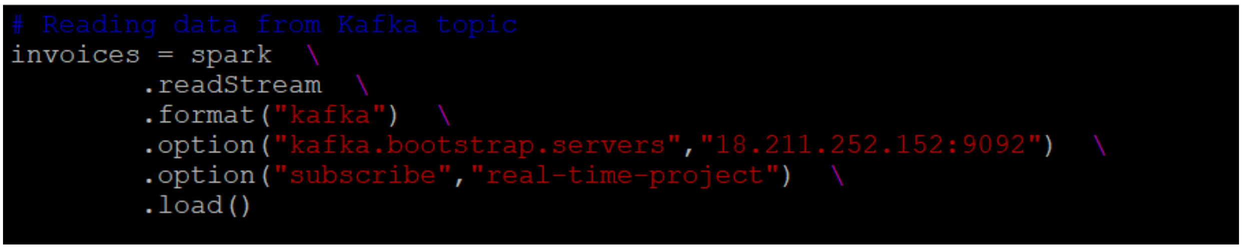
Retail Data Analysis Using Spark Streaming and Kakfa

Create Python script to read the data streams from Kafka topic and process the input data streams into the resultant JSON files

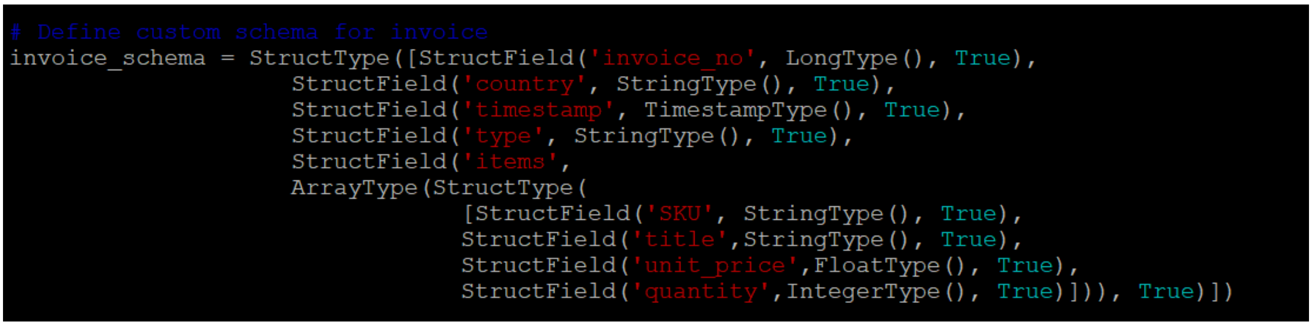
spark-streaming.py

Import the necessary dependencies and initialise the Spark Session and read the data from Kafka topic (by providing bootstrap server, port and topic name)



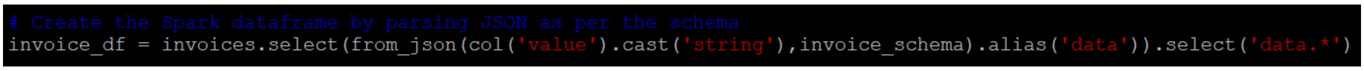
```
# Reading data from Kafka topic
invoices = spark  \
        .readStream  \
        .format("kafka")  \
        .option("kafka.bootstrap.servers","18.211.252.152:9092")  \
        .option("subscribe","real-time-project")  \
        .load()
```

Define a custom schema for the input data stream



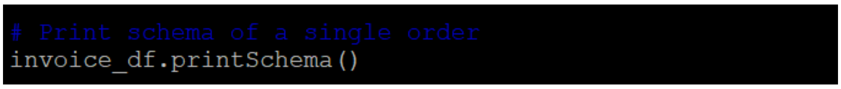
```
# Define custom schema for invoice
invoice_schema = StructType([StructField('invoice_no', LongType(), True),
                    StructField('country', StringType(), True),
                    StructField('timestamp', TimestampType(), True),
                    StructField('type', StringType(), True),
                    StructField('items',
                    ArrayType(StructType(
                                [StructField('SKU', StringType(), True),
                                StructField('title',StringType(), True),
                                StructField('unit_price',FloatType(), True),
                                StructField('quantity',IntegerType(), True)])), True)])
```

Parse the raw JSON data as per the schema and store the into ***invoice_df*** dataframe



```
# Create the Spark dataframe by parsing JSON as per the schema
invoice_df = invoices.select(from_json(col('value').cast('string'),invoice_schema).alias('data')).select('data.*')
```

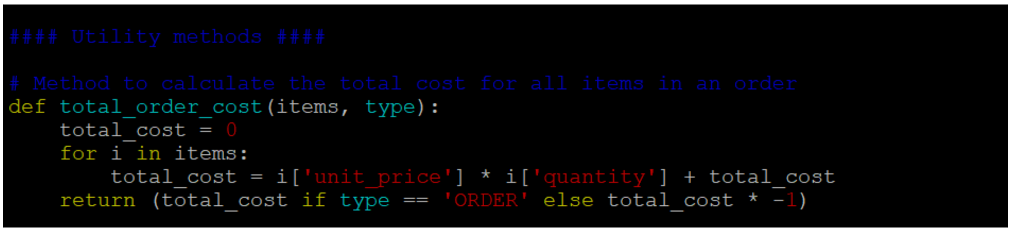
Print schema of a single order invoice



```
# Print schema of a single order
invoice_df.printSchema()
```

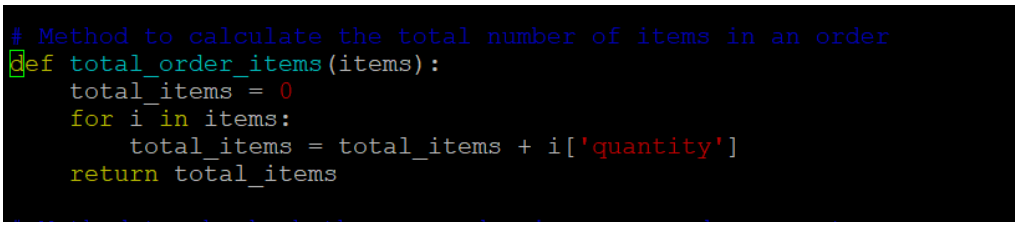
Write utility methods to calculate the additional columns:

- **total_order_cost ():** Calculate the total cost for all items in an order (unit_price * quantity)



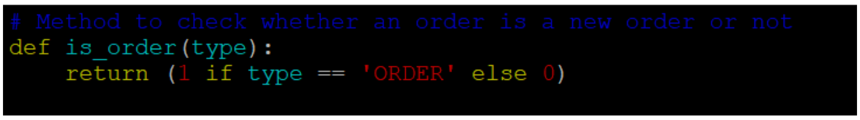
```
#### Utility methods ####

# Method to calculate the total cost for all items in an order
def total_order_cost(items, type):
    total_cost = 0
    for i in items:
        total_cost = i['unit_price'] * i['quantity'] + total_cost
    return (total_cost if type == 'ORDER' else total_cost * -1)
```

- **total_order_items ():** Calculate the total number of items in an order by adding the quantity



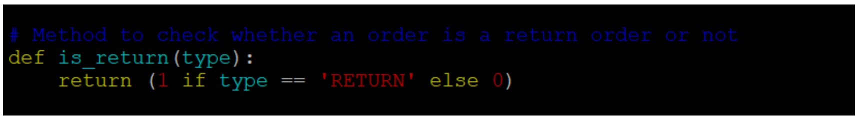
```
# Method to calculate the total number of items in an order
def total_order_items(items):
    total_items = 0
    for i in items:
        total_items = total_items + i['quantity']
    return total_items
```

- **is_order ():** Return 1 if the order is a new order



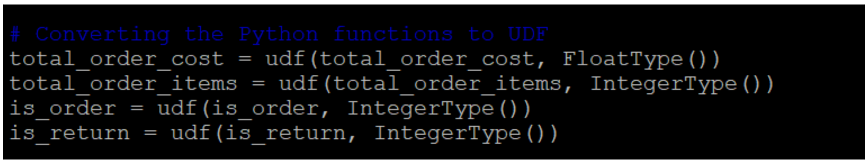
```
# Method to check whether an order is a new order or not
def is_order(type):
    return (1 if type == 'ORDER' else 0)
```

- **is_return ():** Return 1 if the order is a return



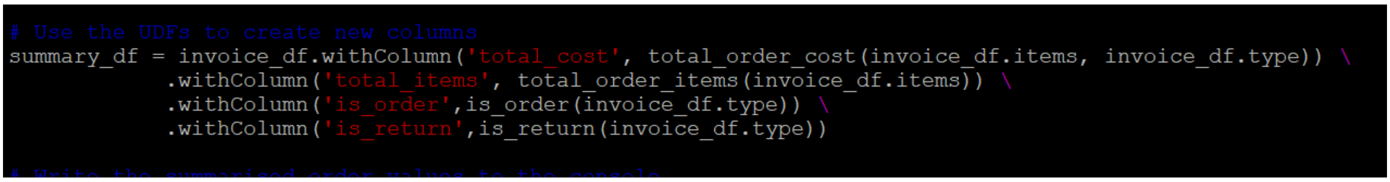
```
# Method to check whether an order is a return order or not
def is_return(type):
    return (1 if type == 'RETURN' else 0)
```

