PROJECT -2

Retail Business Analytics

Submitted by:
Santhosh N
(KB23021)

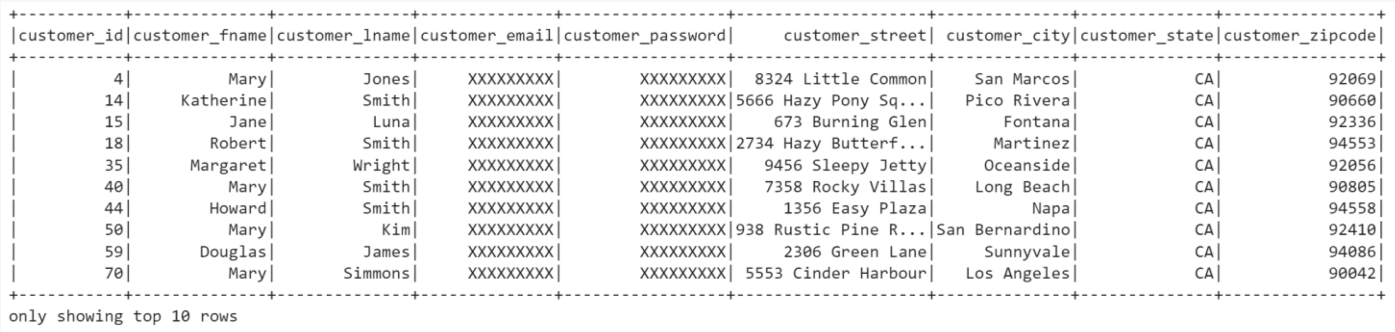
Scenario - 1

1. Explore the customer records saved in the “customers tab delimited” directory on HDFS.

a) Show the client information for those who live in California.

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

# Create a Spark session
spark = SparkSession.builder.appName("California_Customers").getOrCreate()

input_directory = "/content/customers-tab-delimited/part-m-00000"
output_directory = "/content/solution/scenario1/result"
output_file = "cal_customers.csv"

#Schema
customer_schema = StructType([
    StructField("customer_id", IntegerType(), True),
    StructField("customer_fname", StringType(), True),
    StructField("customer_lname", StringType(), True),
    StructField("customer_email", StringType(), True),
    StructField("customer_password", StringType(), True),
    StructField("customer_street", StringType(), True),
    StructField("customer_city", StringType(), True),
    StructField("customer_state", StringType(), True),
    StructField("customer_zipcode", StringType(), True)
])


# Load data from HDFS
data =
spark.read.option("delimiter","\t").csv(input_directory,schema=customer_schema)


# Filter customers from California
california_customers = data.filter(data.customer_state == "CA")
california_customers.show(10)
```

#Output:



```
+-----------+--------------+--------------+--------------+-----------------+--------------------+--------------+--------------+----------------+
|customer_id|customer_fname|customer_lname|customer_email|customer_password|     customer_street| customer_city|customer_state|customer_zipcode|
+-----------+--------------+--------------+--------------+-----------------+--------------------+--------------+--------------+----------------+
|          4|          Mary|         Jones|     XXXXXXXXX|        XXXXXXXXX|  8324 Little Common|    San Marcos|            CA|           92069|
|         14|     Katherine|         Smith|     XXXXXXXXX|        XXXXXXXXX|5666 Hazy Pony Sq...|   Pico Rivera|            CA|           90660|
|         15|          Jane|          Luna|     XXXXXXXXX|        XXXXXXXXX|    673 Burning Glen|       Fontana|            CA|           92336|
|         18|        Robert|         Smith|     XXXXXXXXX|        XXXXXXXXX|2734 Hazy Butterf...|      Martinez|            CA|           94553|
|         35|      Margaret|        Wright|     XXXXXXXXX|        XXXXXXXXX|   9456 Sleepy Jetty|     Oceanside|            CA|           92056|
|         40|          Mary|         Smith|     XXXXXXXXX|        XXXXXXXXX|   7358 Rocky Villas|    Long Beach|            CA|           90805|
|         44|        Howard|         Smith|     XXXXXXXXX|        XXXXXXXXX|     1356 Easy Plaza|          Napa|            CA|           94558|
|         50|          Mary|           Kim|     XXXXXXXXX|        XXXXXXXXX|938 Rustic Pine R...|San Bernardino|            CA|           92410|
|         59|       Douglas|         James|     XXXXXXXXX|        XXXXXXXXX|     2306 Green Lane|     Sunnyvale|            CA|           94086|
|         70|          Mary|       Simmons|     XXXXXXXXX|        XXXXXXXXX| 5553 Cinder Harbour|   Los Angeles|            CA|           90042|
+-----------+--------------+--------------+--------------+-----------------+--------------------+--------------+--------------+----------------+
only showing top 10 rows
```

b) Include the customer's entire name in the output

```
# Extract full names
california_customers = california_customers.withColumn("full_name", concat_ws(" ",
california_customers["customer_fname"], california_customers["customer_lname"]))
california_customers.show(10)
```

```
+-----------+--------------+--------------+--------------+-----------------+--------------------+--------------+--------------+----------------+---------------+
|customer_id|customer_fname|customer_lname|customer_email|customer_password|     customer_street| customer_city|customer_state|customer_zipcode|      full_name|
+-----------+--------------+--------------+--------------+-----------------+--------------------+--------------+--------------+----------------+---------------+
|          4|          Mary|         Jones|     XXXXXXXXX|        XXXXXXXXX|  8324 Little Common|    San Marcos|            CA|           92069|     Mary Jones|
|         14|     Katherine|         Smith|     XXXXXXXXX|        XXXXXXXXX|5666 Hazy Pony Sq...|   Pico Rivera|            CA|           90660|Katherine Smith|
|         15|          Jane|          Luna|     XXXXXXXXX|        XXXXXXXXX|    673 Burning Glen|       Fontana|            CA|           92336|      Jane Luna|
|         18|        Robert|         Smith|     XXXXXXXXX|        XXXXXXXXX|2734 Hazy Butterf...|      Martinez|            CA|           94553|   Robert Smith|
|         35|      Margaret|        Wright|     XXXXXXXXX|        XXXXXXXXX|   9456 Sleepy Jetty|     Oceanside|            CA|           92056|Margaret Wright|
|         40|          Mary|         Smith|     XXXXXXXXX|        XXXXXXXXX|   7358 Rocky Villas|    Long Beach|            CA|           90805|     Mary Smith|
|         44|        Howard|         Smith|     XXXXXXXXX|        XXXXXXXXX|     1356 Easy Plaza|          Napa|            CA|           94558|   Howard Smith|
|         50|          Mary|           Kim|     XXXXXXXXX|        XXXXXXXXX|938 Rustic Pine R...|San Bernardino|            CA|           92410|       Mary Kim|
|         59|       Douglas|         James|     XXXXXXXXX|        XXXXXXXXX|     2306 Green Lane|     Sunnyvale|            CA|           94086|  Douglas James|
|         70|          Mary|       Simmons|     XXXXXXXXX|        XXXXXXXXX| 5553 Cinder Harbour|   Los Angeles|            CA|           90042|   Mary Simmons|
+-----------+--------------+--------------+--------------+-----------------+--------------------+--------------+--------------+----------------+---------------+
only showing top 10 rows
```

