Kafka Assignment

Siddharth Kumar

October 2, 2025

Contents

Chapter 1

Task 1: Kafka Cluster Setup and Topic Operations

1.1 Topic Creation

```
1 #path-> /usr/odp/3.2.2.0-1/kafka/bin
2 ./kafka-topics.sh --create --bootstrap-server stg-xxx:6667 \
3 --replication-factor 3 --partitions 10 --topic test_topic
```

Explanation
- **Line 2:** Executes the `./kafka-topics.sh` script with the `--create` action, connecting to the Kafka cluster via the specified `--bootstrap-server` at `stg-xxx:6667`.
- **Line 3:** Defines the topic properties: sets the `--replication-factor` to 3 for fault tolerance, divides the topic into 10 `--partitions` for parallelism, and names it `--topic test_topic`.

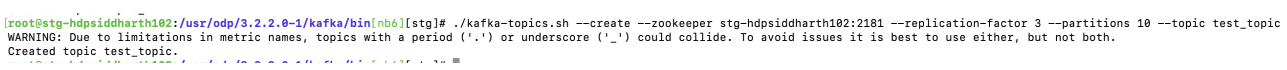

Figure 1.1: Kafka topic creation output

1.2 Topic Description

```
1 #path-> /usr/odp/3.2.2.0-1/kafka/bin
2 ./kafka-topics.sh --describe --bootstrap-server stg-xxx:6667 --
     topic test_topic
```

Explanation

- **Line 2:** Executes the `./kafka-topics.sh` script with the `--describe` action to fetch details about an existing topic. It connects to the cluster via the `--bootstrap-server stg-xxx:6667` and specifies the target topic as `--topic test_topic`.

```
^Croot@stg-hdpsiddharth102:/usr/odp/3.2.2.0-1/kafka/bin[nb6][stg]# ./kafka-topics.sh --describe --bootstrap-server stg-hdpsiddharth102:6667
opic test_topic
Topic: test_topic       TopicId: 9pFEK8R2SD20cXH1hQ9dgg PartitionCount: 10      ReplicationFactor: 3    Configs: compression.type=producer,
insync.replicas=1,segment.bytes=1073741824,max.message.bytes=1000000,index.interval.bytes=4096,retention.bytes=-1,segment.index.bytes=10485
        Topic: test_topic       Partition: 0    Leader: 1002    Replicas: 1002,1001,1003        Isr: 1002,1001,1003
        Topic: test_topic       Partition: 1    Leader: 1003    Replicas: 1003,1002,1001        Isr: 1003,1002,1001
        Topic: test_topic       Partition: 2    Leader: 1001    Replicas: 1001,1003,1002        Isr: 1001,1003,1002
        Topic: test_topic       Partition: 3    Leader: 1002    Replicas: 1002,1003,1001        Isr: 1002,1003,1001
        Topic: test_topic       Partition: 4    Leader: 1003    Replicas: 1003,1001,1002        Isr: 1003,1001,1002
        Topic: test_topic       Partition: 5    Leader: 1001    Replicas: 1001,1002,1003        Isr: 1001,1002,1003
        Topic: test_topic       Partition: 6    Leader: 1002    Replicas: 1002,1001,1003        Isr: 1002,1001,1003
        Topic: test_topic       Partition: 7    Leader: 1003    Replicas: 1003,1002,1001        Isr: 1003,1002,1001
        Topic: test_topic       Partition: 8    Leader: 1001    Replicas: 1001,1003,1002        Isr: 1001,1003,1002
        Topic: test_topic       Partition: 9    Leader: 1002    Replicas: 1002,1003,1001        Isr: 1002,1003,1001
```

Figure 1.2: Kafka describe

1.3 Produce and Consume Data

```
1  # Produce data
2  #path-> /usr/odp/3.2.2.0-1/kafka/bin
3  ./kafka-console-producer.sh \
4  > --bootstrap-server stg-XXX:6667 \
5  > --topic test_topic \
6  > --property  parse.key=true \
7  > --property key.separator=":"
8
9
10 # Consume data
11 #path-> /usr/odp/3.2.2.0-1/kafka/bin
12 ./kafka-console-consumer.sh \
13 > --bootstrap-server stg-xxx:6667 \
14 > --topic test_topic \
15 > --from-beginning \
16 > --group test_consumer_group \
17 > --property print.key=true \
18 > --property print.partition=true \
19 > --property print.offset=true \
20 > --property print.timestamp=true
```

Explanation

Produce Data (Lines 3-7):
- **Line 3:** Executes the `./kafka-console-producer.sh` script, which starts an interactive command-line tool to send messages to a Kafka topic.
- **Line 4-5:** Connects to the cluster via `--bootstrap-server stg-XXX:6667` and specifies that messages should be sent to the `--topic test_topic`.
- **Line 6-7:** Configures the producer's properties. `parse.key=true` tells it to split input into a key and a value, and `key.separator=":"` sets the colon as the delimiter between the key and the value.

