

## DSC650-Week-07-Mukherjee

**Author:** Chitramoy Mukherjee

**Date:** 01/27/2024

**Exercise 1:** Topic Creation and Verification in Kafka (On One Terminal Only)

**Exercise 1:** Create a Kafka topic named 'my-topic'.

```
/opt/kafka_2.13-2.8.1/bin/kafka-topics.sh --create --topic my-topic --bootstrap-server localhost:9092
```

**Exercise 2:** List the topics to verify that 'my-topic' has been successfully created.

```
/opt/kafka_2.13-2.8.1/bin/kafka-topics.sh --list --bootstrap-server localhost:9092
```

```
root@80f9f96f054e:/# /opt/kafka_2.13-2.8.1/bin/kafka-topics.sh --create --topic my-topic --bootstrap-server localhost:9092
Created topic my-topic.
root@80f9f96f054e:/# /opt/kafka_2.13-2.8.1/bin/kafka-topics.sh --list --bootstrap-server localhost:9092
my-topic
root@80f9f96f054e:/#
```

**Exercise 3:** In the first terminal, start a Kafka consumer.

```
/opt/kafka_2.13-2.8.1/bin/kafka-console-consumer.sh --topic my-topic --from-beginning --bootstrap-server localhost:9092
```

**Exercise 4:** In the second terminal, start a Kafka producer:

```
/opt/kafka_2.13-2.8.1/bin/kafka-console-producer.sh --topic my-topic --bootstrap-server localhost:9092
```

```
chitramoy@bigdata: ~/dsc650-infra/bellevue-bigdata/kafka
$ docker ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED      STATUS      PORTS      NAMES
chitramoy@bigdata:~/dsc650-infra/bellevue-bigdata/hadoop-hive-spark-hbase$ cd ..
chitramoy@bigdata:~/dsc650-infra/bellevue-bigdata$ ls -ltr
total 16
drwxrwxr-x 2 chitramoy chitramoy 4096 Dec  5 01:26 solr
drwxr-xr-x 3 root      root      4096 Dec  5 01:29 nifi
drwxrwxr-x 6 chitramoy chitramoy 4096 Dec  5 01:31 hadoop-hive-spark-hbase
drwxrwxr-x 4 chitramoy chitramoy 4096 Jan 27 12:57 kafka
chitramoy@bigdata:~/dsc650-infra/bellevue-bigdata$ cd kafka
chitramoy@bigdata:~/dsc650-infra/bellevue-bigdata/kafka$ sudo docker-compose up -d
Starting kafka_kafka_1 ... done
Starting kafka_zookeeper_1 ... done
chitramoy@bigdata:~/dsc650-infra/bellevue-bigdata/kafka$ sudo docker exec -it kafka_kafka_1 bash
root@80f9f96f054e:/# /opt/kafka_2.13-2.8.1/bin/kafka-topics.sh --list --bootstrap-server localhost:9092
>hello world
>test
>Bellevue
>Chitramoy
>DSC_650 Week-7 Assignment
>

root@80f9f96f054e:/# /opt/kafka_2.13-2.8.1/bin/kafka-topics.sh --create --topic my-topic --bootstrap-server localhost:9092
Exception in thread "main" java.lang.UnrecognizedOptionException: bootstr is not a recognized option
    at joptsimple.OptionException.unrecognizedOption(OptionException.java:108)
    at joptsimple.OptionParser.handleLongOptionToken(OptionParser.java:510)
    at joptsimple.OptionParserState$2.handleArgument(OptionParserState.java:56)
    at joptsimple.OptionParser.parse(OptionParser.java:396)
    at kafka.admin.TopicCommand$TopicCommandOptions.<init>(TopicCommand.scala:691)
    at kafka.admin.TopicCommand$.main(TopicCommand.scala:52)
    at kafka.admin.TopicCommand.main(TopicCommand.scala)
root@80f9f96f054e:/# /opt/kafka_2.13-2.8.1/bin/kafka-topics.sh --create --topic my-topic --bootstrap-server localhost:9092
Created topic my-topic.
root@80f9f96f054e:/# /opt/kafka_2.13-2.8.1/bin/kafka-topics.sh --list --bootstrap-server localhost:9092
my-topic
root@80f9f96f054e:/# /opt/kafka_2.13-2.8.1/bin/kafka-console-consumer.sh --topic my-topic --from-beginning --bootstrap-server localhost:9092
^CProcessed a total of 0 messages
root@80f9f96f054e:/# /opt/kafka_2.13-2.8.1/bin/kafka-console-consumer.sh --topic my-topic --from-beginning --bootstrap-server localhost:9092
my-topic --from-beginning --bootstrap-server localhost:9092

hello world
test
Bellevue
Chitramoy
DSC_650 Week-7 Assignment
```