```
# Dropping first Name & Last Name Column
Final_results = california_customers.drop("customer_fname", "customer_lname")

#Rearranging Columns
sorting_columns = ["customer_id", "full_name", "customer_email",
"customer_password","customer_street", "customer_city", "customer_state",
"customer_zipcode"]
Final_results = california_customers.select(*sorting_columns)
Final_results.show(5)
```

#Output:

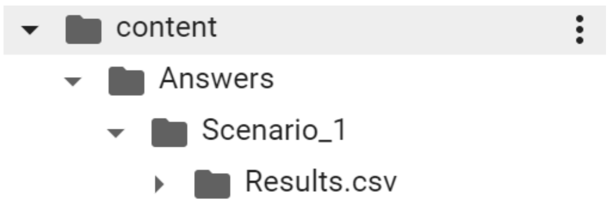
```
+-----------+---------------+--------------+-----------------+--------------------+-------------+--------------+----------------+
|customer_id|      full_name|customer_email|customer_password|     customer_street|customer_city|customer_state|customer_zipcode|
+-----------+---------------+--------------+-----------------+--------------------+-------------+--------------+----------------+
|          4|     Mary Jones|     XXXXXXXXX|        XXXXXXXXX|  8324 Little Common|   San Marcos|            CA|           92069|
|         14|Katherine Smith|     XXXXXXXXX|        XXXXXXXXX|5666 Hazy Pony Sq...|  Pico Rivera|            CA|           90660|
|         15|      Jane Luna|     XXXXXXXXX|        XXXXXXXXX|    673 Burning Glen|      Fontana|            CA|           92336|
|         18|   Robert Smith|     XXXXXXXXX|        XXXXXXXXX|2734 Hazy Butterf...|     Martinez|            CA|           94553|
|         35|Margaret Wright|     XXXXXXXXX|        XXXXXXXXX|   9456 Sleepy Jetty|    Oceanside|            CA|           92056|
+-----------+---------------+--------------+-----------------+--------------------+-------------+--------------+----------------+
only showing top 5 rows
```

c) Save the results in the result/scenario1/solution folder

```
Final_results
# Save the DataFrame as a CSV file
output_path="/content/drive/MyDrive/data-files/Answers/Scenario_1/Results.csv"
Final_results.write.csv(output_path, header=True, mode="overwrite")

print(f"DataFrame saved as CSV at: {output_path}")
```

```
    DataFrame saved as CSV at: /content/drive/MyDrive/data-files/Answers/Scenario_1/Results.csv
```

#Output:

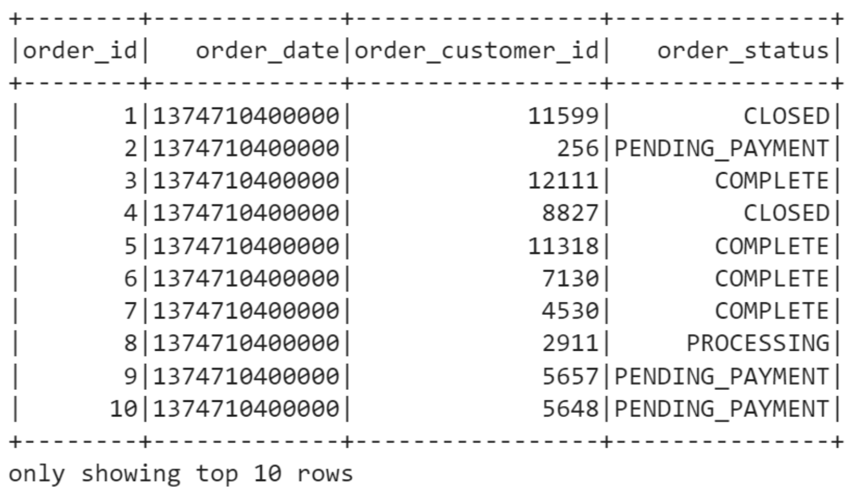


Scenario - 2

2. Explore the order records saved in the “orders parquet” directory on HDFS

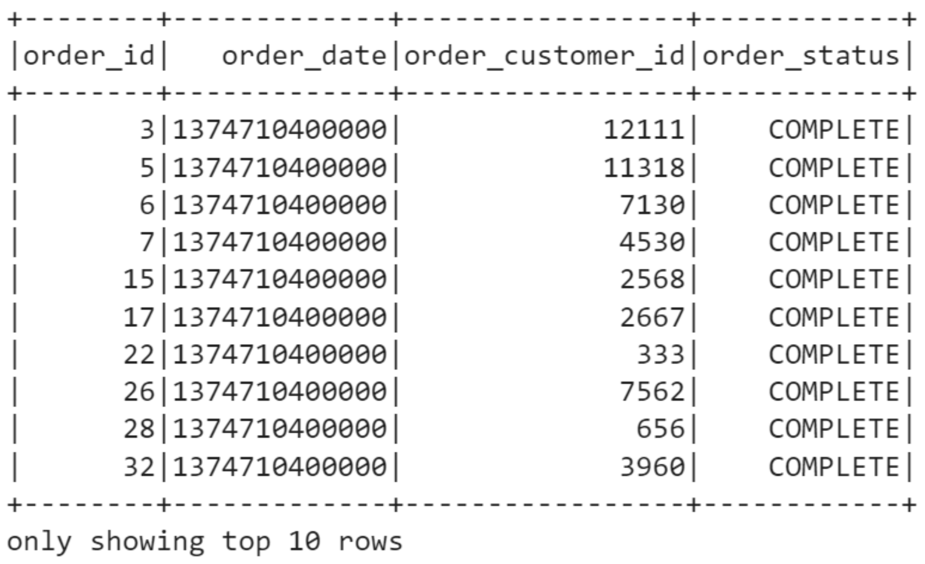
a) Show all orders with the order status value “COMPLETE”.

