Retail Data Analysis
Code Logic

Logic for Python Script ‘spark-streaming.py’

Setting up the system dependencies for Cloudera distribution by importing necessary libraries, modules and the path variables

```
import os
import sys

os.environ["PYSPARK_PYTHON"] = "/opt/cloudera/parcels/Anaconda/bin/python"
os.environ["JAVA_HOME"] = "/usr/java/jdk1.8.0_232-cloudera/jre"
os.environ["SPARK_HOME"]="/opt/cloudera/parcels/SPARK2-2.3.0.cloudera2-
1.cdh5.13.3.p0.316101/lib/spark2/"
os.environ["PYLIB"] = os.environ["SPARK_HOME"] + "/python/lib"
sys.path.insert(0, os.environ["PYLIB"] +"/py4j-0.10.6-src.zip")
sys.path.insert(0, os.environ["PYLIB"] +"/pyspark.zip")

from pyspark.sql import SparkSession
from pyspark.sql.functions import *
from pyspark.sql.types import *
```

Writing the Python functions, which contain the logic for the UDFs

1. Total Cost UDF - To calculate the total income from every invoice I needed to calculate the income from sale of each product, so I multiplied the unit price of the product with the quantity of the product purchased. The sum of this cost across the products in that invoice gives me the total cost of the order. I also made sure that if the transaction is a return transaction, the total cost is negative.

```
def find_total_order_cost(items, trn_type):
    if items is not None:
        total_cost = 0
        item_price = 0
        for item in items:
           item_price = (item['quantity'] * item['unit_price'])
            total_cost = total_cost + item_price
            item_price = 0

        if trn_type == "RETURN":
            return total_cost * -1
        else:
            return total_cost
```

2. Total Items UDF - To calculate the number of products in every invoice I added the quantity ordered of each product in that invoice

```
def find_total_item_count(items):
    if items is not None:
        total_count = 0
        for item in items:
            total_count = total_count + item['quantity']
        return total_count
```

3. Is Order UDF - To determine if invoice is for an order or not I used an if-else statement

```
def flag_isOrder(trn_type):
    if trn_type == "ORDER":
        return(1)
    else:
        return(0)
```

4. Is Return UDF - To determine if invoice is for a return or not I used an if-else statement

```
def flag_isReturn(trn_type):
    if trn_type == "RETURN":
        return(1)
    else:
        return(0)
```

Initialising the Spark session and setting the log level to error as a good practice

```
spark = SparkSession  \
    .builder  \
    .appName("spark-streaming")  \
    .getOrCreate()
spark.sparkContext.setLogLevel('ERROR')
```

Reading input data from Kafka mentioning the details of the Kafka broker, such as bootstrap server, port and topic name

```
orderRawData = spark.readStream \
    .format("kafka") \
    .option("kafka.bootstrap.servers", "18.211.252.152:9092") \
    .option("startingOffsets","earliest") \
    .option("failOnDataLoss", "false") \
    .option("subscribe", "real-time-project") \
    .load()
```

Defining JSON schema of each order, using appropriate datatypes and StrucField in the case of the item attributes

```
jsonSchema = StructType() \
    .add("invoice_no", LongType()) \
    .add("country", StringType()) \
    .add("timestamp", TimestampType()) \
    .add("type", StringType()) \
    .add("items", ArrayType(StructType([
    StructField("SKU", StringType()),
    StructField("title", StringType()),
    StructField("unit_price", FloatType()),
    StructField("quantity", IntegerType()),
])))
```

Reading the raw JSON data from Kafka as ‘order stream’ by casting it to string and storing it into the alias ‘data’

```
orderStream = orderRawData.select(from_json(col("value").cast("string"),
jsonSchema).alias("data")).select("data.*")
```

Defining the UDFs by Converting the Python functions I defined earlier, and assigning the appropriate return datatype

```
sum_total_order_cost = udf(find_total_order_cost, FloatType())
sum_total_item_count = udf(find_total_item_count, IntegerType())
sum_isOrder = udf(flag_isOrder, IntegerType())
sum_isReturn = udf (flag_isReturn, IntegerType())
```

Calculating the additional columns according to the required input values

```
expandedOrderStream = orderStream \
    .withColumn("total_cost", sum_total_order_cost(orderStream.items,
orderStream.type)) \
    .withColumn("total_items", sum_total_item_count(orderStream.items)) \
    .withColumn("is_order", sum_isOrder(orderStream.type)) \
    .withColumn("is_return", sum_isReturn(orderStream.type))
```

Writing the summarised input values to console, using 'append' output method and applying truncate as false and setting the processing time to 1 minute

```
extendedOrderQuery = expandedOrderStream \
    .select("invoice_no", "country", "timestamp", "total_cost", "total_items",
"is_order", "is_return") \
    .writeStream \
    .outputMode("append") \
    .format("console") \
    .option("truncate", "false") \
    .trigger(processingTime = "1 minute") \
    .start()
```

Calculating time-based KPIs (Total sale volume, OPM, Rate of return, Average transaction size) having tumbling window of one minute and watermark of one minute.

```
aggStreamByTime = expandedOrderStream \
    .withWatermark("timestamp","1 minute") \
    .groupBy(window("timestamp", "1 minute", "1 minute")) \
    .agg(sum("total_cost").alias("total_sale_volume"),
         count("invoice_no").alias("OPM"),
         avg("is_return").alias("rate_of_return"),
         avg("total_cost").alias("average_transaction_size")
        ) \
    .select("window", "OPM", "total_sale_volume", "average_transaction_size",
"rate_of_return")
```

