Code Logic - Retail Data Analysis

1. Code logic used for performing the Retail data Analysis is as below.

Inclusion/import of the Cloudera Distribution and other PySpark environment setting

```
# PySpark Program for Retail Data Analysis Project
# import of the required modules
import os
import sys
from pyspark.sql import SparkSession
from pyspark.sql.functions import *
from pyspark.sql.types import *

# PySpark environment variable declarations
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")
```

Creating the spark context and Syntax definition of the JSON File

```
# Spark Session Context
spark = SparkSession \
    .builder \
    .appName("RetailDataAnalysisProject") \
    .getOrCreate()
spark.sparkContext.setLogLevel('ERROR')
```

readStream creation Reading the data from Kafka

```
# Reading order data from Kafka provided Bootstrap
orderRaw = spark \
    .readStream \
    .format("kafka") \
    .option("kafka.bootstrap.servers", "18.211.252.152:9092") \
    .option("startingOffsets", "latest") \
    .option("subscribe","real-time-project") \
    .load() \
```

Defining the schema for JSON file for the incoming data

```
# Defining schema for JSON format as per the expected incoming data
jsonSchema = StructType() \
   .add("invoice_no", LongType()) \
   .add("timestamp", TimestampType()) \
   .add("type", StringType()) \
   .add("country", StringType()) \
   .add("items", ArrayType(StructType([
   StructField("SKU", StringType()),
   StructField("title", StringType()),
   StructField("unit_price", DoubleType()),
   StructField("quantity", IntegerType())])))
```

DataFrame input creation with Scheme applying

```
# Creating dataframe from input data after applying the schema
orderStream = orderRaw.select(from_json(col("value").cast("string"),
jsonSchema).alias("order")).select("order.*")
```

Definition of the UDF functions for is Order, Is return, total items and total cost calculations and conversion to UDF types and calculation of the columns

```
# Defining functions to interpret the required columns
# is Order Checking
def is_order(order):
   return 1 if (order == "ORDER") else 0
# is return checking
def is_return(order):
   return 1 if (order == "RETURN") else 0
# Calculation of total items
def total_items(items):
   return len(items)
# Calculating the total cost of the times
def total_cost(items, order_type):
   Sum = 0
   for item in items:
      Sum = Sum + item.unit_price * item.quantity if (order_type == "ORDER") else (
            Sum - (item.unit_price * item.quantity))
   return Sum

# UDF conversion
isOrder = udf(is_order, IntegerType())
isReturn = udf(is_return, IntegerType())
totalItems = udf(total_items, IntegerType())
totalCost = udf(total_cost, DoubleType())

# Calculating columns - applying UDF
```

```
orderStream = orderStream.withColumn("is_order", isOrder(orderStream.type)) \
.withColumn("is_return", isReturn(orderStream.type)) \
.withColumn("total_items", totalItems(orderStream.items)) \
.withColumn("total_cost", totalCost(orderStream.items, orderStream.type))
```