```
input_path = "/content/drive/MyDrive/data-files/orders_parquet/741ca897-c70e-4633-
b352-5dc3414c5680.parquet"
output_path = "/content/drive/MyDrive/data-files/Answers/ Scenario_2/Results.
parquet"

# Read the parquet data
orders = spark.read.parquet(input_path)
orders.show(10)
```



```
+--------+-------------+-----------------+---------------+
|order_id|   order_date|order_customer_id|   order_status|
+--------+-------------+-----------------+---------------+
|       1|1374710400000|            11599|         CLOSED|
|       2|1374710400000|              256|PENDING_PAYMENT|
|       3|1374710400000|            12111|       COMPLETE|
|       4|1374710400000|             8827|         CLOSED|
|       5|1374710400000|            11318|       COMPLETE|
|       6|1374710400000|             7130|       COMPLETE|
|       7|1374710400000|             4530|       COMPLETE|
|       8|1374710400000|             2911|     PROCESSING|
|       9|1374710400000|             5657|PENDING_PAYMENT|
|      10|1374710400000|             5648|PENDING_PAYMENT|
+--------+-------------+-----------------+---------------+
only showing top 10 rows
```

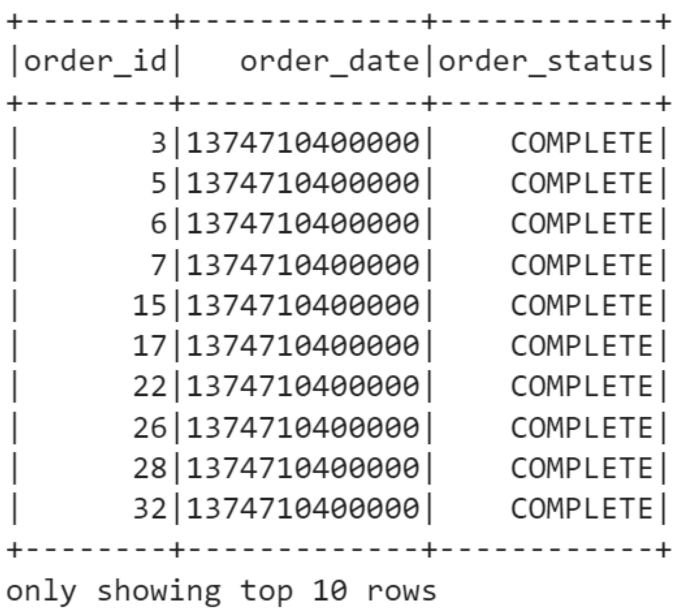
```
# Filter orders with order status "COMPLETE"
completed_orders = orders.filter(col("order_status") == "COMPLETE")
completed_orders.show()

#Output:
```



```
+--------+-------------+-----------------+------------+
|order_id|   order_date|order_customer_id|order_status|
+--------+-------------+-----------------+------------+
|       3|1374710400000|            12111|    COMPLETE|
|       5|1374710400000|            11318|    COMPLETE|
|       6|1374710400000|             7130|    COMPLETE|
|       7|1374710400000|             4530|    COMPLETE|
|      15|1374710400000|             2568|    COMPLETE|
|      17|1374710400000|             2667|    COMPLETE|
|      22|1374710400000|              333|    COMPLETE|
|      26|1374710400000|             7562|    COMPLETE|
|      28|1374710400000|              656|    COMPLETE|
|      32|1374710400000|             3960|    COMPLETE|
+--------+-------------+-----------------+------------+
only showing top 10 rows
```

b) Include order number, order date, and current situation in the output.

```python
# Include order number, order date and current situation
include_columns = ["order_id", "order_date", "order_status"]
filtered_orders_selected = completed_orders.select(*include_columns)
filtered_orders_selected.show(10)

#Output:
```



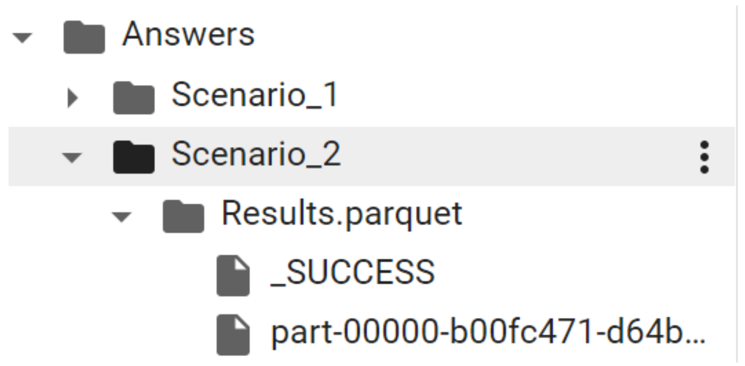
```
+--------+-------------+------------+
|order_id|   order_date|order_status|
+--------+-------------+------------+
|       3|1374710400000|    COMPLETE|
|       5|1374710400000|    COMPLETE|
|       6|1374710400000|    COMPLETE|
|       7|1374710400000|    COMPLETE|
|      15|1374710400000|    COMPLETE|
|      17|1374710400000|    COMPLETE|
|      22|1374710400000|    COMPLETE|
|      26|1374710400000|    COMPLETE|
|      28|1374710400000|    COMPLETE|
|      32|1374710400000|    COMPLETE|
+--------+-------------+------------+
only showing top 10 rows
```

c) Save the data in the "result/scenario2/solution" directory on HDFS.

```python
# Save the filtered data as a parquet file
filtered_orders_selected.write.parquet(output_path, mode="overwrite")
print(f"Filtered data saved to: {output_path}")
```

```
Filtered data saved to: /content/drive/MyDrive/data-files/Answers/Scenario_2/Results.parquet
```

```
#Output:
```



```
Answers
    Scenario_1
    Scenario_2
        Results.parquet
            _SUCCESS
            part-00000-b00fc471-d64b...
```

Scenario - 3

3. Explore the customer records saved in the “customers tab delimited” directory on HDFS.

a) Produce a list of all consumers who live in the city of "Caguas".The result should only contain records with the value "Caguas" for the customer city.

```
#Schema
customer_schema = StructType([
    StructField("customer_id", IntegerType(), True),
    StructField("customer_fname", StringType(), True),
    StructField("customer_lname", StringType(), True),
    StructField("customer_email", StringType(), True),
    StructField("customer_password", StringType(), True),
    StructField("customer_street", StringType(), True),
    StructField("customer_city", StringType(), True),
    StructField("customer_state", StringType(), True),
    StructField("customer_zipcode", StringType(), True)
])

