.183"...IP address where Mysql running. If it is running on same machine where kafka installed so use “localhost”.
4. "database.port": "3306", database port number.
5. "database.user": "ganeshSql"-----user name
6. "database.password": "Mahity@123", —-password of username.
7. "database.server.id": "1", —- you will find it on mysql
8. "database.server.name": "Mysql" ——---give any unique name.
9. "topic.prefix": "MyApp" —--------it is important. Give any unique name. Its important bcoz the topic name which will be automatically created is start from this name.

   Eg. In my case the topic created as, MyApp.new.Call_Strikes.

   Prefix.databasename.tablename

10 & 11."database.whitelist": "new",

   "database.include.list": "new", —--------the database name.

12 &13."table.include.list": "new.Call_Strikes",

   "table.whitelist": "new.Call_Strikes", —------------table name.

14."database.history.kafka.topic": "schema-changes.new", —-give anyname.database name.

15."database.history": "io.debezium.relational.history.MemoryDatabaseHistory —copy as it is.

16. "schema.history.internal.kafka.bootstrap.servers": "localhost:9092", —--broker port. Where kafka broker is running. You can find it on server.properties file.

17. "schema.history.internal.kafka.topic": "testTopic", ——---- topic name, optional

18. "include.schema.changes": "true",

    "tasks.max": "1",

    "decimal.handling.mode": "precise",

    "bigint.unsigned.handling.mode": "long",

    "max.batch.size": "2048",

    "Max.queue.size":"8192",

    "poll.interval.ms": "500",

    "connect.timeout.ms": "30000",-----------------------give as it is.

19."snapshot.mode": "SCHEMA_ONLY_RECOVERY", —- it will recover all data and snapshot it.

20."snapshot.new.tables": "parallel",

"snapshot.locking.mode": "minimal", —--------give as it is.

21."snapshot.include.collection.list": "new.Call_Strikes", —--databasename.tablename