Consume Data (Lines 12-20):
- **Line 12:** Executes the `./kafka-console-consumer.sh` script, which starts a tool to read messages from a Kafka topic and print them to the console.
- **Line 13-14:** Connects to the same Kafka cluster and specifies it will read from `--topic test_topic`.
- **Line 15:** The `--from-beginning` flag instructs the consumer to read all existing messages in the topic, starting from the earliest offset.
- **Line 16:** Assigns this consumer to the `--group test_consumer_group`. Kafka uses consumer groups to track which messages have been read (offsets).
- **Line 17-20:** Sets properties to enrich the output. These flags instruct the consumer to print not just the message value, but also its metadata: the `key`, `partition`, `offset`, and `timestamp`.

```
root@stg-hdpsiddharth102:/usr/odp/3.2.2.0-1/kafka/bin[nb6][stg]# ./kafka-console-producer.sh \
> --bootstrap-server stg-hdpsiddharth102:6667 \
> --topic test_topic \
> --property parse.key=true \
> --property key.separator=":"
>user1:hello world
>user2:hello sid
>user1:hhihii
>user3:qwerty
>user2:poiuyt
>
```

Figure 1.3: Kafka producer

```
root@stg-hdpsiddharth104:/usr/odp/3.2.2.0-1/kafka/bin[nb6][stg]# ./kafka-console-consumer.sh \
> --bootstrap-server stg-hdpsiddharth102:6667 \
> --topic test_topic \
> --from-beginning \
> --group test_consumer_group \
> --property print.key=true \
> --property print.partition=true \
> --property print.offset=true \
> --property print.timestamp=true
CreateTime:1759214670283        Partition:4     Offset:0        user1   hello world
CreateTime:1759214691312        Partition:9     Offset:0        user2   hello sid
CreateTime:1759214704874        Partition:4     Offset:1        user1   hhihii
CreateTime:1759214749138        Partition:5     Offset:0        user3   qwerty
CreateTime:1759214767365        Partition:9     Offset:1        user2   poiuyt
```

Figure 1.4: Kafka consumer

1.4 Alter Partitions

```
1 #path-> /usr/odp/3.2.2.0-1/kafka/bin
2 ./kafka-topics.sh --alter \
3 > --bootstrap-server stg-xxx:9092 \
4 > --topic test_topic --partitions 20
```

Explanation

- **Line 2:** Executes the `./kafka-topics.sh` script with the `--alter` action. This command is used to modify the properties of an existing topic, such as increasing its partition count.

```
root@stg-hdpsiddharth102:/usr/odp/3.2.2.0-1/kafka/bin[nb6][stg]# ./kafka-topics.sh --describe --bootstrap-server stg-hdpsiddharth102:6667 -
ic test_topic
Topic: test_topic       TopicId: 9pFEK8R2SD20cXH1hQ9dgg PartitionCount: 20      ReplicationFactor: 3    Configs: compression.type=producer,
insync.replicas=1,segment.bytes=1073741824,max.message.bytes=1000000,index.interval.bytes=4096,retention.bytes=-1,segment.index.bytes=10485
        Topic: test_topic       Partition: 0    Leader: 1002    Replicas: 1002,1001,1003        Isr: 1002,1001,1003
        Topic: test_topic       Partition: 1    Leader: 1003    Replicas: 1003,1002,1001        Isr: 1003,1002,1001
        Topic: test_topic       Partition: 2    Leader: 1001    Replicas: 1001,1003,1002        Isr: 1001,1003,1002
        Topic: test_topic       Partition: 3    Leader: 1002    Replicas: 1002,1003,1001        Isr: 1002,1003,1001
        Topic: test_topic       Partition: 4    Leader: 1003    Replicas: 1003,1001,1002        Isr: 1003,1001,1002
        Topic: test_topic       Partition: 5    Leader: 1001    Replicas: 1001,1002,1003        Isr: 1001,1002,1003
        Topic: test_topic       Partition: 6    Leader: 1002    Replicas: 1002,1001,1003        Isr: 1002,1001,1003
        Topic: test_topic       Partition: 7    Leader: 1003    Replicas: 1003,1002,1001        Isr: 1003,1002,1001
        Topic: test_topic       Partition: 8    Leader: 1001    Replicas: 1001,1003,1002        Isr: 1001,1003,1002
        Topic: test_topic       Partition: 9    Leader: 1002    Replicas: 1002,1003,1001        Isr: 1002,1003,1001
        Topic: test_topic       Partition: 10   Leader: 1003    Replicas: 1003,1002,1001        Isr: 1003,1002,1001
        Topic: test_topic       Partition: 11   Leader: 1001    Replicas: 1001,1003,1002        Isr: 1001,1003,1002
        Topic: test_topic       Partition: 12   Leader: 1002    Replicas: 1002,1003,1001        Isr: 1002,1003,1001
        Topic: test_topic       Partition: 13   Leader: 1003    Replicas: 1003,1001,1002        Isr: 1003,1001,1002
        Topic: test_topic       Partition: 14   Leader: 1001    Replicas: 1001,1002,1003        Isr: 1001,1002,1003
        Topic: test_topic       Partition: 15   Leader: 1002    Replicas: 1002,1001,1003        Isr: 1002,1001,1003
        Topic: test_topic       Partition: 16   Leader: 1003    Replicas: 1003,1002,1001        Isr: 1003,1002,1001
        Topic: test_topic       Partition: 17   Leader: 1001    Replicas: 1001,1003,1002        Isr: 1001,1003,1002
        Topic: test_topic       Partition: 18   Leader: 1002    Replicas: 1002,1003,1001        Isr: 1002,1003,1001
        Topic: test_topic       Partition: 19   Leader: 1003    Replicas: 1003,1001,1002        Isr: 1003,1001,1002
```

