Name: Royyuru Sai Prasanna Gangadhar **Roll Number**: ME19B190

Problem statement:

Stream the data stored on the GCS bucket into Kafka by breaking the data into batches of 10 records that are written to Kafka separated by a sleep time of 10 seconds until 100 records are written. Use Spark Streaming to read from Kafka every 5 seconds and emit the count of rows seen in the last 10 seconds.

Step 1: Install the necessary dependencies, using the following commands and download kafka-2.13-3.40 ".tgz" folder and extract it.

```
me19b190@me19b190-ass8:~$ sudo apt update && sudo apt install -y tmux vim openjdk-8-jdk wget curl git python3-pip
Hit:1 http://us-central1.gce.archive.ubuntu.com/ubuntu focal InRelease
Hit:2 http://us-central1.gce.archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:3 http://us-central1.gce.archive.ubuntu.com/ubuntu focal-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu focal-security InRelease
Reading package lists... Done
Building dependency tree
Reading state information... Done
16 packages can be upgraded. Run 'apt list --upgradable' to see them.
Reading package lists... Done
Building dependency tree
Reading state information... Done
curl is already the newest version (7.68.0-1ubuntu2.18).
git is already the newest version (1:2.25.1-1ubuntu3.10).
tmux is already the newest version (3.0a-2ubuntu0.4).
vim is already the newest version (2:8.1.2269-1ubuntu5.12).
wget is already the newest version (1.20.3-1ubuntu2).
openjdk-8-jdk is already the newest version (8u362-ga-0ubuntu1~20.04.1).
python3-pip is already the newest version (20.0.2-5ubuntu1.8).
The following packages were automatically installed and are no longer required:
  libatasmart4 libblockdev-fs2 libblockdev-loop2 libblockdev-part-err2 libblockdev-part2 libblockdev-swap2 libblockdev-utils2 libblockdev2 libmbim-glib4
  libmbim-proxy libmm-glib0 libnuma1 libparted-fs-resize0 libqmi-glib5 libqmi-proxy libudisks2-0 libxmlb2 usb-modeswitch usb-modeswitch-data
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 16 not upgraded.
me19b190@me19b190-ass8:~$
```

```
me19b190@me19b190-ass8:~$ pip install kafka-python pyspark pyspark[sql]
Requirement already satisfied: kafka-python in ./.local/lib/python3.8/site-packages (2.0.2)
Requirement already satisfied: pyspark in ./.local/lib/python3.8/site-packages (3.3.2)
Requirement already satisfied: py4j==0.10.9.5 in ./.local/lib/python3.8/site-packages (from pyspark) (0.10.9.5)
```

```
me19b190@me19b190-ass8:~$ wget https://dlcdn.apache.org/kafka/3.4.0/kafka_2.13-3.4.0.tgz # Get Kafka Sources
--2023-03-31 22:39:16--  https://dlcdn.apache.org/kafka/3.4.0/kafka_2.13-3.4.0.tgz
Resolving dlcdn.apache.org (dlcdn.apache.org)... 151.101.2.132, 2a04:4e42::644
Connecting to dlcdn.apache.org (dlcdn.apache.org)|151.101.2.132|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 106290956 (101M) [application/x-gzip]
Saving to: 'kafka_2.13-3.4.0.tgz.1'

kafka_2.13-3.4.0.tgz.1              100%[==========================================================================>] 101.37M   183MB/s    in 0.6s

2023-03-31 22:39:16 (183 MB/s) - 'kafka_2.13-3.4.0.tgz.1' saved [106290956/106290956]
```

