NAME: Mohammad Hussam (2303.KHI.DEG.020)

PAIRING WITH : MAVIA ALAM KHAN (2303.KHI.DEG.017)

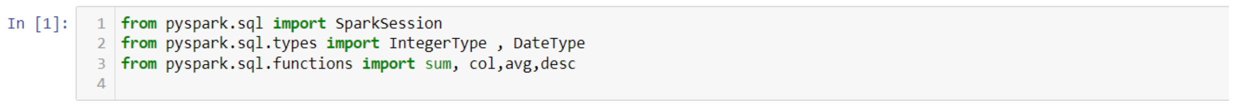
&

AQSA TAUHEED(2303.KHI.DEG.011)

ASSIGNMENT NO : 5.1

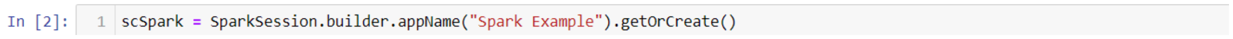
SOLUTION:

Importing libraries



```
In [1]:
from pyspark.sql import SparkSession
from pyspark.sql.types import IntegerType , DateType
from pyspark.sql.functions import sum, col,avg,desc

```

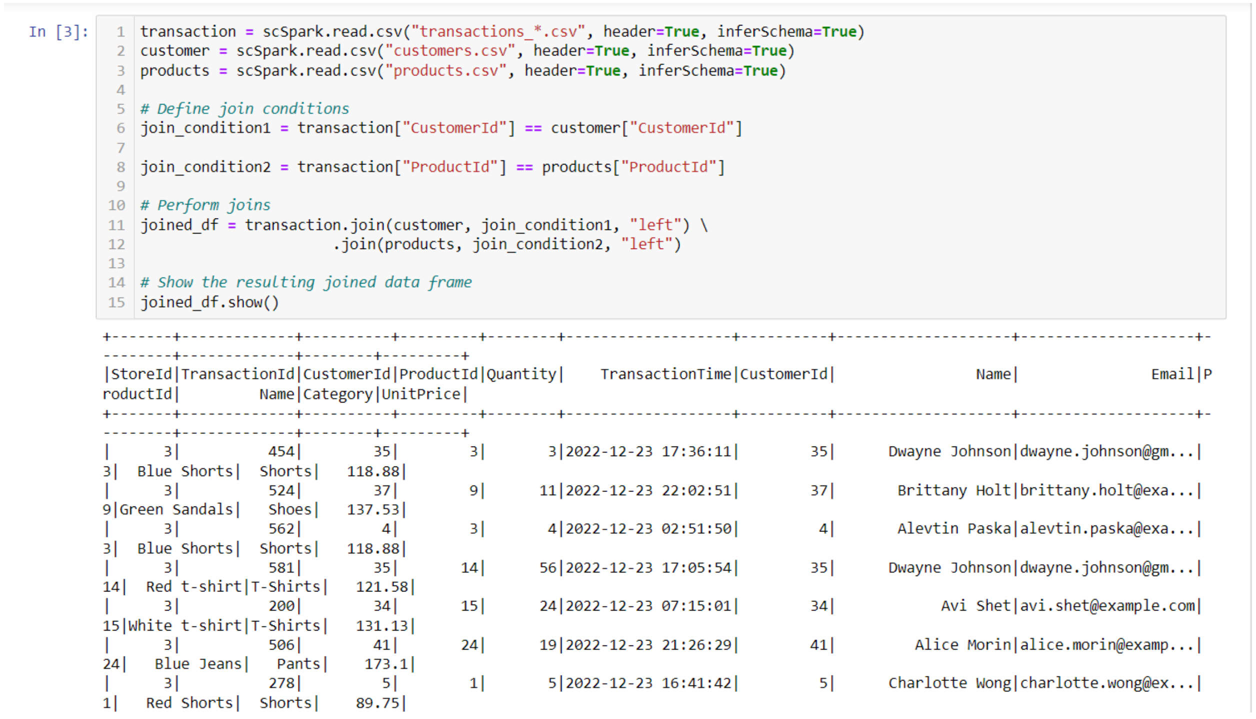


```
In [2]:
scSpark = SparkSession.builder.appName("Spark Example").getOrCreate()
```

