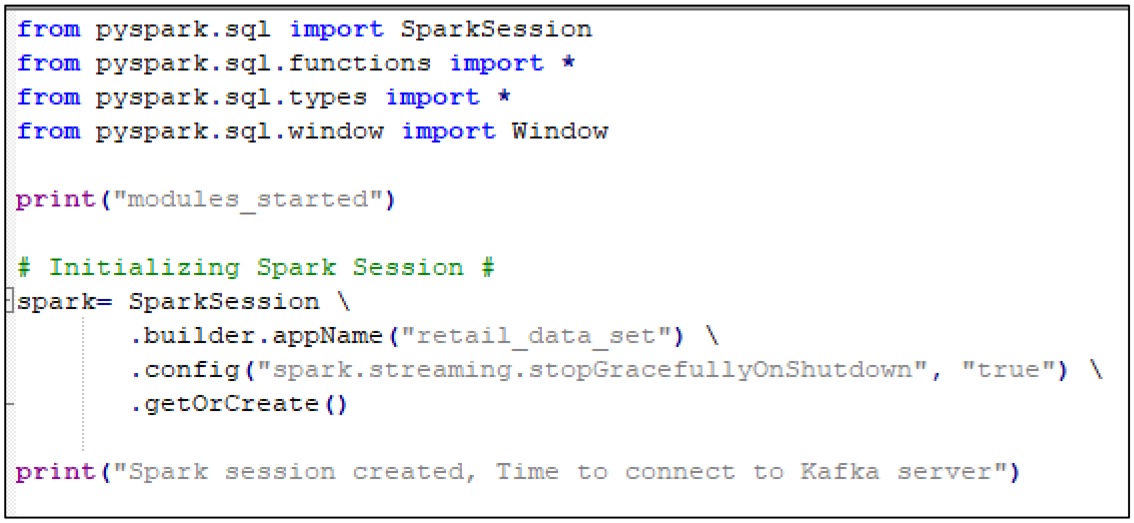
Code Logic - Retail Data Analysis

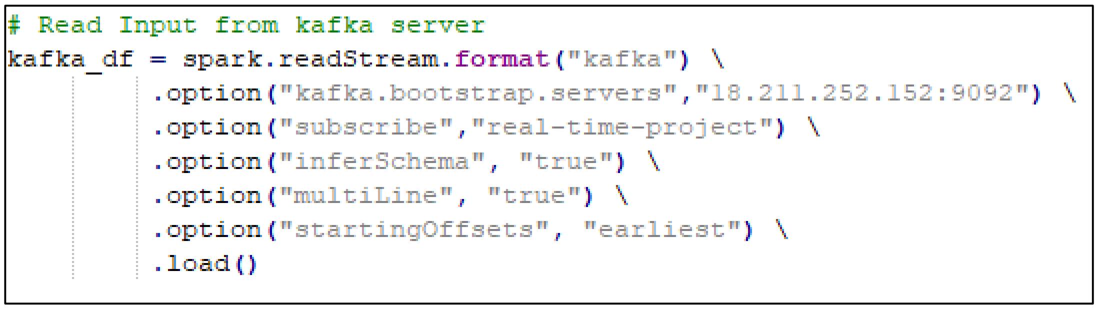
In this document, overall steps are mentioned to solve the assignment in the executable python file.

Step 1: Importing the required modules and creating spark session



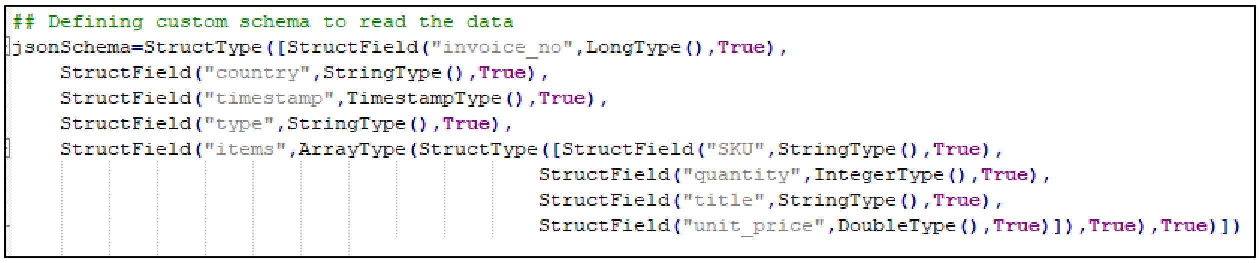
```python
from pyspark.sql import SparkSession
from pyspark.sql.functions import *
from pyspark.sql.types import *
from pyspark.sql.window import Window

print("modules_started")

# Initializing Spark Session #
spark= SparkSession \
        .builder.appName("retail_data_set") \
        .config("spark.streaming.stopGracefullyOnShutdown", "true") \
        .getOrCreate()

print("Spark session created, Time to connect to Kafka server")
```