# Load data from HDFS
data =
spark.read.option("delimiter","\t").csv(input_directory,schema=customer_schema)

# Filter customers who live in the city of "Caguas"
Cagus_customers = data.filter(data["customer_city"] == "Caguas")

# Define paths
output_path = "/content/drive/MyDrive/data-files/Answers/Scenario_3/Result.parquet"

# Save the filtered data as a parquet file
Cagus_customers.write.parquet(output_path, mode="overwrite")
Cagus_customers.show(10)
```

#Output:

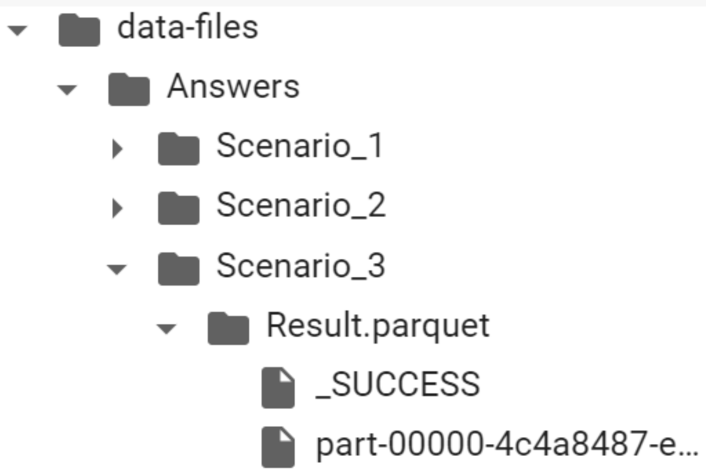
```
+-----------+--------------+--------------+--------------+-----------------+--------------------+-------------+--------------+
|customer_id|customer_fname|customer_lname|customer_email|customer_password|     customer_street|customer_city|customer_state|
+-----------+--------------+--------------+--------------+-----------------+--------------------+-------------+--------------+
|          3|           Ann|         Smith|     XXXXXXXXX|        XXXXXXXXX|3422 Blue Pioneer...|       Caguas|            PR|
|          5|        Robert|        Hudson|     XXXXXXXXX|        XXXXXXXXX|10 Crystal River ...|       Caguas|            PR|
|          7|       Melissa|        Wilcox|     XXXXXXXXX|        XXXXXXXXX|9453 High Concession|       Caguas|            PR|
|          9|          Mary|         Perez|     XXXXXXXXX|        XXXXXXXXX| 3616 Quaking Street|       Caguas|            PR|
|         11|          Mary|       Huffman|     XXXXXXXXX|        XXXXXXXXX|    3169 Stony Woods|       Caguas|            PR|
|         13|          Mary|       Baldwin|     XXXXXXXXX|        XXXXXXXXX|7922 Iron Oak Gar...|       Caguas|            PR|
|         16|       Tiffany|         Smith|     XXXXXXXXX|        XXXXXXXXX|      6651 Iron Port|       Caguas|            PR|
|         19|     Stephanie|      Mitchell|     XXXXXXXXX|        XXXXXXXXX|3543 Red Treasure...|       Caguas|            PR|
|         21|       William|     Zimmerman|     XXXXXXXXX|        XXXXXXXXX|3323 Old Willow M...|       Caguas|            PR|
|         24|          Mary|         Smith|     XXXXXXXXX|        XXXXXXXXX| 9417 Emerald Towers|       Caguas|            PR|
+-----------+--------------+--------------+--------------+-----------------+--------------------+-------------+--------------+
only showing top 10 rows
```

b) Save the results in the result/scenario3/solution folder

```
print(f"Filtered data saved to: {output_path}")
```

```
Filtered data saved to: /content/drive/MyDrive/data-files/Answers/Scenario_3/Result.parquet
```

```
#Output:
```



```
data-files
  Answers
    Scenario_1
    Scenario_2
    Scenario_3
      Result.parquet
        _SUCCESS
        part-00000-4c4a8487-e...
```

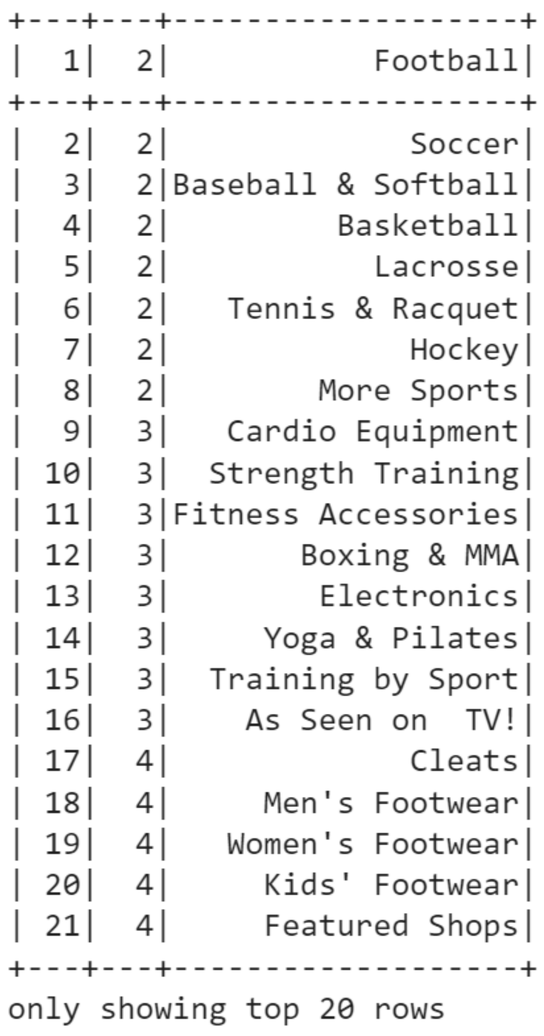
Scenario - 4

4. Explore the order records saved in the "categories" directory on HDFS.

a) Save the result files in CSV format.

```
input_path = "/content/drive/MyDrive/data-files/categories"
output_path = "/content/drive/MyDrive/data-files/Answers/Scenario_4/Results.csv"

# Reading the data
categories_df = spark.read.csv(input_path ,header =True)
categories_df.show()
```

#Output:



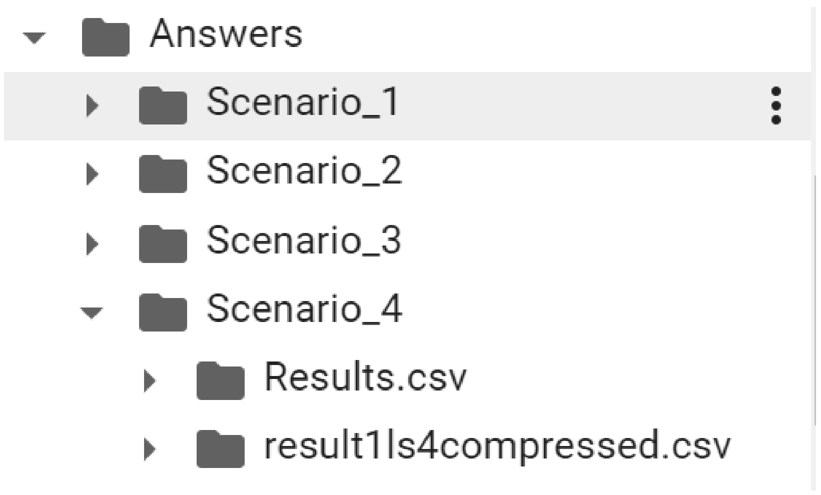
```
+---+---+-------------------+
|  1|  2|           Football|
+---+---+-------------------+
|  2|  2|             Soccer|
|  3|  2|Baseball & Softball|
|  4|  2|         Basketball|
|  5|  2|           Lacrosse|
|  6|  2|   Tennis & Racquet|
|  7|  2|             Hockey|
|  8|  2|        More Sports|
|  9|  3|   Cardio Equipment|
| 10|  3|  Strength Training|
| 11|  3|Fitness Accessories|
| 12|  3|       Boxing & MMA|
| 13|  3|        Electronics|
| 14|  3|     Yoga & Pilates|
| 15|  3|  Training by Sport|
| 16|  3|    As Seen on  TV!|
| 17|  4|             Cleats|
| 18|  4|     Men's Footwear|
| 19|  4|   Women's Footwear|
| 20|  4|     Kids' Footwear|
| 21|  4|     Featured Shops|
+---+---+-------------------+
only showing top 20 rows
```

b) Use lz4 compression to compress the output. Save the data in the result /scenario4 /solution directory on HDFS.