Figure 1.5: Kafka describe

1.5 Verifying data is produced to newer partitions

```
#path-> /usr/odp/3.2.2.0-1/kafka/bin
./kafka-consumer-groups.sh \
> --bootstrap-server stg-xxx:6667 \
> --describe \
> --group test_consumer_group
```

```
root@stg-hdpsiddharth104:/usr/odp/3.2.2.0-1/kafka/bin[nb6][stg]# ./kafka-consumer-groups.sh --bootstrap-server stg-hdpsiddharth102:6667 \
> --describe \
> --group test_consumer_group

Consumer group 'test_consumer_group' has no active members.

GROUP               TOPIC           PARTITION  CURRENT-OFFSET  LOG-END-OFFSET  LAG             CONSUMER-ID     HOST            CLIENT-ID
test_consumer_group test_topic      15         0               0               0               -               -               -
test_consumer_group test_topic      14         0               0               0               -               -               -
test_consumer_group test_topic      13         0               0               0               -               -               -
test_consumer_group test_topic      12         0               0               0               -               -               -
test_consumer_group test_topic      19         0               0               0               -               -               -
test_consumer_group test_topic      18         1               1               0               -               -               -
test_consumer_group test_topic      17         1               1               0               -               -               -
test_consumer_group test_topic      16         0               0               0               -               -               -
test_consumer_group test_topic      7          0               0               0               -               -               -
test_consumer_group test_topic      6          0               0               0               -               -               -
test_consumer_group test_topic      5          1               1               0               -               -               -
test_consumer_group test_topic      4          2               2               0               -               -               -
test_consumer_group test_topic      11         0               0               0               -               -               -
test_consumer_group test_topic      10         0               0               0               -               -               -
test_consumer_group test_topic      9          4               4               0               -               -               -
test_consumer_group test_topic      8          0               0               0               -               -               -
test_consumer_group test_topic      3          1               1               0               -               -               -
test_consumer_group test_topic      2          1               1               0               -               -               -
test_consumer_group test_topic      1          1               1               0               -               -               -
test_consumer_group test_topic      0          2               2               0               -               -               -
```

Figure 1.6: Kafka consumer group describe

1.6 Stop Broker and Check Cluster Health

```
#path-> /usr/odp/3.2.2.0-1/kafka/bin
./kafka-topic.sh \
> --bootstrap-server stg-xxx:6667 \
> --describe \
> --topic test_topic

#zookeeper shell
ls /brokers/ids
```

```
root@stg-hdpsiddharth102:/usr/odp/3.2.2.0-1/kafka/bin[nb6][stg]# ./kafka-topics.sh --describe --bootstrap-server stg-hdpsiddharth102:6667 --topi
ic test_topic
Topic: test_topic       TopicId: 9pFEK8R2SD20cXH1hQ9dgg PartitionCount: 20      ReplicationFactor: 3    Configs: compression.type=producer,min.
insync.replicas=1,segment.bytes=1073741824,max.message.bytes=1000000,index.interval.bytes=4096,retention.bytes=-1,segment.index.bytes=10485760
        Topic: test_topic       Partition: 0    Leader: 1002    Replicas: 1002,1001,1003        Isr: 1002,1003
        Topic: test_topic       Partition: 1    Leader: 1003    Replicas: 1003,1002,1001        Isr: 1003,1002
        Topic: test_topic       Partition: 2    Leader: 1003    Replicas: 1001,1003,1002        Isr: 1003,1002
        Topic: test_topic       Partition: 3    Leader: 1002    Replicas: 1002,1003,1001        Isr: 1002,1003
        Topic: test_topic       Partition: 4    Leader: 1003    Replicas: 1003,1001,1002        Isr: 1003,1002
        Topic: test_topic       Partition: 5    Leader: 1002    Replicas: 1001,1002,1003        Isr: 1002,1003
        Topic: test_topic       Partition: 6    Leader: 1002    Replicas: 1002,1001,1003        Isr: 1002,1003
        Topic: test_topic       Partition: 7    Leader: 1003    Replicas: 1003,1002,1001        Isr: 1003,1002
        Topic: test_topic       Partition: 8    Leader: 1003    Replicas: 1001,1003,1002        Isr: 1003,1002
        Topic: test_topic       Partition: 9    Leader: 1002    Replicas: 1002,1003,1001        Isr: 1002,1003
        Topic: test_topic       Partition: 10   Leader: 1003    Replicas: 1003,1002,1001        Isr: 1003,1002
        Topic: test_topic       Partition: 11   Leader: 1003    Replicas: 1001,1003,1002        Isr: 1003,1002
        Topic: test_topic       Partition: 12   Leader: 1002    Replicas: 1002,1003,1001        Isr: 1002,1003
        Topic: test_topic       Partition: 13   Leader: 1003    Replicas: 1003,1001,1002        Isr: 1003,1002
        Topic: test_topic       Partition: 14   Leader: 1002    Replicas: 1001,1002,1003        Isr: 1002,1003
        Topic: test_topic       Partition: 15   Leader: 1002    Replicas: 1002,1001,1003        Isr: 1002,1003
        Topic: test_topic       Partition: 16   Leader: 1003    Replicas: 1003,1002,1001        Isr: 1003,1002
        Topic: test_topic       Partition: 17   Leader: 1003    Replicas: 1001,1003,1002        Isr: 1003,1002
        Topic: test_topic       Partition: 18   Leader: 1002    Replicas: 1002,1003,1001        Isr: 1002,1003
        Topic: test_topic       Partition: 19   Leader: 1003    Replicas: 1003,1001,1002        Isr: 1003,1002
```