Step 2: Reading data from Kafka server



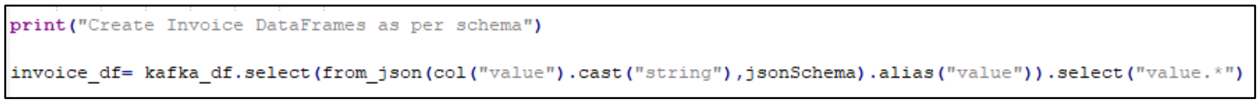
```python
# Read Input from kafka server
kafka_df = spark.readStream.format("kafka") \
        .option("kafka.bootstrap.servers","18.211.252.152:9092") \
        .option("subscribe","real-time-project") \
        .option("inferSchema", "true") \
        .option("multiLine", "true") \
        .option("startingOffsets", "earliest") \
        .load()
```

Step 3: Defining the schema for the json input data



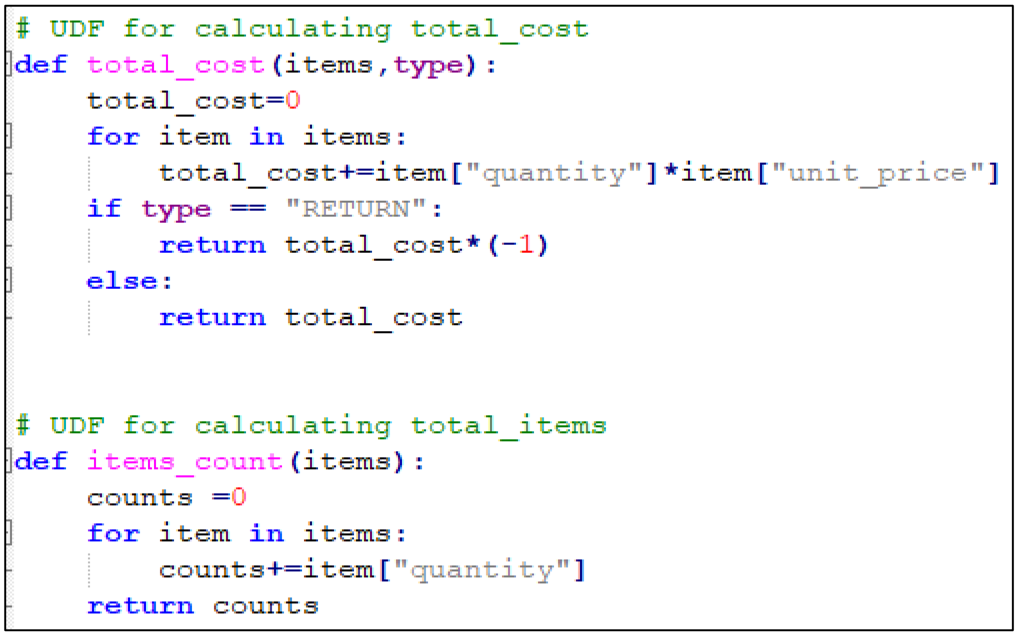
```python
## Defining custom schema to read the data
jsonSchema=StructType([StructField("invoice_no",LongType(),True),
    StructField("country",StringType(),True),
    StructField("timestamp",TimestampType(),True),
    StructField("type",StringType(),True),
    StructField("items",ArrayType(StructType([StructField("SKU",StringType(),True),
                                              StructField("quantity",IntegerType(),True),
                                              StructField("title",StringType(),True),
                                              StructField("unit_price",DoubleType(),True)]),True),True)])
```