```
categories_df.write.csv(output_path, mode="overwrite")

print(f"Filtered data saved to: {output_path}")

input_path = "/content/drive/MyDrive/data-files/Answers/Scenario_4/Results.csv"
category_df = spark.read.option("header", "true").csv(input_path)

output1_path = "/content/drive/MyDrive/data-files /Answers /Scenario_4
/result1ls4compressed.csv"
category_df.write.option("compression", "lz4").option("header", "true")
.csv(output1_path)
```

#Output:

```
Filtered data saved to: /content/drive/MyDrive/data-files/Answers/Scenario_4/Results.csv
```

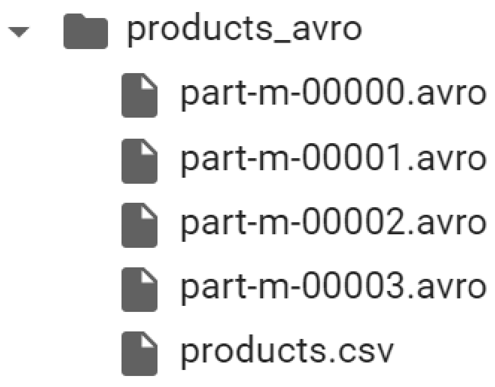


Scenario - 5

5. Explore the customer records saved in the “products_avro" directory on HDFS.

- **Include the products with a price of more than 1000.0 in the output**
- **Remove data from the table if the product price is lesser than 1000.0.**
- **Save the results in the result/scenario5/solution folder.**

```
from pyspark.sql import SparkSession

# Creating the SparkSession
spark = SparkSession.builder.appName("AvroToCSV").getOrCreate()

avro_path = "/content/drive/MyDrive/data-files/products_avro"

# Read Avro data from the specified path
avro_df = spark.read.format("avro").load(avro_path)
avro_df_flat = avro_df.selectExpr("*")

csv_output_path = "/content/drive/MyDrive/data-files/products_avro/products.csv"

# Creating the Csv file from Avro
avro_df_flat.write.mode("overwrite").option("header", "true").csv(csv_output_path)

spark.stop()
```



```
products_avro
    part-m-00000.avro
    part-m-00001.avro
    part-m-00002.avro
    part-m-00003.avro
    products.csv
```

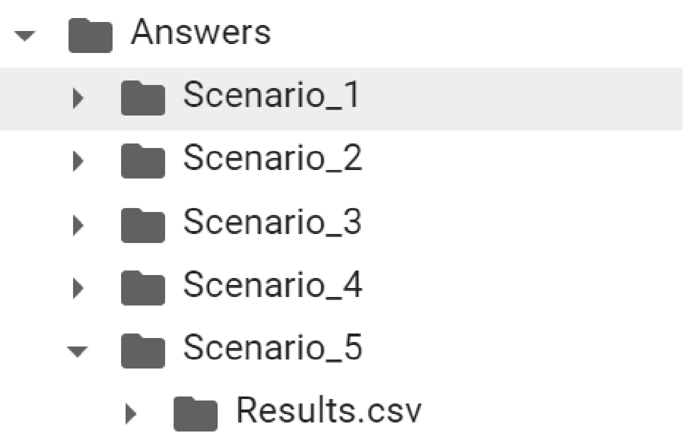
```
input_path = "/content/drive/MyDrive/data-files/products_avro/products.csv"
output_path = "/content/drive/MyDrive/data-files/Answers/Scenario_5/Result"

input_path = "/content/drive/MyDrive/data-files/products_avro/products.csv"
df = spark.read.csv(input_path, header=True, inferSchema=True)
df.show(1000)


# Read the CSV data
product_df = spark.read.option("header", "true").csv(input_path)
```

```
+----------+-------------------+--------------------+-------------------+-------------+--------------------+
|product_id|product_category_id|        product_name|product_description|product_price|       product_image|
+----------+-------------------+--------------------+-------------------+-------------+--------------------+
|      1009|                 45|Diamond Fear No E...|               null|       599.99|http://images.acm...|
|      1010|                 46|DBX Vector Series...|               null|        19.98|http://images.acm...|
|      1011|                 46|Old Town Canoe Sa...|               null|       499.99|http://images.acm...|
|      1012|                 46|Pelican Trailblaz...|               null|       299.99|http://images.acm...|
|      1013|                 46|Perception Sport ...|               null|       349.99|http://images.acm...|
|      1014|                 46|O'Brien Men's Neo...|               null|        49.98|http://images.acm...|

|      1046|                 47|Quest 15 FT Tramp...|               null|       499.99|http://images.acm...|
|      1047|                 47|Under Armour Men'...|               null|        34.99|http://images.acm...|
|      1048|                 47|"Spalding Beast 6...|               null|      1099.99|http://images.acm...|
|      1049|                 47|Under Armour Boys...|               null|        29.99|http://images.acm...|
|      1050|                 47|McDavid HEX Exten...|               null|        29.99|http://images.acm...|
|      1051|                 47|Garmin Forerunner...|               null|       249.99|http://images.acm...|
|      1052|                 47|"Lifetime Elite 5...|               null|       399.99|http://images.acm...|
|      1053|                 47|Garmin Women's Fo...|               null|       129.99|http://images.acm...|
|      1054|                 47|"Spalding NBA 54"...|               null|       699.99|http://images.acm...|
|      1055|                 47|Nike Women's Pro ...|               null|        31.97|http://images.acm...|
|      1056|                 47|Garmin vivofit Fi...|               null|       169.99|http://images.acm...|
|      1103|                 49|Quest 12' x 12' D...|               null|       149.99|http://images.acm...|
|      1104|                 49|GoPro HERO3+ Blac...|               null|       399.99|http://images.acm...|
|      1105|                 49|ASICS Women's GEL...|               null|       119.99|http://images.acm...|
|      1106|                 50|Majestic Youth 20...|               null|         60.0|http://images.acm...|
|      1107|                 50|Majestic Youth 20...|               null|         60.0|http://images.acm...|
|      1108|                 50|Majestic Youth 20...|               null|         60.0|http://images.acm...|
+----------+-------------------+--------------------+-------------------+-------------+--------------------+
```

```python
# Filter products with price greater than 1000.0
MoreThan100_df = df.filter(col("product_price") > 1000.0)