Two join conditions are defined:

join_condition1 specifies that the "CustomerId" column in the transaction DataFrame should match the "CustomerId" column in the customer DataFrame.

join_condition2 specifies that the "ProductId" column in the transaction DataFrame should match the "ProductId" column in the products DataFrame.



```
In [3]:
transaction = scSpark.read.csv("transactions_*.csv", header=True, inferSchema=True)
customer = scSpark.read.csv("customers.csv", header=True, inferSchema=True)
products = scSpark.read.csv("products.csv", header=True, inferSchema=True)

# Define join conditions
join_condition1 = transaction["CustomerId"] == customer["CustomerId"]

join_condition2 = transaction["ProductId"] == products["ProductId"]

# Perform joins
joined_df = transaction.join(customer, join_condition1, "left") \
                      .join(products, join_condition2, "left")

# Show the resulting joined data frame
joined_df.show()
```

```
+-------+-------------+----------+---------+--------+-------------------+----------+--------------------+--------------------+-
--------+-------------+--------+---------+
|StoreId|TransactionId|CustomerId|ProductId|Quantity|    TransactionTime|CustomerId|                Name|               Email|P
roductId|         Name|Category|UnitPrice|
+-------+-------------+----------+---------+--------+-------------------+----------+--------------------+--------------------+-
--------+-------------+--------+---------+
|      3|          454|        35|        3|       3|2022-12-23 17:36:11|        35|      Dwayne Johnson|dwayne.johnson@gm...|
3|  Blue Shorts|  Shorts|   118.88|
|      3|          524|        37|        9|      11|2022-12-23 22:02:51|        37|       Brittany Holt|brittany.holt@exa...|
9|Green Sandals|   Shoes|   137.53|
|      3|          562|         4|        3|       4|2022-12-23 02:51:50|         4|       Alevtin Paska|alevtin.paska@exa...|
3|  Blue Shorts|  Shorts|   118.88|
|      3|          581|        35|       14|      56|2022-12-23 17:05:54|        35|      Dwayne Johnson|dwayne.johnson@gm...|
14|  Red t-shirt|T-Shirts|   121.58|
|      3|          200|        34|       15|      24|2022-12-23 07:15:01|        34|            Avi Shet|avi.shet@example.com|
15|White t-shirt|T-Shirts|   131.13|
|      3|          506|        41|       24|      19|2022-12-23 21:26:29|        41|         Alice Morin|alice.morin@examp...|
24|   Blue Jeans|   Pants|    173.1|
|      3|          278|         5|        1|       5|2022-12-23 16:41:42|         5|      Charlotte Wong|charlotte.wong@ex...|
1|   Red Shorts|  Shorts|    89.75|
```