Step 4: Defining data-frame for the data based on schema defined above

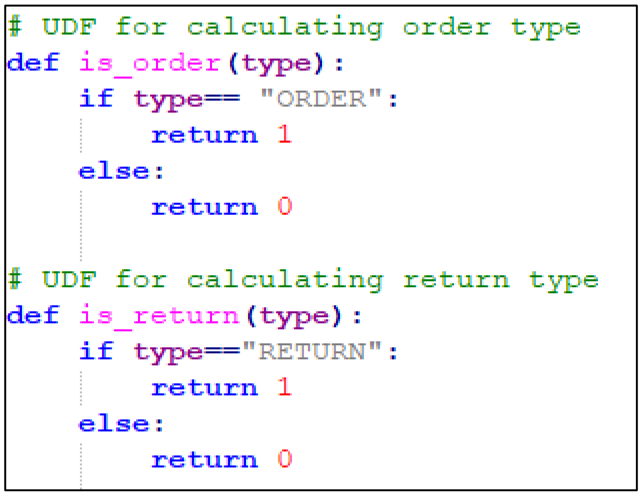


```
print("Create Invoice DataFrames as per schema")

invoice_df= kafka_df.select(from_json(col("value").cast("string"),jsonSchema).alias("value")).select("value.*")
```

Step 5: Calculating and defining the additional 4 columns (total_cost, items_count, is_order, is_return) using user defined functions(UDF's)



```
# UDF for calculating total_cost
def total_cost(items,type):
    total_cost=0
    for item in items:
        total_cost+=item["quantity"]*item["unit_price"]
    if type == "RETURN":
        return total_cost*(-1)
    else:
        return total_cost


# UDF for calculating total_items
def items_count(items):
    counts =0
    for item in items:
        counts+=item["quantity"]
    return counts
```



```
# UDF for calculating order type
def is_order(type):
    if type== "ORDER":
        return 1
    else:
        return 0

# UDF for calculating return type
def is_return(type):
    if type=="RETURN":
        return 1
    else:
        return 0
```

Step-6: Converting UDF’s to utilize the functions defining the datatype and adding the columns into data-frame

```
# Converting to UDF’s with the utility functions
totalcost= udf(total_cost,DoubleType())
totalitems = udf(items_count,IntegerType())
isorder = udf(is_order,IntegerType())
isreturn = udf(is_return,IntegerType())


print("Writing the addtional columns to kafka data stream")

invoice_df = invoice_df \
        .withColumn("total_cost", totalcost(invoice_df.items, invoice_df.type)) \
        .withColumn("total_items", totalitems(invoice_df.items)) \
        .withColumn("is_order", isorder(invoice_df.type)) \
        .withColumn("is_return", isreturn(invoice_df.type))
```

Step-7: Writing data to console with 1 Minute Interval

```
# Writing the Intermediary data into Console

retailstream = invoice_df \
       .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()
```

Step-8: Calculate time based KPI’s with watermark and grouping by window timestamp of 1minute and write stream data into json file

```
## Calculating time based KPI
timebasedKPIS = invoice_df \
            .withWatermark("timestamp","1 minute") \
            .groupby(window("timestamp", "1 minute", "1 minute")) \
            .agg(count("invoice_no").alias("OPM"),
                 sum("total_cost").alias("total_sales_volume"),
                 avg("total_cost").alias("average_transaction_size"),
                 avg("is_return").alias("rate_of_return")) \
            .select("window", "OPM", "total_sales_volume", "average_transaction_size", "rate_of_return")

# write stream data in json format for time based KPIs
timebasedKPIS_output = timebasedKPIS \
    .writeStream \
    .outputMode("Append") \
    .format("json") \
    .option("format","append") \
    .option("path", "time_KPI") \
    .option("checkpointLocation", "time-KPI") \
    .option("truncate", "False") \
    .trigger(processingTime="1 minute") \
    .start()
```