**Exercise 5:** Run a performance test on the producer using the Kafka producer performance test script with provided arguments.

```
/opt/kafka_2.13-2.8.1/bin/kafka-producer-perf-test.sh --topic my-topic --num-records 50000 --record-size 100 --throughput 1000 --producer-props bootstrap.servers=localhost:9092 key.serializer=org.apache.kafka.common.serialization.StringSerializer value.serializer=org.apache.kafka.common.serialization.StringSerializer
```

**Exercise 6:** Following the producer test, run a consumer performance test on ‘my-topic’:

```
/opt/kafka_2.13-2.8.1/bin/kafka-consumer-perf-test.sh --broker-list localhost:9092 --topic my-topic --messages 50000
```

```
root@859121cfff686:/# /opt/kafka_2.13-2.8.1/bin/kafka-producer-perf-test.sh --topic my-topic --num-records 50000 --record-size 100 --throughput 1000 --producer-props bootstrap.servers=localhost:9092 key.serializer=org.apache.kafka.common.serialization.StringSerializer value.serializer=org.apache.kafka.common.serialization.StringSerializer
[2024-01-28 18:42:20,192] WARN [Producer clientId=producer-1] Error while fetching metadata with correlation id 1 : {my-topic=LEADER_NOT_AVAILABLE} (org.apache.kafka.clients.NetworkClient)
[2024-01-28 18:42:20,301] WARN [Producer clientId=producer-1] Error while fetching metadata with correlation id 3 : {my-topic=LEADER_NOT_AVAILABLE} (org.apache.kafka.clients.NetworkClient)
[2024-01-28 18:42:20,406] WARN [Producer clientId=producer-1] Error while fetching metadata with correlation id 4 : {my-topic=LEADER_NOT_AVAILABLE} (org.apache.kafka.clients.NetworkClient)
5001 records sent, 1000.0 records/sec (0.10 MB/sec), 18.0 ms avg latency, 914.0 ms max latency.
5006 records sent, 1001.0 records/sec (0.10 MB/sec), 1.0 ms avg latency, 9.0 ms max latency.
5004 records sent, 1000.6 records/sec (0.10 MB/sec), 0.8 ms avg latency, 9.0 ms max latency.
5004 records sent, 1000.4 records/sec (0.10 MB/sec), 0.8 ms avg latency, 9.0 ms max latency.
5006 records sent, 1001.2 records/sec (0.10 MB/sec), 0.7 ms avg latency, 7.0 ms max latency.
5002 records sent, 1000.2 records/sec (0.10 MB/sec), 0.7 ms avg latency, 8.0 ms max latency.
5001 records sent, 1000.2 records/sec (0.10 MB/sec), 0.7 ms avg latency, 11.0 ms max latency.
5002 records sent, 1000.2 records/sec (0.10 MB/sec), 0.6 ms avg latency, 6.0 ms max latency.
5001 records sent, 1000.2 records/sec (0.10 MB/sec), 0.7 ms avg latency, 10.0 ms max latency.
50000 records sent, 999.740068 records/sec (0.10 MB/sec), 2.48 ms avg latency, 914.00 ms max latency, 1 ms 50th, 2 ms 95th, 87 ms 99th, 91 ms 99.9th.
root@859121cfff686:/# /opt/kafka_2.13-2.8.1/bin/kafka-consumer-perf-test.sh --broker-list localhost:9092 --topic my-topic --messages 50000
^C
root@859121cfff686:/# /opt/kafka_2.13-2.8.1/bin/kafka-consumer-perf-test.sh --broker-list localhost:9092 --topic my-topic --messages 50000
start.time, end.time, data.consumed.in.MB, MB.sec, data.consumed.in.nMsg, nMsg.sec, rebalance.time.ms, fetch.time.ms, fetch.MB.sec, fetch.nMsg.sec
2024-01-28 18:47:43:915, 2024-01-28 18:47:45:636, 4.7684, 2.7707, 50000, 29052.8762, 1348, 373, 12.7838, 134048.2574
root@859121cfff686:/#
```

## Result Description:

The performance test started on January 28, 2024, at 18:47:43:915 and ended at 18:47:45:636. During the test, 4.7684 megabytes of data were consumed at a rate of 2.7707 megabytes per second. A total of 50,000 messages were consumed at a rate of 29,052.8762 messages per second. The rebalancing process took 1,348 milliseconds. The time taken for message fetching was 373 milliseconds. The rate of fetching data was 12.7838 megabytes per second. The rate of fetching messages was 134,048.2574 messages per second.

This output provides insights into the performance characteristics of the Kafka system during the specific test scenario, including data consumption rates, rebalancing times, and message fetching metrics.