Figure 1.7: Kafka describe after one broker down

```
WatchedEvent state:SyncConnected type:None path:null
[zk: localhost:2181(CONNECTED) 0] ls /brokers/ids
[1002, 1003]
```

Figure 1.8: Broker id's in zookeeper

1.7 Again, produce and consume data for the topic test_topic

```
# Produce data
#path-> /usr/odp/3.2.2.0-1/kafka/bin
./kafka-console-producer.sh \
> --bootstrap-server stg-XXX:6667 \
> --topic test_topic \
> --property  parse.key=true \
> --property key.separator=":"


# Consume data
#path-> /usr/odp/3.2.2.0-1/kafka/bin
./kafka-console-consumer.sh \
> --bootstrap-server stg-xxx:6667 \
> --topic test_topic \
> --from-beginning \
> --group test_consumer_group \
> --property print.key=true \
> --property print.partition=true \
> --property print.offset=true \
> --property print.timestamp=true
```

```
root@stg-hdpsiddharth102:/usr/odp/3.2.2.0-1/kafka/bin[nb6][stg]# ./kafka-console-producer.sh --bootstrap-server stg-hdpsiddharth102:6667 --
c test_topic --property parse.key=true --property key.separator=":"
>down1:xwcwc
>down2:cdcwc
>down3:wcwecw
>down6:cvwsv
>
```

Figure 1.9: Kafka producer

```
root@stg-hdpsiddharth104:/usr/odp/3.2.2.0-1/kafka/bin[nb6][stg]# ./kafka-console-consumer.sh --bootstrap-server stg-hdpsiddharth102:6667 --topic test_topic --from-beginning --group test
umer_group --property print.key=true --property print.partition=true --property print.offset=true --property print.timestamp=true
CreateTime:1759229901853        Partition:6     Offset:0        down1   xwcwc
CreateTime:1759229905967        Partition:8     Offset:0        down2   cdcwc
CreateTime:1759229909544        Partition:6     Offset:1        down3   wcwecw
CreateTime:1759229921079        Partition:16    Offset:0        down6   cvwsv
```

