databricks ProjectP1(7)

(https://databricks.com)

Access key and Secret Access key

```
ls /FileStore/shared_uploads/yashbuty07@gmail.com/30cdbiotaws00124_accessKeys__1_-2.csv
```

```
aws_keys=spark.read.format('csv').option('header','true').option('inferschema','true').load('/FileStore/shared_uploads/yas
hbuty07@gmail.com/30cdbiotaws00124_accessKeys__1_-1.csv')
aws_keys.columns
```

```
ak=aws_keys.select('Access key ID').take(1)[0]['Access key ID']
sk=aws_keys.select('Secret access key').take(1)[0]['Secret access key']
```

Encryption Code

```
import urllib
encoded=urllib.parse.quote(string=sk,safe="")
```

Mounting S3 Buckets on Databricks enviournment

```
#bucket1
aws_bucket1='aws-bucky7'
mount_name1='/mnt/bucket1'

source_url1="s3a://{0}:{1}@{2}".format(ak,encoded,aws_bucket1)



dbutils.fs.mount(source_url1,mount_name1)
```

```
#bucket2
aws_bucket2='aws-bucky-8'
mount_name2='/mnt/b-8new'

source_url2="s3a://{0}:{1}@{2}".format(ak,encoded,aws_bucket2)
```

```
dbutils.fs.mount(source_url2,mount_name2)
```

```
aws_s3_df1=spark.read.format('csv').option('header','true').option('inferschema','true').load('/mnt/bucket1/P1 data.csv')
aws_s3_df1.show()
```

Importing the client data

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np
```

```
df1=spark.read.format('csv').option('header','true').option('inferschema','true').load('/mnt/bucket1/P1 data.csv')
```

```
display(df1)
```

Table

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segmen
1	32298	CA-2012-124891	2012-07-31	2012-07-31	Same Day	RH-19495	Rick Hansen	Consum
2	26341	IN-2013-77878	2013-02-05	2013-02-07	Second Class	JR-16210	Justin Ritter	Corporat
3	25330	IN-2013-71249	2013-10-17	2013-10-18	First Class	CR-12730	Craig Reiter	Consum
4	13524	ES-2013-1579342	2013-01-28	2013-01-30	First Class	KM-16375	Katherine Murray	Home O
5	47221	SG-2013-4320	2013-11-05	2013-11-06	Same Day	RH-9495	Rick Hansen	Consum
6	22732	IN-2013-42360	2013-06-28	2013-07-01	Second Class	JM-15655	Jim Mitchum	Corporat
7	30570	IN-2011-81826	2011-11-07	2011-11-09	First Class	TS-21340	Toby Swindell	Consum

7,340 rows | Truncated data

```
display(df1.head(7))
```

Table

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment
1	32298	CA-2012-124891	2012-07-31	2012-07-31	Same Day	RH-19495	Rick Hansen	Consumer
2	26341	IN-2013-77878	2013-02-05	2013-02-07	Second Class	JR-16210	Justin Ritter	Corporate
3	25330	IN-2013-71249	2013-10-17	2013-10-18	First Class	CR-12730	Craig Reiter	Consumer
4	13524	ES-2013-1579342	2013-01-28	2013-01-30	First Class	KM-16375	Katherine Murray	Home Office
5	47221	SG-2013-4320	2013-11-05	2013-11-06	Same Day	RH-9495	Rick Hansen	Consumer
6	22732	IN-2013-42360	2013-06-28	2013-07-01	Second Class	JM-15655	Jim Mitchum	Corporate
7	30570	IN-2011-81826	2011-11-07	2011-11-09	First Class	TS-21340	Toby Swindell	Consumer

7 rows