Converting the Python functions to UDF



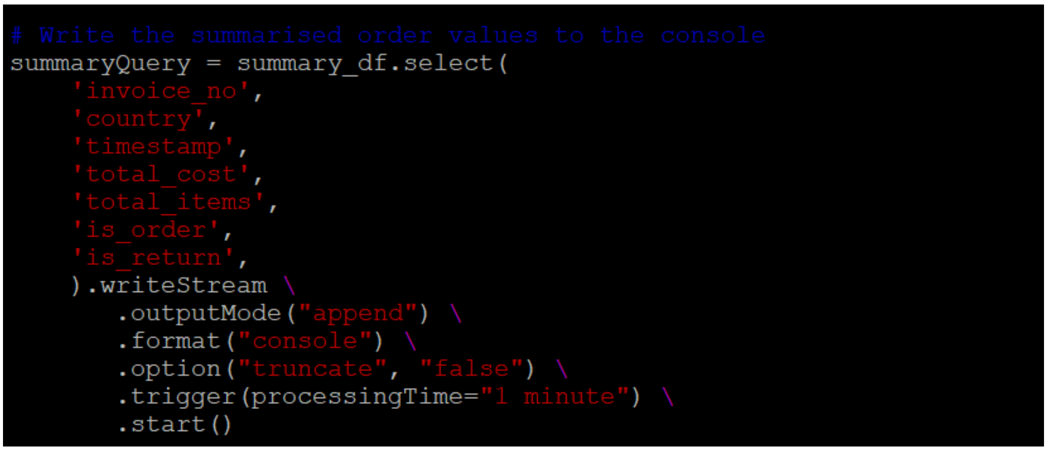
```
# Converting the Python functions to UDF
total_order_cost = udf(total_order_cost, FloatType())
total_order_items = udf(total_order_items, IntegerType())
is_order = udf(is_order, IntegerType())
is_return = udf(is_return, IntegerType())
```

Use the UDFs to create 4 new columns: total_cost, total_items, is_order, is_return and store in ***summary_df*** dataframe



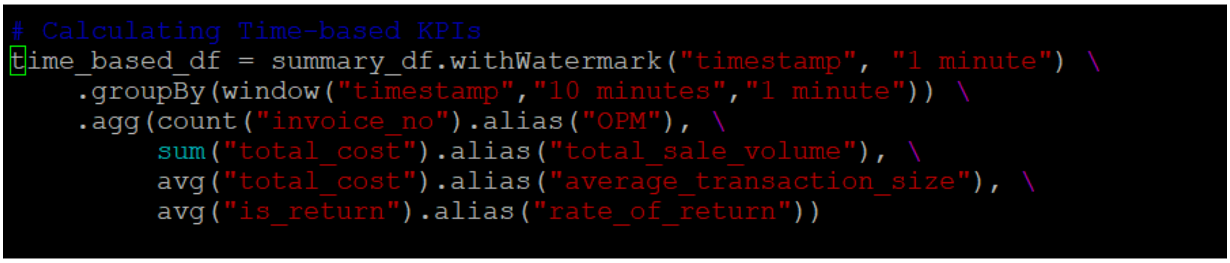
```
# Use the UDFs to create new columns
summary_df = invoice_df.withColumn('total_cost', total_order_cost(invoice_df.items, invoice_df.type)) \
            .withColumn('total_items', total_order_items(invoice_df.items)) \
            .withColumn('is_order',is_order(invoice_df.type)) \
            .withColumn('is_return',is_return(invoice_df.type))

# Write the summarised order values to the console
```

Write the summarised input values to console for each one-minute window



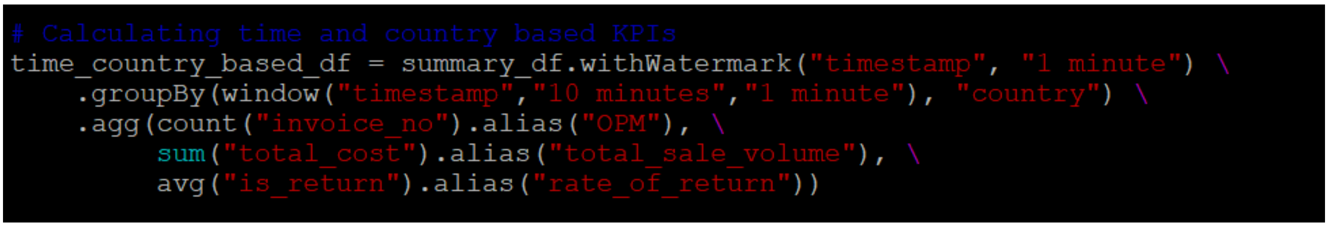
```
# Write the summarised order values to the console
summaryQuery = summary_df.select(
    'invoice_no',
    'country',
    'timestamp',
    'total_cost',
    'total_items',
    'is_order',
    'is_return',
    ).writeStream \
       .outputMode("append") \
       .format("console") \
       .option("truncate", "false") \
       .trigger(processingTime="1 minute") \
       .start()
```

Calculate the time-based KPIs with tumbling window of one minute on orders and store in ***time_based_df*** dataframe



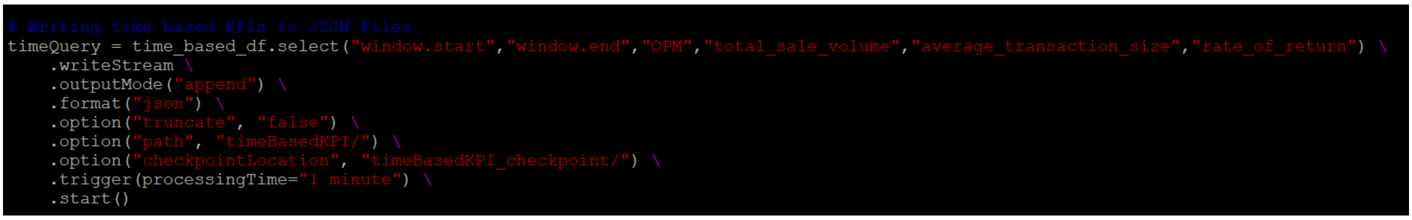
```
# Calculating Time-based KPIs
time_based_df = summary_df.withWatermark("timestamp", "1 minute") \
    .groupBy(window("timestamp","10 minutes","1 minute")) \
    .agg(count("invoice_no").alias("OPM"), \
         sum("total_cost").alias("total_sale_volume"), \
         avg("total_cost").alias("average_transaction_size"), \
         avg("is_return").alias("rate_of_return"))
```

Calculate the time-and country-based KPIs with tumbling window of one minute on orders and store in ***time_country_based_df*** dataframe.



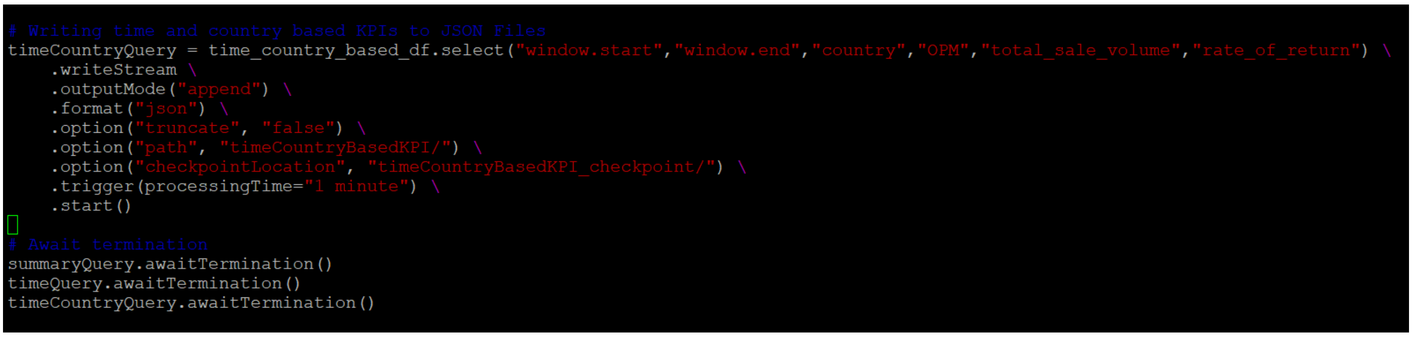
```
# Calculating time and country based KPIs
time_country_based_df = summary_df.withWatermark("timestamp", "1 minute") \
    .groupBy(window("timestamp","10 minutes","1 minute"), "country") \
    .agg(count("invoice_no").alias("OPM"), \
         sum("total_cost").alias("total_sale_volume"), \
         avg("is_return").alias("rate_of_return"))
```

Writing all the time-based KPIs to JSON files at HDFS directory: timeBasedKPI/



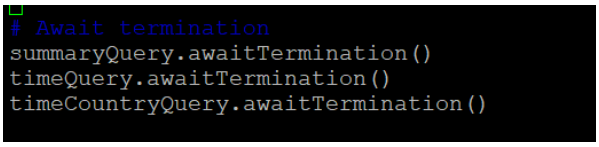
```
# Writing time based KPIs to JSON Files
timeQuery = time_based_df.select("window.start","window.end","OPM","total_sale_volume","average_transaction_size","rate_of_return") \
    .writeStream \
    .outputMode("append") \
    .format("json") \
    .option("truncate", "false") \
    .option("path", "timeBasedKPI/") \
    .option("checkpointLocation", "timeBasedKPI_checkpoint/") \
    .trigger(processingTime="1 minute") \
    .start()
```

Writing all the time-and country-based KPIs to JSON files at HDFS directory: timeCountryBasedKPI/