Figure 1.10: Kafka consumer

1.8 Describe the topic again and observe the changes

```
#path-> /usr/odp/3.2.2.0-1/kafka/bin
./kafka-topic.sh \
> --bootstrap-server stg-xxx:6667 \
> --describe \
> --topic test_
```

```
root@stg-hdpsiddharth102:/usr/odp/3.2.2.0-1/kafka/bin[nb6][stg]# ./kafka-topics.sh --describe --bootstrap-server stg-hdpsiddharth102:6667 -
ic test_topic
Topic: test_topic       TopicId: 9pFEK8R2SD20cXH1hQ9dgg PartitionCount: 20      ReplicationFactor: 3    Configs: compression.type=producer,
insync.replicas=1,segment.bytes=1073741824,max.message.bytes=1000000,index.interval.bytes=4096,retention.bytes=-1,segment.index.bytes=1048
        Topic: test_topic       Partition: 0    Leader: 1002    Replicas: 1002,1001,1003        Isr: 1002,1003
        Topic: test_topic       Partition: 1    Leader: 1003    Replicas: 1003,1002,1001        Isr: 1003,1002
        Topic: test_topic       Partition: 2    Leader: 1003    Replicas: 1001,1003,1002        Isr: 1003,1002
        Topic: test_topic       Partition: 3    Leader: 1002    Replicas: 1002,1003,1001        Isr: 1002,1003
        Topic: test_topic       Partition: 4    Leader: 1003    Replicas: 1003,1001,1002        Isr: 1003,1002
        Topic: test_topic       Partition: 5    Leader: 1002    Replicas: 1001,1002,1003        Isr: 1002,1003
        Topic: test_topic       Partition: 6    Leader: 1002    Replicas: 1002,1001,1003        Isr: 1002,1003
        Topic: test_topic       Partition: 7    Leader: 1003    Replicas: 1003,1002,1001        Isr: 1003,1002
        Topic: test_topic       Partition: 8    Leader: 1003    Replicas: 1001,1003,1002        Isr: 1003,1002
        Topic: test_topic       Partition: 9    Leader: 1002    Replicas: 1002,1003,1001        Isr: 1002,1003
        Topic: test_topic       Partition: 10   Leader: 1003    Replicas: 1003,1002,1001        Isr: 1003,1002
        Topic: test_topic       Partition: 11   Leader: 1003    Replicas: 1001,1003,1002        Isr: 1003,1002
        Topic: test_topic       Partition: 12   Leader: 1002    Replicas: 1002,1003,1001        Isr: 1002,1003
        Topic: test_topic       Partition: 13   Leader: 1003    Replicas: 1003,1001,1002        Isr: 1003,1002
        Topic: test_topic       Partition: 14   Leader: 1002    Replicas: 1001,1002,1003        Isr: 1002,1003
        Topic: test_topic       Partition: 15   Leader: 1002    Replicas: 1002,1001,1003        Isr: 1002,1003
        Topic: test_topic       Partition: 16   Leader: 1003    Replicas: 1003,1002,1001        Isr: 1003,1002
        Topic: test_topic       Partition: 17   Leader: 1003    Replicas: 1001,1003,1002        Isr: 1003,1002
        Topic: test_topic       Partition: 18   Leader: 1002    Replicas: 1002,1003,1001        Isr: 1002,1003
        Topic: test_topic       Partition: 19   Leader: 1003    Replicas: 1003,1001,1002        Isr: 1003,1002
root@stg-hdpsiddharth102:/usr/odp/3.2.2.0-1/kafka/bin[nb6][stg]#
```

Figure 1.11: Kafka describe

1.9 Under-Replicated Partitions

```
#path-> /usr/odp/3.2.2.0-1/kafka/bin
./kafka-topics.sh \
> --describe \
> --bootstrap-server stg-xxx:6667 \
> --under-replicated-partitions
```

Explanation

- Executes the `./kafka-topics.sh` script to `--describe` topics, connecting to the cluster via `--bootstrap-server stg-xxx:6667`. The command includes the `--under-replicated-partition` flag, which acts as a filter to display only the topics that currently have partitions with fewer active replicas than their configured replication factor. This is a common command for diagnosing the health of a Kafka cluster.

```
root@stg-hdpsiddharth102:/usr/odp/3.2.2.0-1/kafka/bin[nb6][stg]# ./kafka-topics.sh --describe --bootstrap-server stg-hdpsiddharth102:6667 -
ic test_topic --under-replicated-partitions
        Topic: test_topic       Partition: 0    Leader: 1002    Replicas: 1002,1001,1003        Isr: 1002,1003
        Topic: test_topic       Partition: 1    Leader: 1003    Replicas: 1003,1002,1001        Isr: 1003,1002
        Topic: test_topic       Partition: 2    Leader: 1003    Replicas: 1001,1003,1002        Isr: 1003,1002
        Topic: test_topic       Partition: 3    Leader: 1002    Replicas: 1002,1003,1001        Isr: 1002,1003
        Topic: test_topic       Partition: 4    Leader: 1003    Replicas: 1003,1001,1002        Isr: 1003,1002
        Topic: test_topic       Partition: 5    Leader: 1002    Replicas: 1001,1002,1003        Isr: 1002,1003
        Topic: test_topic       Partition: 6    Leader: 1002    Replicas: 1002,1001,1003        Isr: 1002,1003
        Topic: test_topic       Partition: 7    Leader: 1003    Replicas: 1003,1002,1001        Isr: 1003,1002
        Topic: test_topic       Partition: 8    Leader: 1003    Replicas: 1001,1003,1002        Isr: 1003,1002
        Topic: test_topic       Partition: 9    Leader: 1002    Replicas: 1002,1003,1001        Isr: 1002,1003
        Topic: test_topic       Partition: 10   Leader: 1003    Replicas: 1003,1002,1001        Isr: 1003,1002
        Topic: test_topic       Partition: 11   Leader: 1003    Replicas: 1001,1003,1002        Isr: 1003,1002
        Topic: test_topic       Partition: 12   Leader: 1002    Replicas: 1002,1003,1001        Isr: 1002,1003
        Topic: test_topic       Partition: 13   Leader: 1003    Replicas: 1003,1001,1002        Isr: 1003,1002
        Topic: test_topic       Partition: 14   Leader: 1002    Replicas: 1001,1002,1003        Isr: 1002,1003
        Topic: test_topic       Partition: 15   Leader: 1002    Replicas: 1002,1001,1003        Isr: 1002,1003
        Topic: test_topic       Partition: 16   Leader: 1003    Replicas: 1003,1002,1001        Isr: 1003,1002
        Topic: test_topic       Partition: 17   Leader: 1003    Replicas: 1001,1003,1002        Isr: 1003,1002
        Topic: test_topic       Partition: 18   Leader: 1002    Replicas: 1002,1003,1001        Isr: 1002,1003
        Topic: test_topic       Partition: 19   Leader: 1003    Replicas: 1003,1001,1002        Isr: 1003,1002
```