# Define the output directory
output_path = "/content/drive/MyDrive/data-files/Answers/Scenario_5/Results.csv"

# Save the filtered DataFrame as CSV
MoreThan100_df.write.csv(output_path, header=True, mode="overwrite")

MoreThan100_df.show()
```

```
+----------+-------------------+--------------------+-------------------+-------------+--------------------+
|product_id|product_category_id|        product_name|product_description|product_price|       product_image|
+----------+-------------------+--------------------+-------------------+-------------+--------------------+
|      1048|                 47|"Spalding Beast 6...|               null|      1099.99|http://images.acm...|
+----------+-------------------+--------------------+-------------------+-------------+--------------------+
```

#Output:



```
Answers
  Scenario_1
  Scenario_2
  Scenario_3
  Scenario_4
  Scenario_5
    Results.csv
```

Scenario - 6

6. Explore the order records saved in the “products_avro” directory on HDFS.

- **Only products with a price of more than 1000.0 should be in the output**
- **The pattern "Treadmill" appears in the product name**
- **Save the data in the result/scenario6/solution directory on HDFS**

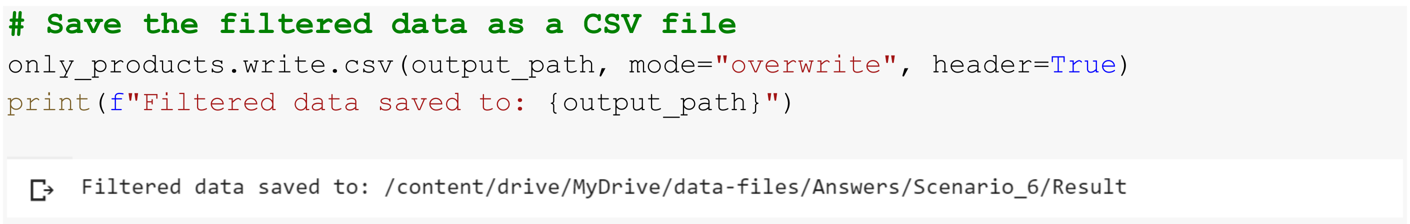
```
input_path = "/content/drive/MyDrive/data-files/products_avro/products.csv"
output_path = "/content/drive/MyDrive/data-files/Answers/Scenario_6/Result"

# Read the CSV data
only_df = spark.read.option("header", "true").csv(input_path)
only_df.show(1000)
```

```
+----------+-------------------+--------------------+-------------------+------------+--------------------+
|product_id|product_category_id|        product_name|product_description|product_price|       product_image|
+----------+-------------------+--------------------+-------------------+------------+--------------------+
|      1009|                 45|Diamond Fear No E...|               null|      599.99|http://images.acm...|
|      1010|                 46|DBX Vector Series...|               null|       19.98|http://images.acm...|
|      1011|                 46|Old Town Canoe Sa...|               null|      499.99|http://images.acm...|
|      1012|                 46|Pelican Trailblaz...|               null|      299.99|http://images.acm...|
|      1013|                 46|Perception Sport ...|               null|      349.99|http://images.acm...|
|      1014|                 46|O'Brien Men's Neo...|               null|       49.98|http://images.acm...|

|      1046|                 47|Quest 15 FT Tramp...|               null|      499.99|http://images.acm...|
|      1047|                 47|Under Armour Men'...|               null|       34.99|http://images.acm...|
|      1048|                 47|"Spalding Beast 6...|               null|     1099.99|http://images.acm...|
|      1049|                 47|Under Armour Boys...|               null|       29.99|http://images.acm...|
|      1050|                 47|McDavid HEX Exten...|               null|       29.99|http://images.acm...|
|      1051|                 47|Garmin Forerunner...|               null|      249.99|http://images.acm...|
|      1052|                 47|"Lifetime Elite 5...|               null|      399.99|http://images.acm...|
|      1053|                 47|Garmin Women's Fo...|               null|      129.99|http://images.acm...|
|      1054|                 47|"Spalding NBA 54"...|               null|      699.99|http://images.acm...|
|      1055|                 47|Nike Women's Pro ...|               null|       31.97|http://images.acm...|
|      1056|                 47|Garmin vivofit Fi...|               null|      169.99|http://images.acm...|
|      1103|                 49|Quest 12' x 12' D...|               null|      149.99|http://images.acm...|
|      1104|                 49|GoPro HERO3+ Blac...|               null|      399.99|http://images.acm...|
|      1105|                 49|ASICS Women's GEL...|               null|      119.99|http://images.acm...|
|      1106|                 50|Majestic Youth 20...|               null|        60.0|http://images.acm...|
|      1107|                 50|Majestic Youth 20...|               null|        60.0|http://images.acm...|
|      1108|                 50|Majestic Youth 20...|               null|        60.0|http://images.acm...|
+----------+-------------------+--------------------+-------------------+------------+--------------------+
```

```
# Filter products with a price greater than 1000.0 and containing
"Treadmill" in product name
only_products = only_df.filter((col("product_price") > 1000.0) &col ("product_name")
.contains("Treadmill"))
only_products.show()
```

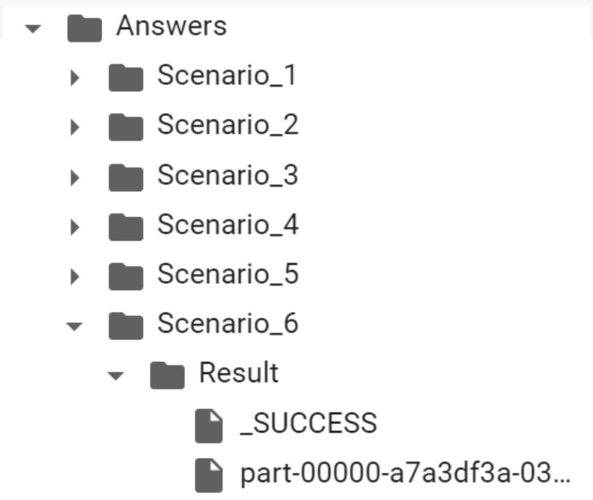
#Output:

```
+----------+-------------------+------------+-------------------+-------------+-------------+
|product_id|product_category_id|product_name|product_description|product_price|product_image|
+----------+-------------------+------------+-------------------+-------------+-------------+
+----------+-------------------+------------+-------------------+-------------+-------------+
```



```
# Save the filtered data as a CSV file
only_products.write.csv(output_path, mode="overwrite", header=True)
print(f"Filtered data saved to: {output_path}")
```