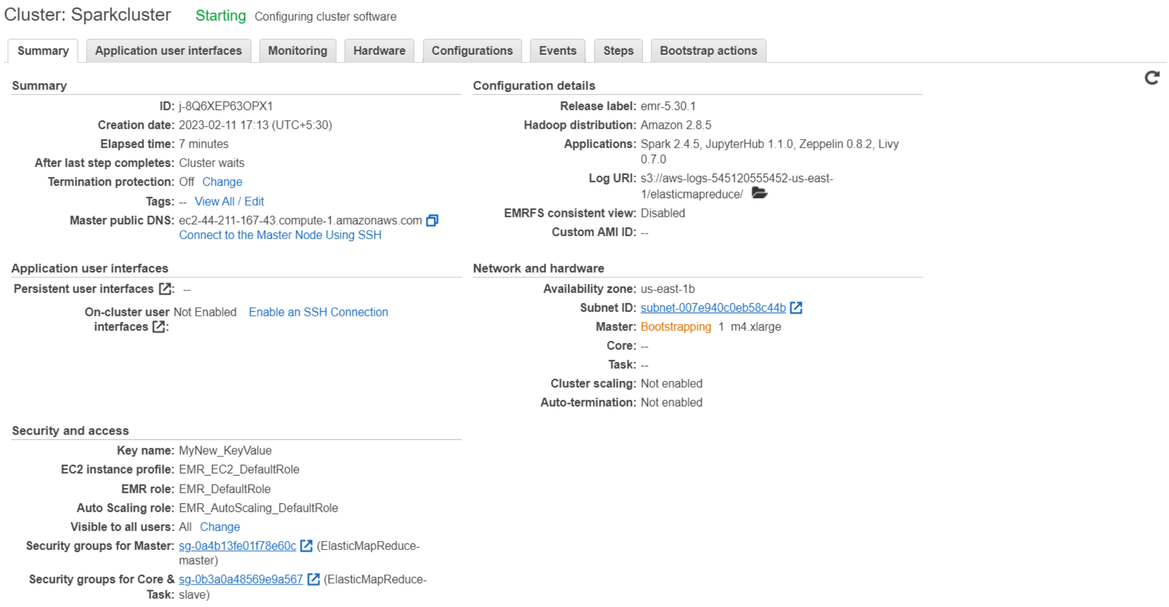
```
# Writing time and country based KPIs to JSON Files
timeCountryQuery = time_country_based_df.select("window.start","window.end","country","OPM","total_sale_volume","rate_of_return") \
    .writeStream \
    .outputMode("append") \
    .format("json") \
    .option("truncate", "false") \
    .option("path", "timeCountryBasedKPI/") \
    .option("checkpointLocation", "timeCountryBasedKPI_checkpoint/") \
    .trigger(processingTime="1 minute") \
    .start()

# Await termination
summaryQuery.awaitTermination()
timeQuery.awaitTermination()
timeCountryQuery.awaitTermination()
```

Await termination



```
# Await termination
summaryQuery.awaitTermination()
timeQuery.awaitTermination()
timeCountryQuery.awaitTermination()
```

Spark EMR Cluster Configuration



Cluster: Sparkcluster Starting Configuring cluster software

Summary

ID: j-8Q6XEP63OPX1
Creation date: 2023-02-11 17:13 (UTC+5:30)
Elapsed time: 7 minutes
After last step completes: Cluster waits
Termination protection: Off Change
Tags: -- View All / Edit
Master public DNS: ec2-44-211-167-43.compute-1.amazonaws.com
Connect to the Master Node Using SSH

Configuration details

Release label: emr-5.30.1
Hadoop distribution: Amazon 2.8.5
Applications: Spark 2.4.5, JupyterHub 1.1.0, Zeppelin 0.8.2, Livy 0.7.0
Log URI: s3://aws-logs-545120555452-us-east-1/elasticmapreduce/
EMRFS consistent view: Disabled
Custom AMI ID: --

Application user interfaces

Persistent user interfaces: --
On-cluster user interfaces: Not Enabled Enable an SSH Connection

Network and hardware

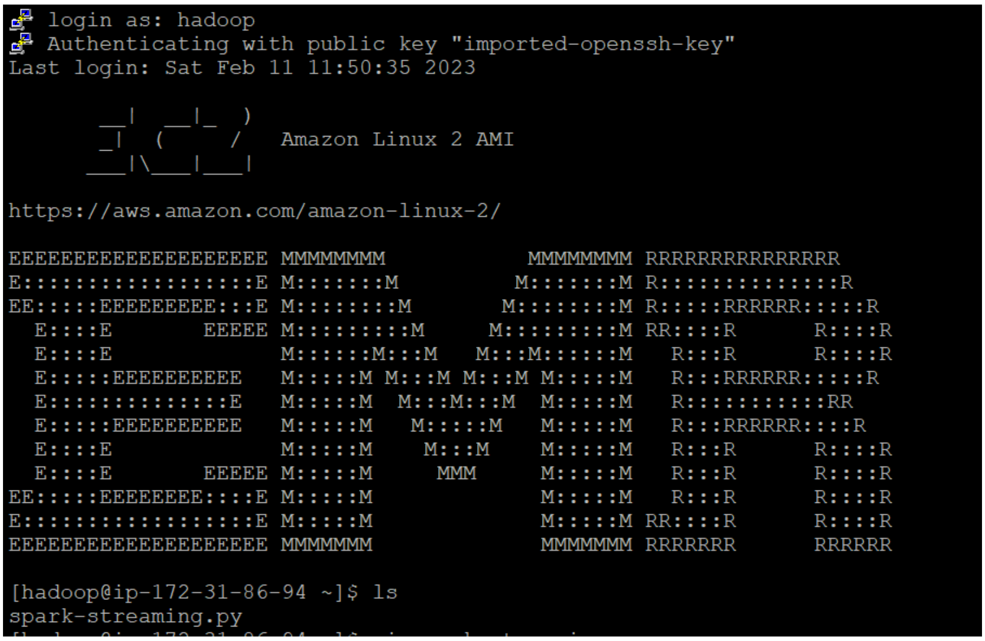
Availability zone: us-east-1b
Subnet ID: subnet-007e940c0eb58c44b
Master: Bootstrapping 1 m4.xlarge
Core: --
Task: --
Cluster scaling: Not enabled
Auto-termination: Not enabled

Security and access

Key name: MyNew_KeyValue
EC2 instance profile: EMR_EC2_DefaultRole
EMR role: EMR_DefaultRole
Auto Scaling role: EMR_AutoScaling_DefaultRole
Visible to all users: All Change
Security groups for Master: sg-0a4b13fe01f78e60c (ElasticMapReduce-master)
Security groups for Core & Task: sg-0b3a0a48569e9a567 (ElasticMapReduce-slave)

SSH into the EMR shell

Transfer the spark-streaming.py using WinSCP to local EMR master node.

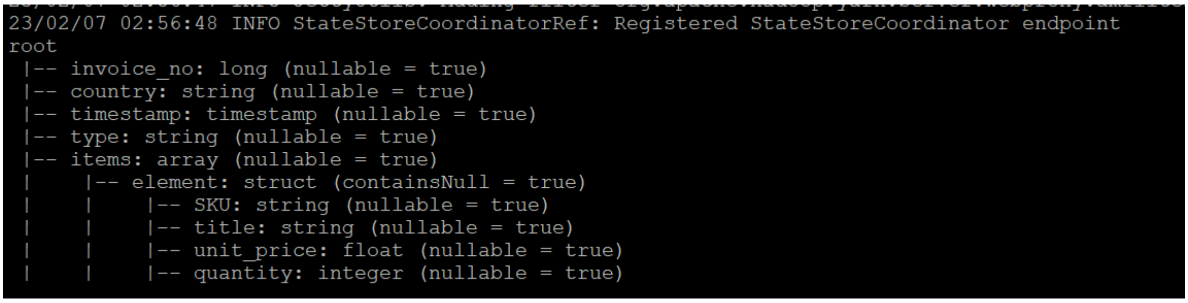


```
login as: hadoop
Authenticating with public key "imported-openssh-key"
Last login: Sat Feb 11 11:50:35 2023

       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/

EEEEEEEEEEEEEEEEEEEE MMMMMMMM           MMMMMMMM RRRRRRRRRRRRRRR
E::::::::::::::::::E M:::::::M         M:::::::M R::::::::::::::R
EE:::::EEEEEEEEE:::E M::::::::M       M::::::::M R:::::RRRRRR:::::R
  E::::E       EEEEE M:::::::::M     M:::::::::M RR::::R      R::::R
  E::::E             M::::::M:::M   M:::M::::::M   R:::R      R::::R
  E:::::EEEEEEEEEE   M:::::M M:::M M:::M M:::::M   R:::RRRRRR:::::R
  E::::::::::::::E   M:::::M  M:::M:::M  M:::::M   R:::::::::::RR
  E:::::EEEEEEEEEE   M:::::M   M:::::M   M:::::M   R:::RRRRRR::::R
  E::::E             M:::::M    M:::M    M:::::M   R:::R      R::::R
  E::::E       EEEEE M:::::M     MMM     M:::::M   R:::R      R::::R
EE:::::EEEEEEEE::::E M:::::M             M:::::M   R:::R      R::::R
E::::::::::::::::::E M:::::M             M:::::M RR::::R      R::::R
EEEEEEEEEEEEEEEEEEEE MMMMMMM             MMMMMMM RRRRRRR      RRRRRR

[hadoop@ip-172-31-86-94 ~]$ ls
spark-streaming.py
```

