Final Assignment (Part 2) - Creating Streaming Data Pipelines using Kafka

Estimated time needed: **45** minutes.

About This SN Labs Cloud IDE

This Skills Network Labs Cloud IDE provides a hands-on environment for course and project related labs. It utilizes Theia, an open-source IDE (Integrated Development Environment) platform, that can be run on desktop or on the cloud. To complete this lab, we will be using the Cloud IDE based on Theia and Kafka and MySQL database running in a Docker container. You will also need an instance of DB2 running in IBM Cloud.

Important Notice about this lab environment

Please be aware that sessions for this lab environment are not persistent. A new environment is created for you every time you connect to this lab. Any data you may have saved in the earlier session will get lost. Please plan to complete these labs in a single session to avoid losing your data.

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.

Note - Screenshots

Throughout this lab you will be prompted to take screenshots and save them on your own device. These screenshots will need to be uploaded for peer review in the next section of the course. You can use various free screengrabbing tools or your operating system's shortcut keys (Alt + PrintScreen in Windows, for example) to capture the required screenshots.

Exercise 1 - Prepare the lab environment

Before you start the assignment, complete the following steps to set up the lab:

- Step 1: Download Kafka.

```
1
```

```
wget https://archive.apache.org/dist/kafka/2.8.0/kafka_2.12-2.8.0.tgz
```

Copied!

- Step 2: Extract Kafka.

```
1
```

```
tar -xzf kafka_2.12-2.8.0.tgz
```

Copied!

- Step 3: Start MySQL server.

```
1
```

```
start_mysql
```

Copied!

- Step 4: 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.

```
1
```

```
mysql --host=127.0.0.1 --port=3306 --user=root --password=Mjk0NDQtcnNhbm5h
```

Copied!

- Step 5: Create a database named `tolldata`.

At the 'mysql>' prompt, run the command below to create the database.

```
1
```

```
create database tolldata;
```

Copied!

- Step 6: Create a table named `livetolldata` with the schema to store the data generated by the traffic simulator.

Run the following command to create the table:

```
1
2
3
```

```
use tolldata;

create table livetolldata(timestamp datetime,vehicle_id int,vehicle_type char(15),toll_plaza_id smallint);
```

Copied!

This is the table where you would store all the streamed data that comes from kafka. Each row is a record of when a vehicle has passed through a certain toll plaza along with its type and anonymized id.

- Step 7: Disconnect from MySQL server.

```
1
```

```
exit
```

Copied!

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

```
1
```

```
1. python3 -m pip install kafka-python
```

Copied!

This python module will help you to communicate with kafka server. It can used to send and receive messages from kafka.

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

```
1. 1
```

```
1. python3 -m pip install mysql-connector-python==8.0.31
```

Copied!

This python module will help you to interact with mysql server.

Exercise 2 - Start Kafka

Task 2.1 - Start Zookeeper

Start zookeeper server.

Take a screenshot of the command you run.

Name the screenshot `start_zookeeper.jpg`. (Images can be saved with either the .jpg or .png extension.)

Task 2.2 - Start Kafka server

Start Kafka server

Take a screenshot of the command you run.

Name the screenshot `start_kafka.jpg`. (Images can be saved with either the .jpg or .png extension.)

Task 2.3 - Create a topic named `toll`

Create a Kakfa topic named `toll`

Take a screenshot of the command you run.

Name the screenshot `create_toll_topic.jpg`. (Images can be saved with either the .jpg or .png extension.)

Task 2.4 - Download the Toll Traffic Simulator

Download the `toll_traffic_generator.py` from the url given below using ‘wget’.

```
https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DB0250EN-SkillsNetwork/labs/Final%20Assignment/toll_traffic_generator.py
```

Open the code using the theia editor using the “Menu –> File –>Open” option.

Take a screenshot of the task code.

Name the screenshot `download_simulator.jpg`. (Images can be saved with either the .jpg or .png extension.)

Task 2.5 - Configure the Toll Traffic Simulator

Open the `toll_traffic_generator.py` and set the topic to `toll`.

Take a screenshot of the task code with the topic clearly visible.

Name the screenshot `configure_simulator.jpg`. (Images can be saved with either the .jpg or .png extension.)

Task 2.6 - Run the Toll Traffic Simulator

Run the `toll_traffic_generator.py`.

Hint : `python3 <pythonfilename>` runs a python program on the theia lab.

Take a screenshot of the output of the simulator.

Name the screenshot `simulator_output.jpg`. (Images can be saved with either the .jpg or .png extension.)

Task 2.7 - Configure `streaming_data_reader.py`

Download the `streaming_data_reader.py` from the url below using 'wget'.

```
https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DB0250EN-SkillsNetwork/labs/Final%20Assignment/streaming_data_reader.py
```

Open the `streaming_data_reader.py` and modify the following details so that the program can connect to your mysql server.

```
TOPIC

DATABASE

USERNAME

PASSWORD
```

Take a screenshot of the code you modified.

Name the screenshot `streaming_reader_code.jpg`. (Images can be saved with either the .jpg or .png extension.)

Task 2.8 - Run `streaming_data_reader.py`

Run the `streaming_data_reader.py`

Take a screenshot of the output of the streaming_data_reader.py.

Name the screenshot `data_reader_output.jpg`. (Images can be saved with either the .jpg or .png extension.)

```
1. 1

1. python3 streaming_data_reader.py
```

Copied!

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`.

Take a screenshot of the command and the output.

Name the screenshot `output_rows.jpg`. (Images can be saved with either the .jpg or .png extension.)

This concludes the assignment.

Authors

Ramesh Sannareddy

Other Contributors

Rav Ahuja

Change Log

Date (YYYY-MM-DD)	Version	Changed By	Change Description
2021-08-16	0.1	Ramesh Sannareddy	Created initial version
2021-12-01	0.2	Jeff Grossman	Added copy code blocks
2022-09-21	0.3	Appalabhaktula Hema	Updated code blocks
2022-11-10	0.4	Appalabhaktula Hema	Corrected instructions