Shraddha Patel A20499171

CSP554—Big Data Technologies

Assignment #9

Readings

Read (From the Free Books and Chapters section of our blackboard site):

- Learning Spark, Ch. 8 (pp. 207-234)
- Spark: The Definitive Guide (pp.26-32)

Worth: 5 points + 5 points extra credit

Due by the start of the next class period

Assignments should be uploaded via the Blackboard portal

Exercise 1) 5 points

Read the article "Real-time stream processing for Big Data" available on the blackboard in the 'Articles' section and then answer the following questions:

a) (1.25 points) What is the Kappa architecture and how does it differ from the lambda architecture?

Answer:

The Kappa architecture do all its computation in stream Processing System alone and perform all the computation again by replaying historical data only when the business logic change. This architecture does not recompute all data in batch layer periodically.

Difference:

The lambda architecture needs two processor for batch processing and stream of the data. This architecture has high latency due huge volume of batch processing

The kappa architecture uses single processor for streaming and batch processing. It reprocesses only when the business logic changes. This architecture has low latency.

b) (1.25 points) What are the advantages and drawbacks of pure streaming versus micro-batch real-time processing systems?

Answer:

Advantages:

Pure streaming systems have low latency

Micro-batch Realtime processing systems have high latency.

Disadvantages:

Pure streaming systems have High cost per item

Micro batch real-time processing systems have efficient unparalleled resources.

c) (1.25 points) In few sentences describe the data processing pipeline in Storm.

Answer:

The data processing pipeline is referring to as topology or application in storm. The topology is a graph which consists of nodes (spouts and bolts) and edges (data flow between nodes).

The spouts nodes responsible for take the data and initiate the data flow in topology. These nodes also transmit tuples downstream to bolt nodes where they execute processing, write data to eternal storage, or sent tuples to another bolt.

d) (1.25 points) How does Spark streaming shift the Spark batch processing approach to work on real-time data streams?

Answer:

Spark streaming shift the Spark batch processing approach to work on real-time data streams by dividing the the stream of incoming data into small batches, transform it into RDDs which then be processed in usual way. Spark streaming does the data flow and distribution automatically.

Exercise 2) 5 points (extra credit; if you don't want to try or if you try and can't get things to work, this won't impact your score negatively)

Step A – Start an EMR cluster

Step B – Copy the Kafka software to the EMR master node

Step C – Install the Kafka software and start it

Enter the following command:

```
tar -xzf kafka_2.13-3.0.0.tgz
```

Then enter this command:

```
pip install kafka-python
```

Now enter the following commands into the terminal:

```
cd kafka_2.13-3.0.0

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

Step D – Prepare to run Kafka producers and consumers

Step E – Create a Kafka topic

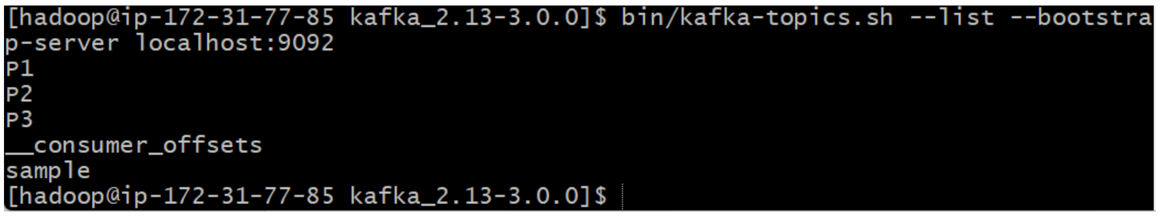
In the Producer-Term, enter the following command:

```
cd kafka_2.13-3.0.0

bin/kafka-topics.sh --create --replication-factor 1 --partitions 1 --bootstrap-server
localhost:9092 --topic sample
```

To list the topics that you created you can enter the following into the Producer-Term (note some default topics already exist):

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



```
[hadoop@ip-172-31-77-85 kafka_2.13-3.0.0]$ bin/kafka-topics.sh --list --bootstra
p-server localhost:9092
P1
P2
P3
__consumer_offsets
sample
[hadoop@ip-172-31-77-85 kafka_2.13-3.0.0]$
```

a)

In the Producer-Term (or some other way) write a small program, call it ‘put.py’, using the vi text or some other way of putting a python program onto the EMR master node. If you like you could use a text editor on your PC/MAC to write the program and then scp it over to your EMR master name.

This program should implement a kafka producer that writes three messages to the topic ‘sample’. Recall that you need to convert values and keys to type bytes. The three messages should have keys and values as follows:

Key	Value
‘MYID’	Your student id
‘MYNAME’	Your name
‘MYEYECOLOR’	Your eye color (make it up if you can’t remember)

Execute this program in the Producer-Term, use the command line (you might need to provide a full pathname depending on where your python program is such as /home/hadoop/someplace/put.py):