Step 2: Open a new terminal session using "tmux" and split the window pane into two and run the following commands to start the zookeeper and kafka.

```
eerConfig)
[2023-03-31 22:12:25,250] INFO Starting server (org.apache.zookeeper.server.ZooKeeperServerMain)
[2023-03-31 22:12:25,275] INFO ServerMetrics initialized with provider org.apache.zookeeper.metrics.impl.DefaultMetricsProvider@3551a94 (org.apache.zookeeper.se
rver.ServerMetrics)
[2023-03-31 22:12:25,286] INFO zookeeper.snapshot.trust.empty : false (org.apache.zookeeper.server.persistence.FileTxnSnapLog)
[2023-03-31 22:12:25,308] INFO  (org.apache.zookeeper.server.ZooKeeperServer)
[2023-03-31 22:12:25,309] INFO   ______                  _                                           (org.apache.zookeeper.server.ZooKeeperServer)
[2023-03-31 22:12:25,309] INFO  |___  /                 | |                                          (org.apache.zookeeper.server.ZooKeeperServer)
[2023-03-31 22:12:25,309] INFO     / /    ___     ___   | | __   ___    ___   _ __     ___   _ __    (org.apache.zookeeper.server.ZooKeeperServer)
[2023-03-31 22:12:25,309] INFO    / /    / _ \   / _ \  | |/ /  / _ \  / _ \ | '_ \   / _ \ | '__| (org.apache.zookeeper.server.ZooKeeperServer)
[2023-03-31 22:12:25,309] INFO   / /__  | (_) | | (_) | |   <  |  __/ |  __/ | |_) | |  __/ | |     (org.apache.zookeeper.server.ZooKeeperServer)
[2023-03-31 22:12:25,309] INFO  /_____|  \___/   \___/  |_|\_\  \___|  \___| | .__/   \___| |_| (org.apache.zookeeper.server.ZooKeeperServer)
```

```
[2023-03-31 22:12:25,493] INFO zookeeper audit is disabled. (org.apache.zookeeper.audit.ZKAuditProvider)
[2023-03-31 22:14:46,801] INFO Creating new log file: log.1 (org.apache.zookeeper.server.persistence.FileTxnLog)

[2023-03-31 22:14:48,842] INFO [GroupCoordinator 0]: Startup complete. (kafka.coordinator.group.GroupCoordinator)
[2023-03-31 22:14:48,877] INFO [MetadataCache brokerId=0] Updated cache from existing <empty> to latest FinalizedFeaturesAndEpoch(features=Map(), epoch=0). (kaf
ka.server.metadata.ZkMetadataCache)
[2023-03-31 22:14:48,898] INFO [TransactionCoordinator id=0] Starting up. (kafka.coordinator.transaction.TransactionCoordinator)
[2023-03-31 22:14:48,939] INFO [TransactionCoordinator id=0] Startup complete. (kafka.coordinator.transaction.TransactionCoordinator)
[2023-03-31 22:14:48,949] INFO [Transaction Marker Channel Manager 0]: Starting (kafka.coordinator.transaction.TransactionMarkerChannelManager)
[2023-03-31 22:14:49,069] INFO [ExpirationReaper-0-AlterAcls]: Starting (kafka.server.DelayedOperationPurgatory$ExpiredOperationReaper)
[2023-03-31 22:14:49,171] INFO [/config/changes-event-process-thread]: Starting (kafka.common.ZkNodeChangeNotificationListener$ChangeEventProcessThread)
[2023-03-31 22:14:49,186] INFO [SocketServer listenerType=ZK_BROKER, nodeId=0] Enabling request processing. (kafka.network.SocketServer)
[2023-03-31 22:14:49,206] INFO Kafka version: 3.4.0 (org.apache.kafka.common.utils.AppInfoParser)
[2023-03-31 22:14:49,206] INFO Kafka commitId: 2e1947d240607d53 (org.apache.kafka.common.utils.AppInfoParser)
[2023-03-31 22:14:49,206] INFO Kafka startTimeMs: 1680300889194 (org.apache.kafka.common.utils.AppInfoParser)
[2023-03-31 22:14:49,208] INFO [KafkaServer id=0] started (kafka.server.KafkaServer)
[2023-03-31 22:14:49,337] INFO [BrokerToControllerChannelManager broker=0 name=alterPartition]: Recorded new controller, from now on will use node me19b190-ass8
.c.me19b190-project1.internal:9092 (id: 0 rack: null) (kafka.server.BrokerToControllerRequestThread)
[2023-03-31 22:14:49,338] INFO [BrokerToControllerChannelManager broker=0 name=forwarding]: Recorded new controller, from now on will use node me19b190-ass8.c.m
e19b190-project1.internal:9092 (id: 0 rack: null) (kafka.server.BrokerToControllerRequestThread)
```