Step-9: Calculate time-country based KPI's with watermark and grouping by window timestamp of 1minute and country, write stream data into json file

```
# Calculating time and country-based KPIs

timecountryKPIS = invoice_df \
    .withWatermark("timestamp", "1 minute") \
    .groupby(window("timestamp", "1 minute", "1 minute"), "country") \
    .agg(count("invoice_no").alias("OPM"),
         sum("total_cost").alias("total_sales_volume"),
         avg("is_return").alias("rate_of_return")) \
    .select("window", "country", "OPM", "total_sales_volume", "rate_of_return")

## Write stream data in json format for time and country based KPIS
timecountryKPIS_output = timecountryKPIS \
    .writeStream \
    .outputMode("Append") \
    .format("json") \
    .option("format","append") \
    .option("truncate", "false") \
    .option("path", "timecountry_KPI") \
    .option("checkpointLocation","time-country-KPI") \
    .trigger(processingTime="1 minute") \
    .start()
```

Step-10: Waiting for the stream to write data infinitely

```
## Spark to await termination
retailstream.awaitTermination()
timebasedKPIS_output.awaitTermination()
timecountryKPIS_output.awaitTermination()
```

Console Commands

Spark Submit Command to execute the python file for data streaming and writing in json files

"spark-submit --packages org.apache.spark:spark-sql-kafka-0-10_2.11:2.4.5 spark-streaming.py 18.211.252.152 9092 real-time-project "

Using 'mkdir' command to make directories to copy files from hadoop

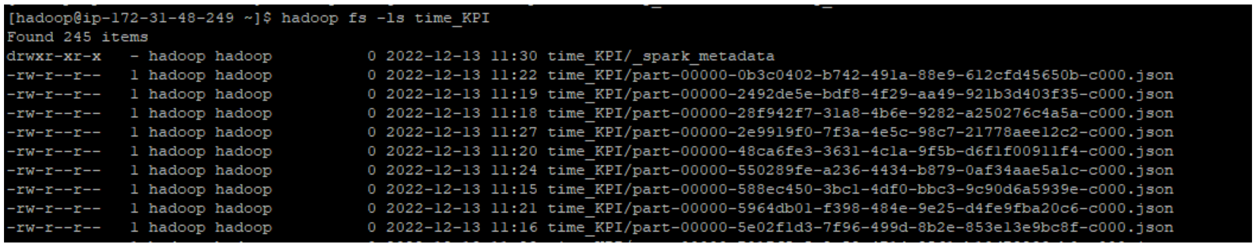
"mkdir time_KPI"
"mkdir timecountry_KPI "

Command to check json files generated as checked in below snapshots

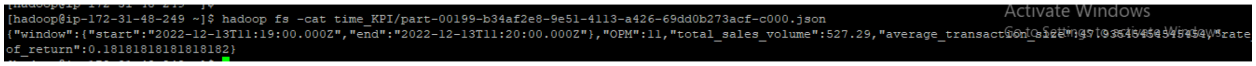
hadoop fs -ls
hadoop fs -ls time_KPI
hadoop fs -ls timecountry_KPI

hadoop fs –cat time_KPI/<*.json file>

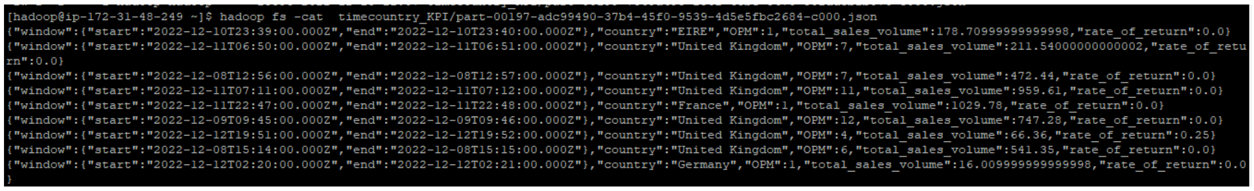
hadoop fs –cat timecountry_KPI/<*.json file>



```
[hadoop@ip-172-31-48-249 ~]$ hadoop fs -ls time_KPI
Found 245 items
drwxr-xr-x   - hadoop hadoop          0 2022-12-13 11:30 time_KPI/_spark_metadata
-rw-r--r--   1 hadoop hadoop          0 2022-12-13 11:22 time_KPI/part-00000-0b3c0402-b742-491a-88e9-612cfd45650b-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-12-13 11:19 time_KPI/part-00000-2492de5e-bdf8-4f29-aa49-921b3d403f35-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-12-13 11:18 time_KPI/part-00000-28f942f7-31a8-4b6e-9282-a250276c4a5a-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-12-13 11:27 time_KPI/part-00000-2e9919f0-7f3a-4e5c-98c7-21778aee12c2-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-12-13 11:20 time_KPI/part-00000-48ca6fe3-3631-4c1a-9f5b-d6f1f00911f4-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-12-13 11:24 time_KPI/part-00000-550289fe-a236-4434-b879-0af34aae5a1c-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-12-13 11:15 time_KPI/part-00000-588ec450-3bc1-4df0-bbc3-9c90d6a5939e-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-12-13 11:21 time_KPI/part-00000-5964db01-f398-484e-9e25-d4fe9fba20c6-c000.json
-rw-r--r--   1 hadoop hadoop          0 2022-12-13 11:16 time_KPI/part-00000-5e02f1d3-7f96-499d-8b2e-853e13e9bc8f-c000.json
```