Spark Submit command to read the data from Kakfa topic

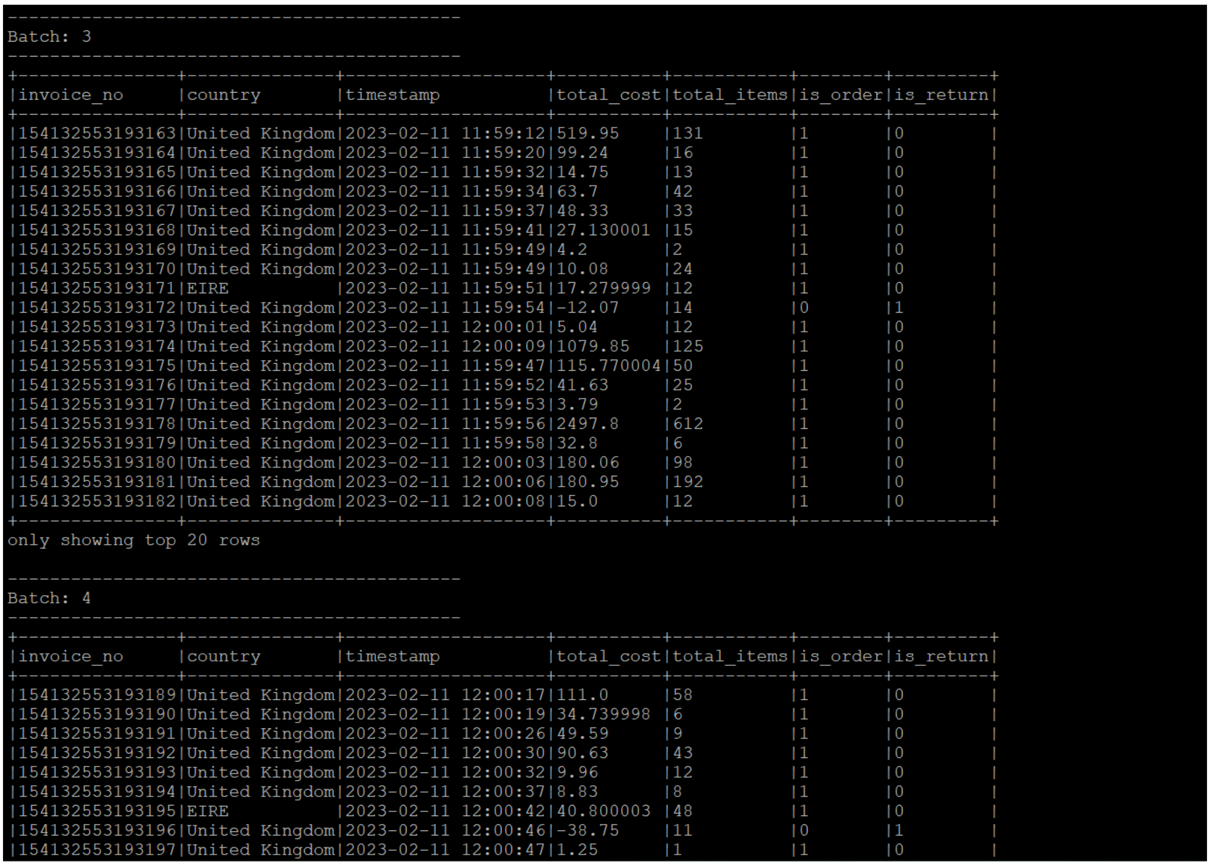
```
export SPARK_KAFKA_VERSION=0.10
spark-submit --packages org.apache.spark:spark-sql-kafka-0-10_2.11:2.4.5 spark-streaming.py
```

Screenshot of schema for a single order



```
23/02/07 02:56:48 INFO StateStoreCoordinatorRef: Registered StateStoreCoordinator endpoint
root
 |-- invoice_no: long (nullable = true)
 |-- country: string (nullable = true)
 |-- timestamp: timestamp (nullable = true)
 |-- type: string (nullable = true)
 |-- items: array (nullable = true)
 |    |-- element: struct (containsNull = true)
 |    |    |-- SKU: string (nullable = true)
 |    |    |-- title: string (nullable = true)
 |    |    |-- unit_price: float (nullable = true)
 |    |    |-- quantity: integer (nullable = true)
```

Screenshot of final summarised values written to the console (for a single order)

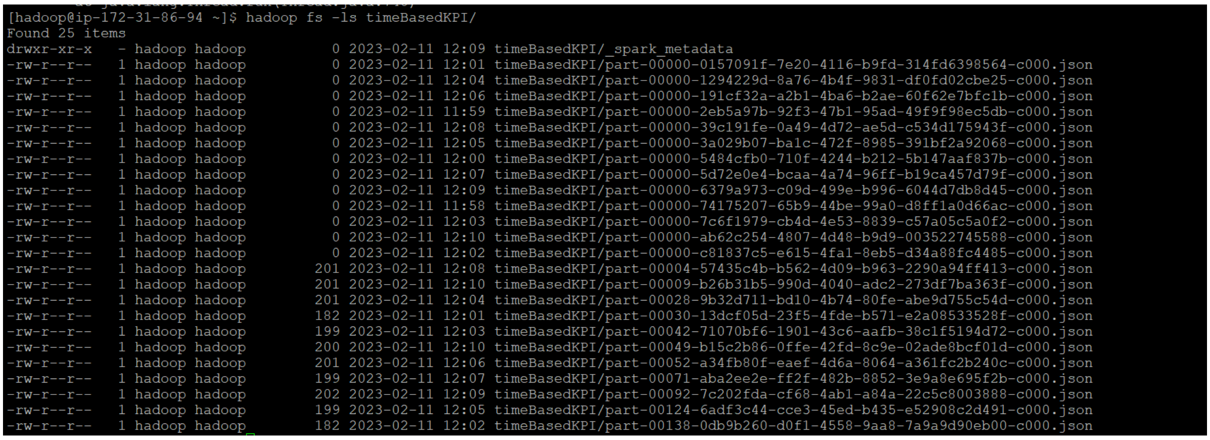


```
-------------------------------------------
Batch: 3
-------------------------------------------
+---------------+-------------+-------------------+----------+-----------+--------+---------+
|invoice_no     |country      |timestamp          |total_cost|total_items|is_order|is_return|
+---------------+-------------+-------------------+----------+-----------+--------+---------+
|154132553193163|United Kingdom|2023-02-11 11:59:12|519.95    |131        |1       |0        |
|154132553193164|United Kingdom|2023-02-11 11:59:20|99.24     |16         |1       |0        |
|154132553193165|United Kingdom|2023-02-11 11:59:32|14.75     |13         |1       |0        |
|154132553193166|United Kingdom|2023-02-11 11:59:34|63.7      |42         |1       |0        |
|154132553193167|United Kingdom|2023-02-11 11:59:37|48.33     |33         |1       |0        |
|154132553193168|United Kingdom|2023-02-11 11:59:41|27.130001 |15         |1       |0        |
|154132553193169|United Kingdom|2023-02-11 11:59:49|4.2       |2          |1       |0        |
|154132553193170|United Kingdom|2023-02-11 11:59:49|10.08     |24         |1       |0        |
|154132553193171|EIRE         |2023-02-11 11:59:51|17.279999 |12         |1       |0        |
|154132553193172|United Kingdom|2023-02-11 11:59:54|-12.07    |14         |0       |1        |
|154132553193173|United Kingdom|2023-02-11 12:00:01|5.04      |12         |1       |0        |
|154132553193174|United Kingdom|2023-02-11 12:00:09|1079.85   |125        |1       |0        |
|154132553193175|United Kingdom|2023-02-11 11:59:47|115.770004|50         |1       |0        |
|154132553193176|United Kingdom|2023-02-11 11:59:52|41.63     |25         |1       |0        |
|154132553193177|United Kingdom|2023-02-11 11:59:53|3.79      |2          |1       |0        |
|154132553193178|United Kingdom|2023-02-11 11:59:56|2497.8    |612        |1       |0        |
|154132553193179|United Kingdom|2023-02-11 11:59:58|32.8      |6          |1       |0        |
|154132553193180|United Kingdom|2023-02-11 12:00:03|180.06    |98         |1       |0        |
|154132553193181|United Kingdom|2023-02-11 12:00:06|180.95    |192        |1       |0        |
|154132553193182|United Kingdom|2023-02-11 12:00:08|15.0      |12         |1       |0        |
+---------------+-------------+-------------------+----------+-----------+--------+---------+
only showing top 20 rows

-------------------------------------------
Batch: 4
-------------------------------------------
+---------------+-------------+-------------------+----------+-----------+--------+---------+
|invoice_no     |country      |timestamp          |total_cost|total_items|is_order|is_return|
+---------------+-------------+-------------------+----------+-----------+--------+---------+
|154132553193189|United Kingdom|2023-02-11 12:00:17|111.0     |58         |1       |0        |
|154132553193190|United Kingdom|2023-02-11 12:00:19|34.739998 |6          |1       |0        |
|154132553193191|United Kingdom|2023-02-11 12:00:26|49.59     |9          |1       |0        |
|154132553193192|United Kingdom|2023-02-11 12:00:30|90.63     |43         |1       |0        |
|154132553193193|United Kingdom|2023-02-11 12:00:32|9.96      |12         |1       |0        |
|154132553193194|United Kingdom|2023-02-11 12:00:37|8.83      |8          |1       |0        |
|154132553193195|EIRE         |2023-02-11 12:00:42|40.800003 |48         |1       |0        |
|154132553193196|United Kingdom|2023-02-11 12:00:46|-38.75    |11         |0       |1        |
|154132553193197|United Kingdom|2023-02-11 12:00:47|1.25      |1          |1       |0        |
```

View the list of time-based KPIs JSON files

Get the list of all files that are created for each 1-minute window

```
hadoop fs -ls timeBasedKPI/
```