Step 3: Open a new terminal session to create a new topic "me19b190-ass8-topic" using the following commands. We are using local host server "localhost:9092" to run locally (not on a server)

```
me19b190@me19b190-ass8:~/kafka_2.13-3.4.0$ bin/kafka-topics.sh --create --topic me19b190-ass8-topic --bootstrap-server localhost:9092
Created topic me19b190-ass8-topic.
me19b190@me19b190-ass8:~/kafka_2.13-3.4.0$ bin/kafka-topics.sh --list --bootstrap-server localhost:9092
me19b190-ass8-topic
test-topic
me19b190@me19b190-ass8:~/kafka_2.13-3.4.0$ bin/kafka-topics.sh --describe --topic me19b190-ass8-topic --bootstrap-server localhost:9092
Topic: me19b190-ass8-topic      TopicId: 55fJMblmS0-cmZ30Xdyrrw PartitionCount: 1       ReplicationFactor: 1    Configs:
        Topic: me19b190-ass8-topic      Partition: 0    Leader: 0       Replicas: 0     Isr: 0
me19b190@me19b190-ass8:~/kafka_2.13-3.4.0$
```

Step 4: Before proceeding to run our producer and consumer files it is important to check if the created topic is working properly (are we able to write/ broadcast etc.), for which we open a new terminal session and split the screen into two panes and run each of the “test producer” and “test consumer” in the two created panes. Kafka topic is the created topic “me19b190-ass8-topic”.

```
me19b190@me19b190-ass8:~/kafka_2.13-3.4.0$ bin/kafka-console-producer.sh --topi | me19b190@me19b190-ass8:~/kafka_2.13-3.4.0$ bin/kafka-console-consumer.sh --topi
c me19b190-ass8-topic --bootstrap-server localhost:9092                         | c me19b190-ass8-topic --from-beginning --bootstrap-server localhost:9092
>I am Gangadhar                                                                 | I am Gangadhar
>This is  a Trail run                                                           | This is  a Trail run
>Exiting...                                                                     | Exiting...
>
```

Thus the created topic is running successfully.

Step 5: Upload a “.csv” to the bucket. A new bucket named “me19b190-ass8” is created and we upload the “Customers.csv” data set which is downloaded from Kaggle. It has 8 columns and 2001 rows. The following figure illustrates the dataset.