22."schema.history.internal.kafka.recovery.poll.interval.ms": "10000",

    "schema.history.internal.kafka.query.timeout.ms": "3000",

    "schema.history.internal.kafka.create.timeout.ms": "30000",

    "schema.history.internal.kafka.recovery.attempts": "1000",

    "schema.history.internal.skip.unparseable.ddl": "false",

    "schema.history.internal.store.only.captured.tables.ddl": "false",

    "schema.history.internal.store.only.captured.databases.ddl": "true",

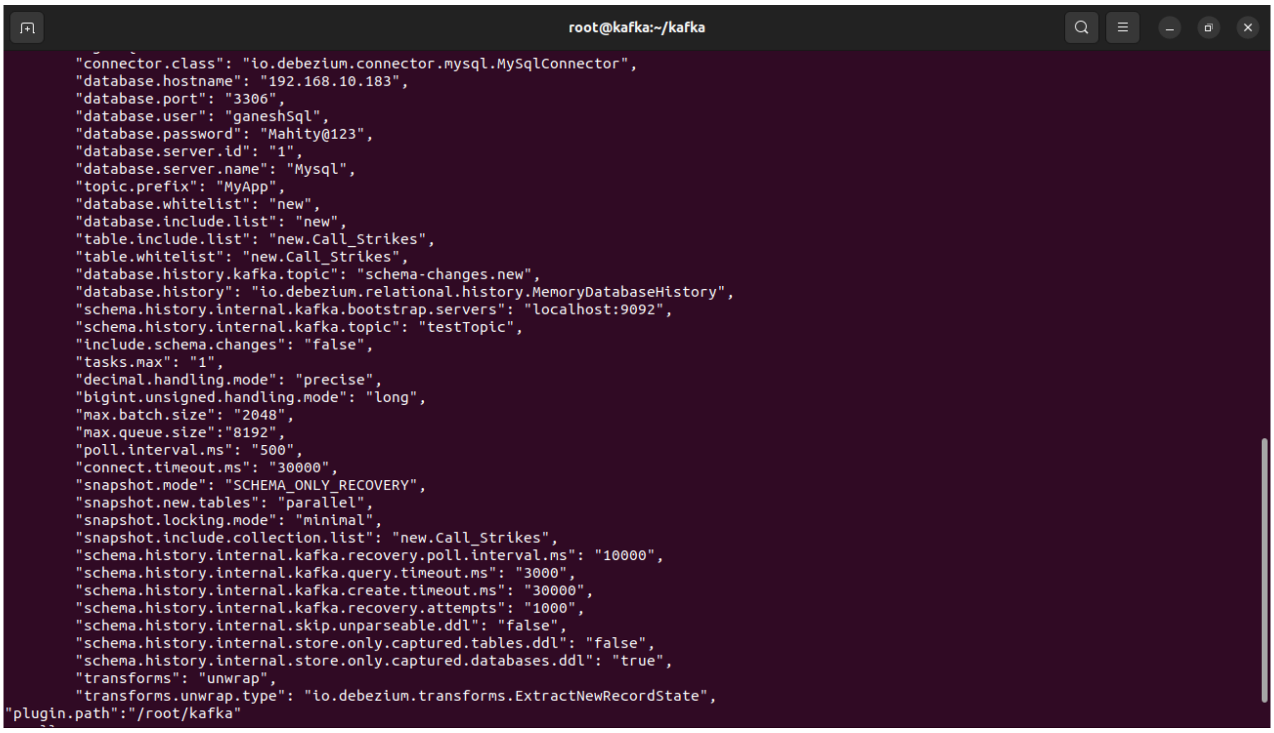
23. "transforms": "unwrap",

    "transforms.unwrap.type": "io.debezium.transforms.ExtractNewRecordState",

    “transforms.unwrap.drop.tombstones”:”false”,

    “transforms.unwrap.delete.handling.mode”: “rewrite”, “Transforms.unwrap.add.fields” : “table,lsn”, —----------It is important bcoz it give you records about data changes, if you delete something in mysql it will show you like ____deleted=true, and that record. So you get idea that this record is deleted.

24."plugin.path":"/root/kafka" —-------path of plugin of debezium.


```
root@kafka:~/kafka
        "connector.class": "io.debezium.connector.mysql.MySqlConnector",
        "database.hostname": "192.168.10.183",
        "database.port": "3306",
        "database.user": "ganeshSql",
        "database.password": "Mahity@123",
        "database.server.id": "1",
        "database.server.name": "Mysql",
        "topic.prefix": "MyApp",
        "database.whitelist": "new",
        "database.include.list": "new",
        "table.include.list": "new.Call_Strikes",
        "table.whitelist": "new.Call_Strikes",
        "database.history.kafka.topic": "schema-changes.new",
        "database.history": "io.debezium.relational.history.MemoryDatabaseHistory",
        "schema.history.internal.kafka.bootstrap.servers": "localhost:9092",
        "schema.history.internal.kafka.topic": "testTopic",
        "include.schema.changes": "false",
        "tasks.max": "1",
        "decimal.handling.mode": "precise",
        "bigint.unsigned.handling.mode": "long",
        "max.batch.size": "2048",
        "max.queue.size":"8192",
        "poll.interval.ms": "500",
        "connect.timeout.ms": "30000",
        "snapshot.mode": "SCHEMA_ONLY_RECOVERY",
        "snapshot.new.tables": "parallel",
        "snapshot.locking.mode": "minimal",
        "snapshot.include.collection.list": "new.Call_Strikes",
        "schema.history.internal.kafka.recovery.poll.interval.ms": "10000",
        "schema.history.internal.kafka.query.timeout.ms": "3000",
        "schema.history.internal.kafka.create.timeout.ms": "30000",
        "schema.history.internal.kafka.recovery.attempts": "1000",
        "schema.history.internal.skip.unparseable.ddl": "false",
        "schema.history.internal.store.only.captured.tables.ddl": "false",
        "schema.history.internal.store.only.captured.databases.ddl": "true",
        "transforms": "unwrap",
        "transforms.unwrap.type": "io.debezium.transforms.ExtractNewRecordState",
"plugin.path":"/root/kafka"
```

img.1.4

==

Dont run now. Run After mysql Config.

5.Restart your Kafka Connect process to pick up the new JAR files.

Run this commands in kafka dir:

```
$ curl -X POST -H "Content-Type: application/json" --data @connector-config.json
    http://localhost:8083/connectors
```

Eg. just run kafka connect command. It will pick up all jar files from debezium.

