Hands-on Lab: Build a Streaming ETL Pipeline using Kafka

Scenario

You are a data engineer at a data analytics consulting company. You have been assigned to a project that aims to de-congest the national highways by analyzing the road traffic data from different toll plazas. As a vehicle passes a toll plaza, the vehicle's data like vehicle_id,vehicle_type,toll_plaza_id and timestamp are streamed to Kafka. Your job is to create a data pipe line that collects the streaming data and loads it into a database.

Objectives

In this assignment you will create a streaming data pipe by performing these steps:

- Start a MySQL Database server.
- Create a table to hold the toll data.
- Start the Kafka server.
- Install the Kafka python driver.
- Install the MySQL python driver.
- Create a topic named toll in kafka.
- Download streaming data generator program.
- Customize the generator program to steam to toll topic.
- Download and customise streaming data consumer.
- Customize the consumer program to write into a MySQL database table.
- Verify that streamed data is being collected in the database table.

Exercise 1 - Prepare the lab environment

- Download Kafka

```
theia@theiadocker-craigtrupp8:/home/project$ wget
https://archive.apache.org/dist/kafka/2.8.0/kafka_2.12-2.8.0.tgz
--2023-09-29 11:12:35--  https://archive.apache.org/dist/kafka/2.8.0/kafka_2.12-2.8.0.tgz
Resolving archive.apache.org (archive.apache.org)... 65.108.204.189, 2a01:4f9:1a:a084::2
Connecting to archive.apache.org (archive.apache.org)|65.108.204.189|:443... connected.
```

```
HTTP request sent, awaiting response... 200 OK
Length: 71542357 (68M) [application/x-gzip]
Saving to: ‘kafka_2.12-2.8.0.tgz’

kafka_2.12-2.8.0.tgz
100%[=====================================================>]  68.23M
18.9MB/s    in 3.6s

2023-09-29 11:12:39 (18.9 MB/s) - ‘kafka_2.12-2.8.0.tgz’ saved [71542357/71542357]
```

- Extract Kafka

```
theia@theiadocker-craigtrupp8:/home/project$ tar -xzf kafka_2.12-2.8.0.tgz
```

- Start MySQL

```
theia@theiadocker-craigtrupp8:/home/project$ start_mysql
Starting your MySQL database....
This process can take up to a minute.

MySQL database started, waiting for all services to be ready....

Your MySQL database is now ready to use and available with username: root
password: MjE4NS1jcmFpZ3Ry

You can access your MySQL database via:
 • The browser at:
https://craigtrupp8-8080.theiadocker-3-labs-prod-theiak8s-4-tor01.proxy.cog
nitiveclass.ai
 • CommandLine: mysql --host=127.0.0.1 --port=3306 --user=root
--password=MjE4NS1jcmFpZ3Ry
```

- Connect to the mysql server, using the command below. Make sure you use the password given to you when the MySQL server starts.
    - Please make a note or record of the password because you will need it later.

```
theia@theiadocker-craigtrupp8:/home/project$ mysql --host=127.0.0.1
--port=3306 --user=root --password=MjE4NS1jcmFpZ3Ry
mysql: [Warning] Using a password on the command line interface can be
```

```
insecure.
Welcome to the MySQL monitor.  Commands end with ; or \g.
```

```
Your MySQL connection id is 56
Server version: 8.0.22 MySQL Community Server - GPL

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input
statement.
```

```
mysql> create database tolldata;
Query OK, 1 row affected (0.01 sec)

mysql> use tolldata;
Database changed
mysql> create table livetolldata(timestamp datetime,vehicle_id
int,vehicle_type char(15),toll_plaza_id smallint);
Query OK, 0 rows affected (0.03 sec)
```

- Step 7: Disconnect from MySQL server.

```
mysql> exit
Bye
```

- Step 8: Install the python module kafka-python using the pip command.

```
theia@theiadocker-craigtrupp8:/home/project$ python3 -m pip install
kafka-python
Collecting kafka-python
  Downloading
https://files.pythonhosted.org/packages/75/68/dcb0db055309f680ab2931a3eeb22
d865604b638acf8c914bedf4c1a0c8c/kafka_python-2.0.2-py2.py3-none-any.whl
(246kB)
```

```
    100% |████████████████████████████████| 256kB 6.0MB/s
Installing collected packages: kafka-python
Successfully installed kafka-python-2.0.2
```