Writing the time-based KPIs data to HDFS - HDFS into JSON files for each one-minute window, using 'append' output mode, setting truncate as false, and specifying the HDFS output path for both the KPI files and for their checkpoints. Ten 1-minute window batches were taken.

```
queryByTime = aggStreamByTime.writeStream \
    .format("json") \
    .outputMode("append") \
    .option("truncate","false") \
    .option("path","/user/ec2-user/time_kpi") \
    .option("checkpointLocation","/user/ec2-user/time_kpi_checkpoints") \
    .trigger(processingTime="1 minute") \
    .start()
```

Calculating time-and-country-based KPIs (Total sale volume, OPM, Rate of return) having tumbling window of one minute and watermark of one minute. Here I grouped by window and country both.

```
aggStreamByCountry = expandedOrderStream \
    .withWatermark("timestamp", "1 minute") \
    .groupBy(window("timestamp", "1 minute", "1 minute"), "country") \
    .agg(sum("total_cost").alias("total_sale_volume"),
         count("invoice_no").alias("OPM"),
         avg("is_return").alias("rate_of_return")) \
    .select("window", "country", "OPM", "total_sale_volume", "rate_of_return")
```

Writing the the time-and-country-based KPIs data to HDFS into JSON files for each one-minute window, using 'append' output mode, setting truncate as false, and specifying the HDFS output path for both the KPI files and for their checkpoints. Ten 1-minute window batches were taken.

```
queryByCountry = aggStreamByCountry.writeStream \
    .format("json") \
    .outputMode("append") \
    .option("truncate","false") \
    .option("path","/user/ec2-user/country_kpi") \
    .option("checkpointLocation","/user/ec2-user/country_kpi_checkpoints") \
    .trigger(processingTime="1 minute") \
    .start()
```

Indicating Spark to await termination

```
extendedOrderQuery.awaitTermination()
queryByCountry.awaitTermination()
queryByTime.awaitTermination()
```

Console Commands

I started by logging into the ec2 instance as 'ec2-user'

Next, I downloaded the Spark-SQL-Kafka jar file. This jar is used to run the Spark Streaming-Kafka codes

```
wget https://ds-spark-sql-kafka-jar.s3.amazonaws.com/spark-sql-kafka-0-10_2.11-
2.3.0.jar
```

Next, I created the 'spark-streaming.py' file having the code discussed above

```
vi spark-streaming.py
```

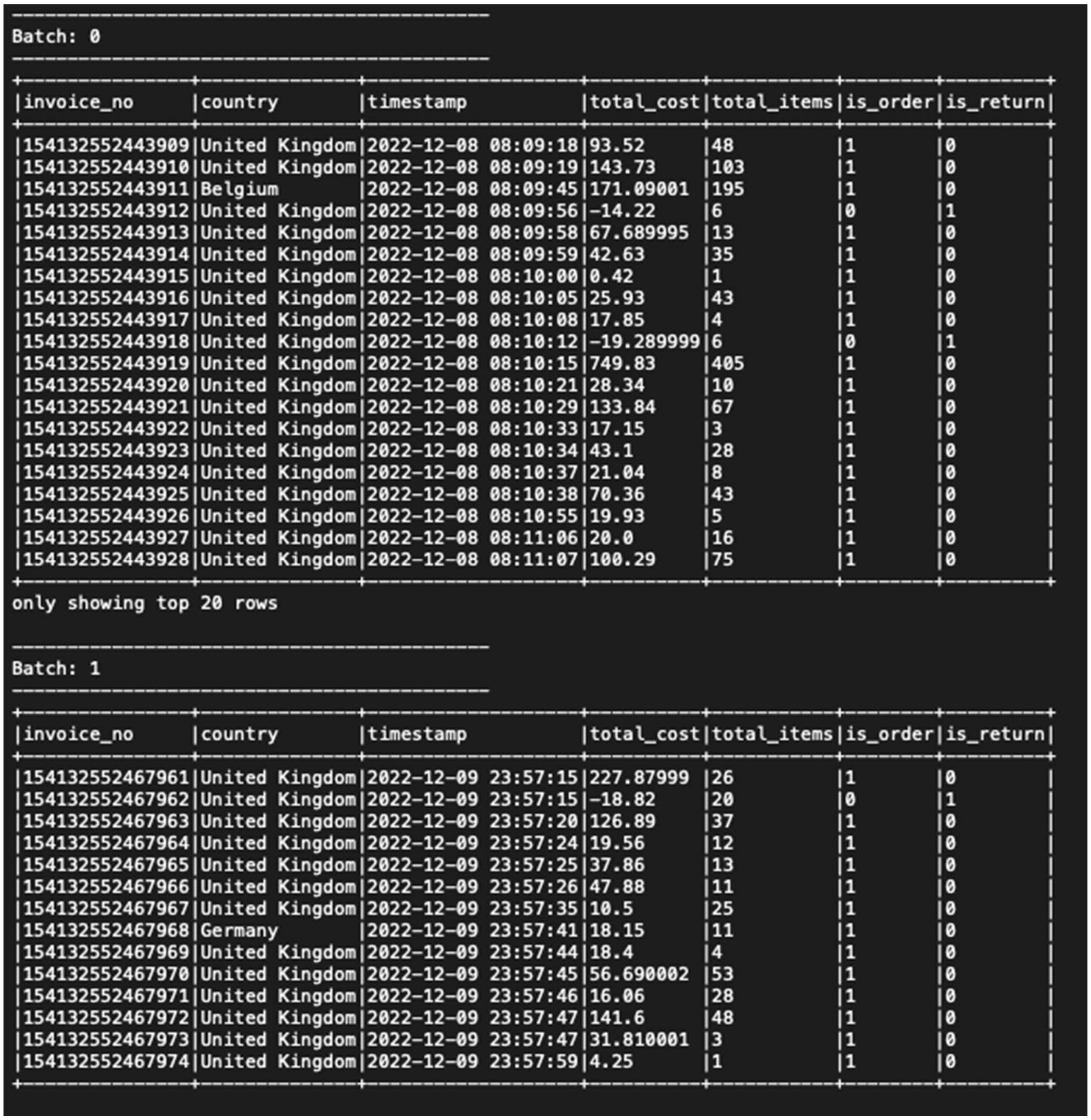
Next, I set the Kafka Version using the following command

```
export SPARK_KAFKA_VERSION=0.10
```