(use this command for delete connector if needed= curl -X DELETE
    localhost:8083/connectors/inventory-connector)

==

Mysql Config

Open mysql with “mysql -u root -p”

1.Run = mysql>CREATE USER 'ganeshSql'@'%' IDENTIFIED BY 'Mahity@123';

Eg. ganeshSql is username. % for connecting all IP address from it. Last is password.

2.mysql> GRANT SELECT, RELOAD, SHOW DATABASES, REPLICATION SLAVE, REPLICATION CLIENT ON *.* TO 'ganeshSql';

Note. you can give some extra grants . in mysql “show grants;” will give you list of grants, you can give any grants just adding comma.

3.mysql> FLUSH PRIVILEGES;

4.SELECT variable_value as "BINARY LOGGING STATUS (log-bin) ::"

FROM performance_schema.global_variables WHERE variable_name='log_bin';

5.show global variables like '%GTID%';

See gtid_mode and ENFORCE_GTID_CONSISTENCY is on or not.

6.SET @@GLOBAL.GTID_MODE = OFF;

7.SET @@GLOBAL.GTID_MODE = OFF_PERMISSIVE;

8.SET @@GLOBAL.GTID_MODE = ON_PERMISSIVE;

9.SET @@GLOBAL.ENFORCE_GTID_CONSISTENCY = ON;

10.SET @@GLOBAL.GTID_MODE = ON;

11.SET SESSION interactive_timeout = 30;

12.SET SESSION wait_timeout=30;

13.SET binlog_rows_query_log_events=ON;

14.show global variables where variable_name = 'binlog_row_value_options';

15.set @@global.binlog_row_value_options="" ;

Exit from mysql.

Go to the cd /etc/my.cnf file. And set this value.

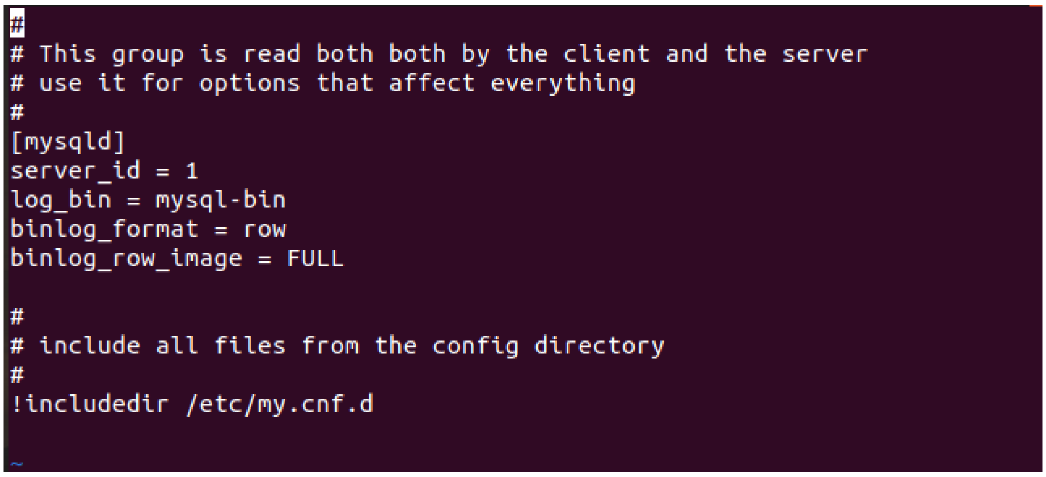
[mysqld]

server_id = 1

log_bin = mysql-bin

binlog_format = row

binlog_row_image = FULL



```
#
# This group is read both both by the client and the server
# use it for options that affect everything
#
[mysqld]
server_id = 1
log_bin = mysql-bin
binlog_format = row
binlog_row_image = FULL

#
# include all files from the config directory
#
!includedir /etc/my.cnf.d

~
```

img.1.5

==

Go to the cd kafka/config. Open server.properties file.Uncomment this or modify it.

broker.id=0

listeners=PLAINTEXT://localhost:9092

advertised.listeners=PLAINTEXT://localhost:9092

listener.security.protocol.map=PLAINTEXT:PLAINTEXT,SSL:SSL,SASL_PLAINTEXT:SASL_PLAINTEXT,SASL_SSL:SASL_SSL

num.network.threads=3

num.io.threads=8

socket.send.buffer.bytes=102400

socket.receive.buffer.bytes=102400

socket.request.max.bytes=104857600

log.dirs=/tmp/kafka-logs

num.partitions=1

num.recovery.threads.per.data.dir=1

offsets.topic.replication.factor=1

transaction.state.log.replication.factor=1

transaction.state.log.min.isr=1

log.flush.interval.messages=10000

zookeeper.connect=localhost:2181

And remaining all needed.