**Exercise 7:** Create a topic, this time partitioned and replicated across all three Kafka instances:

```
/opt/kafka_2.13-2.8.1/bin/kafka-topics.sh --create --topic my-partitioned-topic --replication-factor 3 --partitions 3 --bootstrap-server localhost:9092
```

**Exercise 8:** Conduct the producer and consumer performance tests on the new topic, observing differences:

```
/opt/kafka_2.13-2.8.1/bin/kafka-producer-perf-test.sh --topic my-partitioned-topic --num-records 50000 --record-size 100 --throughput 1000 --producer-props bootstrap.servers=localhost:9092 key.serializer=org.apache.kafka.common.serialization.StringSerializer value.serializer=org.apache.kafka.common.serialization.StringSerializer
```

Followed by:

```
/opt/kafka_2.13-2.8.1/bin/kafka-consumer-perf-test.sh --broker-list localhost:9092 --topic my-partitioned-topic --messages 50000
```

```
root@859121cfff686:/# exit
exit
chitramoy@bigdata:~/dsc650-infra/bellevue-bigdata/kafka$ sudo docker-compose scale kafka=3
WARNING: The scale command is deprecated. Use the up command with the --scale flag instead.
Starting kafka_kafka_1 ... done
Creating kafka_kafka_2 ... done
Creating kafka_kafka_3 ... done
chitramoy@bigdata:~/dsc650-infra/bellevue-bigdata/kafka$ sudo docker exec -it kafka_kafka_1 bash
root@859121cfff686:/# /opt/kafka_2.13-2.8.1/bin/kafka-topics.sh --create --topic my-partitioned-topic --replication-factor 3 --partitions 3 --bootstrap-server localhost:9092
Created topic my-partitioned-topic.
root@859121cfff686:/# /opt/kafka_2.13-2.8.1/bin/kafka-producer-perf-test.sh --topic my-partitioned-topic --num-records 50000 --record-size 100 --throughput 1000 --producer-props bootstrap.servers=localhost:9092 key.serializer=org.apache.kafka.common.serialization.StringSerializer value.serializer=org.apache.kafka.common.serialization.StringSerializer
[2024-01-28 18:51:18,394] WARN [Producer clientId=producer-1] Error while fetching metadata with correlation id 1 : {my-partitioned-topic=LEADER_NOT_AVAILABLE}
5001 records sent, 1000.0 records/sec (0.10 MB/sec), 11.1 ms avg latency, 638.0 ms max latency.
5010 records sent, 1002.0 records/sec (0.10 MB/sec), 1.0 ms avg latency, 10.0 ms max latency.
5001 records sent, 1000.2 records/sec (0.10 MB/sec), 0.8 ms avg latency, 9.0 ms max latency.
5004 records sent, 1000.4 records/sec (0.10 MB/sec), 0.8 ms avg latency, 16.0 ms max latency.
5003 records sent, 1000.4 records/sec (0.10 MB/sec), 0.7 ms avg latency, 8.0 ms max latency.
5001 records sent, 1000.2 records/sec (0.10 MB/sec), 0.7 ms avg latency, 5.0 ms max latency.
5002 records sent, 1000.4 records/sec (0.10 MB/sec), 0.7 ms avg latency, 6.0 ms max latency.
5002 records sent, 1000.2 records/sec (0.10 MB/sec), 0.7 ms avg latency, 9.0 ms max latency.
5001 records sent, 1000.2 records/sec (0.10 MB/sec), 0.7 ms avg latency, 12.0 ms max latency.
50000 records sent, 999.780048 records/sec (0.10 MB/sec), 1.79 ms avg latency, 638.00 ms max latency, 1 ms 50th, 2 ms 95th, 69 ms 99th, 78 ms 99.9th.
bash: ation.StringSerializer: command not found
root@859121cfff686:/# /opt/kafka_2.13-2.8.1/bin/kafka-consumer-perf-test.sh --broker-list localhost:9092 --topic my-partitioned-topic --messages 50000
start.time, end.time, data.consumed.in.MB, MB.sec, data.consumed.in.nMsg, nMsg.sec, rebalance.time.ms, fetch.time.ms, fetch.MB.sec, fetch.nMsg.sec
WARNING: Exiting before consuming the expected number of messages: timeout (10000 ms) exceeded. You can use the --timeout option to increase the timeout.
2024-01-28 18:52:29:437, 2024-01-28 18:52:39:488, 0.0000, 0.0000, 0, 0.0000, 473, 9578, 0.0000, 0.0000
root@859121cfff686:/#
```

## Result Description:

These exercises involve creating a partitioned and replicated Kafka topic and then performing producer and consumer performance tests on that topic. The partitioning and replication configurations are crucial for scalability, fault tolerance, and ensuring high availability in a distributed Kafka environment. It is vital to comprehend the proper way to partition data, as this knowledge is essential for enhancing the performance and reliability of your Kafka configuration. The performance tests help assess the throughput and efficiency of the Kafka producers and consumers under specified conditions. For larger data sets, partitioning proves to be significantly more optimal, particularly in terms of parallel computing and fault tolerance.