Finally, I ran the spark2-submit command, specifying the jar and python file

```
spark2-submit --jars spark-sql-kafka-0-10_2.11-2.3.0.jar spark-streaming.py
```

Example table - Final Summarised Input Values

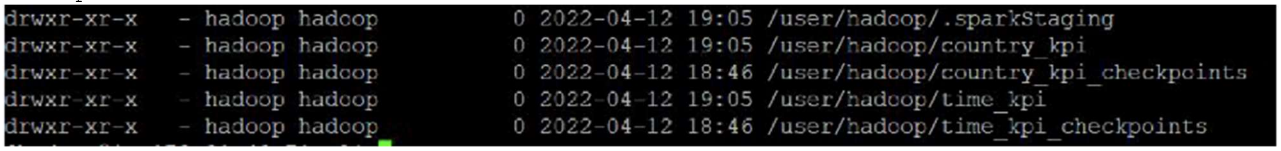


```
-------------------------------------------
Batch: 0
-------------------------------------------
+---------------+--------------+-------------------+----------+-----------+--------+---------+
|invoice_no     |country       |timestamp          |total_cost|total_items|is_order|is_return|
+---------------+--------------+-------------------+----------+-----------+--------+---------+
|154132552443909|United Kingdom|2022-12-08 08:09:18|93.52     |48         |1       |0        |
|154132552443910|United Kingdom|2022-12-08 08:09:19|143.73    |103        |1       |0        |
|154132552443911|Belgium       |2022-12-08 08:09:45|171.09001 |195        |1       |0        |
|154132552443912|United Kingdom|2022-12-08 08:09:56|-14.22    |6          |0       |1        |
|154132552443913|United Kingdom|2022-12-08 08:09:58|67.689995 |13         |1       |0        |
|154132552443914|United Kingdom|2022-12-08 08:09:59|42.63     |35         |1       |0        |
|154132552443915|United Kingdom|2022-12-08 08:10:00|0.42      |1          |1       |0        |
|154132552443916|United Kingdom|2022-12-08 08:10:05|25.93     |43         |1       |0        |
|154132552443917|United Kingdom|2022-12-08 08:10:08|17.85     |4          |1       |0        |
|154132552443918|United Kingdom|2022-12-08 08:10:12|-19.289999|6          |0       |1        |
|154132552443919|United Kingdom|2022-12-08 08:10:15|749.83    |405        |1       |0        |
|154132552443920|United Kingdom|2022-12-08 08:10:21|28.34     |10         |1       |0        |
|154132552443921|United Kingdom|2022-12-08 08:10:29|133.84    |67         |1       |0        |
|154132552443922|United Kingdom|2022-12-08 08:10:33|17.15     |3          |1       |0        |
|154132552443923|United Kingdom|2022-12-08 08:10:34|43.1      |28         |1       |0        |
|154132552443924|United Kingdom|2022-12-08 08:10:37|21.04     |8          |1       |0        |
|154132552443925|United Kingdom|2022-12-08 08:10:38|70.36     |43         |1       |0        |
|154132552443926|United Kingdom|2022-12-08 08:10:55|19.93     |5          |1       |0        |
|154132552443927|United Kingdom|2022-12-08 08:11:06|20.0      |16         |1       |0        |
|154132552443928|United Kingdom|2022-12-08 08:11:07|100.29    |75         |1       |0        |
+---------------+--------------+-------------------+----------+-----------+--------+---------+
only showing top 20 rows

-------------------------------------------
Batch: 1
-------------------------------------------
+---------------+--------------+-------------------+----------+-----------+--------+---------+
|invoice_no     |country       |timestamp          |total_cost|total_items|is_order|is_return|
+---------------+--------------+-------------------+----------+-----------+--------+---------+
|154132552467961|United Kingdom|2022-12-09 23:57:15|227.87999 |26         |1       |0        |
|154132552467962|United Kingdom|2022-12-09 23:57:15|-18.82    |20         |0       |1        |
|154132552467963|United Kingdom|2022-12-09 23:57:20|126.89    |37         |1       |0        |
|154132552467964|United Kingdom|2022-12-09 23:57:24|19.56     |12         |1       |0        |
|154132552467965|United Kingdom|2022-12-09 23:57:25|37.86     |13         |1       |0        |
|154132552467966|United Kingdom|2022-12-09 23:57:26|47.88     |11         |1       |0        |
|154132552467967|United Kingdom|2022-12-09 23:57:35|10.5      |25         |1       |0        |
|154132552467968|Germany       |2022-12-09 23:57:41|18.15     |11         |1       |0        |
|154132552467969|United Kingdom|2022-12-09 23:57:44|18.4      |4          |1       |0        |
|154132552467970|United Kingdom|2022-12-09 23:57:45|56.690002 |53         |1       |0        |
|154132552467971|United Kingdom|2022-12-09 23:57:46|16.06     |28         |1       |0        |
|154132552467972|United Kingdom|2022-12-09 23:57:47|141.6     |48         |1       |0        |
|154132552467973|United Kingdom|2022-12-09 23:57:47|31.810001 |3          |1       |0        |
|154132552467974|United Kingdom|2022-12-09 23:57:59|4.25      |1          |1       |0        |
+---------------+--------------+-------------------+----------+-----------+--------+---------+
```