- Step 9: Install the python module mysql-connector-python using the pip command.

```
theia@theiadocker-craigtrupp8:/home/project$ python3 -m pip install
mysql-connector-python==8.0.31
Collecting mysql-connector-python==8.0.31
  Downloading
https://files.pythonhosted.org/packages/08/1f/42d74bae9dd6dcfec67c9ed0f3fa4
82b1ae5ac5f117ca82ab589ecb3ca19/mysql_connector_python-8.0.31-py2.py3-none-
any.whl (352kB)
    100% |████████████████████████████████| 358kB 4.2MB/s
Collecting protobuf<=3.20.1,>=3.11.0 (from mysql-connector-python==8.0.31)
  Downloading
https://files.pythonhosted.org/packages/32/27/1141a8232723dcb10a595cc0ce432
1dcbbd5215300bf4acfc142343205bf/protobuf-3.19.6-py2.py3-none-any.whl
(162kB)
    100% |████████████████████████████████| 163kB 9.1MB/s
Installing collected packages: protobuf, mysql-connector-python
Successfully installed mysql-connector-python-8.0.31 protobuf-3.19.6
```

Exercise 2 - Start Kafka

Note : All of these kafka options need to be running in separate terminal instances/windows

- Start Zookeeper

```
theia@theiadocker-craigtrupp8:/home/project/kafka_2.12-2.8.0$
bin/zookeeper-server-start.sh config/zookeeper.properties
```

- Start Kafka Server

```
theia@theiadocker-craigtrupp8:/home/project/kafka_2.12-2.8.0$
bin/kafka-server-start.sh config/server.properties
```

- Create a topic named toll

```
theia@theiadocker-craigtrupp8:/home/project/kafka_2.12-2.8.0$
bin/kafka-topics.sh --create --topic toll --bootstrap-server localhost:9092
Created topic toll.
```

- Download the Toll Traffic Simulator

```
theia@theiadocker-craigtrupp8:/home/project/kafka_2.12-2.8.0$ wget
https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DB02
50EN-SkillsNetwork/labs/Final%20Assignment/toll_traffic_generator.py
--2023-09-29 12:09:04--
https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DB02
50EN-SkillsNetwork/labs/Final%20Assignment/toll_traffic_generator.py
Resolving cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud
(cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud)...
169.63.118.104
Connecting to cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud
(cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud)|169.63.118.104
|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 828 [text/x-python]
Saving to: 'toll_traffic_generator.py'

toll_traffic_generator.py
100%[=============================================================>]
828  --.-KB/s    in 0s

2023-09-29 12:09:04 (105 MB/s) - 'toll_traffic_generator.py' saved
[828/828]
```

- Configure and Run Simulator

```
"""
Top Traffic Simulator
"""
from time import sleep, time, ctime
from random import random, randint, choice
from kafka import KafkaProducer
producer = KafkaProducer(bootstrap_servers='localhost:9092')

TOPIC = 'toll'
```

```
VEHICLE_TYPES = ("car", "car", "car", "car", "car", "car", "car", "car",
                 "car", "car", "car", "truck", "truck", "truck",
                 "truck", "van", "van")
for _ in range(100000):
    vehicle_id = randint(10000, 10000000)
    vehicle_type = choice(VEHICLE_TYPES)
    now = ctime(time())
    plaza_id = randint(4000, 4010)
    message = f"{now},{vehicle_id},{vehicle_type},{plaza_id}"
    message = bytearray(message.encode("utf-8"))
    print(f"A {vehicle_type} has passed by the toll plaza {plaza_id} at
{now}.")
    producer.send(TOPIC, message)
    sleep(random() * 2)
```

```
theia@theiadocker-craigtrupp8:/home/project/kafka_2.12-2.8.0$ python3
toll_traffic_generator.py
A car has passed by the toll plaza 4003 at Fri Sep 29 12:11:07 2023.
A car has passed by the toll plaza 4005 at Fri Sep 29 12:11:09 2023.
A car has passed by the toll plaza 4003 at Fri Sep 29 12:11:09 2023.
A car has passed by the toll plaza 4004 at Fri Sep 29 12:11:10 2023.
```

