Command line interface (CLI)

Note - Instead of --zookeeper 127.0.0.1:2181 use --bootstrap-server localhost:9092

* Kafka topics CLI

1. **Create**

kafka-topics.sh --zookeeper 127.0.0.1:2181 --topic <topic-name> --create --partitions <value> --replication-factor <value>

eg kafka-topics.sh --zookeeper 127.0.0.1:2181 -topic first_topic --create --partitions 3 --replication-factor 1

- You can only create as many replicas as the number of brokers
- zookeeper works/binds on port 2181 always.

2. **List** : Lists names of all kafka topics.

kafka-topics.sh --zookeeper 127.0.0.1:2181 --list

3. **Describe** : List details for given topic

kafka-topics.sh --zookeeper 127.0.0.1:2181 --topic first_topic --describ

4. **Delete** : Deletes a topic

kafka-topics.sh --zookeeper 127.0.0.1:2181 --topic first_topic --delete

-marks a topic for deletion.

* Kafka console producer CLI

- To send data to kafka, a producer is required.
- role of producer is to send or write data/messages to kafka topics.

Create

kafka-console-producer --broker-list 127.0.0.1:9092 --topic topic1

> hello world

- You will get a '>' after pressing enter.
- 9092 is the port of kafka
- Press ctrl+c to stop writing messages.

What if a topic does not exist?

- A warning will appear (Leader-not-available).
- kafka will create that topic for you and next time you write there will be no warning/errors because the topic comes in existing list now.
- It is always better to have the topic created first and then write to it as while writing the topics are

created with default properties.
- You can change default properties in server.properties eg. no. of replicas by default etc.

```
kafka-console-producer --broker-list 127.0.0.1:9092 --topic topic3
--producer-property acks=all
```

★ Kafka console consumer CLI

Imp: Apache kafka consumer will consume only those messages which are produced when the consumer was in active state.

- The order of message may change when there are multiple partition. Within a partition the messages will be read in order.

```
kafka-console-consumer --bootstrap-server 127.0.0.1:9092 --topic
first_topic
```

To read from beginning.

```
kafka-console-consumer --bootstrap-server localhost:9092 --topic
topic1 --from-beginning.
```

Note: The order of messages in consumer is not 'total', it is per partition. Order is only guaranteed within a partition. If you try a topic with 1 partition you will see total ordering.

★ Kafka consumers in groups.

```
kafka-console-consumer.sh --bootstrap-server localhost:9092
--topic mytopic --group my-first-group
```

In 3 terminals, run same ↑ command to get 3 consumers in the group. You will see that load is being distributed amongst them. If one of them goes down others will handle the load automatically.

Now, if you want to read from beginning use --from-beginning. Note if you stop the consumer and re-run the command it will not run from beginning now. Because group has been specified and offset has been stored so it will only read newer messages.

```
kafka-console-consumer.sh --bootstrap-server localhost:9092
--topic topic2 --group my-second-group --from-beginning
```

* Kafka consumer groups CLI

list all consumer groups, describe a consumer group, delete consumer group info, or reset consumer group offsets.

LIST - lists all consumer groups

```
kafka-consumer-groups --bootstrap-server localhost:9092 --list
```

DESCRIBE

```
kafka-consumer-groups --bootstrap-server localhost:9092
--describe --group my-second-application
```

RESET-OFFSETS

Resets offset of consumer group. Supports one consumer at a time, the instance should be inactive.

reset specifications:- --to-datetime, --by-period, to-earliest, to-latest, -shift-by, from-file, to-current.

to-earliest → at beginning again

```
kafka-consumer-groups.sh --bootstrap-server localhost:9092
--group my-first-group --reset-offsets --to-earliest --execute
--topic first_topic
```