Now we add total price column in which we multiply quantity & unit price column

In [4]:
```python
join_df = joined_df.withColumn("total price", col("Quantity") * col("UnitPrice"))
join_df.show()
```

```
+-------+-------------+----------+---------+--------+-------------------+----------+--------------------+--------------------+-
--------+-------------+--------+---------+-----------------+
|StoreId|TransactionId|CustomerId|ProductId|Quantity|    TransactionTime|CustomerId|                Name|               Email|P
roductId|         Name|Category|UnitPrice|      total price|
+-------+-------------+----------+---------+--------+-------------------+----------+--------------------+--------------------+-
--------+-------------+--------+---------+-----------------+
|      3|          454|        35|        3|       3|2022-12-23 17:36:11|        35|      Dwayne Johnson|dwayne.johnson@gm...|
3|  Blue Shorts|  Shorts|   118.88|           356.64|
|      3|          524|        37|        9|      11|2022-12-23 22:02:51|        37|       Brittany Holt|brittany.holt@exa...|
9|Green Sandals|   Shoes|   137.53|          1512.83|
|      3|          562|         4|        3|       4|2022-12-23 02:51:50|         4|       Alevtin Paska|alevtin.paska@exa...|
3|  Blue Shorts|  Shorts|   118.88|           475.52|
|      3|          581|        35|       14|      56|2022-12-23 17:05:54|        35|      Dwayne Johnson|dwayne.johnson@gm...|
14|  Red t-shirt|T-Shirts|   121.58|           6808.48|
|      3|          200|        34|       15|      24|2022-12-23 07:15:01|        34|            Avi Shet|avi.shet@example.com|
15|White t-shirt|T-Shirts|   131.13|           3147.12|
|      3|          506|        41|       24|      19|2022-12-23 21:26:29|        41|         Alice Morin|alice.morin@examp...|
24|   Blue Jeans|   Pants|    173.1|            3288.9|
|      3|          278|         5|        1|       5|2022-12-23 16:41:42|         5|     Charlotte Wong|charlotte.wong@ex...|
1|    Red Shorts|  Shorts|    89.75|           448.75|
|      3|          849|        36|       23|      13|2022-12-23 13:22:55|        36|    William Nielsen|william.nielsen@e...|
23| Green Chinos|   Pants|   150.93|1962.0900000000001|
|      3|          992|        34|        7|       3|2022-12-23 16:47:14|        34|            Avi Shet|avi.shet@example.com|
7|White Sandals|   Shoes|   160.96|           482.88|
|      3|          703|        19|        7|      13|2022-12-23 22:36:48|        19|       Alexia Renaud|alexia.renaud@exa...|
7|White Sandals|   Shoes|   160.96|          2092.48|
|      3|          719|        48|       18|      12|2022-12-23 10:11:29|        48|        Amoli Shenoy|amoli.shenoy@exam...|
18|Black t-shirt|T-Shirts|   102.41|           1228.92|
```

The daily total sales for the store with id 1

and the withColumn method is used to create a new column in the DataFrame with the casted values, and the original column is replaced by the new one.

In [5]:
```python
join_df = join_df.withColumn("total price", col("total price").cast("float"))
join_df= join_df.withColumn("TransactionTime", join_df["TransactionTime"].cast(DateType()))
store_df = join_df.filter(join_df["StoreId"] == 1)
daily_total_sales = store_df.groupBy("TransactionTime").agg(sum("total price").alias("TotalSales"))
daily_total_sales.show()
```

```
+---------------+----------------+
|TransactionTime|      TotalSales|
+---------------+----------------+
|     2022-12-23|41264.00012207031|
+---------------+----------------+
```

the mean sales for the store with id 2

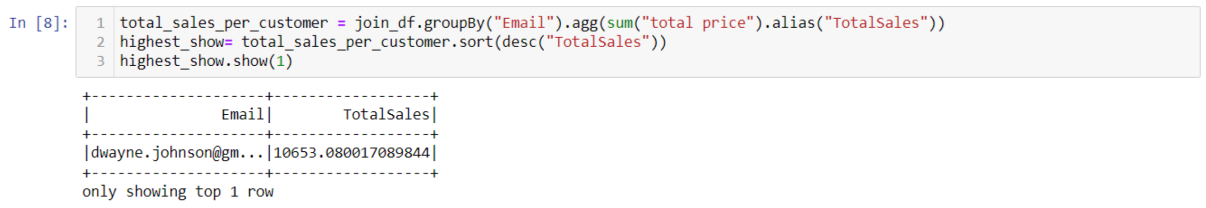
```
+---------------+----------------+
```

In [6]:
```python
store_sales_df = join_df.filter(join_df["StoreId"] == 2)
mean_sales_store2 = store_sales_df.select(avg("total price").alias("MeanSales"))
mean_sales_store2.show()
```

```
+-----------------+
|        MeanSales|
+-----------------+
|513.4598035625382|
+-----------------+
```

the email of the client that spent the most when summing up purchases from all of the stores

the sort method is used on total_sales_per_customer to sort the DataFrame in descending order based on the "TotalSales" column. The desc function specifies the sorting order as descending.