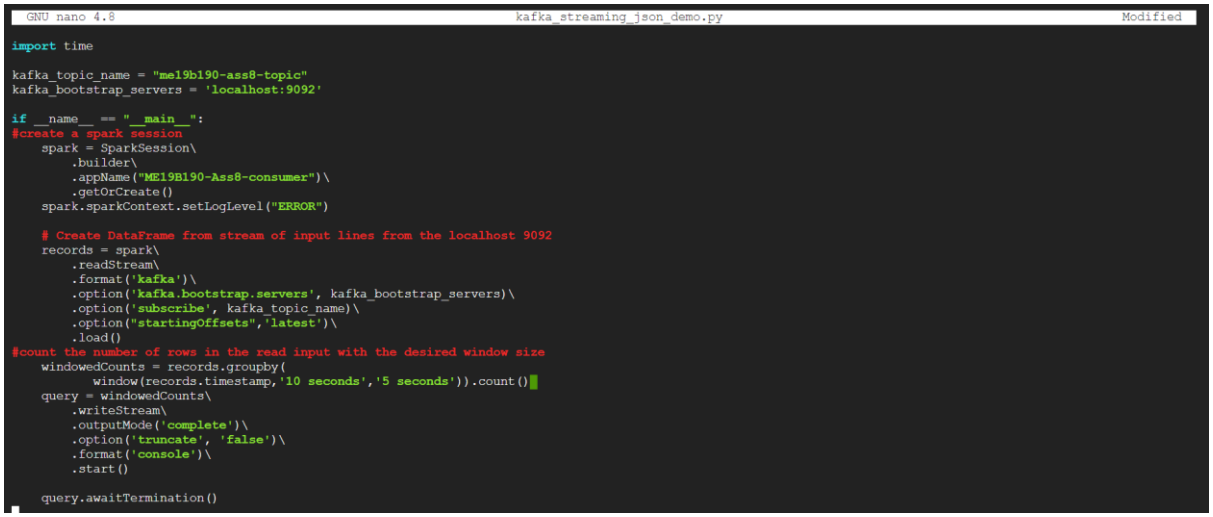
```
+----------+------+---+-----------------+----------------------+-------------+---------------+-----------+
|CustomerID|Gender|Age|Annual Income ($)|Spending Score (1-100)|   Profession|Work Experience|Family Size|
+----------+------+---+-----------------+----------------------+-------------+---------------+-----------+
|         1|  Male| 19|            15000|                    39|   Healthcare|              1|          4|
|         2|  Male| 21|            35000|                    81|     Engineer|              3|          3|
|         3|Female| 20|            86000|                     6|     Engineer|              1|          1|
|         4|Female| 23|            59000|                    77|       Lawyer|              0|          2|
|         5|Female| 31|            38000|                    40|Entertainment|              2|          6|
|         6|Female| 22|            58000|                    76|       Artist|              0|          2|
|         7|Female| 35|            31000|                     6|   Healthcare|              1|          3|
|         8|Female| 23|            84000|                    94|   Healthcare|              1|          3|
|         9|  Male| 64|            97000|                     3|     Engineer|              0|          3|
|        10|Female| 30|            98000|                    72|       Artist|              1|          4|
+----------+------+---+-----------------+----------------------+-------------+---------------+-----------+
only showing top 10 rows
```

Step 6: The following kafka_producer.py file reads the “.csv” file from the bucket and pushes it to the kafka topic “me19b190-ass8-topic”. The producer terminates once it pushes 100 records in batches of 10, which are separated by a sleep time of 10 seconds.

```
 GNU nano 4.8                                  kafka_producer.py                                  Modified

spark = SparkSession.builder.appName("pysparkdf").getOrCreate()   #create a spark session

KAFKA_TOPIC_NAME_CONS = "me19b190-ass8-topic"  #topic name
KAFKA_BOOTSTRAP_SERVERS_CONS = 'localhost:9092' #local host

if __name__ == "__main__":
        storage_client = storage.Client()
        bucket = storage_client.get_bucket('me19b190-bucket-ass8')
        blob = bucket.blob('Customers.csv')
        records_str = blob.download_as_string().decode('utf-8').split('\n')
        records_str = list(csv.reader(records_str, delimiter=","))
        data = records_str[1:-1]                #construct the dataframe
        header = records_str[0]                 #header of the dataframe
        df = spark.createDataFrame(data, header)

        num_records_terminate = 100     #termination number of records = 100
        print("Kafka Producer Application Started ... ")
        #create the producer object
        kafka_producer_obj = KafkaProducer(bootstrap_servers=KAFKA_BOOTSTRAP_SERVERS_CONS,value_serializer=lambda x: dumps(x).encode('utf-8'))
        df = df.withColumn("CustomerID", df['CustomerID'].cast("Integer"))
        i = 11
        while i <= num_records_terminate+1:
                #collect batches of 10 records each and push them to topic
                print("Current records: ")
                records = (df.filter((df.CustomerID >= i - 10) & (df.CustomerID < i)))
                records.show()
                for row in records.rdd.collect():
                        kafka_producer_obj.send(KAFKA_TOPIC_NAME_CONS, row)
                #increment i
                i = i+10
                time.sleep(10) #sleep time of the producer, send data after a wait time of 10 seconds

        print("Kafka Producer Application Completed. ")

^G Get Help  ^O Write Out  ^W Where Is  ^K Cut Text  ^J Justify  ^C Cur Pos  M-U Undo  M-A Mark Text  M-] To Bracket  M-Q Previous
^X Exit      ^R Read File  ^\ Replace   ^U Paste Text ^T To Spell  ^_ Go To Line M-E Redo  M-6 Copy Text  ^Q Where Was  M-W Next
[0] 0:java  1:bash  2:nano  3:bash  4:bash  5:bash  6:bash-  7:bash  8:nano*                "me19b190-ass8" 21:40 31-Mar-2
```