```
Filtered data saved to: /content/drive/MyDrive/data-files/Answers/Scenario_6/Result
```



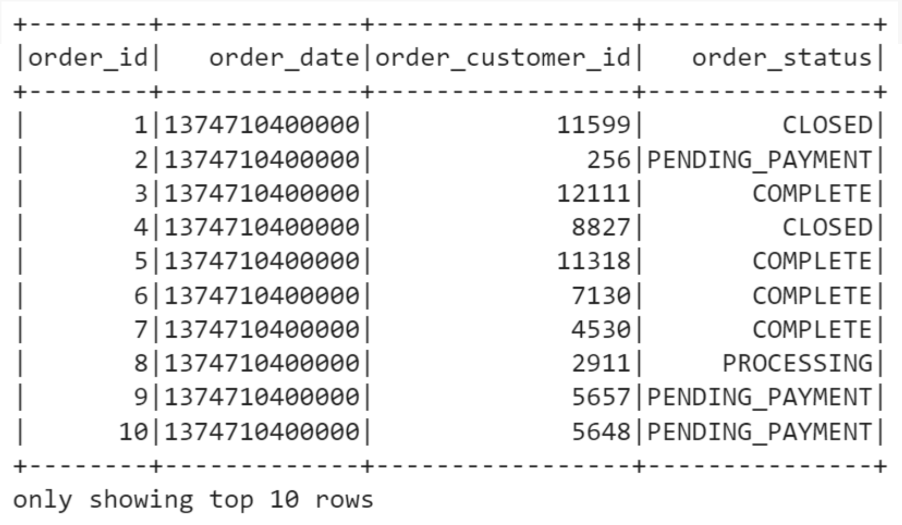
```
Answers
    Scenario_1
    Scenario_2
    Scenario_3
    Scenario_4
    Scenario_5
    Scenario_6
        Result
            _SUCCESS
            part-00000-a7a3df3a-03...
```

Scenario - 7

7. Explore the customer records saved in the “orders parquet” directory on HDFS

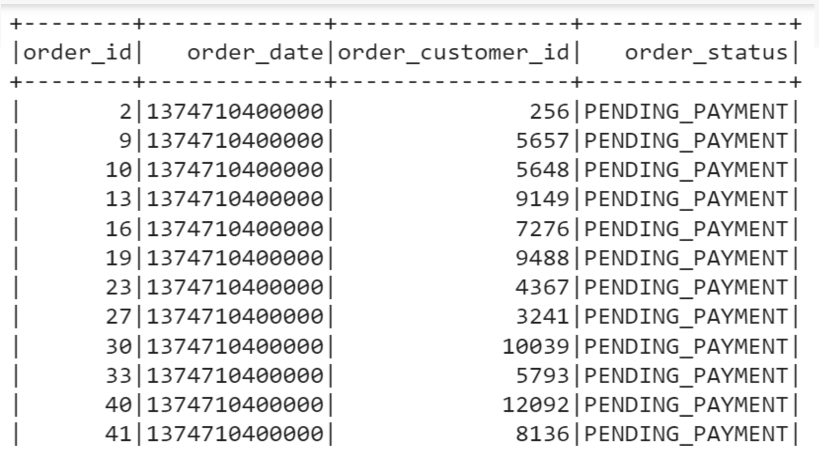
a) Output all PENDING orders in July 2013, Only entries with the order status value of "PENDING" should be included in the result

```
input_path = "/content/drive/MyDrive/data-files/order_parquet/741ca897-c70e-4633-
b352-5dc3414c5680.parquet"
output_path = "/content/drive/MyDrive/data-files/Answers/Scenario_7/Results
.parquet"

# Read the Parquet data
orders_df = spark.read.parquet(input_path)
orders_df.show(10)
```



```
+--------+-------------+-----------------+---------------+
|order_id|   order_date|order_customer_id|   order_status|
+--------+-------------+-----------------+---------------+
|       1|1374710400000|            11599|         CLOSED|
|       2|1374710400000|              256|PENDING_PAYMENT|
|       3|1374710400000|            12111|       COMPLETE|
|       4|1374710400000|             8827|         CLOSED|
|       5|1374710400000|            11318|       COMPLETE|
|       6|1374710400000|             7130|       COMPLETE|
|       7|1374710400000|             4530|       COMPLETE|
|       8|1374710400000|             2911|     PROCESSING|
|       9|1374710400000|             5657|PENDING_PAYMENT|
|      10|1374710400000|             5648|PENDING_PAYMENT|
+--------+-------------+-----------------+---------------+
only showing top 10 rows
```

```
# Filter PENDING orders in July 2013
filtered_orders = orders_df.filter((col("order_status") == "PENDING_PAYMENT") &
(year(from_unixtime(col("order_date") / 1000)) == 2013)
(month(from_unixtime(col("order_date") / 1000)) == 7))
filtered_orders.show(12)