===

In consumer.properties uncomment this.

bootstrap.servers=localhost:9092

===

For run this do as follow

1. Open Terminal. Go to kafka directory and Run.

$ bin/zookeeper-server-start.sh config/zookeeper.properties

It will start Zookeeper.

2. Open 2nd Terminal and Run:

$ bin/kafka-server-start.sh config/server.properties

It will run kafka server or broker.

3.Open 3rd Terminal and run:

$ ./bin/connect-distributed.sh config/connect-distributed.properties

It will run kafka connect.

4.Open 4th Terminal and run:

$ curl -X POST -H "Content-Type: application/json" --data @connector-config.json http://localhost:8083/connectors

It will run that “connector-config.json” file. And See Http status is get 201. Created.

For check it is running run

$ curl -X GET http://localhost:8083/connectors/inventory-connector/status

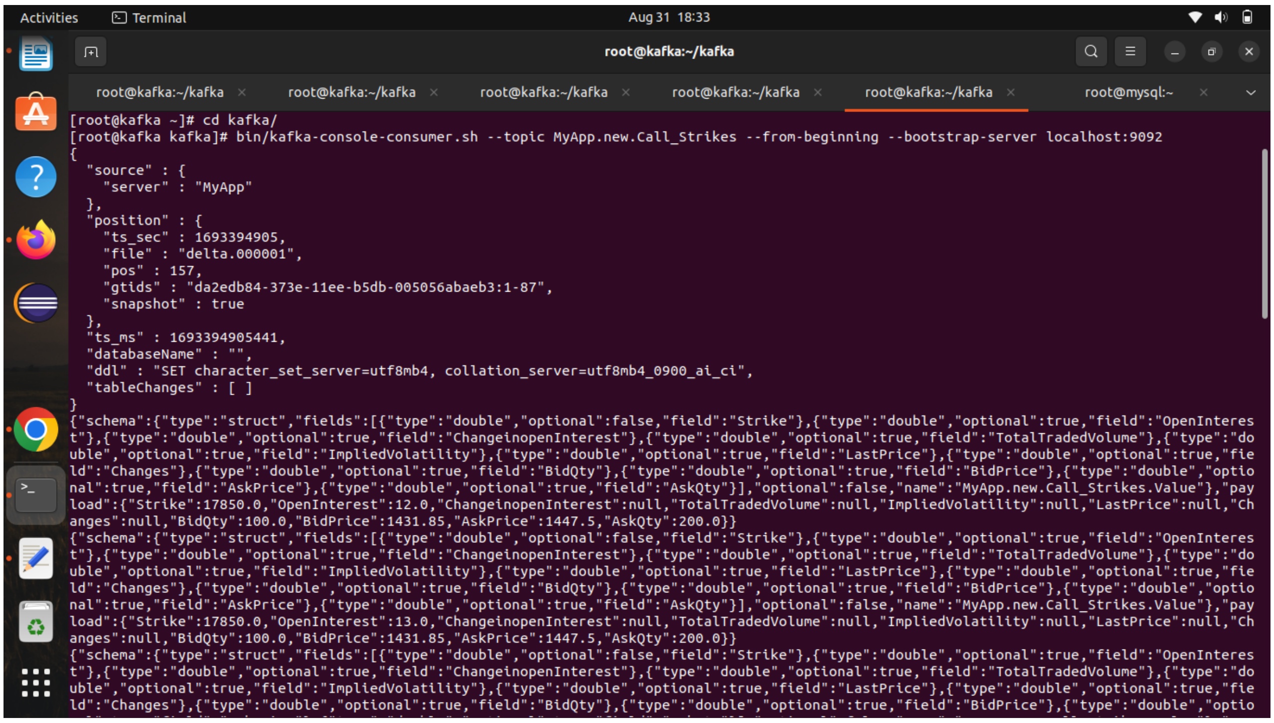
Inventory-connector is my connector name from config file.