Selecting the expression and Write Stream function definition for orderStreams, time based and time – country based KPI calculation for the interval of 1 minute

```
# Write stream for console output as per the expectation (1 minute interval)
orderStream = orderStream.selectExpr("invoice_no", "country", "timestamp", "total_cost",
"total_items", "is_order",
                                "is_return")

orderOutputStream = orderStream \
   .writeStream \
   .outputMode("append") \
   .format("console") \
   .option("truncate", "False") \
   .trigger(processingTime="1 minute") \
   .start()

# Calculating time based KPIs
timeBasedKPIs = orderStream.withWatermark("timestamp", "1 minute") \
   .groupby(window("timestamp", "1 minute")) \
   .agg(count("invoice_no").alias("invoiceNo"),
      sum("total_cost").alias("totalcost"),
      sum("is_return").alias("return"),
      sum("is_order").alias("order")) \
    .withColumn("rate_of_return", col("return") / (col("return") + col("order"))) \
    .withColumn("avg_transaction", col("totalcost") / (col("return") + col("order")))

# write stream for time based KPIs
timeBasedKPIs = timeBasedKPIs.selectExpr("window", "invoiceNo", "totalcost", "avg_transaction",
                                "rate_of_return")

timeBasedKPIsOutput = timeBasedKPIs \
   .writeStream \
   .outputMode("append") \
   .format("json") \
   .option("path", "/user/ec2-user/rdaproj/TimeBasedKPIsOutput") \
   .option("checkpointLocation", "/user/ec2-user/rdaproj/TimeBasedKPI") \
   .option("truncate", "False") \
   .trigger(processingTime="1 minute") \
   .start()
```

```
# Calculating time and country based KPIs
timeAndCountryBasedKPIs = orderStream.withWatermark("timestamp", "1 minute") \
    .groupby(window("timestamp", "1 minute"), "country") \
    .agg(count("invoice_no").alias("invoiceNo"),
         sum("total_cost").alias("totalCost"),
         sum("is_return").alias("return"),
         sum("is_order").alias("order")) \
    .withColumn("rate_of_return", col("return") / (col("return") + col("order")))

# write stream for time and country based KPIs
timeAndCountryBasedKPIs = timeAndCountryBasedKPIs.selectExpr("window", "country",
"invoiceNo", "totalcost",
                                                    "rate_of_return")


timeAndCountryBasedKPIsOutput = timeAndCountryBasedKPIs \
    .writeStream \
    .outputMode("append") \
    .format("json") \
    .option("path", "/user/ec2-user/rdaproj/TimeAndCountryBasedKPIsOutput") \
    .option("checkpointLocation", "/user/ec2-user/rdaproj/TimeAndCountryBased") \
    .option("truncate", "False") \
    .trigger(processingTime="1 minute") \
    .start()
```

Waiting for the termination of stream infinitely

```
# Waiting infinitely to read the data
timeAndCountryBasedKPIsOutput.awaitTermination()
timeBasedKPIsOutput.awaitTermination()
orderOutputStream.awaitTermination()
```

Console Commands and Analysis

List of commands/instruction used as below, post entering ec2 instance with 'ec2-user'

- `wget https://ds-spark-sql-kafka-jar.s3.amazonaws.com/spark-sql-kafka-0-10_2.11-.3.0.jar`
- `Copied the spark-streaming.py file to /user/ec2-user/ location`
- `export SPARK_KAFKA_VERSION=0.10`
- `spark2-submit --jars spark-sql-kafka-0-10_2.11-2.3.0.jar spark-streaming.py`

Output Screens

Summarized Console Output

```
-------------------------------------------
Batch: 59
-------------------------------------------
+---------------+--------------+-------------------+------------------+-----------+--------+---------+
|invoice_no     |country       |timestamp          |total_cost        |total_items|is_order|is_return|
+---------------+--------------+-------------------+------------------+-----------+--------+---------+
|154132547743871|United Kingdom|2022-01-19 11:59:18|86.38000000000001 |9          |1       |0        |
|154132547743872|United Kingdom|2022-01-19 11:59:21|23.35             |2          |1       |0        |
|154132547743873|United Kingdom|2022-01-19 11:59:34|51.65             |4          |1       |0        |
|154132547743874|United Kingdom|2022-01-19 11:59:46|276.81000000000006|4          |1       |0        |
|154132547743875|United Kingdom|2022-01-19 11:59:48|63.16000000000001 |7          |1       |0        |
|154132547743876|United Kingdom|2022-01-19 11:59:51|66.6              |2          |1       |0        |
|154132547743877|United Kingdom|2022-01-19 12:00:00|5.9               |1          |1       |0        |
|154132547743878|United Kingdom|2022-01-19 12:00:06|101.25999999999999|4          |1       |0        |
|154132547743879|United Kingdom|2022-01-19 12:00:10|22.15             |3          |1       |0        |
|154132547743880|United Kingdom|2022-01-19 12:00:14|204.0             |1          |1       |0        |
+---------------+--------------+-------------------+------------------+-----------+--------+---------+
```