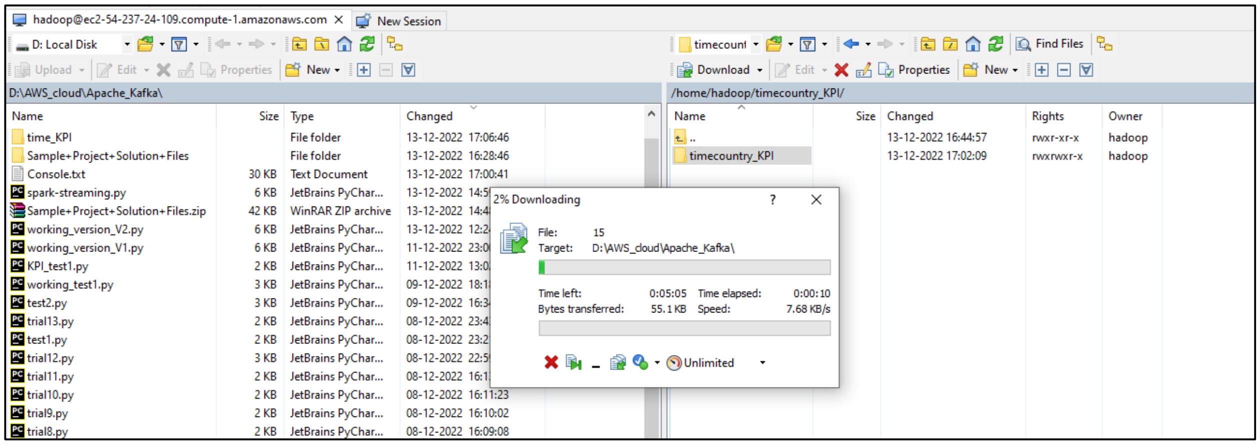


```
[hadoop@ip-172-31-48-249 ~]$ hadoop fs -cat time_KPI/part-00199-b34af2e8-9e51-4113-a426-69dd0b273acf-c000.json
{"window":{"start":"2022-12-13T11:19:00.000Z","end":"2022-12-13T11:20:00.000Z"},"OPM":11,"total_sales_volume":527.29,"average_transaction_size":[illegible]"rate_
of_return":0.18181818181818182}
```



```
[hadoop@ip-172-31-48-249 ~]$ hadoop fs -cat  timecountry_KPI/part-00197-adc99490-37b4-45f0-9539-4d5e5fbc2684-c000.json
{"window":{"start":"2022-12-10T23:39:00.000Z","end":"2022-12-10T23:40:00.000Z"},"country":"EIRE","OPM":1,"total_sales_volume":178.70999999999998,"rate_of_return":0.0}
{"window":{"start":"2022-12-11T06:50:00.000Z","end":"2022-12-11T06:51:00.000Z"},"country":"United Kingdom","OPM":7,"total_sales_volume":211.54000000000002,"rate_of_retu
rn":0.0}
{"window":{"start":"2022-12-08T12:56:00.000Z","end":"2022-12-08T12:57:00.000Z"},"country":"United Kingdom","OPM":7,"total_sales_volume":472.44,"rate_of_return":0.0}
{"window":{"start":"2022-12-11T07:11:00.000Z","end":"2022-12-11T07:12:00.000Z"},"country":"United Kingdom","OPM":11,"total_sales_volume":959.61,"rate_of_return":0.0}
{"window":{"start":"2022-12-11T22:47:00.000Z","end":"2022-12-11T22:48:00.000Z"},"country":"France","OPM":1,"total_sales_volume":1029.78,"rate_of_return":0.0}
{"window":{"start":"2022-12-09T09:45:00.000Z","end":"2022-12-09T09:46:00.000Z"},"country":"United Kingdom","OPM":12,"total_sales_volume":747.28,"rate_of_return":0.0}
{"window":{"start":"2022-12-12T19:51:00.000Z","end":"2022-12-12T19:52:00.000Z"},"country":"United Kingdom","OPM":4,"total_sales_volume":66.36,"rate_of_return":0.25}
{"window":{"start":"2022-12-08T15:14:00.000Z","end":"2022-12-08T15:15:00.000Z"},"country":"United Kingdom","OPM":6,"total_sales_volume":541.35,"rate_of_return":0.0}
{"window":{"start":"2022-12-12T02:20:00.000Z","end":"2022-12-12T02:21:00.000Z"},"country":"Germany","OPM":1,"total_sales_volume":16.009999999999998,"rate_of_return":0.0
}
```