I checked HDFS to make sure the KPI files were present

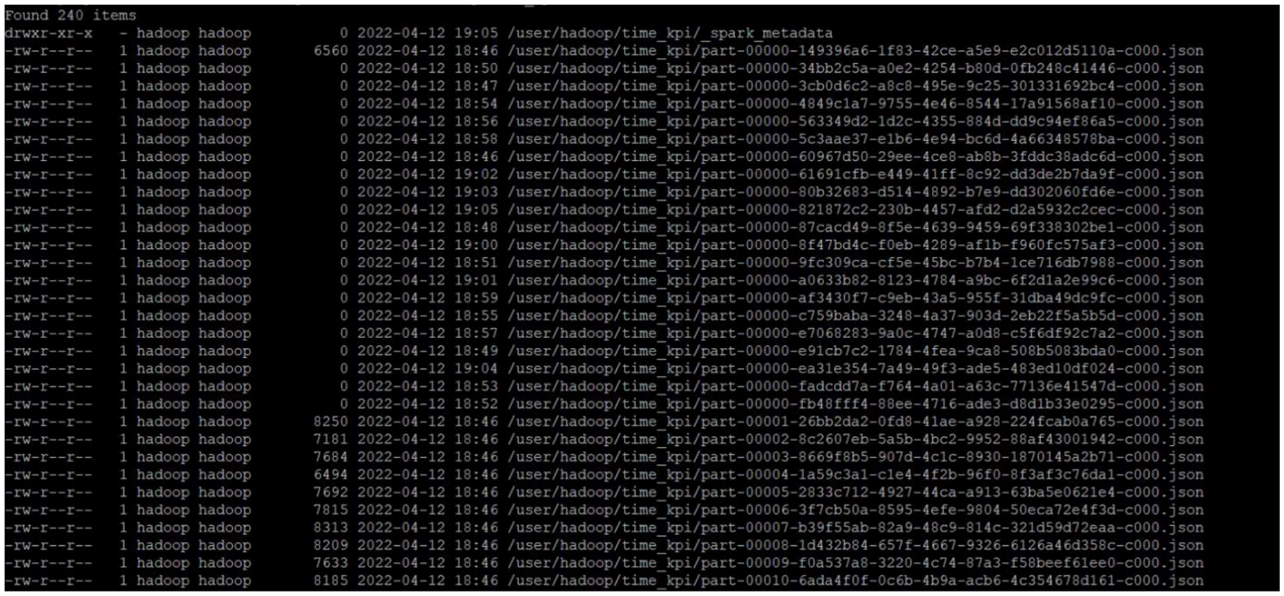
```
hadoop fs -ls /user/ec2-user
```



```
drwxr-xr-x   - hadoop hadoop          0 2022-04-12 19:05 /user/hadoop/.sparkStaging
drwxr-xr-x   - hadoop hadoop          0 2022-04-12 19:05 /user/hadoop/country_kpi
drwxr-xr-x   - hadoop hadoop          0 2022-04-12 18:46 /user/hadoop/country_kpi_checkpoints
drwxr-xr-x   - hadoop hadoop          0 2022-04-12 19:05 /user/hadoop/time_kpi
drwxr-xr-x   - hadoop hadoop          0 2022-04-12 18:46 /user/hadoop/time_kpi_checkpoints
```