Figure 1.12: Kafka describe under-replica

1.10 Again, start the Kaka broker and observe the health

```
#path-> /usr/odp/3.2.2.0-1/kafka/bin
./kafka-topic.sh \
> --bootstrap-server stg-xxx:6667 \
> --describe \
> --topic test_topic

#zookeeper shell
ls /brokers/ids
```

```
WatchedEvent state:SyncConnected type:None path:null
[zk: localhost:2181(CONNECTED) 0] ls /brokers/ids
[1001, 1002, 1003]
```

Figure 1.13: Broker id's in zookeeper

```
root@stg-hdpsiddharth102:/usr/odp/3.2.2.0-1/kafka/bin[nb6][stg]# ./kafka-topics.sh --describe --bootstrap-server stg-hdpsiddharth102:6667 --topic test_topic
Topic: test_topic       TopicId: 9pFEK8R2SD20cXH1hQ9dgg PartitionCount: 20      ReplicationFactor: 3    Configs: compression.type=producer,min.insync.replicas=1,segment.bytes=1073741824,max.message.bytes=1000000,index.interval.bytes=4096,retention.bytes=-1,segment.index.bytes=10485760
        Topic: test_topic       Partition: 0    Leader: 1002    Replicas: 1002,1001,1003        Isr: 1002,1003,1001
        Topic: test_topic       Partition: 1    Leader: 1003    Replicas: 1003,1002,1001        Isr: 1003,1002,1001
        Topic: test_topic       Partition: 2    Leader: 1001    Replicas: 1001,1003,1002        Isr: 1003,1002,1001
        Topic: test_topic       Partition: 3    Leader: 1002    Replicas: 1002,1003,1001        Isr: 1002,1003,1001
        Topic: test_topic       Partition: 4    Leader: 1003    Replicas: 1003,1001,1002        Isr: 1003,1002,1001
        Topic: test_topic       Partition: 5    Leader: 1001    Replicas: 1001,1002,1003        Isr: 1002,1003,1001
        Topic: test_topic       Partition: 6    Leader: 1002    Replicas: 1002,1001,1003        Isr: 1002,1003,1001
        Topic: test_topic       Partition: 7    Leader: 1003    Replicas: 1003,1002,1001        Isr: 1003,1002,1001
        Topic: test_topic       Partition: 8    Leader: 1001    Replicas: 1001,1003,1002        Isr: 1003,1002,1001
        Topic: test_topic       Partition: 9    Leader: 1002    Replicas: 1002,1003,1001        Isr: 1002,1003,1001
        Topic: test_topic       Partition: 10   Leader: 1003    Replicas: 1003,1002,1001        Isr: 1003,1002,1001
        Topic: test_topic       Partition: 11   Leader: 1001    Replicas: 1001,1003,1002        Isr: 1003,1002,1001
        Topic: test_topic       Partition: 12   Leader: 1002    Replicas: 1002,1003,1001        Isr: 1002,1003,1001
        Topic: test_topic       Partition: 13   Leader: 1003    Replicas: 1003,1001,1002        Isr: 1003,1002,1001
        Topic: test_topic       Partition: 14   Leader: 1001    Replicas: 1001,1002,1003        Isr: 1002,1003,1001
        Topic: test_topic       Partition: 15   Leader: 1002    Replicas: 1002,1001,1003        Isr: 1002,1003,1001
        Topic: test_topic       Partition: 16   Leader: 1003    Replicas: 1003,1002,1001        Isr: 1003,1002,1001
        Topic: test_topic       Partition: 17   Leader: 1001    Replicas: 1001,1003,1002        Isr: 1003,1002,1001
        Topic: test_topic       Partition: 18   Leader: 1002    Replicas: 1002,1003,1001        Isr: 1002,1003,1001
        Topic: test_topic       Partition: 19   Leader: 1003    Replicas: 1003,1001,1002        Isr: 1003,1002,1001
root@stg-hdpsiddharth102:/usr/odp/3.2.2.0-1/kafka/bin[nb6][stg]# ./kafka-topics.sh --describe --bootstrap-server stg-hdpsiddharth102:6667 --topic test_topic --under-replicated-partitions
```