Name: Royyuru Sai Prasanna Gangadhar **Roll Number**: ME19B190

Step 7: The below given file is the code for consumer. It reads the data from kafka every 5 seconds and emits the count of rows seen in the last 10 seconds.



```
 GNU nano 4.8                    kafka_streaming_json_demo.py                    Modified

import time

kafka_topic_name = "me19b190-ass8-topic"
kafka_bootstrap_servers = 'localhost:9092'

if __name__ == "__main__":
#create a spark session
    spark = SparkSession\
        .builder\
        .appName("ME19B190-Ass8-consumer")\
        .getOrCreate()
    spark.sparkContext.setLogLevel("ERROR")

    # Create DataFrame from stream of input lines from the localhost 9092
    records = spark\
        .readStream\
        .format('kafka')\
        .option('kafka.bootstrap.servers', kafka_bootstrap_servers)\
        .option('subscribe', kafka_topic_name)\
        .option("startingOffsets","latest")\
        .load()
#count the number of rows in the read input with the desired window size
    windowedCounts = records.groupby(
            window(records.timestamp,'10 seconds','5 seconds')).count()
    query = windowedCounts\
        .writeStream\
        .outputMode('complete')\
        .option('truncate', 'false')\
        .format('console')\
        .start()

    query.awaitTermination()
```

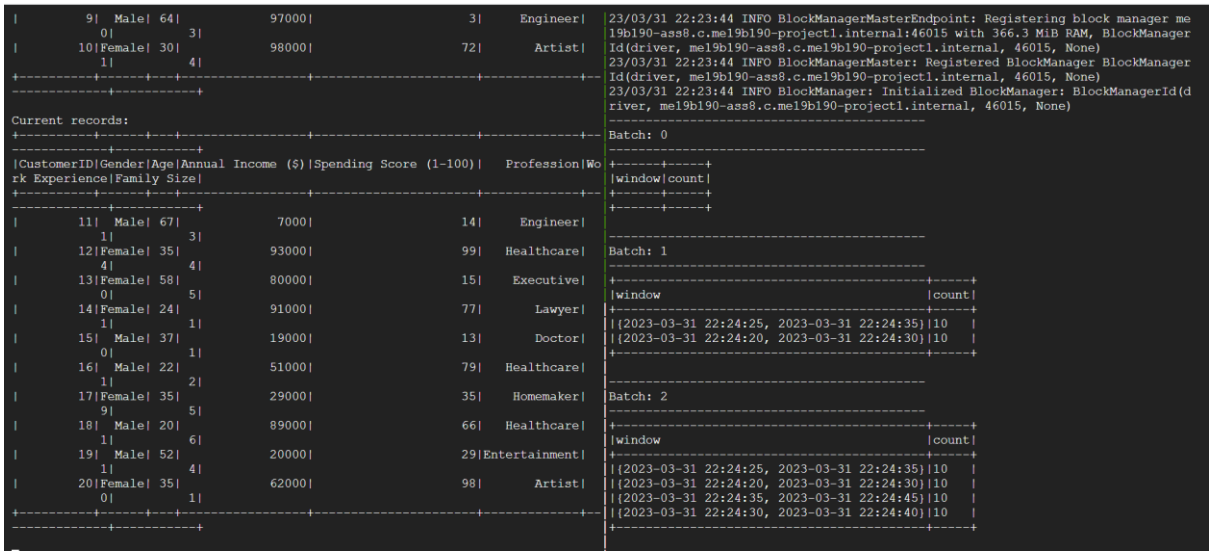
Step 8: Open a new terminal session and spilt the terminal into two panes and run the producer and consumer file separately in each of the panes using the following commands