Copying json files from hadoop to local directory



hadoop@ec2-54-237-24-109.compute-1.amazonaws.com

D:\AWS_cloud\Apache_Kafka\

Name	Size	Type	Changed
time_KPI		File folder	13-12-2022 17:06:46
Sample+Project+Solution+Files		File folder	13-12-2022 16:28:46
Console.txt	30 KB	Text Document	13-12-2022 17:00:41
spark-streaming.py	6 KB	JetBrains PyChar...	13-12-2022 14:5
Sample+Project+Solution+Files.zip	42 KB	WinRAR ZIP archive	13-12-2022 14:4
working_version_V2.py	6 KB	JetBrains PyChar...	13-12-2022 12:2
working_version_V1.py	6 KB	JetBrains PyChar...	11-12-2022 23:0
KPI_test1.py	2 KB	JetBrains PyChar...	11-12-2022 13:0
working_test1.py	3 KB	JetBrains PyChar...	09-12-2022 18:1
test2.py	3 KB	JetBrains PyChar...	09-12-2022 16:3
trial13.py	2 KB	JetBrains PyChar...	08-12-2022 23:4
test1.py	2 KB	JetBrains PyChar...	08-12-2022 23:2
trial12.py	3 KB	JetBrains PyChar...	08-12-2022 22:5
trial11.py	2 KB	JetBrains PyChar...	08-12-2022 16:1
trial10.py	2 KB	JetBrains PyChar...	08-12-2022 16:11:23
trial9.py	2 KB	JetBrains PyChar...	08-12-2022 16:10:02
trial8.py	2 KB	JetBrains PyChar...	08-12-2022 16:09:08

/home/hadoop/timecountry_KPI/

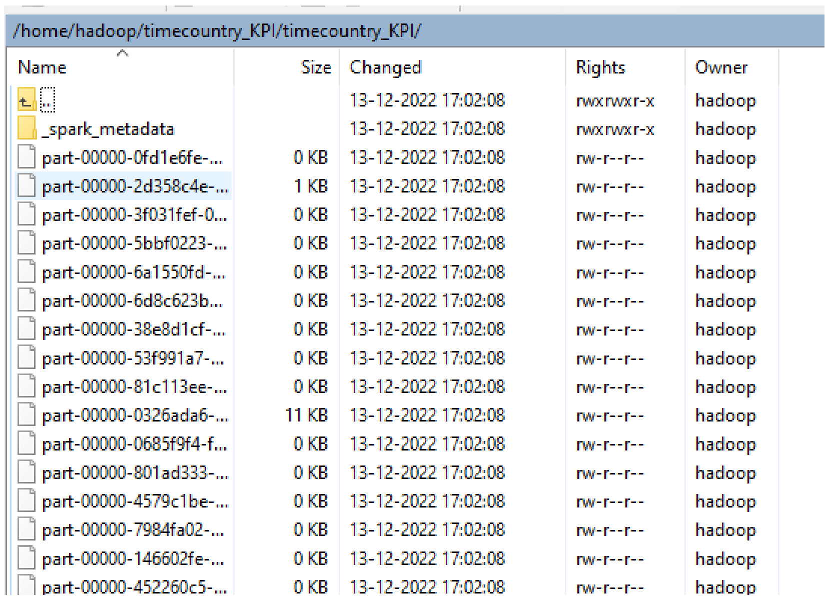
Name	Size	Changed	Rights	Owner
..		13-12-2022 16:44:57	rwxr-xr-x	hadoop
timecountry_KPI		13-12-2022 17:02:09	rwxrwxr-x	hadoop

