Kafka1

Michael Edelman

May 2022

1 Phase 1

1.1 Using the kafka-topics.sh command, show that the kafka1_assignment topic doesn't exist. Record this interaction in a screenshot

Below is a screenshot of the kafka-topics command displaying no topics.

```
mikeede11@LAPTOP-F8JUS0BM:/mnt/c/Kafka2/kafka_2.13-3.1.0$ bin/kafka-topics.sh --bootstrap-server localhost:9092 --list

mikeede11@LAPTOP-F8JUS0BM:/mnt/c/Kafka2/kafka_2.13-3.1.0$
```

1.2 Start the message emitter and redirect to a file. Screenshot the first 5 lines of output.

Below is a screenshot of the output of the message emitter.

```
mikeede11@LAPTOP-F8JUS0BM:/mnt/c/Kafka2$ cat dataForProducer.txt
Will emit alerts every 10 seconds
Press [CTRL+C] to stop...
Current date and time Wed May 25 02:54:57 EDT 2022
emmiting alert 0 at 02:54:57
emmiting alert 1 at 02:55:07
emmiting alert 2 at 02:55:17
emmiting alert 3 at 02:55:27
emmiting alert 4 at 02:55:38
emmiting alert 5 at 02:55:48
mikeede11@LAPTOP-F8JUS0BM:/mnt/c/Kafka2$
```

1.3 Start the producer and screenshot the window (and command) that invokes the producer.

Below is a screenshot of the command used to start and get the producer to read the messages as they manifest in the file.(note: their are some WARN messages, but the producer did end up working fine)

```
mikeede11@LAPTOP-F8JUS0BM:/mnt/c/Kafka2/kafka_2.13-3.1.0$ tail -f dataForProducer.txt | bin/kafka-console-producer.sh --broker-list localhost:9092 --topic kafka1_assign
ment
[2022-05-25 03:21:15,147] WARN [Producer clientId=console-producer] Bootstrap broker localhost:9092 (id: -1 rack: null) disconnected (org.apache.kafka.clients.NetworkCl
ient)
[2022-05-25 03:21:16,159] WARN [Producer clientId=console-producer] Error while fetching metadata with correlation id 2 : {kafka1_assignment=LEADER_NOT_AVAILABLE} (org.
apache.kafka.clients.NetworkClient)
[2022-05-25 03:21:16,334] WARN [Producer clientId=console-producer] Error while fetching metadata with correlation id 4 : {kafka1_assignment=LEADER_NOT_AVAILABLE} (org.
apache.kafka.clients.NetworkClient)
```

1.4 Screenshot an interaction that shows that the topic does exist.

Below is a screenshot that shows that our Producer successfully created a .

```
mikeede11@LAPTOP-F8JUS0BM:/mnt/c/Kafka2/kafka_2.13-3.1.0$ bin/kafka-topics.sh --bootstrap-server localhost:9092 --list
kafka1_assignment
mikeede11@LAPTOP-F8JUS0BM:/mnt/c/Kafka2/kafka_2.13-3.1.0$
```

1.5 Screenshot an interaction that shows that three broker processes are running.

Below is a screenshot using the kafka-topics.sh command to display the Brokers that are active.

```
mikeede11@LAPTOP-F8JUS0BM:/mnt/c/Kafka2/kafka_2.13-3.1.0$ bin/kafka-broker-api-versions.sh --bootstrap-server localhost:9092 | awk '/id/{print $1}'
localhost:9093
localhost:9094
localhost:9092
mikeede11@LAPTOP-F8JUS0BM:/mnt/c/Kafka2/kafka_2.13-3.1.0$
```

1.6 . In a separate window start a consumer, and have it read messages from this topic, starting from the beginning. Screenshot the first 5 lines and last 5 lines of the output.

Below are screenshots of the first few and last few lines of output that the Consumer received.

```
mikeede11@LAPTOP-F8JUS0BM:/mnt/c/Kafka2/kafka_2.13-3.1.0$ bin/kafka-console-consumer.sh --bootstrap-server localhost:9093 --topic kafka1_assignment --from-beginning
emmiting alert 24 at 03:19:31
emmiting alert 25 at 03:19:41
emmiting alert 26 at 03:19:51
emmiting alert 27 at 03:20:01
emmiting alert 28 at 03:20:11
emmiting alert 29 at 03:20:21
```

```
emmiting alert 95 at 03:31:22
emmiting alert 96 at 03:31:32
emmiting alert 97 at 03:31:42
emmiting alert 98 at 03:31:52
emmiting alert 99 at 03:32:02
```

1.7 The consumer should read messages from the topic (from the beginning) and copy these messages to a "sink" file. Screenshot the command(s) that get this going.

Belwo is a screenshot of the Consumer Command.

```
mikeede11@LAPTOP-F8JUS0BM:/mnt/c/Kafka2/kafka_2.13-3.1.0$ bin/kafka-console-consumer.sh --bootstrap-server localhost:9093 --topic kafka1_assignment --from-beginning > c
onsumer_messages.txt
```

1.8 Show the first 5 and last 5 lines of the "sink" file.

Below is the results of the Consumer successfuly receiving messages from the kafka1_assignment topic and storing them in a file. (Note: you may notice that the output did not change from the inital one - this is because I did not put my emitter in an infinite loop but terminated it after 100 iterations. By the time I executed these commands the loop was terminated. You can check my bash file to confirm this. Otherwise the logic is sound and the Consumer was still able

to pull messages off the brokers.)
The beginning:

consumer_messages.txt - Notepad

File Edit Format View Help

```
emmiting alert 24 at 03:19:31
emmiting alert 25 at 03:19:41
emmiting alert 26 at 03:19:51
emmiting alert 27 at 03:20:01
emmiting alert 28 at 03:20:11
emmiting alert 29 at 03:20:21
emmiting alert 30 at 03:20:31
```

The end:

```
emmiting alert 93 at 03:31:02
emmiting alert 94 at 03:31:12
emmiting alert 95 at 03:31:22
emmiting alert 96 at 03:31:32
emmiting alert 97 at 03:31:42
emmiting alert 98 at 03:31:52
emmiting alert 99 at 03:32:02
```

2 Phase 2

2.1 Screenshot/include the Avro schema that you used to implement this phase's requirements.

```
{
  "namespace": "Rule",
  "type": "record",
  "name": "Data",
  "fields": [
    {
      "name": "sensor_id",
      "type": "long"
    },
    {
      "name": "time",
      "type": "long"
    },
    {
      "name": "status",
      "type":  ["null", {
                                "type": "enum",
                                "name": "Alert",
                                "symbols": ["CRITICAL", "MINOR"]
                            }
                            ]
    }
  ]
}
```

2.2 Start the producer which should emit twenty messages. Screenshot the first three and last three messages.

2.3 Start the consumer and have it read the messages per the format shown above. Screenshot some "mix" of the Critical and Minor messages.