```
python put.py
```

Submit the program as your answer to ‘part a’ of this exercise

Program: put.py

```
from kafka import KafkaProducer
producer = KafkaProducer(bootstrap_servers=['localhost:9092'])

data={"MYID": "A20499171" ,"MYNAME": "Shraddha", "MYEYECOLOR":"brown"}
for a , b in data.items():
   a_bytes=bytes (a, 'utf-8')
   b_bytes=bytes (b, 'utf-8')
   producer. Send ('sample', key =a_bytes, value =b_bytes)

producer. Close()
```

b)

In the Consumer-Term, write another small program, call it ‘get.py’, using the vi text or some other way of putting a python program onto the EMR master node.

This program should implement a kafka consumer that reads the messages you wrote previously from the topic ‘sample’ and writes them to the terminal.

The output should look something like this:

Key=MYID, Value='your id number'

Key=MYNAME, Value='your name'

Key=MYEYECOLOR, Value='your eye color'

Execute this program in the Consumer-Term. Use the command line:

```
python get.py
```

Note, if needed you can terminate the program by entering 'ctrl-c'.

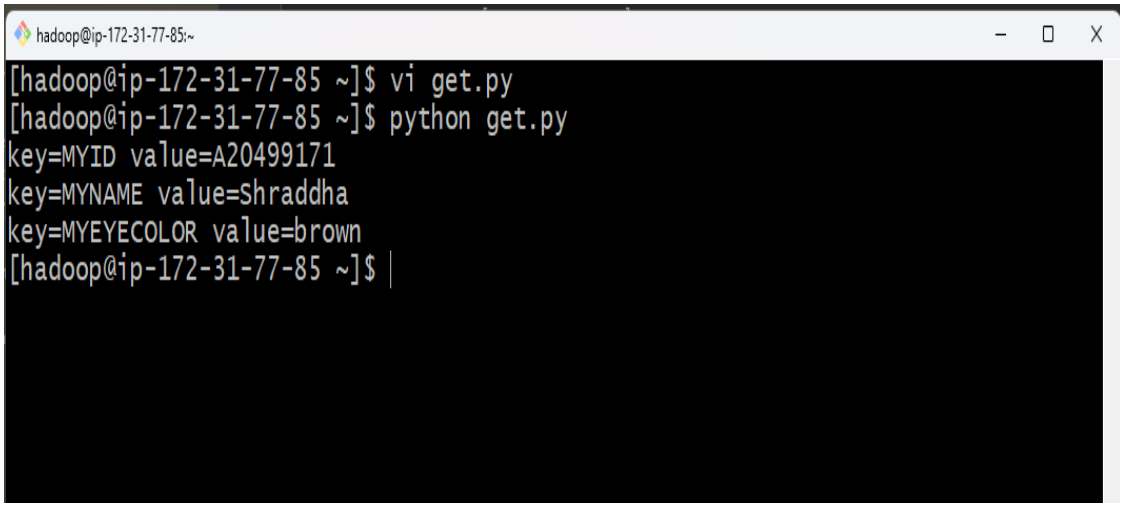
Submit the program and a screenshot of its output as your answer to 'part b' of this exercise.

Program: get.py

```
from kafka import KafkaConsumer
consumer = KafkaConsumer(
   'sample',
    bootstrap_servers=['localhost:9092'],
    auto_offset_reset='earliest',
    enable_auto_commit=True,
    consumer_timeout_ms=10000,
    group_id='my-group')

for message in consumer:
    print("key=%s value=%s" % (message.key.decode('utf-8'),message.value.decode('utf-
8')))

consumer.close()
```


hadoop@ip-172-31-77-85:~

```
[hadoop@ip-172-31-77-85 ~]$ vi get.py
[hadoop@ip-172-31-77-85 ~]$ python get.py
key=MYID value=A20499171
key=MYNAME value=Shraddha
key=MYEYECOLOR value=brown
[hadoop@ip-172-31-77-85 ~]$
```