2% Downloading

File: 15
Target: D:\AWS_cloud\Apache_Kafka\

Time left: 0:05:05 Time elapsed: 0:00:10
Bytes transferred: 55.1 KB Speed: 7.68 KB/s

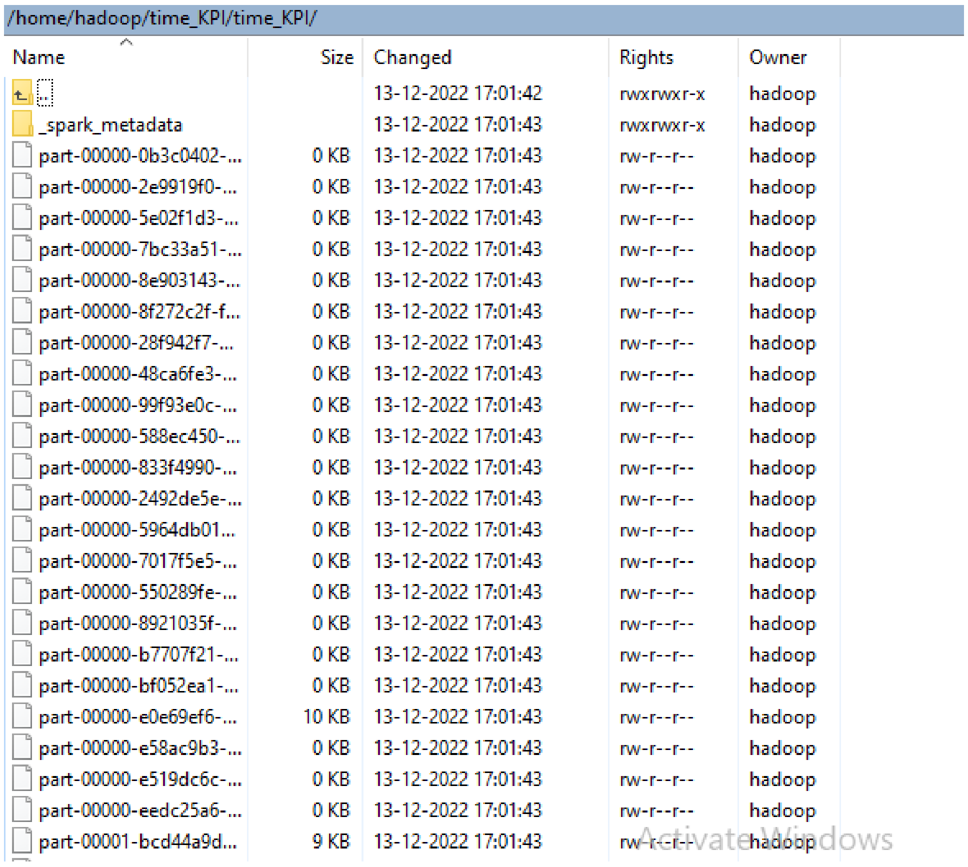
Time based KPI json files



/home/hadoop/timecountry_KPI/timecountry_KPI/

Name	Size	Changed	Rights	Owner
..		13-12-2022 17:02:08	rwxrwxr-x	hadoop
_spark_metadata		13-12-2022 17:02:08	rwxrwxr-x	hadoop
part-00000-0fd1e6fe-...	0 KB	13-12-2022 17:02:08	rw-r--r--	hadoop
part-00000-2d358c4e-...	1 KB	13-12-2022 17:02:08	rw-r--r--	hadoop
part-00000-3f031fef-0...	0 KB	13-12-2022 17:02:08	rw-r--r--	hadoop
part-00000-5bbf0223-...	0 KB	13-12-2022 17:02:08	rw-r--r--	hadoop
part-00000-6a1550fd-...	0 KB	13-12-2022 17:02:08	rw-r--r--	hadoop
part-00000-6d8c623b...	0 KB	13-12-2022 17:02:08	rw-r--r--	hadoop
part-00000-38e8d1cf-...	0 KB	13-12-2022 17:02:08	rw-r--r--	hadoop
part-00000-53f991a7-...	0 KB	13-12-2022 17:02:08	rw-r--r--	hadoop
part-00000-81c113ee-...	0 KB	13-12-2022 17:02:08	rw-r--r--	hadoop
part-00000-0326ada6-...	11 KB	13-12-2022 17:02:08	rw-r--r--	hadoop
part-00000-0685f9f4-f...	0 KB	13-12-2022 17:02:08	rw-r--r--	hadoop
part-00000-801ad333-...	0 KB	13-12-2022 17:02:08	rw-r--r--	hadoop
part-00000-4579c1be-...	0 KB	13-12-2022 17:02:08	rw-r--r--	hadoop
part-00000-7984fa02-...	0 KB	13-12-2022 17:02:08	rw-r--r--	hadoop
part-00000-146602fe-...	0 KB	13-12-2022 17:02:08	rw-r--r--	hadoop
part-00000-452260c5-...	0 KB	13-12-2022 17:02:08	rw-r--r--	hadoop

Time and country based KPI json files



/home/hadoop/time_KPI/time_KPI/

Name	Size	Changed	Rights	Owner
..		13-12-2022 17:01:42	rwxrwxr-x	hadoop