- Download Streaming Reader

```
theia@theiadocker-craigtrupp8:/home/project/kafka_2.12-2.8.0$ wget
https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DB02
50EN-SkillsNetwork/labs/Final%20Assignment/streaming_data_reader.py
--2023-09-29 12:15:33--
https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DB02
50EN-SkillsNetwork/labs/Final%20Assignment/streaming_data_reader.py
Resolving cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud
(cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud)...
169.63.118.104
Connecting to cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud
(cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud)|169.63.118.104
|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1364 (1.3K) [text/x-python]
Saving to: 'streaming_data_reader.py'
```

```
streaming_data_reader.py
100%[=====================================================================>]
1.33K  --.-KB/s    in 0s

2023-09-29 12:15:33 (114 MB/s) - 'streaming_data_reader.py' saved
[1364/1364]
```

- Set Script Database Details

```
"""
Streaming data consumer
"""
from datetime import datetime
from kafka import KafkaConsumer
import mysql.connector

TOPIC='toll'
DATABASE = 'tolldata'
USERNAME = 'root'
PASSWORD = 'MjE4NS1jcmFpZ3Ry'

print("Connecting to the database")
try:
    connection = mysql.connector.connect(host='localhost',
database=DATABASE, user=USERNAME, password=PASSWORD)
except Exception:
    print("Could not connect to database. Please check credentials")
else:
    print("Connected to database")
cursor = connection.cursor()

print("Connecting to Kafka")
consumer = KafkaConsumer(TOPIC)
print("Connected to Kafka")
print(f"Reading messages from the topic {TOPIC}")
for msg in consumer:

    # Extract information from kafka

    message = msg.value.decode("utf-8")

    # Transform the date format to suit the database schema
```

```
    (timestamp, vehcile_id, vehicle_type, plaza_id) = message.split(",")

    dateobj = datetime.strptime(timestamp, '%a %b %d %H:%M:%S %Y')
    timestamp = dateobj.strftime("%Y-%m-%d %H:%M:%S")

    # Loading data into the database table

    sql = "insert into livetolldata values(%s,%s,%s,%s)"
    result = cursor.execute(sql, (timestamp, vehcile_id, vehicle_type,
plaza_id))
    print(f"A {vehicle_type} was inserted into the database")
    connection.commit()
connection.close()
```

- Run Python and Read Streaming Data into MySQL database/table

```
theia@theiadocker-craigtrupp8:/home/project/kafka_2.12-2.8.0$ python3
streaming_data_reader.py
Connecting to the database
Connected to database
Connecting to Kafka
Connected to Kafka
Reading messages from the topic toll
```

- Task 2.9 - Health check of the streaming data pipeline.
- If you have done all the steps till here correctly, the streaming toll data would get stored in the table livetolldata.

- List the top 10 rows in the table livetolldata.

```
theia@theiadocker-craigtrupp8:/home/project$ mysql --host=127.0.0.1
--port=3306 --user=root --password=MjE4NS1jcmFpZ3Ry
mysql: [Warning] Using a password on the command line interface can be
insecure.
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 85
Server version: 8.0.22 MySQL Community Server - GPL

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
```

```
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input
statement.

mysql> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
| tolldata           |
+--------------------+
5 rows in set (0.00 sec)

mysql> USE tolldata;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> SHOW TABLES;
+--------------------+
| Tables_in_tolldata |
+--------------------+
| livetolldata       |
+--------------------+
1 row in set (0.00 sec)

mysql> SELECT * FROM livetolldata LIMIT 10;
+---------------------+------------+--------------+---------------+
| timestamp           | vehicle_id | vehicle_type | toll_plaza_id |
+---------------------+------------+--------------+---------------+
| 2023-09-29 12:18:41 |     273519 | van          |          4005 |
| 2023-09-29 12:18:43 |    5070533 | car          |          4009 |
| 2023-09-29 12:18:43 |    8637788 | car          |          4000 |
| 2023-09-29 12:18:43 |    2302270 | car          |          4004 |
| 2023-09-29 12:18:45 |    8994838 | van          |          4009 |
| 2023-09-29 12:18:47 |    8636006 | car          |          4001 |
| 2023-09-29 12:18:47 |    7036490 | truck        |          4003 |
| 2023-09-29 12:18:48 |    3349586 | car          |          4004 |
```

```
| 2023-09-29 12:18:49 |    7283296 | car          |          4004 |
| 2023-09-29 12:18:49 |    4458162 | car          |          4005 |
+---------------------+------------+--------------+---------------+
10 rows in set (0.00 sec)

mysql>
```