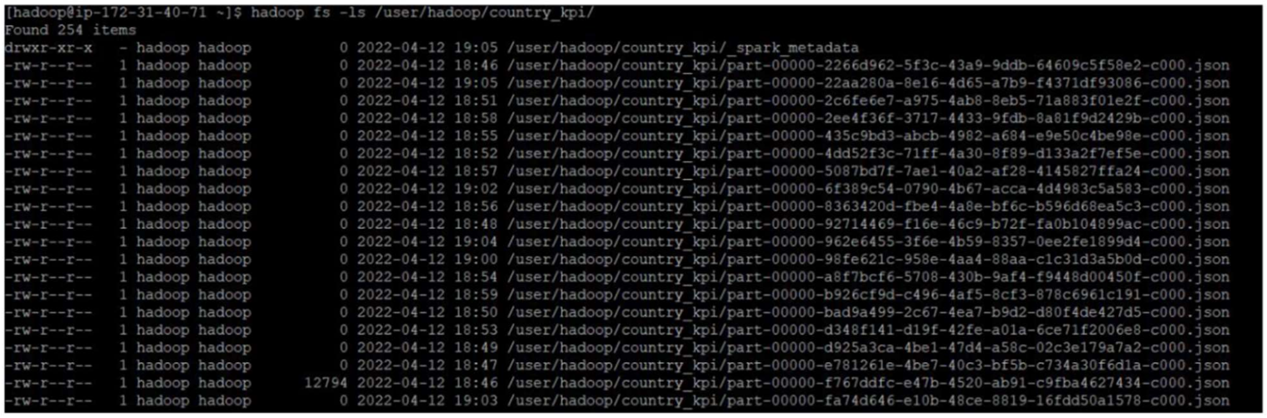
I also checked the folders to see the JSON files

```
hadoop fs -ls /user/ec2-user/time_kpi/
```



```
Found 240 items
drwxr-xr-x   - hadoop hadoop          0 2022-04-12 19:05 /user/hadoop/time_kpi/_spark_metadata
-rw-r--r--   1 hadoop hadoop       6560 2022-04-12 18:46 /user/hadoop/time_kpi/part-00000-149396a6-1f83-42ce-a5e9-e2c012d5110a-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-04-12 18:50 /user/hadoop/time_kpi/part-00000-34bb2c5a-a0e2-4254-b80d-0fb248c41446-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-04-12 18:47 /user/hadoop/time_kpi/part-00000-3cb0d6c2-a8c8-495e-9c25-301331692bc4-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-04-12 18:54 /user/hadoop/time_kpi/part-00000-4849c1a7-9755-4e46-8544-17a91568af10-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-04-12 18:56 /user/hadoop/time_kpi/part-00000-563349d2-1d2c-4355-884d-dd9c94ef86a5-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-04-12 18:58 /user/hadoop/time_kpi/part-00000-5c3aae37-e1b6-4e94-bc6d-4a66348578ba-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-04-12 18:46 /user/hadoop/time_kpi/part-00000-60967d50-29ee-4ce8-ab8b-3fddc38adc6d-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-04-12 19:02 /user/hadoop/time_kpi/part-00000-61691cfb-e449-41ff-8c92-dd3de2b7da9f-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-04-12 19:03 /user/hadoop/time_kpi/part-00000-80b32683-d514-4892-b7e9-dd302060fd6e-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-04-12 19:05 /user/hadoop/time_kpi/part-00000-821872c2-230b-4457-afd2-d2a5932c2cec-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-04-12 18:48 /user/hadoop/time_kpi/part-00000-87cacd49-8f5e-4639-9459-69f338302be1-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-04-12 19:00 /user/hadoop/time_kpi/part-00000-8f47bd4c-f0eb-4289-af1b-f960fc575af3-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-04-12 18:51 /user/hadoop/time_kpi/part-00000-9fc309ca-cf5e-45bc-b7b4-1ce716db7988-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-04-12 19:01 /user/hadoop/time_kpi/part-00000-a0633b82-8123-4784-a9bc-6f2d1a2e99c6-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-04-12 18:59 /user/hadoop/time_kpi/part-00000-af3430f7-c9eb-43a5-955f-31dba49dc9fc-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-04-12 18:55 /user/hadoop/time_kpi/part-00000-c759baba-3248-4a37-903d-2eb22f5a5b5d-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-04-12 18:57 /user/hadoop/time_kpi/part-00000-e7068283-9a0c-4747-a0d8-c5f6df92c7a2-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-04-12 18:49 /user/hadoop/time_kpi/part-00000-e91cb7c2-1784-4fea-9ca8-508b5083bda0-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-04-12 19:04 /user/hadoop/time_kpi/part-00000-ea31e354-7a49-49f3-ade5-483ed10df024-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-04-12 18:53 /user/hadoop/time_kpi/part-00000-fadcdd7a-f764-4a01-a63c-77136e41547d-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-04-12 18:52 /user/hadoop/time_kpi/part-00000-fb48fff4-88ee-4716-ade3-d8d1b33e0295-c000.json
-rw-r--r--   1 hadoop hadoop       8250 2022-04-12 18:46 /user/hadoop/time_kpi/part-00001-26bb2da2-0fd8-41ae-a928-224fcab0a765-c000.json
-rw-r--r--   1 hadoop hadoop       7181 2022-04-12 18:46 /user/hadoop/time_kpi/part-00002-8c2607eb-5a5b-4bc2-9952-88af43001942-c000.json
-rw-r--r--   1 hadoop hadoop       7684 2022-04-12 18:46 /user/hadoop/time_kpi/part-00003-8669f8b5-907d-4c1c-8930-1870145a2b71-c000.json
-rw-r--r--   1 hadoop hadoop       6494 2022-04-12 18:46 /user/hadoop/time_kpi/part-00004-1a59c3a1-c1e4-4f2b-96f0-8f3af3c76da1-c000.json
-rw-r--r--   1 hadoop hadoop       7692 2022-04-12 18:46 /user/hadoop/time_kpi/part-00005-2833c712-4927-44ca-a913-63ba5e0621e4-c000.json
-rw-r--r--   1 hadoop hadoop       7815 2022-04-12 18:46 /user/hadoop/time_kpi/part-00006-3f7cb50a-8595-4efe-9804-50eca72e4f3d-c000.json
-rw-r--r--   1 hadoop hadoop       8313 2022-04-12 18:46 /user/hadoop/time_kpi/part-00007-b39f55ab-82a9-48c9-814c-321d59d72eaa-c000.json
-rw-r--r--   1 hadoop hadoop       8209 2022-04-12 18:46 /user/hadoop/time_kpi/part-00008-1d432b84-657f-4667-9326-6126a46d358c-c000.json
-rw-r--r--   1 hadoop hadoop       7633 2022-04-12 18:46 /user/hadoop/time_kpi/part-00009-f0a537a8-3220-4c74-87a3-f58beef61ee0-c000.json
-rw-r--r--   1 hadoop hadoop       8185 2022-04-12 18:46 /user/hadoop/time_kpi/part-00010-6ada4f0f-0c6b-4b9a-acb6-4c354678d161-c000.json
```