```
from pyspark.sql.functions import col
num_rows = df1.count()
num_cols = len(df1.columns)

print((num_rows, num_cols))
```

```
(51290, 24)
```

```
df1.describe()
```

```
Out[9]: DataFrame[summary: string, Row ID: string, Order ID: string, Ship Mode: string, Customer ID: string, Customer Nam
e: string, Segment: string, City: string, State: string, Country: string, Postal Code: string, Market: string, Region: st
ring, Product ID: string, Category: string, Sub-Category: string, Product Name: string, Sales: string, Quantity: string,
Discount: string, Profit: string, Shipping Cost: string, Order Priority: string]
```

```
df1.printSchema()
```

```
root
 |-- Row ID: integer (nullable = true)
 |-- Order ID: string (nullable = true)
 |-- Order Date: date (nullable = true)
 |-- Ship Date: date (nullable = true)
 |-- Ship Mode: string (nullable = true)
 |-- Customer ID: string (nullable = true)
 |-- Customer Name: string (nullable = true)
 |-- Segment: string (nullable = true)
 |-- City: string (nullable = true)
 |-- State: string (nullable = true)
 |-- Country: string (nullable = true)
 |-- Postal Code: integer (nullable = true)
 |-- Market: string (nullable = true)
 |-- Region: string (nullable = true)
 |-- Product ID: string (nullable = true)
 |-- Category: string (nullable = true)
 |-- Sub-Category: string (nullable = true)
```

```
 |-- Product Name: string (nullable = true)
 |-- Sales: string (nullable = true)
```

Data Cleaning

```
from pyspark.sql.functions import to_date
df2 = df1.withColumn("Order Date", to_date("Order Date", "yyyy-MM-dd"))
display(df2)
```

Table

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segmen
1	32298	CA-2012-124891	2012-07-31	2012-07-31	Same Day	RH-19495	Rick Hansen	Consum
2	26341	IN-2013-77878	2013-02-05	2013-02-07	Second Class	JR-16210	Justin Ritter	Corporat
3	25330	IN-2013-71249	2013-10-17	2013-10-18	First Class	CR-12730	Craig Reiter	Consum
4	13524	ES-2013-1579342	2013-01-28	2013-01-30	First Class	KM-16375	Katherine Murray	Home O
5	47221	SG-2013-4320	2013-11-05	2013-11-06	Same Day	RH-9495	Rick Hansen	Consum
6	22732	IN-2013-42360	2013-06-28	2013-07-01	Second Class	JM-15655	Jim Mitchum	Corporat
7	30570	IN-2011-81826	2011-11-07	2011-11-09	First Class	TS-21340	Toby Swindell	Consum

7,340 rows | Truncated data

```
df2.printSchema()
```

```
root
 |-- Row ID: integer (nullable = true)
 |-- Order ID: string (nullable = true)
 |-- Order Date: date (nullable = true)
 |-- Ship Date: date (nullable = true)
 |-- Ship Mode: string (nullable = true)
 |-- Customer ID: string (nullable = true)
 |-- Customer Name: string (nullable = true)
 |-- Segment: string (nullable = true)
 |-- City: string (nullable = true)
 |-- State: string (nullable = true)
 |-- Country: string (nullable = true)
 |-- Postal Code: integer (nullable = true)
 |-- Market: string (nullable = true)
 |-- Region: string (nullable = true)
 |-- Product ID: string (nullable = true)
 |-- Category: string (nullable = true)
 |-- Sub-Category: string (nullable = true)
 |-- Product Name: string (nullable = true)
 |-- Sales: string (nullable = true)
 |-- Quantity: string (nullable = true)
```