KPI data collection point in HDFS location rdaproj

```
[ec2-user@ip-10-0-0-117 ~]$ hadoop fs -ls /user/ec2-user/rdaproj/
Found 4 items
drwxrwxrwx   - ec2-user ec2-user          0 2022-01-19 11:02 /user/ec2-user/rdaproj/TimeAndCountryBased
drwxr-xr-x   - ec2-user ec2-user          0 2022-01-19 12:06 /user/ec2-user/rdaproj/TimeAndCountryBasedKPIsOutput
drwxrwxrwx   - ec2-user ec2-user          0 2022-01-19 11:01 /user/ec2-user/rdaproj/TimeBasedKPI
drwxr-xr-x   - ec2-user ec2-user          0 2022-01-19 12:06 /user/ec2-user/rdaproj/TimeBasedKPIsOutput
[ec2-user@ip-10-0-0-117 ~]$
```

JSON file collection checking at the HDFS location for time basedKPI

```
[ec2-user@ip-10-0-0-117 ~]$ hadoop fs -ls /user/ec2-user/rdaproj/TimeBasedKPIsOutput
Found 128 items
drwxr-xr-x   - ec2-user ec2-user          0 2022-01-19 12:05 /user/ec2-user/rdaproj/TimeBasedKPIsOutput/_spark_metadata
-rw-r--r--   3 ec2-user ec2-user          0 2022-01-19 11:43 /user/ec2-user/rdaproj/TimeBasedKPIsOutput/part-00000-0120d4a4-f0e2-41a0-b991-6c458fa7a926-c000.json
-rw-r--r--   3 ec2-user ec2-user          0 2022-01-19 11:19 /user/ec2-user/rdaproj/TimeBasedKPIsOutput/part-00000-0889e065-5594-465d-ac69-53b5241ad9b4-c000.json
-rw-r--r--   3 ec2-user ec2-user          0 2022-01-19 11:22 /user/ec2-user/rdaproj/TimeBasedKPIsOutput/part-00000-08bbaaca-6423-4d91-87ed-3406e28c987e-c000.json
-rw-r--r--   3 ec2-user ec2-user          0 2022-01-19 11:17 /user/ec2-user/rdaproj/TimeBasedKPIsOutput/part-00000-0b51918f-cbce-437b-94a6-5f45e2d705e7-c000.json
-rw-r--r--   3 ec2-user ec2-user          0 2022-01-19 11:46 /user/ec2-user/rdaproj/TimeBasedKPIsOutput/part-00000-0f401e33-59ee-4102-9138-f253e6182910-c000.json
-rw-r--r--   3 ec2-user ec2-user          0 2022-01-19 12:05 /user/ec2-user/rdaproj/TimeBasedKPIsOutput/part-00000-141ac7b7-720b-4736-b99f-0ffd235aa9ee-c000.json
-rw-r--r--   3 ec2-user ec2-user          0 2022-01-19 11:42 /user/ec2-user/rdaproj/TimeBasedKPIsOutput/part-00000-1575fed7-d97c-4943-a76e-e67c243b756b-c000.json
-rw-r--r--   3 ec2-user ec2-user          0 2022-01-19 11:18 /user/ec2-user/rdaproj/TimeBasedKPIsOutput/part-00000-164b85bb-97d6-4f64-af1a-04b9611c1b66-c000.json
-rw-r--r--   3 ec2-user ec2-user          0 2022-01-19 11:28 /user/ec2-user/rdaproj/TimeBasedKPIsOutput/part-00000-1fa3b522-24f0-40ea-9bd0-f0898ec365d1-c000.json