Command to execute the consumer file: *"spark-submit --packages org.apache.spark:spark-sql-kafka-0-10_2.12:3.3.2 kafka_streaming_json_demo.py"*

```
me19b190@me19b190-ass8:~/kafka_2.13-3.4.0/ME19B190-ass8-producer_consumer_files
$ spark-submit --packages org.apache.spark:spark-sql-kafka-0-10_2.12:3.3.2 kafk
a_streaming_json_demo.py
```

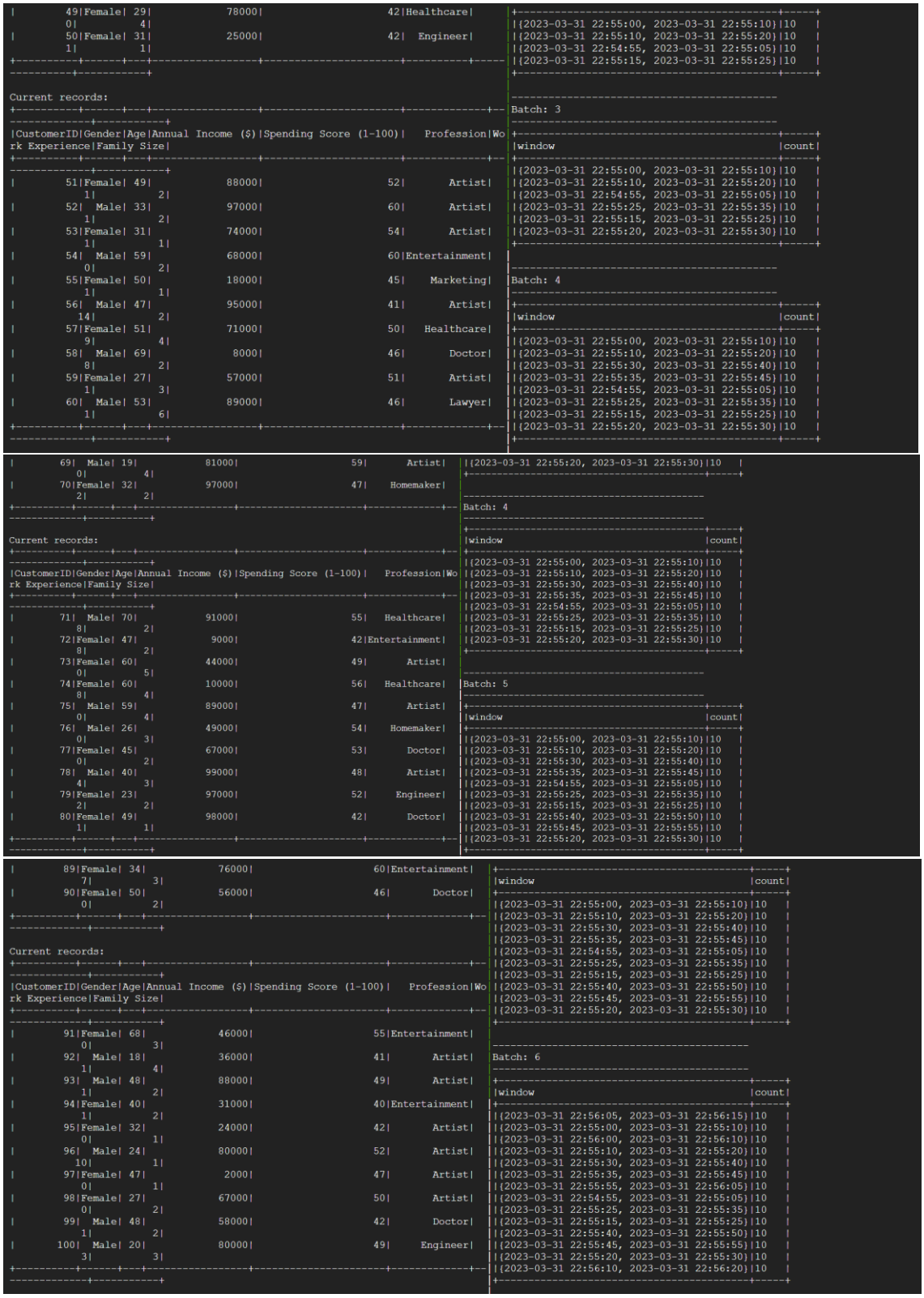
Command to execute the producer file: *"python3 kafka_producer.py"*