5.After this the topic is automatically being created with name of Prefix.databasename.tablename

Open 5th Terminal and run :

bin/kafka-console-consumer.sh --topic MyApp.new.Call_Strikes --from-beginning --bootstrap-server localhost:9092

The data will show you the tables and values in your database. In my case its look like this:



```
[root@kafka ~]# cd kafka/
[root@kafka kafka]# bin/kafka-console-consumer.sh --topic MyApp.new.Call_Strikes --from-beginning --bootstrap-server localhost:9092
{
  "source" : {
    "server" : "MyApp"
  },
  "position" : {
    "ts_sec" : 1693394905,
    "file" : "delta.000001",
    "pos" : 157,
    "gtids" : "da2edb84-373e-11ee-b5db-005056abaeb3:1-87",
    "snapshot" : true
  },
  "ts_ms" : 1693394905441,
  "databaseName" : "",
  "ddl" : "SET character_set_server=utf8mb4, collation_server=utf8mb4_0900_ai_ci",
  "tableChanges" : [ ]
}
{"schema":{"type":"struct","fields":[{"type":"double","optional":false,"field":"Strike"},{"type":"double","optional":true,"field":"OpenInteres
t"},{"type":"double","optional":true,"field":"ChangeinopenInterest"},{"type":"double","optional":true,"field":"TotalTradedVolume"},{"type":"do
uble","optional":true,"field":"ImpliedVolatility"},{"type":"double","optional":true,"field":"LastPrice"},{"type":"double","optional":true,"fie
ld":"Changes"},{"type":"double","optional":true,"field":"BidQty"},{"type":"double","optional":true,"field":"BidPrice"},{"type":"double","optio
nal":true,"field":"AskPrice"},{"type":"double","optional":true,"field":"AskQty"}],"optional":false,"name":"MyApp.new.Call_Strikes.Value"},"pay
load":{"Strike":17850.0,"OpenInterest":12.0,"ChangeinopenInterest":null,"TotalTradedVolume":null,"ImpliedVolatility":null,"LastPrice":null,"Ch
anges":null,"BidQty":100.0,"BidPrice":1431.85,"AskPrice":1447.5,"AskQty":200.0}}
{"schema":{"type":"struct","fields":[{"type":"double","optional":false,"field":"Strike"},{"type":"double","optional":true,"field":"OpenInteres
t"},{"type":"double","optional":true,"field":"ChangeinopenInterest"},{"type":"double","optional":true,"field":"TotalTradedVolume"},{"type":"do
uble","optional":true,"field":"ImpliedVolatility"},{"type":"double","optional":true,"field":"LastPrice"},{"type":"double","optional":true,"fie
ld":"Changes"},{"type":"double","optional":true,"field":"BidQty"},{"type":"double","optional":true,"field":"BidPrice"},{"type":"double","optio
nal":true,"field":"AskPrice"},{"type":"double","optional":true,"field":"AskQty"}],"optional":false,"name":"MyApp.new.Call_Strikes.Value"},"pay
load":{"Strike":17850.0,"OpenInterest":13.0,"ChangeinopenInterest":null,"TotalTradedVolume":null,"ImpliedVolatility":null,"LastPrice":null,"Ch
anges":null,"BidQty":100.0,"BidPrice":1431.85,"AskPrice":1447.5,"AskQty":200.0}}
{"schema":{"type":"struct","fields":[{"type":"double","optional":false,"field":"Strike"},{"type":"double","optional":true,"field":"OpenInteres
t"},{"type":"double","optional":true,"field":"ChangeinopenInterest"},{"type":"double","optional":true,"field":"TotalTradedVolume"},{"type":"do
uble","optional":true,"field":"ImpliedVolatility"},{"type":"double","optional":true,"field":"LastPrice"},{"type":"double","optional":true,"fie
ld":"Changes"},{"type":"double","optional":true,"field":"BidQty"},{"type":"double","optional":true,"field":"BidPrice"},{"type":"double","optio
```