_spark_metadata		13-12-2022 17:01:43	rwxrwxr-x	hadoop
part-00000-0b3c0402-...	0 KB	13-12-2022 17:01:43	rw-r--r--	hadoop
part-00000-2e9919f0-...	0 KB	13-12-2022 17:01:43	rw-r--r--	hadoop
part-00000-5e02f1d3-...	0 KB	13-12-2022 17:01:43	rw-r--r--	hadoop
part-00000-7bc33a51-...	0 KB	13-12-2022 17:01:43	rw-r--r--	hadoop
part-00000-8e903143-...	0 KB	13-12-2022 17:01:43	rw-r--r--	hadoop
part-00000-8f272c2f-f...	0 KB	13-12-2022 17:01:43	rw-r--r--	hadoop
part-00000-28f942f7-...	0 KB	13-12-2022 17:01:43	rw-r--r--	hadoop
part-00000-48ca6fe3-...	0 KB	13-12-2022 17:01:43	rw-r--r--	hadoop
part-00000-99f93e0c-...	0 KB	13-12-2022 17:01:43	rw-r--r--	hadoop
part-00000-588ec450-...	0 KB	13-12-2022 17:01:43	rw-r--r--	hadoop
part-00000-833f4990-...	0 KB	13-12-2022 17:01:43	rw-r--r--	hadoop
part-00000-2492de5e-...	0 KB	13-12-2022 17:01:43	rw-r--r--	hadoop
part-00000-5964db01...	0 KB	13-12-2022 17:01:43	rw-r--r--	hadoop
part-00000-7017f5e5-...	0 KB	13-12-2022 17:01:43	rw-r--r--	hadoop
part-00000-550289fe-...	0 KB	13-12-2022 17:01:43	rw-r--r--	hadoop
part-00000-8921035f-...	0 KB	13-12-2022 17:01:43	rw-r--r--	hadoop
part-00000-b7707f21-...	0 KB	13-12-2022 17:01:43	rw-r--r--	hadoop
part-00000-bf052ea1-...	0 KB	13-12-2022 17:01:43	rw-r--r--	hadoop
part-00000-e0e69ef6-...	10 KB	13-12-2022 17:01:43	rw-r--r--	hadoop
part-00000-e58ac9b3-...	0 KB	13-12-2022 17:01:43	rw-r--r--	hadoop
part-00000-e519dc6c-...	0 KB	13-12-2022 17:01:43	rw-r--r--	hadoop
part-00000-eedc25a6-...	0 KB	13-12-2022 17:01:43	rw-r--r--	hadoop
part-00001-bcd44a9d...	9 KB	13-12-2022 17:01:43	rw-r--r--	hadoop