Figure 1.14: Kafka describe

1.11 Experimenting with test_topic_1 with a replication factor of 1

```
#path-> /usr/odp/3.2.2.0-1/kafka/bin
./kafka-topics.sh \
> --create \
> --bootstrap-server stg-hdpsiddharth102:6667 \
> --replication-factor 1 \
> --partitions 3 \
--topic test_topic_1

./kafka-topics.sh \
> --describe \
> --bootstrap-server stg-hdpsiddharth102:6667 \
> --topic test_topic_1

./kafka-console-producer.sh \
>--bootstrap-server stg-hdpsiddharth102:6667 \
>--topic test_topia_1

kafka stop

./kafka-console-consumer.sh \
> --bootstrap-server stg-hdpsiddharth102:6667 \
> --topic test_topic_1 \
> --from-beginning \
> --group test_consumer_group_1 \
> --property print.key=true \
> --property print.partition=true \
> --property print.offset=true \
> --property print.timestamp=true
```

```
root@stg-hdpsiddharth102:/usr/odp/3.2.2.0-1/kafka/bin[nb6][stg]# ./kafka-topics.sh \
> --create \
> --bootstrap-server stg-hdpsiddharth102:6667 \
> --replication-factor 1 \
> --partitions 3 \
>  --topic test_topic_1
WARNING: Due to limitations in metric names, topics with a period ('.') or underscore ('_') could collide. To avoid issues it is best to use ei
ther, but not both.
Created topic test_topic_1.
root@stg-hdpsiddharth102:/usr/odp/3.2.2.0-1/kafka/bin[nb6][stg]# ./kafka-topics.sh \
> --describe \
> --bootstrap-server stg-hdpsiddharth102:6667 \
> --topic test_topic_1
Topic: test_topic_1     TopicId: RReLBWweQBardmlK2X6Bgw PartitionCount: 3       ReplicationFactor: 1    Configs: compression.type=producer,min.
insync.replicas=1,segment.bytes=1073741824,max.message.bytes=1000000,index.interval.bytes=4096,retention.bytes=-1,segment.index.bytes=10485760
        Topic: test_topic_1     Partition: 0    Leader: 1002    Replicas: 1002  Isr: 1002
        Topic: test_topic_1     Partition: 1    Leader: 1001    Replicas: 1001  Isr: 1001
        Topic: test_topic_1     Partition: 2    Leader: 1003    Replicas: 1003  Isr: 1003
root@stg-hdpsiddharth102:/usr/odp/3.2.2.0-1/kafka/bin[nb6][stg]# ./kafka-console-producer.sh \
> --bootstrap-server stg-hdpsiddharth102:6667 \
> --topic test_topic_1
>hello
>siddharth on thw way!!
>hurray!!
>
```

Figure 1.15: New topic creation and producing data

```
root@stg-hdpsiddharth104:/usr/odp/3.2.2.0-1/kafka/bin[nb6][stg]# ./kafka-console-consumer.sh \
> --bootstrap-server stg-hdpsiddharth102:6667 \
> --topic test_topic_1 \
> --from-beginning
hello
siddharth on thw way!!
hurray!!
```

Figure 1.16: Consuming data from new topic

```
root@stg-hdpsiddharth102:/usr/odp/3.2.2.0-1/kafka/bin[nb6][stg]# ./kafka-topics.sh --describe --bootstrap-server stg-hdpsiddharth102:6667 --top
ic test_topic_1
Topic: test_topic_1     TopicId: RReLBWweQBardmlK2X6Bgw PartitionCount: 3       ReplicationFactor: 1    Configs: compression.type=producer,min.
insync.replicas=1,segment.bytes=1073741824,max.message.bytes=1000000,index.interval.bytes=4096,retention.bytes=-1,segment.index.bytes=10485760
        Topic: test_topic_1     Partition: 0    Leader: 1002    Replicas: 1002  Isr: 1002
        Topic: test_topic_1     Partition: 1    Leader: none    Replicas: 1001  Isr: 1001
        Topic: test_topic_1     Partition: 2    Leader: 1003    Replicas: 1003  Isr: 1003
```

Figure 1.17: Kafka describe with one broker down

```
root@stg-hdpsiddharth104:/usr/odp/3.2.2.0-1/kafka/bin[nb6][stg]# ./kafka-console-consumer.sh --bootstrap-server stg-hdpsiddharth102:6667 --topic test_topic_1 --from-beginning --group test_co
nsumer_group_1 --property print.key=true --property print.partition=true --property print.offset=true --property print.timestamp=true
CreateTime:1759252592104        Partition:2     Offset:0        null    hello
CreateTime:1759252619539        Partition:2     Offset:1        null    hurray!!
CreateTime:1759253450984        Partition:2     Offset:2        null    bjbkn
CreateTime:1759253460199        Partition:2     Offset:3        null    awaesdf
CreateTime:1759253470746        Partition:2     Offset:4        null    bkkjjjjjjj
CreateTime:1759253490049        Partition:2     Offset:5        null    jubbik
CreateTime:1759253447129        Partition:0     Offset:0        null    huhi
CreateTime:1759253455694        Partition:0     Offset:1        null    nonono
CreateTime:1759253465884        Partition:0     Offset:2        null    vujjb kn
CreateTime:1759253475047        Partition:0     Offset:3        null    bnknkn
CreateTime:1759253869167        Partition:0     Offset:4        user1   sid
CreateTime:1759253878254        Partition:2     Offset:6        user2   qwertty
CreateTime:1759253890438        Partition:2     Offset:7        user3   bioo
```

