Apache Kafka Setup and Commands on Ubuntu

1. Setup Apache Kafka in Ubuntu

- First of all, we search and download Kafka from the search engine. I am putting the link of the latest version for now.

Source: https://dlcdn.apache.org/kafka/3.1.0/kafka_2.13-3.1.0.tgz

- After the file is downloaded we need to extract it.

```
tar xzf <file_name>
```

The above command will extract the file.

- We will use kafdrop to visualize Kafka, so we will download maven first and then kafdrop.

```
sudo apt-get install maven
```

KafDrop Links: https://github.com/obsidiandynamics/kafdrop.git

Kafdrop will download as zip. To unzip;

```
unzip <file_name>
```

I put all the files in opt to be organized.

```
mv kaf* /opt/
```

2. Let’s raise Apache Kafka

- In order to run Kafka, we first need to run the **zookeeper**.

```
cd opt/kafka/bin && ./zookeeper-server-start.sh /opt/kafka/config/
zookeeper.properties
```

- Now let’s run our **broker** in a separate terminal.

```
cd opt/kafka/bin && ./kafka-server-start.sh /opt/kafka/config/
server.properties
```

- Let’s create our **topic** where we will write our data in a separate terminal.

```
cd /opt/kafka/bin && ./kafka-topics.sh --bootstrap-server localhost:9092
 --topic <topic_name> --create -- replication-factor <> --config
 retention.ms=<> --partitions <>
```

- To view the topics we have created;

```
cd /opt/kafka/bin && ./kafka-topics.sh --list --bootstrap-server
 localhost:9092
```

- Let’s run **kafdrop** to visualize what we’ve created.

```
cd /opt/kafdrop/target && sudo java -jar <file_name> --
kafka.brokerConnect=localhost:9092
```

- Let’s code a producer with Python.

```
sudo apt-get install pip
sudo pip install confluent-kafka
```

We have downloaded the necessary python libraries.

```
from confluent_kafka import Producer
conf={'bootstrap.servers':"localhost:9092"}
producer=Producer(conf)
def sonuc(err,msg):
    if err is not None:
        print("Hatalı bir gönderim.")
    else:
        print("Gönderme işlemi başarılı.")
producer.produce(<topic_name>,key=<>,value=<>,callback=sonuc,partition=<>)
producer.poll(<>)
```

- Let’s code a consumer with Python.

```
from socket import timeout
from confluent_kafka import Consumer
conf={'bootstrap.servers':"localhost:9092",'group.id':<>}
cons=Consumer(conf)
def basic_loop(consumer,topics):
    try:
        consumer.subscribe(topics)
        while True:
            msg=consumer.poll(timeout=1.0)
            if msg is None: continue
            if msg.error():
                print("Beklenmedik bir hata oluştu.")
            else:
                print("Gelen mesaj:
{}".format(msg.value().decode('utf-8')))
    finally:
        consumer.close()
basic_loop(consumer=cons,topics=["<>"])
```

- We created the consumer and producer. All that remains is to run them.

```
python3 consumer.py
```

```
python3 producer.py
```

- If we want to listen to the topic over kafka;

```
cd /opt/kafka/bin && kafka-console-consumer.sh --bootstrap-server
 localhost:9092 --topic <topic_name> --from-beginning
```

- Let’s do a producer performance test.

```
/opt/kafka/bin && ./kafka-producer-perf-test.sh --topic <> --
num-records <> --throughput <> --record-size <> --producer-props
 bootstrap.servers=localhost:9092
```

- Let’s do a consumer performance test.

```
/opt/kafka/bin && ./kafka-consumer-perf-test.sh --topic <>  --bootstrap-
server localhost:9092 --messages <> --threads <>
```