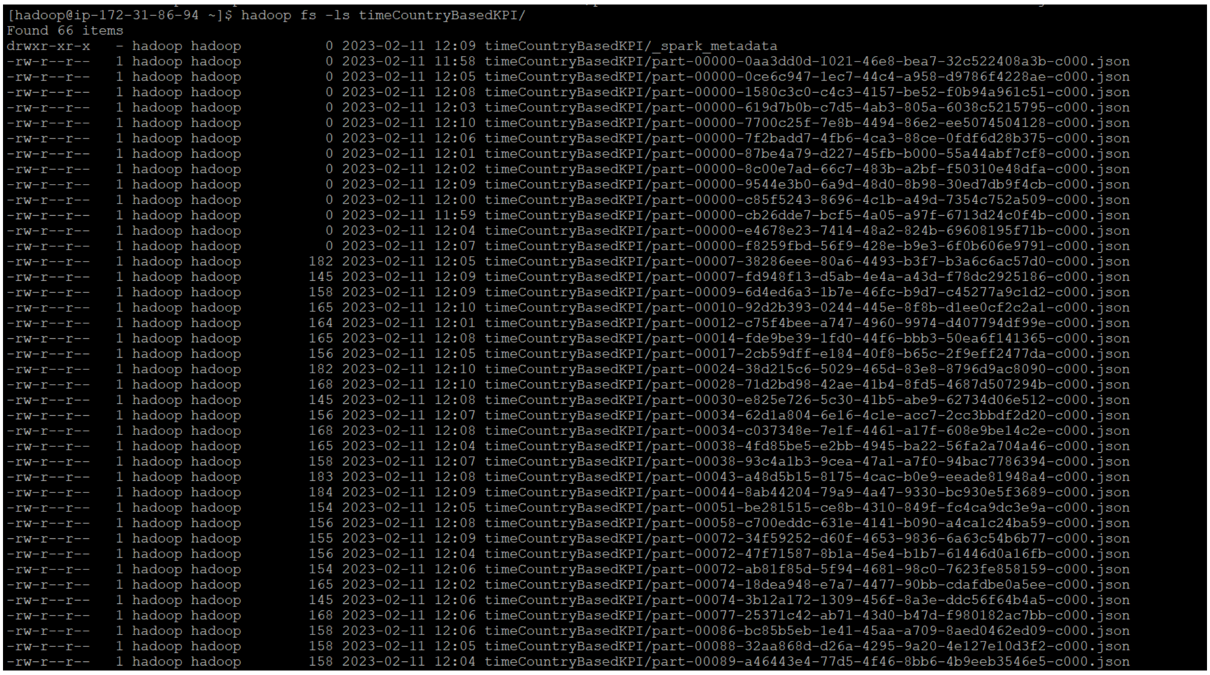


```
[hadoop@ip-172-31-86-94 ~]$ hadoop fs -ls timeBasedKPI/
Found 25 items
drwxr-xr-x   - hadoop hadoop          0 2023-02-11 12:09 timeBasedKPI/_spark_metadata
-rw-r--r--   1 hadoop hadoop          0 2023-02-11 12:01 timeBasedKPI/part-00000-0157091f-7e20-4116-b9fd-314fd6398564-c000.json
-rw-r--r--   1 hadoop hadoop          0 2023-02-11 12:04 timeBasedKPI/part-00000-1294229d-8a76-4b4f-9831-df0fd02cbe25-c000.json
-rw-r--r--   1 hadoop hadoop          0 2023-02-11 12:06 timeBasedKPI/part-00000-191cf32a-a2b1-4ba6-b2ae-60f62e7bfc1b-c000.json
-rw-r--r--   1 hadoop hadoop          0 2023-02-11 11:59 timeBasedKPI/part-00000-2eb5a97b-92f3-47b1-95ad-49f9f98ec5db-c000.json
-rw-r--r--   1 hadoop hadoop          0 2023-02-11 12:08 timeBasedKPI/part-00000-39c191fe-0a49-4d72-ae5d-c534d175943f-c000.json
-rw-r--r--   1 hadoop hadoop          0 2023-02-11 12:05 timeBasedKPI/part-00000-3a029b07-ba1c-472f-8985-391bf2a92068-c000.json
-rw-r--r--   1 hadoop hadoop          0 2023-02-11 12:00 timeBasedKPI/part-00000-5484cfb0-710f-4244-b212-5b147aaf837b-c000.json
-rw-r--r--   1 hadoop hadoop          0 2023-02-11 12:07 timeBasedKPI/part-00000-5d72e0e4-bcaa-4a74-96ff-b19ca457d79f-c000.json
-rw-r--r--   1 hadoop hadoop          0 2023-02-11 12:09 timeBasedKPI/part-00000-6379a973-c09d-499e-b996-6044d7db8d45-c000.json
-rw-r--r--   1 hadoop hadoop          0 2023-02-11 11:58 timeBasedKPI/part-00000-74175207-65b9-44be-99a0-d8ff1a0d66ac-c000.json
-rw-r--r--   1 hadoop hadoop          0 2023-02-11 12:03 timeBasedKPI/part-00000-7c6f1979-cb4d-4e53-8839-c57a05c5a0f2-c000.json
-rw-r--r--   1 hadoop hadoop          0 2023-02-11 12:10 timeBasedKPI/part-00000-ab62c254-4807-4d48-b9d9-003522745588-c000.json
-rw-r--r--   1 hadoop hadoop          0 2023-02-11 12:02 timeBasedKPI/part-00000-c81837c5-e615-4fa1-8eb5-d34a88fc4485-c000.json
-rw-r--r--   1 hadoop hadoop        201 2023-02-11 12:08 timeBasedKPI/part-00004-57435c4b-b562-4d09-b963-2290a94ff413-c000.json
-rw-r--r--   1 hadoop hadoop        201 2023-02-11 12:10 timeBasedKPI/part-00009-b26b31b5-990d-4040-adc2-273df7ba363f-c000.json
-rw-r--r--   1 hadoop hadoop        201 2023-02-11 12:04 timeBasedKPI/part-00028-9b32d711-bd10-4b74-80fe-abe9d755c54d-c000.json
-rw-r--r--   1 hadoop hadoop        182 2023-02-11 12:01 timeBasedKPI/part-00030-13dcf05d-23f5-4fde-b571-e2a08533528f-c000.json
-rw-r--r--   1 hadoop hadoop        199 2023-02-11 12:03 timeBasedKPI/part-00042-71070bf6-1901-43c6-aafb-38c1f5194d72-c000.json
-rw-r--r--   1 hadoop hadoop        200 2023-02-11 12:10 timeBasedKPI/part-00049-b15c2b86-0ffe-42fd-8c9e-02ade8bcf01d-c000.json
-rw-r--r--   1 hadoop hadoop        201 2023-02-11 12:06 timeBasedKPI/part-00052-a34fb80f-eaef-4d6a-8064-a361fc2b240c-c000.json
-rw-r--r--   1 hadoop hadoop        199 2023-02-11 12:07 timeBasedKPI/part-00071-aba2ee2e-ff2f-482b-8852-3e9a8e695f2b-c000.json
-rw-r--r--   1 hadoop hadoop        202 2023-02-11 12:09 timeBasedKPI/part-00092-7c202fda-cf68-4ab1-a84a-22c5c8003888-c000.json
-rw-r--r--   1 hadoop hadoop        199 2023-02-11 12:05 timeBasedKPI/part-00124-6adf3c44-cce3-45ed-b435-e52908c2d491-c000.json
-rw-r--r--   1 hadoop hadoop        182 2023-02-11 12:02 timeBasedKPI/part-00138-0db9b260-d0f1-4558-9aa8-7a9a9d90eb00-c000.json
```