-rw-r--r--   3 ec2-user ec2-user          0 2022-01-19 11:15 /user/ec2-user/rdaproj/TimeBasedKPIsOutput/part-00000-1fc99d8b-8f64-493c-b5d7-18d9de9edebb-c000.json
-rw-r--r--   3 ec2-user ec2-user          0 2022-01-19 11:38 /user/ec2-user/rdaproj/TimeBasedKPIsOutput/part-00000-22a57a82-9cba-449a-94d1-13679ab890f7-c000.json
-rw-r--r--   3 ec2-user ec2-user          0 2022-01-19 11:53 /user/ec2-user/rdaproj/TimeBasedKPIsOutput/part-00000-2316a6c8-e4e8-41bc-ae42-aae82b529b1a-c000.json
-rw-r--r--   3 ec2-user ec2-user          0 2022-01-19 11:55 /user/ec2-user/rdaproj/TimeBasedKPIsOutput/part-00000-2c866d23-9436-4c2c-b0d3-c0a64e3cfbf7-c000.json
-rw-r--r--   3 ec2-user ec2-user          0 2022-01-19 11:49 /user/ec2-user/rdaproj/TimeBasedKPIsOutput/part-00000-32d3ca6b-3330-422e-972c-79a103f216c5-c000.json
-rw-r--r--   3 ec2-user ec2-user          0 2022-01-19 11:21 /user/ec2-user/rdaproj/TimeBasedKPIsOutput/part-00000-341f47a4-6bd7-4c50-bf38-e253472c01c8-c000.json
-rw-r--r--   3 ec2-user ec2-user          0 2022-01-19 11:59 /user/ec2-user/rdaproj/TimeBasedKPIsOutput/part-00000-36179ba9-c4de-404d-8e43-b85604ad50a2-c000.json
-rw-r--r--   3 ec2-user ec2-user          0 2022-01-19 11:41 /user/ec2-user/rdaproj/TimeBasedKPIsOutput/part-00000-3627dbdd-eda2-4cb4-bb72-8668a9d800fa-c000.json
-rw-r--r--   3 ec2-user ec2-user          0 2022-01-19 12:06 /user/ec2-user/rdaproj/TimeBasedKPIsOutput/part-00000-3b8fa98b-c3bc-4fc5-b04d-59927f930dd1-c000.json
-rw-r--r--   3 ec2-user ec2-user          0 2022-01-19 11:32 /user/ec2-user/rdaproj/TimeBasedKPIsOutput/part-00000-3bc837a0-52de-40c1-8951-51d59fc0e84d-c000.json
-rw-r--r--   3 ec2-user ec2-user          0 2022-01-19 11:33 /user/ec2-user/rdaproj/TimeBasedKPIsOutput/part-00000-4e30861d-abe6-47a2-bb15-f4faf4a0eec9-c000.json
-rw-r--r--   3 ec2-user ec2-user          0 2022-01-19 11:36 /user/ec2-user/rdaproj/TimeBasedKPIsOutput/part-00000-5220062e-bc44-4d9c-a246-fd96ae0a4d9d-c000.json
-rw-r--r--   3 ec2-user ec2-user          0 2022-01-19 11:03 /user/ec2-user/rdaproj/TimeBasedKPIsOutput/part-00000-53cd1a58-684a-44cd-8627-e0054bab3eea-c000.json
-rw-r--r--   3 ec2-user ec2-user          0 2022-01-19 11:06 /user/ec2-user/rdaproj/TimeBasedKPIsOutput/part-00000-5bf57a48-6a5c-4602-9c50-f257056f1924-c000.json
-rw-r--r--   3 ec2-user ec2-user          0 2022-01-19 11:51 /user/ec2-user/rdaproj/TimeBasedKPIsOutput/part-00000-5e92fc9a-867c-4ccb-8e78-73f4315ec0b8-c000.json
```