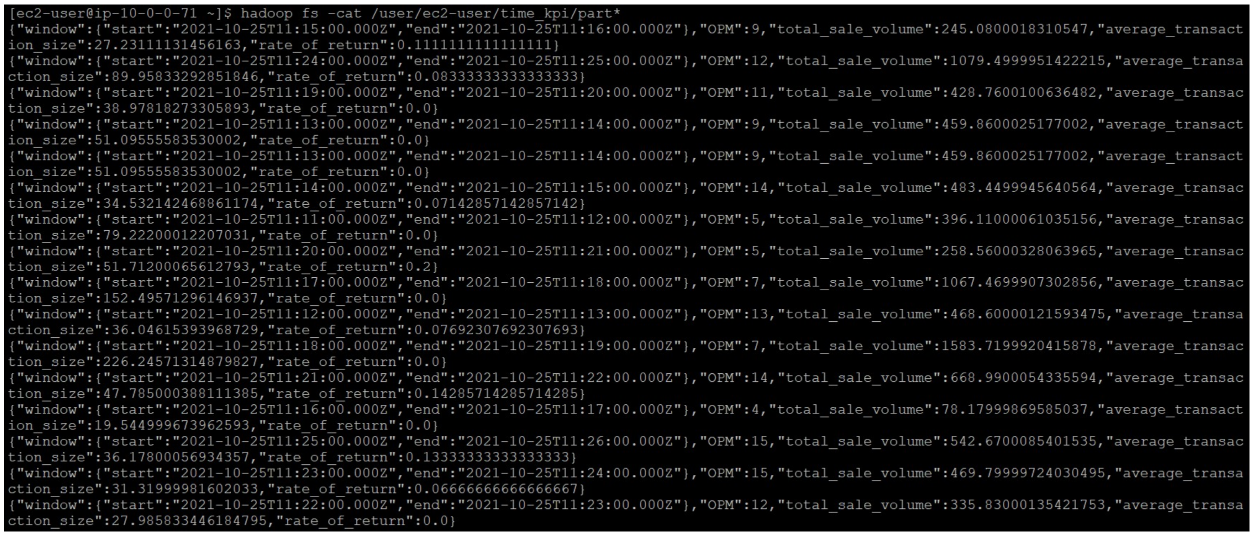
```
hadoop fs -ls /user/ec2-user/country_kpi/
```



```
[hadoop@ip-172-31-40-71 ~]$ hadoop fs -ls /user/hadoop/country_kpi/
Found 254 items
drwxr-xr-x   - hadoop hadoop          0 2022-04-12 19:05 /user/hadoop/country_kpi/_spark_metadata
-rw-r--r--   1 hadoop hadoop          0 2022-04-12 18:46 /user/hadoop/country_kpi/part-00000-2266d962-5f3c-43a9-9ddb-64609c5f58e2-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-04-12 19:05 /user/hadoop/country_kpi/part-00000-22aa280a-8e16-4d65-a7b9-f4371df93086-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-04-12 18:51 /user/hadoop/country_kpi/part-00000-2c6fe6e7-a975-4ab8-8eb5-71a883f01e2f-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-04-12 18:58 /user/hadoop/country_kpi/part-00000-2ee4f36f-3717-4433-9fdb-8a81f9d2429b-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-04-12 18:55 /user/hadoop/country_kpi/part-00000-435c9bd3-abcb-4982-a684-e9e50c4be98e-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-04-12 18:52 /user/hadoop/country_kpi/part-00000-4dd52f3c-71ff-4a30-8f89-d133a2f7ef5e-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-04-12 18:57 /user/hadoop/country_kpi/part-00000-5087bd7f-7ae1-40a2-af28-4145827ffa24-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-04-12 19:02 /user/hadoop/country_kpi/part-00000-6f389c54-0790-4b67-acca-4d4983c5a583-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-04-12 18:56 /user/hadoop/country_kpi/part-00000-8363420d-fbe4-4a8e-bf6c-b596d68ea5c3-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-04-12 18:48 /user/hadoop/country_kpi/part-00000-92714469-f16e-46c9-b72f-fa0b104899ac-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-04-12 19:04 /user/hadoop/country_kpi/part-00000-962e6455-3f6e-4b59-8357-0ee2fe1899d4-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-04-12 19:00 /user/hadoop/country_kpi/part-00000-98fe621c-958e-4aa4-88aa-c1c31d3a5b0d-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-04-12 18:54 /user/hadoop/country_kpi/part-00000-a8f7bcf6-5708-430b-9af4-f9448d00450f-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-04-12 18:59 /user/hadoop/country_kpi/part-00000-b926cf9d-c496-4af5-8cf3-878c6961c191-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-04-12 18:50 /user/hadoop/country_kpi/part-00000-bad9a499-2c67-4ea7-b9d2-d80f4de427d5-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-04-12 18:53 /user/hadoop/country_kpi/part-00000-d348f141-d19f-42fe-a01a-6ce71f2006e8-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-04-12 18:49 /user/hadoop/country_kpi/part-00000-d925a3ca-4be1-47d4-a58c-02c3e179a7a2-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-04-12 18:47 /user/hadoop/country_kpi/part-00000-e781261e-4be7-40c3-bf5b-c734a30f6d1a-c000.json
-rw-r--r--   1 hadoop hadoop      12794 2022-04-12 18:46 /user/hadoop/country_kpi/part-00000-f767ddfc-e47b-4520-ab91-c9fba4627434-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-04-12 19:03 /user/hadoop/country_kpi/part-00000-fa74d646-e10b-48ce-8819-16fdd50a1578-c000.json
```