View the list of time-and-Country based KPIs JSON files

Get the list of all files that are created for each 1-minute window

```
hadoop fs -ls timeCountryBasedKPI/
```



```
[hadoop@ip-172-31-86-94 ~]$ hadoop fs -ls timeCountryBasedKPI/
Found 66 items
drwxr-xr-x   - hadoop hadoop          0 2023-02-11 12:09 timeCountryBasedKPI/_spark_metadata
-rw-r--r--   1 hadoop hadoop          0 2023-02-11 11:58 timeCountryBasedKPI/part-00000-0aa3dd0d-1021-46e8-bea7-32c522408a3b-c000.json
-rw-r--r--   1 hadoop hadoop          0 2023-02-11 12:05 timeCountryBasedKPI/part-00000-0ce6c947-1ec7-44c4-a958-d9786f4228ae-c000.json
-rw-r--r--   1 hadoop hadoop          0 2023-02-11 12:08 timeCountryBasedKPI/part-00000-1580c3c0-c4c3-4157-be52-f0b94a961c51-c000.json
-rw-r--r--   1 hadoop hadoop          0 2023-02-11 12:03 timeCountryBasedKPI/part-00000-619d7b0b-c7d5-4ab3-805a-6038c5215795-c000.json
-rw-r--r--   1 hadoop hadoop          0 2023-02-11 12:10 timeCountryBasedKPI/part-00000-7700c25f-7e8b-4494-86e2-ee5074504128-c000.json
-rw-r--r--   1 hadoop hadoop          0 2023-02-11 12:06 timeCountryBasedKPI/part-00000-7f2badd7-4fb6-4ca3-88ce-0fdf6d28b375-c000.json
-rw-r--r--   1 hadoop hadoop          0 2023-02-11 12:01 timeCountryBasedKPI/part-00000-87be4a79-d227-45fb-b000-55a44abf7cf8-c000.json
-rw-r--r--   1 hadoop hadoop          0 2023-02-11 12:02 timeCountryBasedKPI/part-00000-8c00e7ad-66c7-483b-a2bf-f50310e48dfa-c000.json
-rw-r--r--   1 hadoop hadoop          0 2023-02-11 12:09 timeCountryBasedKPI/part-00000-9544e3b0-6a9d-48d0-8b98-30ed7db9f4cb-c000.json
-rw-r--r--   1 hadoop hadoop          0 2023-02-11 12:00 timeCountryBasedKPI/part-00000-c85f5243-8696-4c1b-a49d-7354c752a509-c000.json
-rw-r--r--   1 hadoop hadoop          0 2023-02-11 11:59 timeCountryBasedKPI/part-00000-cb26dde7-bcf5-4a05-a97f-6713d24c0f4b-c000.json
-rw-r--r--   1 hadoop hadoop          0 2023-02-11 12:04 timeCountryBasedKPI/part-00000-e4678e23-7414-48a2-824b-69608195f71b-c000.json
-rw-r--r--   1 hadoop hadoop          0 2023-02-11 12:07 timeCountryBasedKPI/part-00000-f8259fbd-56f9-428e-b9e3-6f0b606e9791-c000.json
-rw-r--r--   1 hadoop hadoop        182 2023-02-11 12:05 timeCountryBasedKPI/part-00007-38286eee-80a6-4493-b3f7-b3a6c6ac57d0-c000.json
-rw-r--r--   1 hadoop hadoop        145 2023-02-11 12:09 timeCountryBasedKPI/part-00007-fd948f13-d5ab-4e4a-a43d-f78dc2925186-c000.json
-rw-r--r--   1 hadoop hadoop        158 2023-02-11 12:09 timeCountryBasedKPI/part-00009-6d4ed6a3-1b7e-46fc-b9d7-c45277a9c1d2-c000.json
-rw-r--r--   1 hadoop hadoop        165 2023-02-11 12:10 timeCountryBasedKPI/part-00010-92d2b393-0244-445e-8f8b-d1ee0cf2c2a1-c000.json
-rw-r--r--   1 hadoop hadoop        164 2023-02-11 12:01 timeCountryBasedKPI/part-00012-c75f4bee-a747-4960-9974-d407794df99e-c000.json
-rw-r--r--   1 hadoop hadoop        165 2023-02-11 12:08 timeCountryBasedKPI/part-00014-fde9be39-1fd0-44f6-bbb3-50ea6f141365-c000.json
-rw-r--r--   1 hadoop hadoop        156 2023-02-11 12:05 timeCountryBasedKPI/part-00017-2cb59dff-e184-40f8-b65c-2f9eff2477da-c000.json
-rw-r--r--   1 hadoop hadoop        182 2023-02-11 12:10 timeCountryBasedKPI/part-00024-38d215c6-5029-465d-83e8-8796d9ac8090-c000.json
-rw-r--r--   1 hadoop hadoop        168 2023-02-11 12:10 timeCountryBasedKPI/part-00028-71d2bd98-42ae-41b4-8fd5-4687d507294b-c000.json
-rw-r--r--   1 hadoop hadoop        145 2023-02-11 12:08 timeCountryBasedKPI/part-00030-e825e726-5c30-41b5-abe9-62734d06e512-c000.json
-rw-r--r--   1 hadoop hadoop        156 2023-02-11 12:07 timeCountryBasedKPI/part-00034-62d1a804-6e16-4c1e-acc7-2cc3bbdf2d20-c000.json
-rw-r--r--   1 hadoop hadoop        168 2023-02-11 12:08 timeCountryBasedKPI/part-00034-c037348e-7e1f-4461-a17f-608e9be14c2e-c000.json
-rw-r--r--   1 hadoop hadoop        165 2023-02-11 12:04 timeCountryBasedKPI/part-00038-4fd85be5-e2bb-4945-ba22-56fa2a704a46-c000.json
-rw-r--r--   1 hadoop hadoop        158 2023-02-11 12:07 timeCountryBasedKPI/part-00038-93c4a1b3-9cea-47a1-a7f0-94bac7786394-c000.json
-rw-r--r--   1 hadoop hadoop        183 2023-02-11 12:08 timeCountryBasedKPI/part-00043-a48d5b15-8175-4cac-b0e9-eeade81948a4-c000.json
-rw-r--r--   1 hadoop hadoop        184 2023-02-11 12:09 timeCountryBasedKPI/part-00044-8ab44204-79a9-4a47-9330-bc930e5f3689-c000.json
-rw-r--r--   1 hadoop hadoop        154 2023-02-11 12:05 timeCountryBasedKPI/part-00051-be281515-ce8b-4310-849f-fc4ca9dc3e9a-c000.json
-rw-r--r--   1 hadoop hadoop        156 2023-02-11 12:08 timeCountryBasedKPI/part-00058-c700eddc-631e-4141-b090-a4ca1c24ba59-c000.json
-rw-r--r--   1 hadoop hadoop        155 2023-02-11 12:09 timeCountryBasedKPI/part-00072-34f59252-d60f-4653-9836-6a63c54b6b77-c000.json
-rw-r--r--   1 hadoop hadoop        156 2023-02-11 12:04 timeCountryBasedKPI/part-00072-47f71587-8b1a-45e4-b1b7-61446d0a16fb-c000.json
-rw-r--r--   1 hadoop hadoop        154 2023-02-11 12:06 timeCountryBasedKPI/part-00072-ab81f85d-5f94-4681-98c0-7623fe858159-c000.json
-rw-r--r--   1 hadoop hadoop        165 2023-02-11 12:02 timeCountryBasedKPI/part-00074-18dea948-e7a7-4477-90bb-cdafdbe0a5ee-c000.json
-rw-r--r--   1 hadoop hadoop        145 2023-02-11 12:06 timeCountryBasedKPI/part-00074-3b12a172-1309-456f-8a3e-ddc56f64b4a5-c000.json
-rw-r--r--   1 hadoop hadoop        168 2023-02-11 12:06 timeCountryBasedKPI/part-00077-25371c42-ab71-43d0-b47d-f980182ac7bb-c000.json
-rw-r--r--   1 hadoop hadoop        158 2023-02-11 12:06 timeCountryBasedKPI/part-00086-bc85b5eb-1e41-45aa-a709-8aed0462ed09-c000.json
-rw-r--r--   1 hadoop hadoop        158 2023-02-11 12:05 timeCountryBasedKPI/part-00088-32aa868d-d26a-4295-9a20-4e127e10d3f2-c000.json
-rw-r--r--   1 hadoop hadoop        158 2023-02-11 12:04 timeCountryBasedKPI/part-00089-a46443e4-77d5-4f46-8bb6-4b9eeb3546e5-c000.json
```

Store the time-based KPI JSON files from Hadoop to local machine

It was done in 3-steps:

- Creating a directory ***time_kpi*** on local EMR Master node

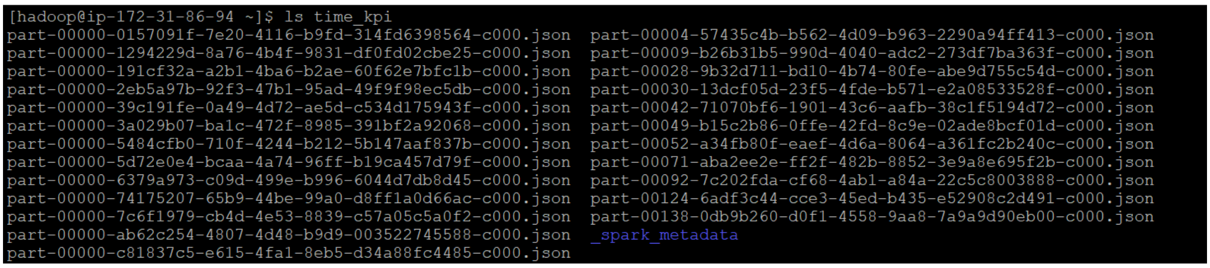
```
mkdir time_kpi
```

- Copy all the files from the timeBasedKPI Hadoop directory to newly created directory on EMR master node

```
hadoop fs -copyToLocal  timeBasedKPI/* time_kpi
```

- View the list of JSON files in time_kpi folder

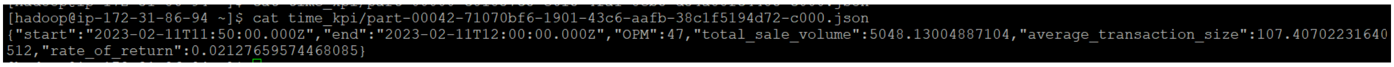
```
ls time_kpi
```



```
[hadoop@ip-172-31-86-94 ~]$ ls time_kpi
part-00000-0157091f-7e20-4116-b9fd-314fd6398564-c000.json  part-00004-57435c4b-b562-4d09-b963-2290a94ff413-c000.json
part-00000-1294229d-8a76-4b4f-9831-df0fd02cbe25-c000.json  part-00009-b26b31b5-990d-4040-adc2-273df7ba363f-c000.json
part-00000-191cf32a-a2b1-4ba6-b2ae-60f62e7bfc1b-c000.json  part-00028-9b32d711-bd10-4b74-80fe-abe9d755c54d-c000.json
part-00000-2eb5a97b-92f3-47b1-95ad-49f9f98ec5db-c000.json  part-00030-13dcf05d-23f5-4fde-b571-e2a08533528f-c000.json
part-00000-39c191fe-0a49-4d72-ae5d-c534d175943f-c000.json  part-00042-71070bf6-1901-43c6-aafb-38c1f5194d72-c000.json
part-00000-3a029b07-ba1c-472f-8985-391bf2a92068-c000.json  part-00049-b15c2b86-0ffe-42fd-8c9e-02ade8bcf01d-c000.json
part-00000-5484cfb0-710f-4244-b212-5b147aaf837b-c000.json  part-00052-a34fb80f-eaef-4d6a-8064-a361fc2b240c-c000.json
part-00000-5d72e0e4-bcaa-4a74-96ff-b19ca457d79f-c000.json  part-00071-aba2ee2e-ff2f-482b-8852-3e9a8e695f2b-c000.json
part-00000-6379a973-c09d-499e-b996-6044d7db8d45-c000.json  part-00092-7c202fda-cf68-4ab1-a84a-22c5c8003888-c000.json
part-00000-74175207-65b9-44be-99a0-d8ff1a0d66ac-c000.json  part-00124-6adf3c44-cce3-45ed-b435-e52908c2d491-c000.json
part-00000-7c6f1979-cb4d-4e53-8839-c57a05c5a0f2-c000.json  part-00138-0db9b260-d0f1-4558-9aa8-7a9a9d90eb00-c000.json
part-00000-ab62c254-4807-4d48-b9d9-003522745588-c000.json  _spark_metadata
part-00000-c81837c5-e615-4fa1-8eb5-d34a88fc4485-c000.json
```