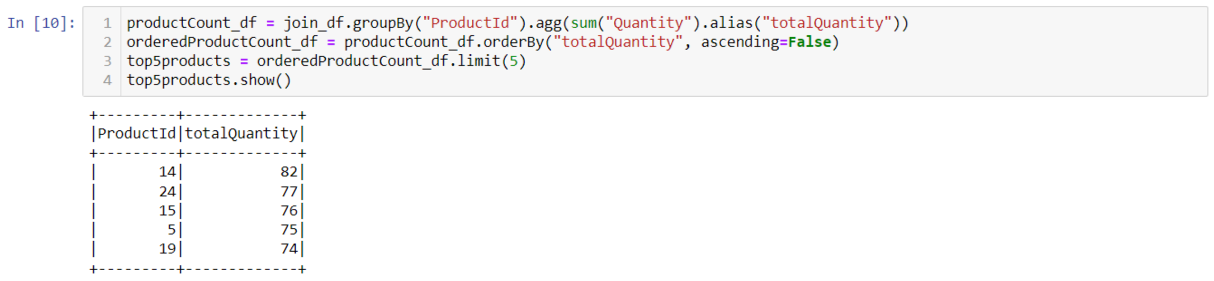


In [8]:

```
total_sales_per_customer = join_df.groupBy("Email").agg(sum("total price").alias("TotalSales"))
highest_show= total_sales_per_customer.sort(desc("TotalSales"))
highest_show.show(1)
```

```
+--------------------+------------------+
|               Email|        TotalSales|
+--------------------+------------------+
|dwayne.johnson@gm...|10653.080017089844|
+--------------------+------------------+
only showing top 1 row
```

And 5 products are most frequently bought across all stores

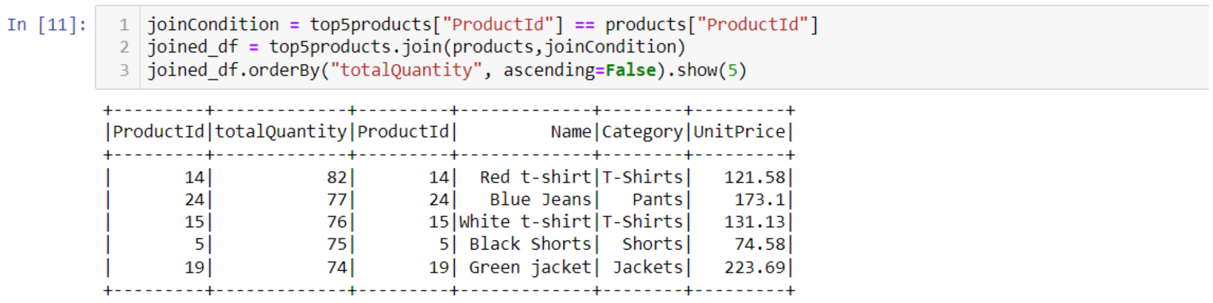


In [10]:

```
productCount_df = join_df.groupBy("ProductId").agg(sum("Quantity").alias("totalQuantity"))
orderedProductCount_df = productCount_df.orderBy("totalQuantity", ascending=False)
top5products = orderedProductCount_df.limit(5)
top5products.show()
```

```
+---------+-------------+
|ProductId|totalQuantity|
+---------+-------------+
|       14|           82|
|       24|           77|
|       15|           76|
|        5|           75|
|       19|           74|
+---------+-------------+
```

Here we print name of the product that purchase most so we join the table



In [11]:

```
joinCondition = top5products["ProductId"] == products["ProductId"]
joined_df = top5products.join(products,joinCondition)
joined_df.orderBy("totalQuantity", ascending=False).show(5)
```

```
+---------+-------------+---------+-------------+--------+---------+
|ProductId|totalQuantity|ProductId|         Name|Category|UnitPrice|
+---------+-------------+---------+-------------+--------+---------+
|       14|           82|       14|  Red t-shirt|T-Shirts|   121.58|
|       24|           77|       24|   Blue Jeans|   Pants|    173.1|
|       15|           76|       15|White t-shirt|T-Shirts|   131.13|
|        5|           75|        5| Black Shorts|  Shorts|    74.58|
|       19|           74|       19| Green jacket| Jackets|   223.69|
+---------+-------------+---------+-------------+--------+---------+
```