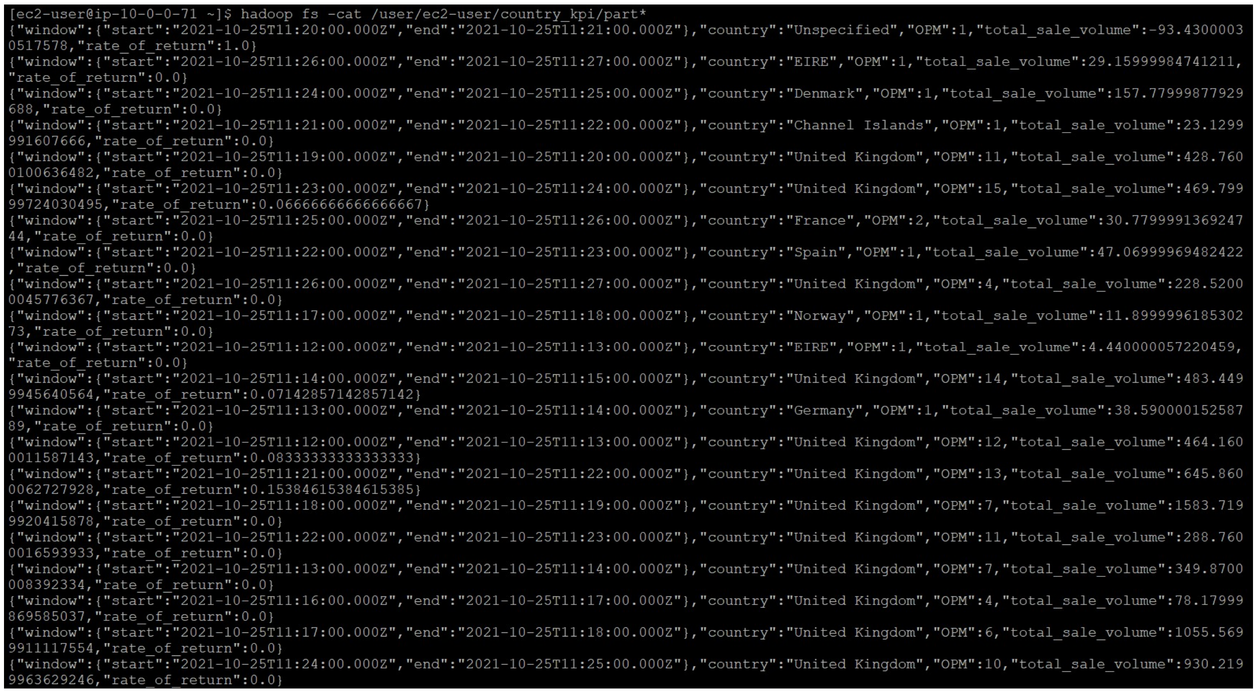
And used ‘cat’ command to take a look at the data

```
hadoop fs -cat /user/ec2-user/time_kpi/part*
```



```
[ec2-user@ip-10-0-0-71 ~]$ hadoop fs -cat /user/ec2-user/time_kpi/part*
{"window":{"start":"2021-10-25T11:15:00.000Z","end":"2021-10-25T11:16:00.000Z"},"OPM":9,"total_sale_volume":245.0800018310547,"average_transact
ion_size":27.23111131456163,"rate_of_return":0.1111111111111111}
{"window":{"start":"2021-10-25T11:24:00.000Z","end":"2021-10-25T11:25:00.000Z"},"OPM":12,"total_sale_volume":1079.4999951422215,"average_transa
ction_size":89.95833292851846,"rate_of_return":0.08333333333333333}
{"window":{"start":"2021-10-25T11:19:00.000Z","end":"2021-10-25T11:20:00.000Z"},"OPM":11,"total_sale_volume":428.7600100636482,"average_transac
tion_size":38.97818273305893,"rate_of_return":0.0}
{"window":{"start":"2021-10-25T11:13:00.000Z","end":"2021-10-25T11:14:00.000Z"},"OPM":9,"total_sale_volume":459.8600025177002,"average_transact
ion_size":51.09555583530002,"rate_of_return":0.0}
{"window":{"start":"2021-10-25T11:13:00.000Z","end":"2021-10-25T11:14:00.000Z"},"OPM":9,"total_sale_volume":459.8600025177002,"average_transact
ion_size":51.09555583530002,"rate_of_return":0.0}
{"window":{"start":"2021-10-25T11:14:00.000Z","end":"2021-10-25T11:15:00.000Z"},"OPM":14,"total_sale_volume":483.4499945640564,"average_transac
tion_size":34.532142468861174,"rate_of_return":0.07142857142857142}
{"window":{"start":"2021-10-25T11:11:00.000Z","end":"2021-10-25T11:12:00.000Z"},"OPM":5,"total_sale_volume":396.11000061035156,"average_transac
tion_size":79.22200012207031,"rate_of_return":0.0}
{"window":{"start":"2021-10-25T11:20:00.000Z","end":"2021-10-25T11:21:00.000Z"},"OPM":5,"total_sale_volume":258.56000328063965,"average_transac
tion_size":51.71200065612793,"rate_of_return":0.2}
{"window":{"start":"2021-10-25T11:17:00.000Z","end":"2021-10-25T11:18:00.000Z"},"OPM":7,"total_sale_volume":1067.4699907302856,"average_transac
tion_size":152.49571296146937,"rate_of_return":0.0}
{"window":{"start":"2021-10-25T11:12:00.000Z","end":"2021-10-25T11:13:00.000Z"},"OPM":13,"total_sale_volume":468.60000121593475,"average_transa
ction_size":36.04615393968729,"rate_of_return":0.07692307692307693}
{"window":{"start":"2021-10-25T11:18:00.000Z","end":"2021-10-25T11:19:00.000Z"},"OPM":7,"total_sale_volume":1583.7199920415878,"average_transac
tion_size":226.24571314879827,"rate_of_return":0.0}
{"window":{"start":"2021-10-25T11:21:00.000Z","end":"2021-10-25T11:22:00.000Z"},"OPM":14,"total_sale_volume":668.9900054335594,"average_transac
tion_size":47.785000388111385,"rate_of_return":0.14285714285714285}
{"window":{"start":"2021-10-25T11:16:00.000Z","end":"2021-10-25T11:17:00.000Z"},"OPM":4,"total_sale_volume":78.17999869585037,"average_transact
ion_size":19.544999673962593,"rate_of_return":0.0}
{"window":{"start":"2021-10-25T11:25:00.000Z","end":"2021-10-25T11:26:00.000Z"},"OPM":15,"total_sale_volume":542.6700085401535,"average_transac
tion_size":36.17800056934357,"rate_of_return":0.13333333333333333}
{"window":{"start":"2021-10-25T11:23:00.000Z","end":"2021-10-25T11:24:00.000Z"},"OPM":15,"total_sale_volume":469.79999724030495,"average_transa
ction_size":31.31999981602033,"rate_of_return":0.06666666666666667}
{"window":{"start":"2021-10-25T11:22:00.000Z","end":"2021-10-25T11:23:00.000Z"},"OPM":12,"total_sale_volume":335.83000135421753,"average_transa
ction_size":27.985833446184795,"rate_of_return":0.0}
```

```
hadoop fs -cat /user/ec2-user/country_kpi/part*
```



```
[ec2-user@ip-10-0-0-71 ~]$ hadoop fs -cat /user/ec2-user/country_kpi/part*
{"window":{"start":"2021-10-25T11:20:00.000Z","end":"2021-10-25T11:21:00.000Z"},"country":"Unspecified","OPM":1,"total_sale_volume":-93.4300003
0517578,"rate_of_return":1.0}
{"window":{"start":"2021-10-25T11:26:00.000Z","end":"2021-10-25T11:27:00.000Z"},"country":"EIRE","OPM":1,"total_sale_volume":29.15999984741211,
"rate_of_return":0.0}