```
from pyspark.sql.functions import first
from pyspark.sql.functions import col


a = df2.groupBy(['Order Date', 'Profit']).count()
display(a.first())
```

```
Row(Order Date=datetime.date(2014, 1, 18), Profit=656.37, count=1)
```

```
from pyspark.sql.functions import col

df2.select([col(c).isNull().alias(c) for c in df2.columns]).show()
```

```
+------+--------+----------+---------+---------+-----------+-------------+-------+-----+-----+-------+-----------+------+
------+----------+--------+------------+------------+-----+--------+--------+------+-------------+--------------+
|Row ID|Order ID|Order Date|Ship Date|Ship Mode|Customer ID|Customer Name|Segment| City|State|Country|Postal Code|Market|
Region|Product ID|Category|Sub-Category|Product Name|Sales|Quantity|Discount|Profit|Shipping Cost|Order Priority|
+------+--------+----------+---------+---------+-----------+-------------+-------+-----+-----+-------+-----------+------+
```

```
------+----------+--------+------------+------------+-----+--------+--------+------+-------------+--------------+
| false|   false|     false|    false|    false|      false|        false|  false|false|false|  false|      false| false|
false|     false|   false|       false|       false|false|   false|   false| false|        false|        false|
| false|   false|     false|    false|    false|      false|        false|  false|false|false|  false|       true| false|
false|     false|   false|       false|       false|false|   false|   false| false|        false|        false|
| false|   false|     false|    false|    false|      false|        false|  false|false|false|  false|       true| false|
false|     false|   false|       false|       false|false|   false|   false| false|        false|        false|
| false|   false|     false|    false|    false|      false|        false|  false|false|false|  false|       true| false|
false|     false|   false|       false|       false|false|   false|   false| false|        false|        false|
| false|   false|     false|    false|    false|      false|        false|  false|false|false|  false|       true| false|
false|     false|   false|       false|       false|false|   false|   false| false|        false|        false|
| false|   false|     false|    false|    false|      false|        false|  false|false|false|  false|       true| false|
false|     false|   false|       false|       false|false|   false|   false| false|        false|        false|
| false|   false|     false|    false|    false|      false|        false|  false|false|false|  false|       true| false|
false|     false|   false|       false|       false|false|   false|   false| false|        false|        false|
```

```
from pyspark.sql.functions import col, sum

df2.select([sum(col(c).isNull().cast("int")).alias(c) for c in df2.columns]).show()
```

```
+------+--------+----------+---------+---------+-----------+-------------+-------+----+-----+-------+-----------+------+-
-----+----------+--------+------------+------------+-----+--------+--------+------+-------------+--------------+
|Row ID|Order ID|Order Date|Ship Date|Ship Mode|Customer ID|Customer Name|Segment|City|State|Country|Postal Code|Market|R
egion|Product ID|Category|Sub-Category|Product Name|Sales|Quantity|Discount|Profit|Shipping Cost|Order Priority|
+------+--------+----------+---------+---------+-----------+-------------+-------+----+-----+-------+-----------+------+-
-----+----------+--------+------------+------------+-----+--------+--------+------+-------------+--------------+
|     0|       0|         0|        0|        0|          0|            0|      0|   0|    0|      0|      41296|     0|
0|         0|       0|           0|           0|    0|       0|       0|     0|            0|             0|
+------+--------+----------+---------+---------+-----------+-------------+-------+----+-----+-------+-----------+------+-
-----+----------+--------+------------+------------+-----+--------+--------+------+-------------+--------------+
```

```
df2 = df2.drop('Postal Code')
```

```
from pyspark.sql.functions import col, sum

display(df2.select([sum(col(c).isNull().cast("int")).alias(c) for c in df2.columns]))
```

Table

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	City
1	0	0	0	0	0	0	0	0	0

1 row

```
from pyspark.sql.functions import col

df3 = df2.withColumn('Ship Mode', col('Ship Mode').cast('string'))
```

```
from pyspark.sql.functions import trim

def remove_leading_spaces(df3):
    for col_name, data_type in df2.dtypes:
        if data_type in ['string', 'char', 'varchar']:
            df = df2.withColumn(col_name, trim(df2[col_name]))
    return df
```