- View the content of some random JSON file in time_kpi folder

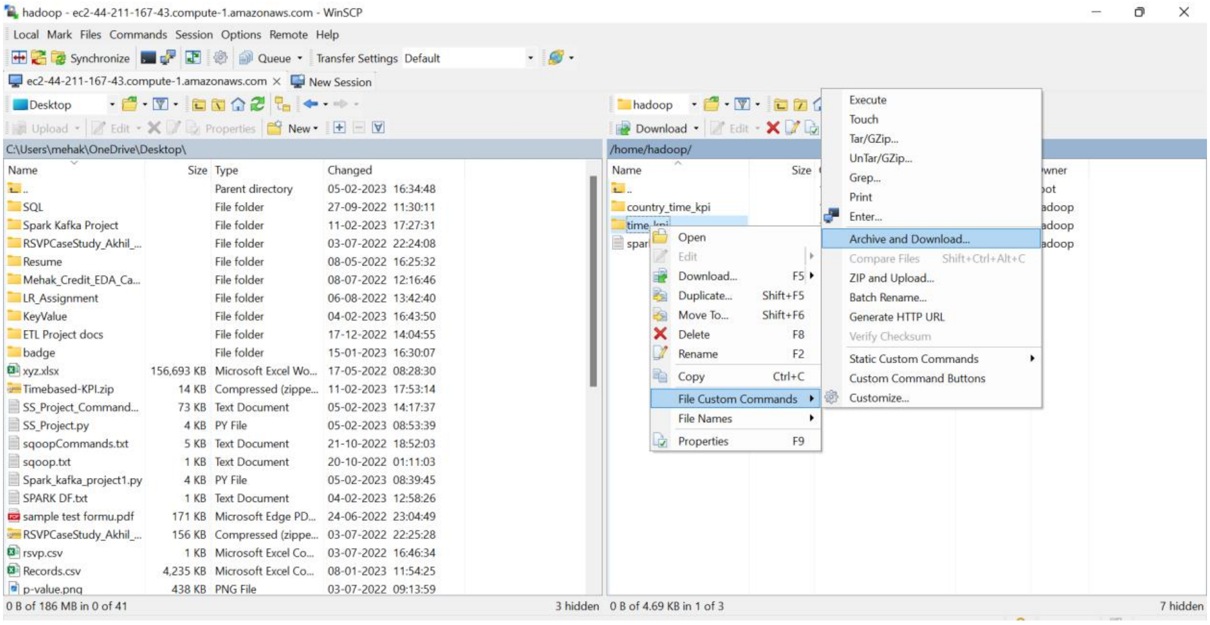
```
cat time_kpi/part-00042-71070bf6-1901-43c6-aafb-38c1f5194d72-c000.json
```



```
[hadoop@ip-172-31-86-94 ~]$ cat time_kpi/part-00042-71070bf6-1901-43c6-aafb-38c1f5194d72-c000.json
{"start":"2023-02-11T11:50:00.000Z","end":"2023-02-11T12:00:00.000Z","OPM":47,"total_sale_volume":5048.13004887104,"average_transaction_size":107.40702231640
512,"rate_of_return":0.02127659574468085}
```

- Use WinSCP to copy the folder from EMR master node to local machine

Right click on the time_kpi folder and select Archive and Download to store the folder to local machine.



hadoop - ec2-44-211-167-43.compute-1.amazonaws.com - WinSCP

ec2-44-211-167-43.compute-1.amazonaws.com

C:\Users\mehak\OneDrive\Desktop\

Name	Size	Type	Changed
..		Parent directory	05-02-2023 16:34:48
SQL		File folder	27-09-2022 11:30:11
Spark Kafka Project		File folder	11-02-2023 17:27:31
RSVPCaseStudy_Akhil_...		File folder	03-07-2022 22:24:08
Resume		File folder	08-05-2022 16:25:32
Mehak_Credit_EDA_Ca...		File folder	08-07-2022 12:16:46
LR_Assignment		File folder	06-08-2022 13:42:40
KeyValue		File folder	04-02-2023 16:43:50
ETL Project docs		File folder	17-12-2022 14:04:55
badge		File folder	15-01-2023 16:30:07
xyz.xlsx	156,693 KB	Microsoft Excel Wo...	17-05-2022 08:28:30
Timebased-KPI.zip	14 KB	Compressed (zippe...	11-02-2023 17:53:14
SS_Project_Command...	73 KB	Text Document	05-02-2023 14:17:37
SS_Project.py	4 KB	PY File	05-02-2023 08:53:39
sqoopCommands.txt	5 KB	Text Document	21-10-2022 18:52:03
sqoop.txt	1 KB	Text Document	20-10-2022 01:11:03
Spark_kafka_project1.py	4 KB	PY File	05-02-2023 08:39:45
SPARK DF.txt	1 KB	Text Document	04-02-2023 12:58:26
sample test formu.pdf	171 KB	Microsoft Edge PD...	24-06-2022 23:04:49
RSVPCaseStudy_Akhil_...	156 KB	Compressed (zippe...	03-07-2022 22:25:28
rsvp.csv	1 KB	Microsoft Excel Co...	03-07-2022 16:46:34
Records.csv	4,235 KB	Microsoft Excel Co...	08-01-2023 11:54:25
p-value.png	438 KB	PNG File	03-07-2022 09:13:59

0 B of 186 MB in 0 of 41 3 hidden

/home/hadoop/

Name	Size	...	...wner
..			...ot
country_time_kpi			...adoop
time_kpi			...adoop
spar...			...adoop

0 B of 4.69 KB in 1 of 3 7 hidden

Store the time-and-country based KPI JSON files from Hadoop to local machine

It was done in 3-steps:

- Creating a directory ***country_time_kpi*** on local EMR Master node

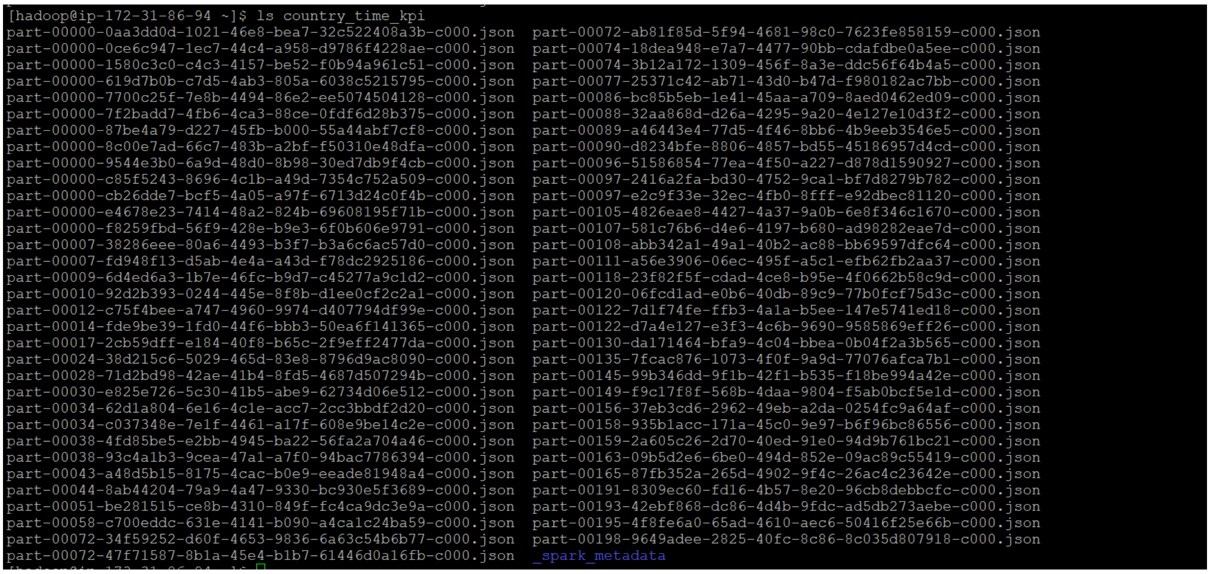
```
mkdir country_time_kpi
```

- Copy all the files from timeCountryBasedKPI Hadoop directory to newly created directory on EMR master node

```
hadoop fs -copyToLocal timeCountryBasedKPI/* country_time_kpi
```