Figure 1.18: Kafka consumer with one broker down

Chapter 2

Task 2: Kafka Health Alert Setup

2.1 Bash Script for Alerts

```
#!/bin/bash
BROKERS_EXPECTED=3                                  # Total brokers in your cluster
KAFKA_BIN_DIR=/usr/odp/3.2.2.0-1/kafka/bin     # Kafka binaries path
ZOOKEEPER_HOST=stg-hdpsiddharth102:2181        # Zookeeper host
TOPICS=("test_topic" "test_topic_1")           # List of topics to check


check_brokers() {
BROKERS_ACTUAL=$($KAFKA_BIN_DIR/../../zookeeper/bin/zkCli.sh -server $ZOOKEEPER_HOST ls /brokers/ids 2>/dev/null | tail -1 | tr -d '[],')
NUM_BROKERS=$(echo $BROKERS_ACTUAL | wc -w)
if [ -z "$BROKERS_ACTUAL" ]; then
    echo "ALERT: Cannot connect to Zookeeper or Kafka brokers!" | tee -a /tmp/kafka_health.log
elif [ "$NUM_BROKERS" -lt "$BROKERS_EXPECTED" ]; then
    echo "ALERT: Some Kafka brokers are down! Expected $BROKERS_EXPECTED, Found $NUM_BROKERS" | tee -a /tmp/kafka_health.log
else
    echo "Kafka brokers OK: $BROKERS_ACTUAL" | tee -a /tmp/kafka_health.log
fi
}

check_urp() {
for topic in "${TOPICS[@]}"; do
URP_COUNT=$($KAFKA_BIN_DIR/kafka-topics.sh --bootstrap-server stg-hdpsiddharth102:6667 --describe --under-replicated-partitions --topic $topic 2>/dev/null | wc -l)
if [ "$URP_COUNT" -gt 0 ]; then
```

```
24 echo "ALERT: $URP_COUNT under-replicated partitions found for
      topic $topic" | tee -a /tmp/kafka_health.log
25 else
26 echo "No under-replicated partitions for topic $topic" | tee -a /
      tmp/kafka_health.log
27 fi
28 done
29 }
30
31 check_brokers
32 check_urp
```

Explanation

Variable Definitions (Lines 2-5):
- **Lines 2-5:** These lines set up configuration variables for the script: the `BROKERS_EXPECTED` count (3), the path to Kafka's binaries (`KAFKA_BIN_DIR`), the Zookeeper connection string (`ZOOKEEPER_HOST`), and an array of `TOPICS` to be monitored.

'check_brokers' Function (Lines 8-18):
- **Line 8:** Defines a function to verify the number of active Kafka brokers.
- **Line 9:** Uses the Zookeeper client (`zkCli.sh`) to query Zookeeper and list the IDs of all currently registered brokers from the `/brokers/ids` znode.
- **Line 10:** Counts the number of active brokers found in the previous step.
- **Lines 11-17:** This block compares the actual number of brokers with the expected number. It prints and logs an `ALERT` to `/tmp/kafka_health.log` if the connection fails or if the broker count is too low; otherwise, it logs an `OK` status.

'check_urp' Function (Lines 20-29):
- **Line 20:** Defines a function to check for Under-Replicated Partitions (URP), which is a key sign of an unhealthy cluster.
- **Line 21:** Starts a loop that iterates through each topic listed in the `TOPICS` array.
- **Line 22:** For each topic, it runs `kafka-topics.sh` with the `--under-replicated-partitions` flag and counts the lines in the output. A count greater than zero indicates that the topic has URPs.
- **Lines 23-27:** If the URP count for a topic is greater than zero, it prints and logs an `ALERT`; otherwise, it logs a healthy status for that topic.

Execution (Lines 31-32):
- **Lines 31-32:** These lines call the two previously defined functions, `check_brokers` and `check_urp`, to execute the health checks.

2.2 Cron Job Setup

```
1 */1 * * * * /usr/local/bin/kafka_health_check.sh >> /var/log/
      kafka_health.log 2>&1
```

```
[root@stg-hdpsiddharth102:/usr/local/bin[nb6][stg]# tail -f /var/log/kafka_health.log
No under-replicated partitions for topic test_topic_1
ALERT: Some Kafka brokers are down! Expected 3, Found 2
ALERT: 20 under-replicated partitions found for topic test_topic
No under-replicated partitions for topic test_topic_1
ALERT: Some Kafka brokers are down! Expected 3, Found 2
ALERT: 20 under-replicated partitions found for topic test_topic
No under-replicated partitions for topic test_topic_1
ALERT: Some Kafka brokers are down! Expected 3, Found 2
ALERT: 20 under-replicated partitions found for topic test_topic
No under-replicated partitions for topic test_topic_1
Kafka brokers OK: 1001 1002 1003
No under-replicated partitions for topic test_topic
No under-replicated partitions for topic test_topic_1
```

Figure 2.1: Alert logs