```
df4 = remove_leading_spaces(df3)
```

```
display(df4.head(3))
```

Table

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment
1	32298	CA-2012-124891	2012-07-31	2012-07-31	Same Day	RH-19495	Rick Hansen	Consumer
2	26341	IN-2013-77878	2013-02-05	2013-02-07	Second Class	JR-16210	Justin Ritter	Corporate
3	25330	IN-2013-71249	2013-10-17	2013-10-18	First Class	CR-12730	Craig Reiter	Consumer

3 rows

```
df4.printSchema()
```

```
root
 |-- Row ID: integer (nullable = true)
 |-- Order ID: string (nullable = true)
 |-- Order Date: date (nullable = true)
 |-- Ship Date: date (nullable = true)
 |-- Ship Mode: string (nullable = true)
 |-- Customer ID: string (nullable = true)
 |-- Customer Name: string (nullable = true)
 |-- Segment: string (nullable = true)
 |-- City: string (nullable = true)
 |-- State: string (nullable = true)
 |-- Country: string (nullable = true)
 |-- Market: string (nullable = true)
 |-- Region: string (nullable = true)
 |-- Product ID: string (nullable = true)
 |-- Category: string (nullable = true)
 |-- Sub-Category: string (nullable = true)
 |-- Product Name: string (nullable = true)
 |-- Sales: string (nullable = true)
 |-- Quantity: string (nullable = true)
 |-- Discount: string (nullable = true)
```

```
from pyspark.sql.functions import count

display(df4.groupBy('Country').agg(count('Order ID').alias('Order ID count')))
```

Table

	Country	Order ID count
1	Chad	2
2	Russia	384
3	Paraguay	12
4	Yemen	30
5	Senegal	112
6	Sweden	203
7	Philippines	681

147 rows

```
display(df4.groupby('Product ID').agg(count('Order ID')))
```

Table

	Product ID	count(Order ID)
1	FUR-CH-10003365	7
2	TEC-MA-10001047	2
3	OFF-ST-10000624	19
4	FUR-HON-10001558	11
5	TEC-CO-10002678	1
6	TEC-MEM-10000374	3
7	FUR-BO-10000847	5

10,000 rows | Truncated data

Data Processing and Analysis

```
# Top 5 countries with hghtest quantity
from pyspark.sql.functions import sum, desc, col, dense_rank
from pyspark.sql.window import Window

w = Window.orderBy(desc('Total Quantity'))

top5q = df4.groupBy('Country') \
         .agg(sum('Quantity').alias('Total Quantity')) \
         .withColumn('rank', dense_rank().over(w)) \
         .filter(col('rank') <= 5) \
         .drop('rank') \
         .orderBy(desc('Total Quantity'))
```

```
display(top5q)
```

Table

	Country	Total Quantity
1	United States	58134.36199999998
2	France	10804
3	Australia	10673
4	Mexico	10011
5	Germany	7745

5 rows

```
# Top 5 Products with hightest Order count
from pyspark.sql.functions import count
from pyspark.sql.window import Window

w = Window.orderBy(desc('Order Count'))

top5p = df4.groupBy('Product ID') \
         .agg(count('Order ID').alias('Order Count')) \
         .withColumn('rank', dense_rank().over(w)) \
         .filter(col('rank') <= 5) \
         .drop('rank') \
         .orderBy(desc('Order Count'))
```

```
display(top5p)
```

Table

	Product ID	Order Count
1	OFF-AR-10003651	35
2	OFF-AR-10003829	31
3	OFF-BI-10002799	30
4	OFF-BI-10003708	30
5	FUR-CH-10003354	28
6	OFF-BI-10002570	27

6 rows

