**STARTING ZOOKEEPER**

zookeeper-server-start.sh config/zookeeper.properties

**STARTING KAFKA SERVER**

kafka-server-start.sh config/server.properties

**CREATE**

kafka-topics.sh --bootstrap-server localhost:9092 -topic topic1 --create --partitions 2 --replication-factor 1

Created topic topic1.

**LIST**

kafka-topics.sh --bootstrap-server localhost:9092 --list

quickstart-events

topic1

**DELETE**

kafka-topics.sh --bootstrap-server localhost:9092 -topic quickstart-events --delete

kafka-topics.sh --bootstrap-server localhost:9092 --list

topic1

**DESCRIBE**

kafka-topics.sh --bootstrap-server localhost:9092 -topic topic1 --describe

Topic: topic1 TopicId: zx4N83EaRIOmTxpRTH9r9Q PartitionCount: 2 ReplicationFactor: 1 Configs:

Topic: topic1 Partition: 0 Leader: 0 Replicas: 0 Isr: 0

Topic: topic1 Partition: 1 Leader: 0 Replicas: 0 Isr: 0

**PRODUCER**

kafka-console-producer.sh --bootstrap-server localhost:9092 --topic topic1

kafka-console-producer.sh --bootstrap-server localhost:9092 --topic topic1 --producer-property acks=all

**PRODUCER WITH TOPIC THAT DOES NOT EXISTS GIVES A WARNING**

kafka-console-producer.sh --bootstrap-server localhost:9092 --topic topic2

>helloo

[2023-01-28 14:07:09,171] WARN [Producer clientId=console-producer] Error while fetching metadata with correlation id 4 : {topic2=LEADER\_NOT\_AVAILABLE} (org.apache.kafka.clients.NetworkClient)

**CONSUMER**

kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic topic1

ho

hii

(only message sent when the consumer became active are shown. For all messages from start use from-beginining)

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

hello

this is harry

how r u today

hello world

ho

hii

**CONSUMER GROUPS** – create 2 consumers use same group, you will see that load is being distributed amongst the consumers

kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic topic1 --group my-first-group

kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic topic2 --group my-first-group

Now, if you want to read from beginning use --from-beginning. Note that ifyou stop the consumer and rerun the same command it will not read from beginning now. Because the group has been specified and offsets have been commited in kafka so only new messages will be read now

kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic topic1 --group my-second-group --from-beginning

**CONSUMER GROUPS CLI**

**LIST ALL GROUPS**

kafka-consumer-groups.sh --bootstrap-server localhost:9092 --list

my-first-group

my-second-group

**DESCRIBE**

harshali15@Harshalis-Air kafka\_2.13-3.3.1 % kafka-consumer-groups.sh --bootstrap-server localhost:9092 --describe --group my-second-group

Consumer group 'my-second-group' has no active members.

GROUP TOPIC PARTITION CURRENT-OFFSET LOG-END-OFFSET LAG CONSUMER-ID HOST CLIENT-ID

my-second-group topic1 0 61 61 0 - - -

my-second-group topic1 1 0 0 0 - - -

harshali15@Harshalis-Air kafka\_2.13-3.3.1 % kafka-consumer-groups.sh --bootstrap-server localhost:9092 --describe --group my-first-group

Consumer group 'my-first-group' has no active members.

GROUP TOPIC PARTITION CURRENT-OFFSET LOG-END-OFFSET LAG CONSUMER-ID HOST CLIENT-ID

my-first-group topic2 0 0 0 0 - - -

my-first-group topic1 1 0 0 0 - - -

my-first-group topic2 2 0 0 0 - - -

my-first-group topic2 1 20 20 0 - - -

my-first-group topic1 0 60 61 1 - - -

**RESET OFFSETS**

kafka-consumer-groups.sh --bootstrap-server localhost:9092 --group my-first-group --reset-offsets --to-earliest --execute --topic topic1

GROUP TOPIC PARTITION NEW-OFFSET

my-first-group topic1 0 0

my-first-group topic1 1 0

bytes.usc.edu/home/csci585

bytes.usc.edu/csci585/home