- View the list of JSON files in country_time_kpi folder

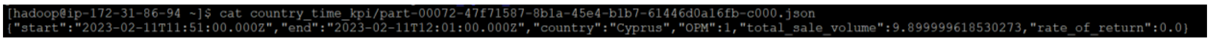
```
ls country_time_kpi
```



```
[hadoop@ip-172-31-86-94 ~]$ ls country_time_kpi
part-00000-0aa3dd0d-1021-46e8-bea7-32c522408a3b-c000.json  part-00072-ab81f85d-5f94-4681-98c0-7623fe858159-c000.json
part-00000-0ce6c947-1ec7-44c4-a958-d9786f4228ae-c000.json  part-00074-18dea948-e7a7-4477-90bb-cdafdbe0a5ee-c000.json
part-00000-1580c3c0-c4c3-4157-be52-f0b94a961c51-c000.json  part-00074-3b12a172-1309-456f-8a3e-ddc56f64b4a5-c000.json
part-00000-619d7b0b-c7d5-4ab3-805a-6038c5215795-c000.json  part-00077-25371c42-ab71-43d0-b47d-f980182ac7bb-c000.json
part-00000-7700c25f-7e8b-4494-86e2-ee5074504128-c000.json  part-00086-bc85b5eb-1e41-45aa-a709-8aed0462ed09-c000.json
part-00000-7f2badd7-4fb6-4ca3-88ce-0fdf6d28b375-c000.json  part-00088-32aa868d-d26a-4295-9a20-4e127e10d3f2-c000.json
part-00000-87be4a79-d227-45fb-b000-55a44abf7cf8-c000.json  part-00089-a46443e4-77d5-4f46-8bb6-4b9eeb3546e5-c000.json
part-00000-8c00e7ad-66c7-483b-a2bf-f50310e48dfa-c000.json  part-00090-d8234bfe-8806-4857-bd55-45186957d4cd-c000.json
part-00000-9544e3b0-6a9d-48d0-8b98-30ed7db9f4cb-c000.json  part-00096-51586854-77ea-4f50-a227-d878d1590927-c000.json
part-00000-c85f5243-8696-4c1b-a49d-7354c752a509-c000.json  part-00097-2416a2fa-bd30-4752-9ca1-bf7d8279b782-c000.json
part-00000-cb26dde7-bcf5-4a05-a97f-6713d24c0f4b-c000.json  part-00097-e2c9f33e-32ec-4fb0-8fff-e92dbec81120-c000.json
part-00000-e4678e23-7414-48a2-824b-69608195f71b-c000.json  part-00105-4826eae8-4427-4a37-9a0b-6e8f346c1670-c000.json
part-00000-f8259fbd-56f9-428e-b9e3-6f0b606e9791-c000.json  part-00107-581c76b6-d4e6-4197-b680-ad98282eae7d-c000.json
part-00007-38286eee-80a6-4493-b3f7-b3a6c6ac57d0-c000.json  part-00108-abb342a1-49a1-40b2-ac88-bb69597dfc64-c000.json
part-00007-fd948f13-d5ab-4e4a-a43d-f78dc2925186-c000.json  part-00111-a56e3906-06ec-495f-a5c1-efb62fb2aa37-c000.json
part-00009-6d4ed6a3-1b7e-46fc-b9d7-c45277a9c1d2-c000.json  part-00118-23f82f5f-cdad-4ce8-b95e-4f0662b58c9d-c000.json
part-00010-92d2b393-0244-445e-8f8b-d1ee0cf2c2a1-c000.json  part-00120-06fcd1ad-e0b6-40db-89c9-77b0fcf75d3c-c000.json
part-00012-c75f4bee-a747-4960-9974-d407794df99e-c000.json  part-00122-7d1f74fe-ffb3-4a1a-b5ee-147e5741ed18-c000.json
part-00014-fde9be39-1fd0-44f6-bbb3-50ea6f141365-c000.json  part-00122-d7a4e127-e3f3-4c6b-9690-9585869eff26-c000.json
part-00017-2cb59dff-e184-40f8-b65c-2f9eff2477da-c000.json  part-00130-da171464-bfa9-4c04-bbea-0b04f2a3b565-c000.json
part-00024-38d215c6-5029-465d-83e8-8796d9ac8090-c000.json  part-00135-7fcac876-1073-4f0f-9a9d-77076afca7b1-c000.json
part-00028-71d2bd98-42ae-41b4-8fd5-4687d507294b-c000.json  part-00145-99b346dd-9f1b-42f1-b535-f18be994a42e-c000.json
part-00030-e825e726-5c30-41b5-abe9-62734d06e512-c000.json  part-00149-f9c17f8f-568b-4daa-9804-f5ab0bcf5e1d-c000.json
part-00034-62d1a804-6e16-4c1e-acc7-2cc3bbdf2d20-c000.json  part-00156-37eb3cd6-2962-49eb-a2da-0254fc9a64af-c000.json
part-00034-c037348e-7e1f-4461-a17f-608e9be14c2e-c000.json  part-00158-935b1acc-171a-45c0-9e97-b6f96bc86556-c000.json
part-00038-4fd85be5-e2bb-4945-ba22-56fa2a704a46-c000.json  part-00159-2a605c26-2d70-40ed-91e0-94d9b761bc21-c000.json
part-00038-93c4a1b3-9cea-47a1-a7f0-94bac7786394-c000.json  part-00163-09b5d2e6-6be0-494d-852e-09ac89c55419-c000.json
part-00043-a48d5b15-8175-4cac-b0e9-eeade81948a4-c000.json  part-00165-87fb352a-265d-4902-9f4c-26ac4c23642e-c000.json
part-00044-8ab44204-79a9-4a47-9330-bc930e5f3689-c000.json  part-00191-8309ec60-fd16-4b57-8e20-96cb8debbcfc-c000.json
part-00051-be281515-ce8b-4310-849f-fc4ca9dc3e9a-c000.json  part-00193-42ebf868-dc86-4d4b-9fdc-ad5db273aebe-c000.json
part-00058-c700eddc-631e-4141-b090-a4ca1c24ba59-c000.json  part-00195-4f8fe6a0-65ad-4610-aec6-50416f25e66b-c000.json
part-00072-34f59252-d60f-4653-9836-6a63c54b6b77-c000.json  part-00198-9649adee-2825-40fc-8c86-8c035d807918-c000.json
part-00072-47f71587-8b1a-45e4-b1b7-61446d0a16fb-c000.json  _spark_metadata
```

- View the content of some random JSON file in country_time_kpi folder

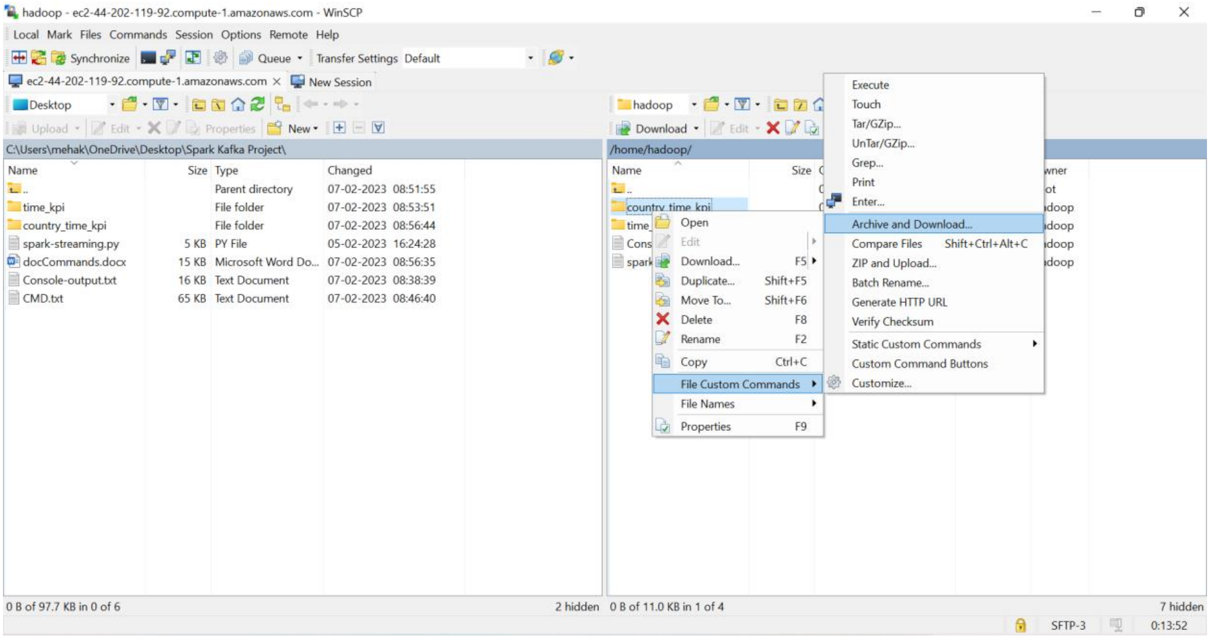
```
cat country_time_kpi/part-00072-47f71587-8b1a-45e4-b1b7-61446d0a16fb-c000.json
```



```
[hadoop@ip-172-31-86-94 ~]$ cat country_time_kpi/part-00072-47f71587-8b1a-45e4-b1b7-61446d0a16fb-c000.json
{"start":"2023-02-11T11:51:00.000Z","end":"2023-02-11T12:01:00.000Z","country":"Cyprus","OPM":1,"total_sale_volume":9.899999618530273,"rate_of_return":0.0}
```

- Use WinSCP to copy the timeCountryBasedKPI folder from EMR master node to local machine

Right click on the country_time_kpi folder and select Archive and Download to store the folder to local machine.



hadoop - ec2-44-202-119-92.compute-1.amazonaws.com - WinSCP

C:\Users\mehak\OneDrive\Desktop\Spark Kafka Project\

Name	Size	Type	Changed
..		Parent directory	07-02-2023 08:51:55
time_kpi		File folder	07-02-2023 08:53:51
country_time_kpi		File folder	07-02-2023 08:56:44
spark-streaming.py	5 KB	PY File	05-02-2023 16:24:28
docCommands.docx	15 KB	Microsoft Word Do...	07-02-2023 08:56:35
Console-output.txt	16 KB	Text Document	07-02-2023 08:38:39
CMD.txt	65 KB	Text Document	07-02-2023 08:46:40

0 B of 97.7 KB in 0 of 6 — 2 hidden

/home/hadoop/

0 B of 11.0 KB in 1 of 4 — 7 hidden