```
from pyspark.sql.functions import sum, desc

top5= df4.groupBy('Product Name') \
         .agg(sum('Profit').alias('Total Profit')) \
         .orderBy(desc('Total Profit')) \
         .limit(5)


display(top5)
```

Table

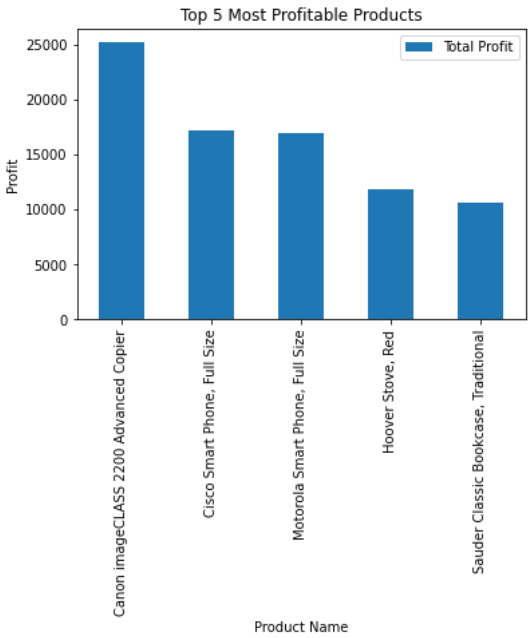
	Product Name	Total Profit
1	Canon imageCLASS 2200 Advanced Copier	25199.928
2	Cisco Smart Phone, Full Size	17238.5206
3	Motorola Smart Phone, Full Size	17027.112999999998
4	Hoover Stove, Red	11807.969
5	Sauder Classic Bookcase, Traditional	10672.072999999999

5 rows

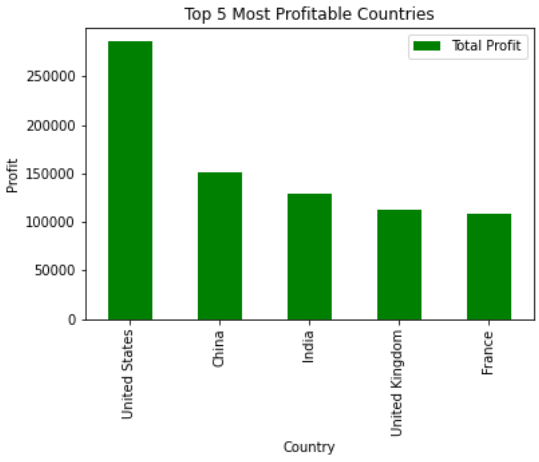
```
type(top5)

Out[83]: pyspark.sql.dataframe.DataFrame


top5.write.mode('overwrite').csv('/mnt/b-8new/top5profit(1)',header=True)
```

Data Representation

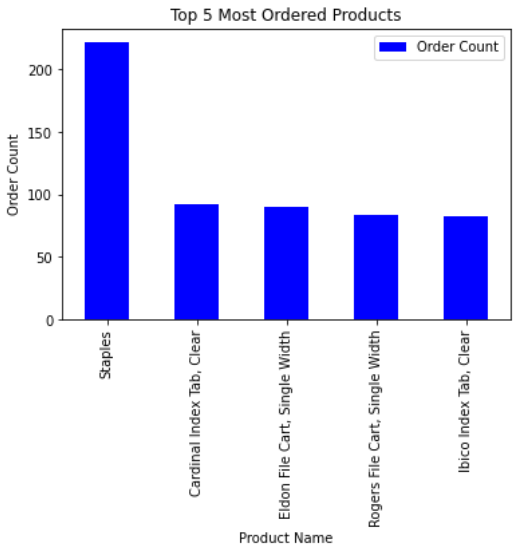
```
# TOP 5 PRODUCT BY TOTAL PROFIT



import pyspark.sql.functions as F
import matplotlib.pyplot as plt

top5b = df4.groupBy('Product Name').agg(F.sum('Profit').alias('Total Profit')).orderBy(F.desc('Total
Profit')).limit(5).toPandas()