#Output:
```

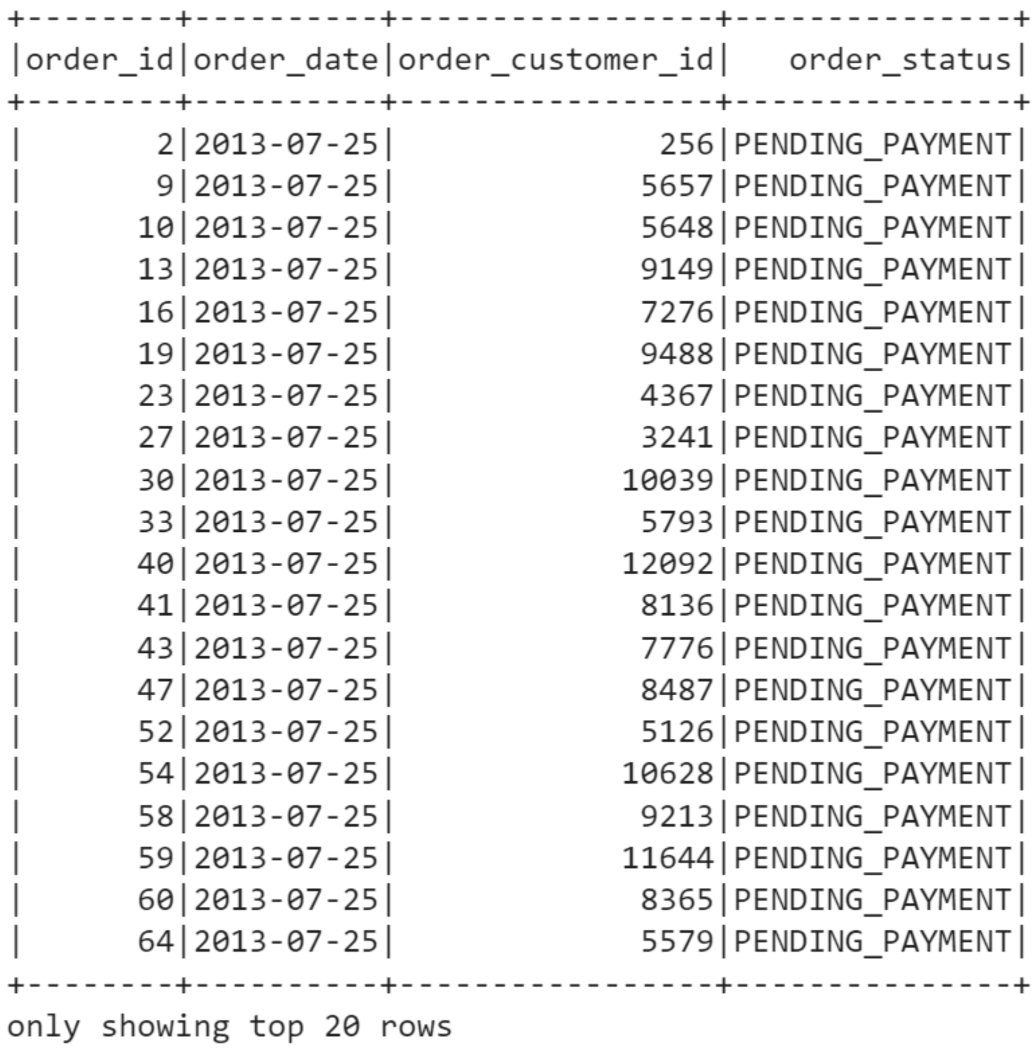


```
+--------+-------------+-----------------+---------------+
|order_id|   order_date|order_customer_id|   order_status|
+--------+-------------+-----------------+---------------+
|       2|1374710400000|              256|PENDING_PAYMENT|
|       9|1374710400000|             5657|PENDING_PAYMENT|
|      10|1374710400000|             5648|PENDING_PAYMENT|
|      13|1374710400000|             9149|PENDING_PAYMENT|
|      16|1374710400000|             7276|PENDING_PAYMENT|
|      19|1374710400000|             9488|PENDING_PAYMENT|
|      23|1374710400000|             4367|PENDING_PAYMENT|
|      27|1374710400000|             3241|PENDING_PAYMENT|
|      30|1374710400000|            10039|PENDING_PAYMENT|
|      33|1374710400000|             5793|PENDING_PAYMENT|
|      40|1374710400000|            12092|PENDING_PAYMENT|
|      41|1374710400000|             8136|PENDING_PAYMENT|
```

b) Order date should be in the YYY-MM-DD format

```
# Select relevant columns and format order date
organizing_columns = ["order_id", "order_date", "order_customer_id","order_status"]

calendar_format =
filtered_orders.select(*organizing_columns).withColumn("order_date", date_format
(from_unixtime(col("order_date") / 1000), "yyyy-MM-dd"))

calendar_format.show()
```

#Output:

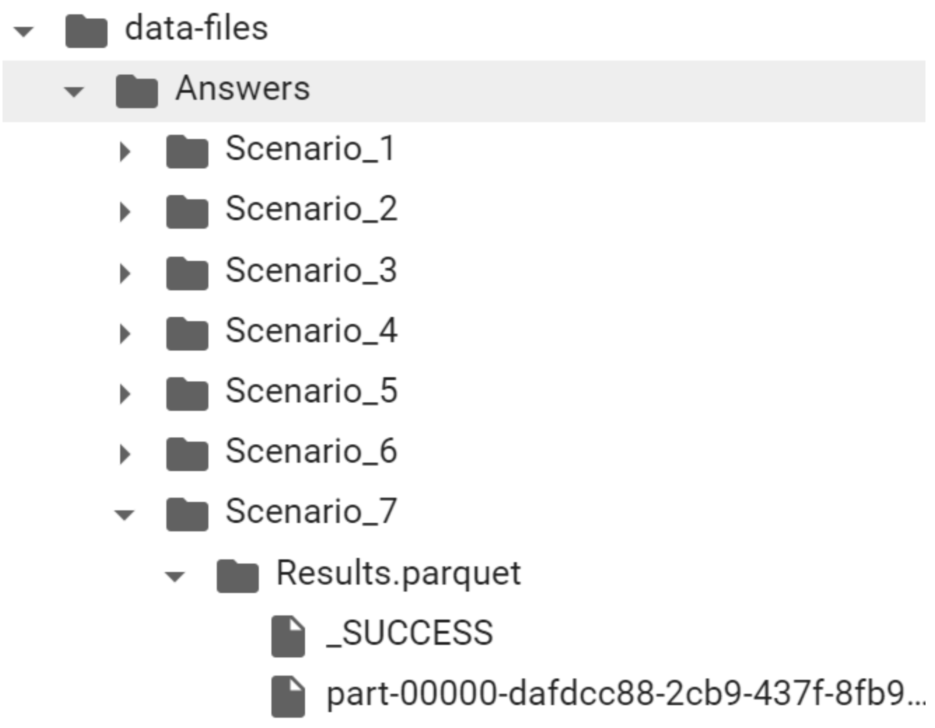


```
+--------+----------+-----------------+---------------+
|order_id|order_date|order_customer_id|   order_status|
+--------+----------+-----------------+---------------+
|       2|2013-07-25|              256|PENDING_PAYMENT|
|       9|2013-07-25|             5657|PENDING_PAYMENT|
|      10|2013-07-25|             5648|PENDING_PAYMENT|
|      13|2013-07-25|             9149|PENDING_PAYMENT|
|      16|2013-07-25|             7276|PENDING_PAYMENT|
|      19|2013-07-25|             9488|PENDING_PAYMENT|
|      23|2013-07-25|             4367|PENDING_PAYMENT|
|      27|2013-07-25|             3241|PENDING_PAYMENT|
|      30|2013-07-25|            10039|PENDING_PAYMENT|
|      33|2013-07-25|             5793|PENDING_PAYMENT|
|      40|2013-07-25|            12092|PENDING_PAYMENT|
|      41|2013-07-25|             8136|PENDING_PAYMENT|
|      43|2013-07-25|             7776|PENDING_PAYMENT|
|      47|2013-07-25|             8487|PENDING_PAYMENT|
|      52|2013-07-25|             5126|PENDING_PAYMENT|
|      54|2013-07-25|            10628|PENDING_PAYMENT|
|      58|2013-07-25|             9213|PENDING_PAYMENT|
|      59|2013-07-25|            11644|PENDING_PAYMENT|
|      60|2013-07-25|             8365|PENDING_PAYMENT|
|      64|2013-07-25|             5579|PENDING_PAYMENT|
+--------+----------+-----------------+---------------+
only showing top 20 rows
```

c) Save the results in the result/scenario7/solution folder

```
# Save the filtered data as a Parquet file
calendar_format.write.parquet(output_path, mode="overwrite")
print(f"Filtered data saved to: {output_path}")

#Output:
```

```
Filtered data saved to: /content/drive/MyDrive/data-files/Answers/Scenario_7/Results.parquet
```



```
data-files
  Answers
    Scenario_1
    Scenario_2
    Scenario_3
    Scenario_4
    Scenario_5
    Scenario_6
    Scenario_7
      Results.parquet
        _SUCCESS
        part-00000-dafdcc88-2cb9-437f-8fb9...
```