Step 9: The output of the producer(left) and consumer(right) is shown below. The producer prints the current records (separated by a sleep time of 10 seconds) it is pushing to the kafka and the consumer reads the data from kafka every 5 seconds and emits the count of rows seen in the last 10 seconds.



```
|         9|  Male| 64|            97000|                      3|     Engineer|
|           0|         3|
|        10|Female| 30|            98000|                     72|       Artist|
|           1|         4|
+----------+------+---+-----------------+-----------------------+-------------+--
-------------+----------+

Current records:
+----------+------+---+-----------------+-----------------------+-------------+--
-------------+----------+
|CustomerID|Gender|Age|Annual Income ($)|Spending Score (1-100)|   Profession|Wo
rk Experience|Family Size|
+----------+------+---+-----------------+-----------------------+-------------+--
-------------+----------+
|        11|  Male| 67|             7000|                     14|     Engineer|
|           1|         3|
|        12|Female| 35|            93000|                     99|   Healthcare|
|           4|         4|
|        13|Female| 58|            80000|                     15|    Executive|
|           0|         5|
|        14|Female| 24|            91000|                     77|       Lawyer|
|           1|         1|
|        15|  Male| 37|            19000|                     13|       Doctor|
|           0|         1|
|        16|  Male| 22|            51000|                     79|   Healthcare|
|           1|         2|
|        17|Female| 35|            29000|                     35|    Homemaker|
|           9|         5|
|        18|  Male| 20|            89000|                     66|   Healthcare|
|           1|         6|
|        19|  Male| 52|            20000|                     29|Entertainment|
|           1|         4|
|        20|Female| 35|            62000|                     98|       Artist|
|           0|         1|
+----------+------+---+-----------------+-----------------------+-------------+--
-------------+----------+
```

```
23/03/31 22:23:44 INFO BlockManagerMasterEndpoint: Registering block manager me
19b190-ass8.c.me19b190-project1.internal:46015 with 366.3 MiB RAM, BlockManager
Id(driver, me19b190-ass8.c.me19b190-project1.internal, 46015, None)
23/03/31 22:23:44 INFO BlockManagerMaster: Registered BlockManager BlockManager
Id(driver, me19b190-ass8.c.me19b190-project1.internal, 46015, None)
23/03/31 22:23:44 INFO BlockManager: Initialized BlockManager: BlockManagerId(d
river, me19b190-ass8.c.me19b190-project1.internal, 46015, None)
-------------------------------------------
Batch: 0
-------------------------------------------
+------+-----+
|window|count|
+------+-----+
+------+-----+

-------------------------------------------
Batch: 1
-------------------------------------------
+------------------------------------------+-----+
|window                                    |count|
+------------------------------------------+-----+
|{2023-03-31 22:24:25, 2023-03-31 22:24:35}|10   |
|{2023-03-31 22:24:20, 2023-03-31 22:24:30}|10   |
+------------------------------------------+-----+

-------------------------------------------
Batch: 2
-------------------------------------------
+------------------------------------------+-----+
|window                                    |count|
+------------------------------------------+-----+
|{2023-03-31 22:24:25, 2023-03-31 22:24:35}|10   |
|{2023-03-31 22:24:20, 2023-03-31 22:24:30}|10   |
|{2023-03-31 22:24:35, 2023-03-31 22:24:45}|10   |
|{2023-03-31 22:24:30, 2023-03-31 22:24:40}|10   |
+------------------------------------------+-----+
```



Step 10: Clean up the logs using the following command *"rm -rf /tmp/kafka-logs /tmp/zookeeper /tmp/kraft-combined-logs"*

```
me19b190@me19b190-ass8:~/kafka_2.13-3.4.0/ME19B190-ass8-producer_consumer_files$
 rm -rf /tmp/kafka-logs /tmp/zookeeper /tmp/kraft-combined-logs
me19b190@me19b190-ass8:~/kafka_2.13-3.4.0/ME19B190-ass8-producer_consumer_files$
```