top5b.plot(kind='bar', x='Product Name', y='Total Profit', title='Top 5 Most Profitable Products')
plt.xlabel('Product Name')
plt.ylabel('Profit')
plt.show()
```



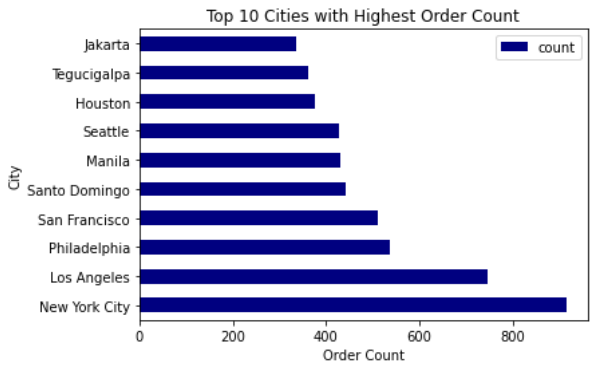

```
#TOP 5 COUNTRY BY TOTAL PROFIT

top5 = df4.groupBy('Country').agg(F.sum('Profit').alias('Total Profit')).orderBy(F.desc('Total
Profit')).limit(5).toPandas()

top5.plot(kind='bar', x='Country', y='Total Profit', title='Top 5 Most Profitable Countries', color='green')
plt.xlabel('Country')
plt.ylabel('Profit')
plt.show()
```



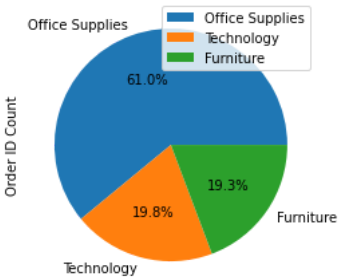
```
#TOP 5 PRODUCT BY TOTAL ORDER


import pyspark.sql.functions as F
import matplotlib.pyplot as plt

top5 = df4.groupBy('Product Name').agg(F.countDistinct('Order ID').alias('Order Count')).orderBy(F.desc('Order
Count')).limit(5).toPandas()

top5.plot(kind='bar', x='Product Name', y='Order Count', title='Top 5 Most Ordered Products', color='blue')
plt.xlabel('Product Name')
plt.ylabel('Order Count')
plt.show()
```



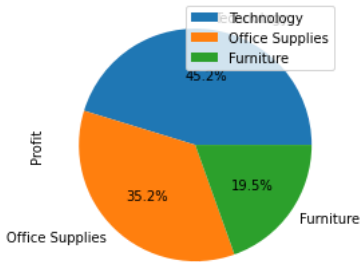

```
# TOP 10 CITY BY TOTAL ORDER

import pyspark.sql.functions as F
import matplotlib.pyplot as plt

top10 = df4.groupBy('City').count().orderBy(F.desc('count')).limit(10).toPandas()

top10.plot(kind='barh', x='City', y='count', title='Top 10 Cities with Highest Order Count', color='navy')
plt.xlabel('Order Count')
plt.ylabel('City')
plt.show()
```



```
# TOTAL ORDER BY CATEGORY

from pyspark.sql.functions import desc
import pyspark.sql.functions as F
import matplotlib.pyplot as plt

pdf1 = df4.groupBy('Category') \
        .agg(F.count('Order ID').alias('Order ID Count')) \
        .sort(desc('Order ID Count')) \
        .limit(5) \
        .toPandas()

pdf1.plot(kind='pie', y='Order ID Count', labels=pdf1['Category'], autopct='%1.1f%%', subplots=True)
plt.show()
```




```
# TOTAL PROFIT BY CATEGORY



from pyspark.sql.functions import desc
import pyspark.sql.functions as F
import matplotlib.pyplot as plt

pdf2 = df4.groupBy('Category') \
        .agg(F.sum('Profit').alias('Profit')) \
        .sort(desc('Profit')) \
        .limit(5) \
        .toPandas()

pdf2.plot(kind='pie', y='Profit', labels=pdf2['Category'], autopct='%1.1f%%', subplots=True)
plt.show()
```



Writing the processed data to Output S3 bucket