Checking on the data part of the JSON File

```
[ec2-user@ip-10-0-0-117 ~]$ hadoop fs -cat /user/ec2-user/rdaproj/TimeBasedKPIsOutput/part*
{"window":{"start":"2022-01-19T11:53:00.000Z","end":"2022-01-19T11:54:00.000Z"},"invoiceNo":5,"totalcost":308.58000000000004,"avg_transaction":61.71600000000001,"rate_of_return":0.0}
{"window":{"start":"2022-01-19T11:07:00.000Z","end":"2022-01-19T11:08:00.000Z"},"invoiceNo":6,"totalcost":403.95000000000005,"avg_transaction":67.325,"rate_of_return":0.0}
{"window":{"start":"2022-01-19T11:21:00.000Z","end":"2022-01-19T11:22:00.000Z"},"invoiceNo":13,"totalcost":6251.049999999999,"avg_transaction":480.8499999999997,"rate_of_return":0.0}
{"window":{"start":"2022-01-19T11:59:00.000Z","end":"2022-01-19T12:00:00.000Z"},"invoiceNo":9,"totalcost":612.26,"avg_transaction":68.02888888888889,"rate_of_return":0.0}
{"window":{"start":"2022-01-19T11:14:00.000Z","end":"2022-01-19T11:15:00.000Z"},"invoiceNo":9,"totalcost":370.8,"avg_transaction":41.2,"rate_of_return":0.0}
{"window":{"start":"2022-01-19T11:44:00.000Z","end":"2022-01-19T11:45:00.000Z"},"invoiceNo":7,"totalcost":889.1499999999999,"avg_transaction":127.02142857142856,"rate_of_return":0.142857142
85714285}
{"window":{"start":"2022-01-19T11:55:00.000Z","end":"2022-01-19T11:56:00.000Z"},"invoiceNo":8,"totalcost":315.65999999999997,"avg_transaction":39.457499999999996,"rate_of_return":0.125}
{"window":{"start":"2022-01-19T11:18:00.000Z","end":"2022-01-19T11:19:00.000Z"},"invoiceNo":16,"totalcost":1196.0100000000002,"avg_transaction":74.75062500000001,"rate_of_return":0.0}
{"window":{"start":"2022-01-19T11:40:00.000Z","end":"2022-01-19T11:41:00.000Z"},"invoiceNo":6,"totalcost":266.62,"avg_transaction":44.43666666666667,"rate_of_return":0.16666666666666666}
{"window":{"start":"2022-01-19T11:09:00.000Z","end":"2022-01-19T11:10:00.000Z"},"invoiceNo":13,"totalcost":1324.8100000000002,"avg_transaction":101.90846153846155,"rate_of_return":0.0}
{"window":{"start":"2022-01-19T11:16:00.000Z","end":"2022-01-19T11:17:00.000Z"},"invoiceNo":13,"totalcost":632.1000000000001,"avg_transaction":48.623076923076924,"rate_of_return":0.076923076
92307693}
{"window":{"start":"2022-01-19T11:36:00.000Z","end":"2022-01-19T11:37:00.000Z"},"invoiceNo":10,"totalcost":228.0,"avg_transaction":22.8,"rate_of_return":0.0}
{"window":{"start":"2022-01-19T11:37:00.000Z","end":"2022-01-19T11:38:00.000Z"},"invoiceNo":10,"totalcost":314.02,"avg_transaction":31.401999999999997,"rate_of_return":0.0}
{"window":{"start":"2022-01-19T12:01:00.000Z","end":"2022-01-19T12:02:00.000Z"},"invoiceNo":6,"totalcost":127.51,"avg_transaction":21.251666666666667,"rate_of_return":0.0}
{"window":{"start":"2022-01-19T11:02:00.000Z","end":"2022-01-19T11:03:00.000Z"},"invoiceNo":13,"totalcost":1530.58,"avg_transaction":117.73692307692308,"rate_of_return":0.0}
{"window":{"start":"2022-01-19T11:26:00.000Z","end":"2022-01-19T11:27:00.000Z"},"invoiceNo":6,"totalcost":466.53,"avg_transaction":77.755,"rate_of_return":0.0}
{"window":{"start":"2022-01-19T11:49:00.000Z","end":"2022-01-19T11:50:00.000Z"},"invoiceNo":9,"totalcost":287.63,"avg_transaction":31.95888888888889,"rate_of_return":0.0}
{"window":{"start":"2022-01-19T11:03:00.000Z","end":"2022-01-19T11:04:00.000Z"},"invoiceNo":14,"totalcost":1047.8899999999999,"avg_transaction":74.84928571428571,"rate_of_return":0.0}
{"window":{"start":"2022-01-19T11:05:00.000Z","end":"2022-01-19T11:06:00.000Z"},"invoiceNo":13,"totalcost":1019.3700000000001,"avg_transaction":78.41307692307693,"rate_of_return":0.07692307
692307693}
```

Transfer of data from HDFS to Local using -copyToLocal command

- hadoop fs -copyToLocal /user/ec2-user/rdaproj/TimeBasedKPIsOutput /home/ec2-user/Timebased-KPI

- hadoop fs -copyToLocal /user/ec2-user/rdaproj/TimeAndCountryBasedKPIsOutput /home/ec2-user/Country-and-timebased-KPI
-

PS: WinSCP used to copy the files from EC2-Instance to local machine.