{"window":{"start":"2021-10-25T11:24:00.000Z","end":"2021-10-25T11:25:00.000Z"},"country":"Denmark","OPM":1,"total_sale_volume":157.77999877929
688,"rate_of_return":0.0}
{"window":{"start":"2021-10-25T11:21:00.000Z","end":"2021-10-25T11:22:00.000Z"},"country":"Channel Islands","OPM":1,"total_sale_volume":23.1299
991607666,"rate_of_return":0.0}
{"window":{"start":"2021-10-25T11:19:00.000Z","end":"2021-10-25T11:20:00.000Z"},"country":"United Kingdom","OPM":11,"total_sale_volume":428.760
0100636482,"rate_of_return":0.0}
{"window":{"start":"2021-10-25T11:23:00.000Z","end":"2021-10-25T11:24:00.000Z"},"country":"United Kingdom","OPM":15,"total_sale_volume":469.799
99724030495,"rate_of_return":0.06666666666666667}
{"window":{"start":"2021-10-25T11:25:00.000Z","end":"2021-10-25T11:26:00.000Z"},"country":"France","OPM":2,"total_sale_volume":30.7799991369247
44,"rate_of_return":0.0}
{"window":{"start":"2021-10-25T11:22:00.000Z","end":"2021-10-25T11:23:00.000Z"},"country":"Spain","OPM":1,"total_sale_volume":47.06999969482422
,"rate_of_return":0.0}
{"window":{"start":"2021-10-25T11:26:00.000Z","end":"2021-10-25T11:27:00.000Z"},"country":"United Kingdom","OPM":4,"total_sale_volume":228.5200
0045776367,"rate_of_return":0.0}
{"window":{"start":"2021-10-25T11:17:00.000Z","end":"2021-10-25T11:18:00.000Z"},"country":"Norway","OPM":1,"total_sale_volume":11.8999996185302
73,"rate_of_return":0.0}
{"window":{"start":"2021-10-25T11:12:00.000Z","end":"2021-10-25T11:13:00.000Z"},"country":"EIRE","OPM":1,"total_sale_volume":4.440000057220459,
"rate_of_return":0.0}
{"window":{"start":"2021-10-25T11:14:00.000Z","end":"2021-10-25T11:15:00.000Z"},"country":"United Kingdom","OPM":14,"total_sale_volume":483.449
9945640564,"rate_of_return":0.07142857142857142}
{"window":{"start":"2021-10-25T11:13:00.000Z","end":"2021-10-25T11:14:00.000Z"},"country":"Germany","OPM":1,"total_sale_volume":38.590000152587
89,"rate_of_return":0.0}
{"window":{"start":"2021-10-25T11:12:00.000Z","end":"2021-10-25T11:13:00.000Z"},"country":"United Kingdom","OPM":12,"total_sale_volume":464.160
0011587143,"rate_of_return":0.08333333333333333}
{"window":{"start":"2021-10-25T11:21:00.000Z","end":"2021-10-25T11:22:00.000Z"},"country":"United Kingdom","OPM":13,"total_sale_volume":645.860
0062727928,"rate_of_return":0.15384615384615385}
{"window":{"start":"2021-10-25T11:18:00.000Z","end":"2021-10-25T11:19:00.000Z"},"country":"United Kingdom","OPM":7,"total_sale_volume":1583.719
9920415878,"rate_of_return":0.0}
{"window":{"start":"2021-10-25T11:22:00.000Z","end":"2021-10-25T11:23:00.000Z"},"country":"United Kingdom","OPM":11,"total_sale_volume":288.760
0016593933,"rate_of_return":0.0}
{"window":{"start":"2021-10-25T11:13:00.000Z","end":"2021-10-25T11:14:00.000Z"},"country":"United Kingdom","OPM":7,"total_sale_volume":349.8700
008392334,"rate_of_return":0.0}
{"window":{"start":"2021-10-25T11:16:00.000Z","end":"2021-10-25T11:17:00.000Z"},"country":"United Kingdom","OPM":4,"total_sale_volume":78.17999
869585037,"rate_of_return":0.0}
{"window":{"start":"2021-10-25T11:17:00.000Z","end":"2021-10-25T11:18:00.000Z"},"country":"United Kingdom","OPM":6,"total_sale_volume":1055.569
9911117554,"rate_of_return":0.0}
{"window":{"start":"2021-10-25T11:24:00.000Z","end":"2021-10-25T11:25:00.000Z"},"country":"United Kingdom","OPM":10,"total_sale_volume":930.219
9963629246,"rate_of_return":0.0}
```

Transfer of files from CDH Instance on AWS to my system, using WinSCP

First, I needed to transfer the JSON files from HDFS into the the EC2 system

I created directories for time-based and then time-and-country-based KPIs as ec2-user. Using the ‘get’ command I copied the contents of the output folders into the EC2 system.

```
mkdir timebased-KPI
hadoop fs -get /user/ec2-user/time_kpi /home/ec2-user/timebased-KPI
```

```
mkdir country-and-timebased-KPI
hadoop fs -get /user/ec2-user/country_kpi /home/ec2-user/country-and-timebased-KPI
```

Thereafter I used WinSCP to establish a connection between the EC2 instance and my local file system to transfer all the required files into my system.