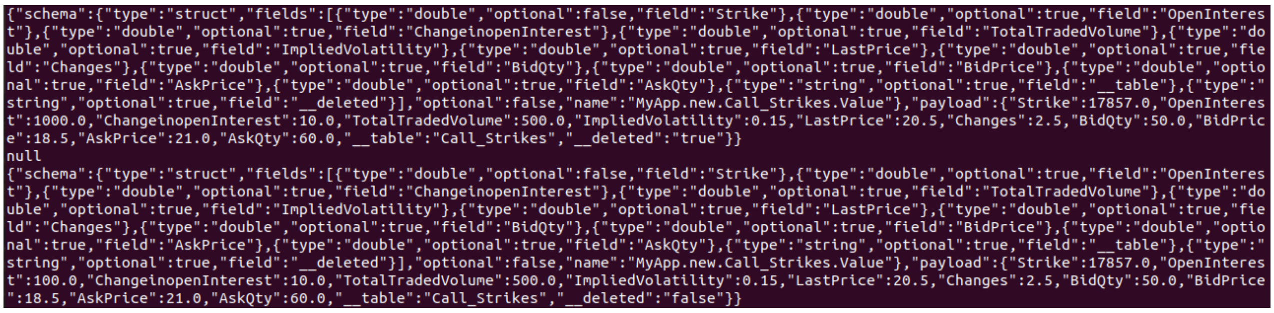
img.1.6

The result after adding

“transforms.unwrap.drop.tombstones”:”false”,

“transforms.unwrap.delete.handling.mode”: “rewrite”,

“Transforms.unwrap.add.fields” : “table,lsn”,



```
{"schema":{"type":"struct","fields":[{"type":"double","optional":false,"field":"Strike"},{"type":"double","optional":true,"field":"OpenInteres
t"},{"type":"double","optional":true,"field":"ChangeinopenInterest"},{"type":"double","optional":true,"field":"TotalTradedVolume"},{"type":"do
uble","optional":true,"field":"ImpliedVolatility"},{"type":"double","optional":true,"field":"LastPrice"},{"type":"double","optional":true,"fie
ld":"Changes"},{"type":"double","optional":true,"field":"BidQty"},{"type":"double","optional":true,"field":"BidPrice"},{"type":"double","optio
nal":true,"field":"AskPrice"},{"type":"double","optional":true,"field":"AskQty"},{"type":"string","optional":true,"field":"__table"},{"type":"
string","optional":true,"field":"__deleted"}],"optional":false,"name":"MyApp.new.Call_Strikes.Value"},"payload":{"Strike":17857.0,"OpenInteres
t":1000.0,"ChangeinopenInterest":10.0,"TotalTradedVolume":500.0,"ImpliedVolatility":0.15,"LastPrice":20.5,"Changes":2.5,"BidQty":50.0,"BidPric
e":18.5,"AskPrice":21.0,"AskQty":60.0,"__table":"Call_Strikes","__deleted":"true"}}
null
{"schema":{"type":"struct","fields":[{"type":"double","optional":false,"field":"Strike"},{"type":"double","optional":true,"field":"OpenInteres
t"},{"type":"double","optional":true,"field":"ChangeinopenInterest"},{"type":"double","optional":true,"field":"TotalTradedVolume"},{"type":"do
uble","optional":true,"field":"ImpliedVolatility"},{"type":"double","optional":true,"field":"LastPrice"},{"type":"double","optional":true,"fie
ld":"Changes"},{"type":"double","optional":true,"field":"BidQty"},{"type":"double","optional":true,"field":"BidPrice"},{"type":"double","optio
nal":true,"field":"AskPrice"},{"type":"double","optional":true,"field":"AskQty"},{"type":"string","optional":true,"field":"__table"},{"type":"
string","optional":true,"field":"__deleted"}],"optional":false,"name":"MyApp.new.Call_Strikes.Value"},"payload":{"Strike":17857.0,"OpenInteres
t":100.0,"ChangeinopenInterest":10.0,"TotalTradedVolume":500.0,"ImpliedVolatility":0.15,"LastPrice":20.5,"Changes":2.5,"BidQty":50.0,"BidPrice
":18.5,"AskPrice":21.0,"AskQty":60.0,"__table":"Call_Strikes","__deleted":"false"}}
```

img.1.7

Note: All this command are run on kafka dir.

Note: Make sure you run the kafka command as shown. First zookeeper -> kafka server->kafka connect -> curl post command -> kafka consumer.